

# A literature study on adoption of open source software

## **IS Development and Implementation in a Business Context Autumn 2016**

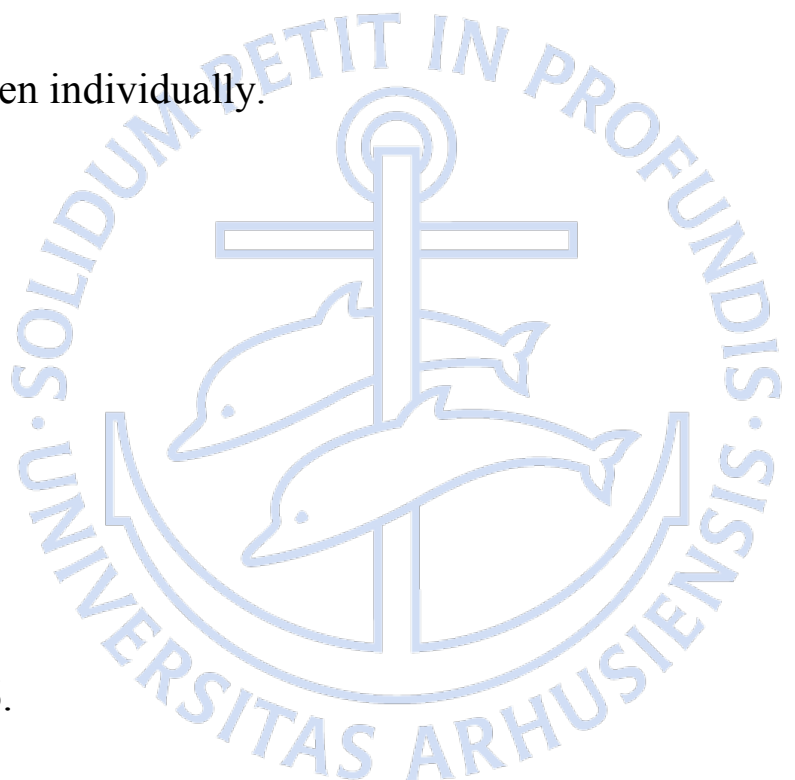
Author: Dmitrij Petrov, Stud. ID: 201601280, Exam ID: 100516

Professors: Nikolaus Obwegeser & Bjarne R. Schlichter

Period: 29.9.2016 – 18.12.2016

This course report has been written individually.

Submitted on 08 December 2016.



## Contents

1	Introduction .....	1
1.1	Research problem .....	2
1.2	Delimitations.....	2
1.3	Structure .....	4
1.4	Methodology .....	5
2	Free and Proprietary Software .....	7
2.1	Free/Libre and Open-Source Software.....	7
2.2	Proprietary software.....	10
2.3	Comparison between two types .....	11
2.3.1	Business Models .....	13
2.3.2	Intellectual property and software licensing .....	14
2.3.3	Benefits of FLOSS software.....	16
3	Barriers of FLOSS adoption.....	21
3.1	Literature review.....	21
3.2	Technology-Organization-Environment(-Individual) framework .....	25
3.2.1	Technological inhibitors.....	26
3.2.2	Organizational barriers .....	29
3.2.3	Environmental challenges.....	30
3.2.4	Individual factors .....	32
3.3	FLOSS in Organizations.....	33
3.3.1	Healthcare sector.....	35
4	Discussion .....	37
5	Implications.....	40
6	Conclusion .....	41
7	References.....	43
8	Abbreviations.....	51

9	Appendix .....	52
---	----------------	----

## Table of figures

Figure 1 shows structure of this report.....	3
Figure 2 shows two categories of software. Some terms depicted here will be clarified at later point in this chapter. ....	9
Figure 3 shows different phases of gathering articles that are reviewed.....	23
Figure 4 displays number of documents per year. ....	23
Figure 5 illustrates different sectors that are included in this review.....	24
Figure 6 summarizes individual barriers from the scientific literature into 4 primary dimensions and 19 secondary factors. An exapnded look is shown in Figure 8 in appendix.....	27
Figure 7 is how my research protocol looks like at the very end. ....	52
Figure 8 provides an expanded and detailed look on categorization of FLOSS integration challenges. ....	53
Figure 9 shows here only an excerpt from the Excel sheet. The full matrix is provided in the supplement. It should be noted that in many cases confusing and improper labels of individual barriers (e.g. “switching costs/TCO/sunk costs”) have been taken into accooount later when creating a final mind map seen in Figure 8. Therefore this is sheet needs to be considered as “very raw” version of my inhibiting categorries.....	54
Table 1 shows a self-made characterization of free and non-free software. “Plus” sign means advantage, “minus” sign is disadvantage and “plus-minus” refers to both depending on perspective for the company (essentially neither positive nor negative). The “target” icon on the left shows whether this characteristic can be or is relevant to the organization. Only those are evaluated, which are relevant for the adoption in the company.....	20

# 1 Introduction

In 2011 Marc Andreessen, an acclaimed investor and co-founder of Netscape browser, famously postulated in his essay that “*software is eating the world*” [O8]. In a piece he rightfully argued that software, upon which for example the Internet is build, is going to reshape every corner of industries and economies, leaving those behind who do not adapt. Not coincidently nowadays for example General Electric, an industrial conglomerate, wants to become “*top ten software company by 2020*” [O15].

Although proprietary (closed-source) type of software is still leading in firm’s software portfolios, its share is decreasing according to the 2010 Gartner survey among over 500 IT leaders in 11 countries [O35] [O2]. On the contrary, what has been rising is the use of so-called *free/libre and open-source software* (FLOSS), which can be viewed as a philosophical movement in the computer science [B3] [27].

Indeed, over the last 15 to 20 years, this category has become a major source and driver of innovation in the information technology (IT) and not only there [O4] [O7]. It has changed the software tools, created new business models, developed new processes of managing communities and engineers and last but not least gave new possibilities of litigation in courts [O28]. Software products such as Linux Kernel, Apache Hadoop or Google’s Android (AOSP) platform have altered many aspects of everyday life and launched new opportunities – be it in the cloud & big data, mobile or “Internet of Things”. All these projects grew to an unparalleled popularity, proved by their market share, and sparked creation of new businesses offering complimentary products and services.

What differentiates among many other aspects FLOSS programs from their proprietary counterparts is having their source code – machine instructions – available on the Internet, in the plain text. Thus, everybody can download and use such software free of charge and be able to modify it to further improve its functionality. With this core feature, free software became an integral and vital part of the 350 billion USD software industry [O21]. As a result, many start-ups and even large enterprises such as Huawei or Google today almost entirely rely on open technologies [O28].

As noted in journal articles by [20], [25], [15] or [57], the field of open-source research is still relatively young and many of its aspects such as FLOSS adoption<sup>1</sup> in various domains such as finance or education continue to be nowadays not thoroughly investigated in the empirical

---

<sup>1</sup> In my study *adoption* refers to the deployment and use of FLOSS technologies in the actual organizational environment (the software).

literature [27]. Nonetheless, because OSS becoming in recent years an emerging topic among both researchers and practitioners, many articles have been written analyzing and describing how to adopt this new software paradigm into the organizations.

Even though largely there is an agreement that benefits of free software (e.g. being gratis to use) overweight its costs (e.g. for the enterprise support and organizational change), FLOSS persists to be only partially adopted in many large, proprietary-dependent, organizations. Thus, as the use, quality and availability of open-source have risen dramatically [25] [67], there is a stream of literature which analyses these adoption challenges in various contexts and from different angles. Correspondingly, in this study I want to conduct a literature review on the subject of what barriers companies in the public and private sector face in order to successfully integrate such free software in their environments. Based on existing studies that explore these inhibiting factors, which prevent FLOSS from becoming the ultimate choice, my goal is to attempt to structure them in a new fashion based on the *Technology, Organization and Environment* (TOE) framework [59].

### 1.1 Research problem

Given on my objective, my study is focusing on answering following three research questions which are developed:

RQ1. What advantages and disadvantages FLOSS offers compared to the traditional closed-source software?

RQ2. What hinders its adoption in organizations?

RQ3. How does healthcare industry perceive such a new software paradigm?

### 1.2 Delimitations

To stay within the boundaries of the course “Information Systems (IS) Development and Implementation in a Business Context”, it is necessary to clarify and restrict this research study.

At first, even though there are differences between *open-source software* (OSS) and *free software*, which I later explain in chapter 2.1, I am considering them – for my literature study – equivalent in meaning in order to remain at a higher level of perspective for my objective. Hence, here I do not further investigate differences in these two terms as many researchers use both phrases interchangeably. In fact, to overcome any potential issues, I use expression *free(/libre) and open-source software* (FLOSS/FOSS) that captures both contexts too.

Furthermore, when talking about integration of FLOSS into firm’s environment, for my literature search I do not delimit myself to any specific industry, business sector or geographic region and a

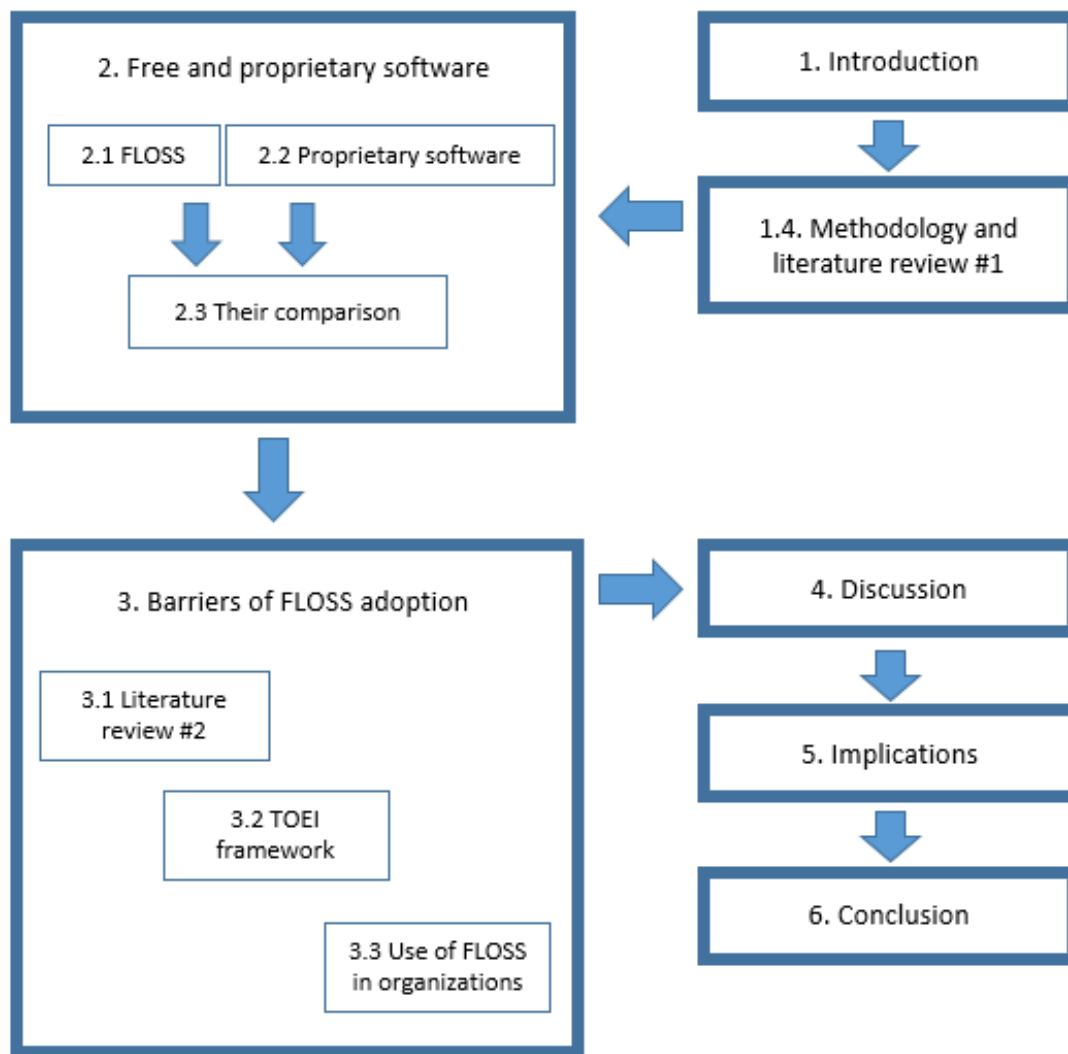


Figure 1 shows structure of this report.

country. As a consequence, due to zero costs for the right to use FLOSS technology, this study also includes works by researchers from developing countries because firms and public institutions have there a strong financial motivation to adopt such innovation [53] [74].

Moreover, contrary for example to articles by [20] or [25] who focus themselves on “*software-intensive organizations*” and small and medium-sized business, I also do not restrict myself to a specific group of organizations which are my unit of analysis. Meaning, my study includes also those whose primary business activities are not related to the development of software, e.g. in the healthcare or public sector. In these industries, the software merely supports such enterprises in conducting their day-to-day operations and it is only used for internal purposes with no goals of additional commercial or non-commercial distribution.

As described by [45], companies can benefit with FLOSS in five various ways. However, for my study, I am not going to focus on adopting development practices (so-called “inner-source”),

creating a relationship with communities or publishing internally developed software as an OSS. On the contrary, I delimit myself only to adoption of free computer programs for corporate applications and their development processes (i.e. using FLOSS during software development).

Lastly, I narrow myself to barriers which are described and relevant for the primary adoption of FLOSS. Whereas this type, as explained by [15], [12] and [54], is about firm's initial decision happening at the organizational level ('managers go ahead'), the secondary adoption would on the contrary involve individuals embracing such free software in the actual implementation process. Given that adoption of free software in many cases happens bottom-up [74], from IT professionals, my goal in this study is to investigate what challenges managers see in their initial reasoning about FLOSS integration [20] [27].

### 1.3 Structure

This report, containing both exploratory and descriptive elements, is divided into two major sections as depicted in the Figure 1. The goal of section two is to provide the reader a point of reference for two major categories of software, namely the free/libre and open-source and the proprietary one. By introducing necessary terminology to establish a basic understanding of the field, I outline important characteristics in order to later be able to compare them and derive benefits and burdens FLOSS can bring to an organization. This is done by developing a table, subsequently addressing the first research question at this stage. Therefore, this section will be to a large degree narrative and descriptive in nature.

Here my primary sources will include various books and other historically relevant literature by authors such as Richard Stallman and Eric Raymond [B5] who were at the beginnings of open-source movement, back in the 1980s and 1990s. These first-hand, original materials supplemented by interviews and video recordings [O36] [O23] will provide the reader a deeper insight and an understanding into the software industry, focusing myself heavily on FLOSS category.

Furthermore, secondary sources for example by [B4] and [B6] that analyze, interpret and discuss above materials are used to gain a broader overview on the subject [O22]. This is mainly applied for the comparison where important features of free and proprietary software are identified, compared and finally presented, see Table 1. This information comes from the reputable journal articles and by authors such as [37].

Based on such table, chapter three then proceeds to the core of my report and asks what the barriers are that hinder the use and adoption of such non-proprietary computer programs in the enterprises. Thus, in light of research question number two, by extending previous exploratory literature

research through a collection of new both primary and secondary resources, the study discusses challenges which can influence FLOSS adoption decision. To structure these barriers, a small extension is applied to the TOE framework that “*identifies three aspects of an enterprise's context that influence the process by which it adopts and implements a technological innovation*” [60] [59]. Consequently, by mapping these challenges to the TOE(I) theory, a new Figure 6 is derived showing what inhibiting factors various authors in the field have explored and found out. Next, my literature study briefly investigates what the current level of FLOSS institutionalization in corporations is and also discusses adoption of open-source software in the healthcare industry.

Lastly, before wrapping-up this literature review with a conclusion, chapter three leads to a discussion of my limitations and outcomes and analysis of implications for both practitioners and researchers.

#### 1.4 Methodology

In order to answer questions outlined in section 1.1, a methodology of this study has to be established. Indeed, as seen in Figure 1, the literature search is conducted twice. For the chapter two, which has only a purpose of introduction to the topic, I do not follow any replicatable process as I mainly explain and describe the field. Thus, my reliability in this chapter – albeit supported with highly-cited scientific literature coming from primary and secondary sources – suffers from not being systematically repeatable. In addition to this, there is the subjectivity and a bias due to my previous professional and current FLOSS developer experiences. As the main goal of the second chapter is to compare both types of software from a point of view of the organization, I base my findings on different sources used in previous sub-sections.

In the chapter three, however, the study attempts to closely follow guidelines that are “*particularly tailored for information systems research*”, written by [58] and [61] in which authors describe how to conduct a systematic literature review in the IS context. My study being (potentially) a starting point for all scientists in the field of FLOSS adoption, it has to exhibit certain characteristic – mainly in terms of scope and rigour – in order to be considered of high value and providing a foundation for the subsequent research [58]. Indeed, such a literature review has a mission to gather available knowledge and evidence from the field and organize and present this information in a new systematic way to other researchers [61].

According to [58], scientists have to follow these eight steps which I also describe in the context of my study:



1. Purpose of the literature review: The goal is to identify, name, and organize studies on adoption of FLOSS and its barriers for the organizations.
2. Protocol: Being my limitation, working alone on this study, no further agreement between several researchers is necessary. However, a protocol and a database of the research is naturally established, see Figure 7 and Appendix 9.
3. Search for the literature: After describing why do companies need to consider free software in the first place – going towards my goals – it is necessary to be transparent, detailed, consistent and rigorous in the choice of the literature. [RQ1]
4. Practical screen: What are criteria for the inclusion of a particular document?
5. Quality appraisal: What are criteria for the exclusion of these papers?
6. Data extraction: Once studies have been thoroughly read, papers are categorized and ‘coded’ so that data (i.e. OSS adoption barriers) can be extracted. [RQ2]
7. Papers analysis: By using and extending TOE framework, my objective is to organize barriers in a structural way – i.e. those previously extracted from the task six. [RQ3]
8. Writing the review: The aim is to report above steps in the sufficient detail.

As a result of abovementioned goals and steps to conduct, my literature review examines qualitative data from studies using a general inductive approach [68]. This, according to Thomas (2006), primarily uses “*detailed readings of raw data to derive concepts, themes, or a model through interpretations made from the raw data by an evaluator*” [68]. Thus, from the sheer size of the available literature which a sample of is gathered and inspected, the inductive approach allows me to find “*frequent, dominant, or significant*” themes with relation to adoption barriers that scientists have been able to observe in their findings and across many studies [68].

Subsequently, by coding my data, I categorize these underlying integration challenges based on the technology, organization and environmental framework [59] [68]. Therefore, it will allow me to gain a meaning from a complex set of literature works “*through the development of summary themes*” presented by individual barriers [68]. Ultimately, from the raw data and based on the TOE framework, I develop categories that “*convey key themes and processes*” which inhibit companies to integrate such free software [68].

More information on how exactly this literature review is done is provided in section 3.1.

## 2 Free and Proprietary Software

Because of various terminologies used across open-source research and in this study, it is first necessary to define and distinguish these, relatively similar, terms. Subsequently, the goals of this chapter are to characterize free/libre & open-source and proprietary (non-free) software and be able to compare them. In doing so, the chapter should answer a key question: What benefits FLOSS has compared to the traditional closed-source computer programs used in companies?

### 2.1 Free/Libre and Open-Source Software

The open-source started to gain traction in the mid- and early-1980s, right at the time when personal computers become accessible and could be used by the wider public [O28]. All along the focus of the new and rising industry has been on the hardware where companies were fiercely competing and software was seen just an add-on on top of it [B4]. Therefore, manufacturers encouraged technicians and scientists to share their improvements to programs with a hope of benefiting clients who would become more loyal to these builder corporations. Due to the Internet not being widespread and software written for a specific machine and its architectural platform, “*it was therefore in the manufacturer's [own] interests for machine-specific code and knowledge [about it] to spread as widely as possible*” [B4].

Yet as the IT industry was slowly taking off with innovations both in hardware platforms and in how the software has been produced (so-called ‘write once and run everywhere’ paradigm), the sharing of it became undesirable [B4]. Producers of hardware started to enforce their copyright so that their customers – not being able to inspect and modify the source code – could not improve the computer programs themselves. Consequently, by using a proprietary licensing, clients of such hardware would be forced to pay for additional software features and/or bug fixes.

#### STALLMAN’S BELIEF

In January 1984 a programmer named Richard Stallman resigned, being prevented from modifying a computer program of the new printer, from his position at MIT AI Lab in order “*to develop an entire operating system, including all associated software, that would not be subject to proprietary licensing*” [O24] [B6]. In fact, because of developing hostility to such closed-source works, he invented GNU General Public License (GPL) family which ensures that programs would remain ‘free’. But not in a sense of ‘free beer’, but rather it would stand for the freedom which it gives to the user [B2]. To realize his newly formed belief in what he called *free software* and accomplishing his new goal of developing entire operating system, he founded Free Software Foundation (FSF) which would promote such ideals and computer innovation.

Free software, as defined by FSF and Richard Stallman, gives their users four essential freedoms:

“the freedom to run it, to study and change it, and to redistribute copies with or without changes” [O29]. However, even though many computer programs are developed publically on the Internet with an access to the source code, they are inherently not free. As an example the founder of FSF talks about systems such as Digital Rights Management (DRM; on the web called *Encrypted Media Extensions*<sup>2</sup>) which restrict abovementioned rights by not allowing the user to change or copy the software or other media [O29].

By the end of 1980s, there were many projects and communities which shared a similar ideology, namely that a source code should be available for everybody free to share, modify and redistribute [B4]. Thus, usually these projects were licensed under the GPL conditions. Because of Richard Stallman’s ideology which (still) considers proprietary source code to be injustice to the user imposing on him malicious functionalities, already at that time many other developers didn’t share his ‘radical’ point of view [O30]. Nevertheless, even today they continue to work on these projects together because in many cases they are able to produce software of very high quality compared to their proprietary counterparts offering same functionality [B4].

As the world turned more attention to this new paradigm, companies were also willing to devote their time, hardware and financial resources to help developing and maintaining such free computer programs [B4]. However, what seemed to be unfortunate naming, developers had to explain that ‘free’ doesn’t stand for the price of software (although it is indeed zero cost) but rather for its freedom, the liberty (from Spanish *libre*). Therefore, many companies which obviously have a commercial purpose have either rejected the freedom part (and thus ignored such software completely; until 2014 including Microsoft) or were faced with the dilemma where they “*wanted to support particular free programs in one aspect of their business, but continue marketing proprietary software in others*” [B4] [B2]. Hence, ‘free’ became a term with an intentional “*anti-business message*” and further split the community into two camps [B2] [O17] [57].

## OPEN SOURCE: A ROOF FOR FREE SOFTWARE

The other community around programmers including Eric Raymond with Tim O’Reilly didn’t share Stallman’s views with regard to the injustice to the user as they have never perceived it that way [B6]. Their desire grew to create an alternative word which avoids using ‘free’ due to misunderstandings while at the same time keeps the spirit of such software making it “*viable in the business world*” too [B4] [14] [27].

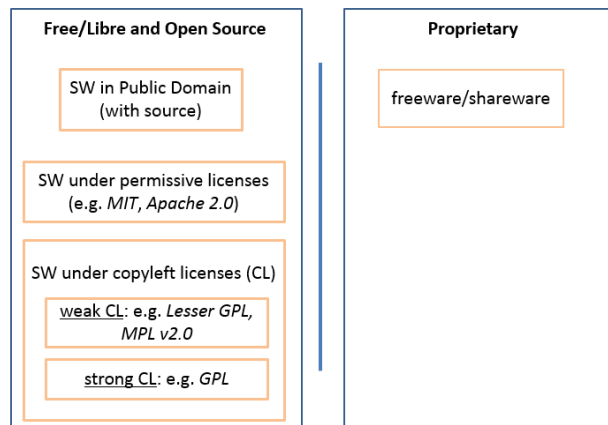
For marketing purposes, in 1998 they coined a new phrase ‘open-source’ and founded Open Source

---

<sup>2</sup> “Encrypted Media Extensions”, <https://w3c.github.io/encrypted-media/>, Retrieved on 08 Dec. 16

Initiative (OSI) to promote such computer programs in the society without enforcing moral binding contracts with freedom which FSF does through the use GNU General Public License [B4] [O32]. OSI defined that in order to be called the ‘open-source’, a license to the accompanying source code must fulfil their ten requirements<sup>3</sup> and be approved by them [O28]. Besides this legal definition, open-source – looked from a different angle – can be also seen as a development process which uses “*practices of open collaboration for superior results*” [O28].

## 2 Categories of Software (SW)



Self-made but inspired by <https://www.gnu.org/philosophy/categories.html> and Macredie & Miiinyawa (2011)

Figure 2 shows two categories of software. Some terms depicted here will be clarified at later point in this chapter.

Therefore, as noted in my delimitations, strictly speaking *free software* is not the same as *open-source software* [9]. These terms represent two complementary philosophical viewpoints and in fact FSF considers free software (the one ‘under GPL’) to be a subset of OSS, as depicted in Figure 2.

Because the aim of this study is not to deal with these differences in the corporate adoption, an umbrella term *free/libre and open source software* will be preferred to stay neutral while together with separate terms, this expression will be used for meaning the same kind of computer programs. Namely ones that have their source code accessible in the plain text under the open-source license which “*grants individuals, groups, and organizations extensive rights to use, modify, and redistribute the binary and source-code of the original and modified/derived works, without requiring license royalty fees*” [25]. Here my choice is also consistent with other authors such as [20] [57] [85] [86] who agree that there are only minor differences in the meaning and use these terms synonymously.

## FLOSS GROWTH

Ever since founding FSF and OSI, open-source movement has had a significant impact on the whole IT sector: from low-level programs (Linux Kernel, Debian operating system and Android Open Source Project) over web browsers (Firefox and Chromium) to high-level, large scale

<sup>3</sup> “The Open Source Definition”, <https://opensource.org/osd>, Retrieved on 08 Dec. 16

applications such as learning management or enterprise resource planning systems (e.g. Moodle, Dolibarr). What have been until a few years ago unheard of, nowadays software producers such as Microsoft or Netflix release their formerly proprietary computer programs on the Internet in order to spur a deeper community-driven engagement and share their innovations together.

Besides the recognition in the private industry, the public sector – through its many initiatives such as *Gov 2.0* and *Open Data* – has acknowledged this type of technology too and started to embrace and engage with the open-source principles, communities and even trying to adopt them in its governmental procurement [O33] [O5] [16] [8] [53] – albeit facing many challenges.

In 2007 it was estimated that over 800,000 contributors participate in the open-source development [9]. Today, this number is in tens of millions as let alone GitHub.com – the most popular FLOSS web service – hosts over 35 million open-source projects supporting over 15 million members worldwide<sup>4</sup>. Additionally, this must be counted together with many thousands of developers programming projects hosted on private servers (in the open) and other online services such as GitLab.com and SourceForge.net.

As observed, open-source has grown from a purely a hacker-driven cultural ‘thing’ to a widespread part of the software industry and IT at large. As a proof that one is capable of creating revenue streams from FLOSS is a company called Red Hat Inc. which for the full 2016 fiscal year became first \$2 billion OSS enterprise [O34]. Indeed, as [4] further show companies “*do not consider lack of appropriability as an obstacle to profitability*”. What is even more important is that according to the 2015 study, corporations see nowadays FLOSS as a strategic asset which enables them to unleash their speed and agility in the ever competitive markets and boosts their innovation in creating new products and services for their customers [O4] [53] [86].

## 2.2 Proprietary software

On the opposing side of the river, there is non-free, so-called proprietary software. To paraphrase [B6], due to the fact that source code is kept secret, users cannot change and redistribute such computer programs as the end-user license agreement (EULA) they acquire and have to accept does not permit them doing that. This means that software vendor, which can be both an individual as well as a company, provides or otherwise sells the end-user an intangible work that has to be used within certain boundaries and under specific conditions [32].

An early and one of the most famous examples of proprietary software is MS-DOS by Microsoft. In the hacker community there is a powerful principle that sharing information, incl. computer source

---

<sup>4</sup> “About GitHub”, <https://github.com/about>, Retrieved on 08 Dec. 16

code, is ethical and ‘good’ [O19]. However, Bill Gates and Paul Allen, who in the mid-1970s wrote and sold their first computer program to the MITS Corporation, turned this paradigm upside down [O27]. In 1976 Bill Gates, finding out that their sold program was pirated by these hackers [O19], published “An Open Letter to Hobbyists” arguing that software should not be shared without author’s explicit permission. Since then, he developed his company to be a leader in commercializing proprietary source code.

It is important to note that terms ‘commercial’ and ‘proprietary’ have different meanings. Commercial software means that it has a purpose of being sold to the interested third-parties. This can happen for example when the software is offered on the retail shelf or the open-source license does not fit a customer and he requests a commercial one from the vendor. For the latter case, these commercial licenses are usually offered by the same developers and companies who contribute to the open-source version. And because there can be many organizations contributing to the same OSS, customers can choose between several of them and don’t be bound to a single one as it is with the proprietary software vendor. An example of such behavior is Collabora Inc.<sup>5</sup> which offers its own, enterprise ready version of office suit called *Collabora Office*. Interestingly, in one-year period between March 2014 and 2015, developers employed by this company have contributed close to 30% of all code changes in the LibreOffice, an open-source version upon which Collabora Office is build [O13].

Even though it may seem that proprietary software is very homogeneous, there exist different types of it. A commonly known one is ‘freeware’ which has nothing to do with freedom but this time rather with its price. It is in fact distributed gratis to everybody albeit may be limited in terms of functionality. Yet, because the source code is (still) not available, what has been attempted many times to various proprietary programs such as Skype<sup>6</sup> is to reverse-engineered them. Meaning go backwards in the software development life-cycle and create own open-source implementation with regard to non-free functionality. This is of course usually, depending on the country and EULA license, not legally permitted.

### 2.3 Comparison between two types

In 1999 Eric Raymond, who first published a paper and later a book called “The cathedral and the bazaar” [B5], contrasted two development models of open-source he was able to observe. Namely, traditionally “*led by command and control approach*” software projects have been developed as a ‘cathedral’, i.e. they were carefully crafted by a small group of developers [O28] [38]. Such software,

---

<sup>5</sup> <https://www.collabora.com/about-us/open-source/open-source-projects/libreoffice.html>, Retrieved on 08 Dec. 16

<sup>6</sup> <http://skype-open-source2.blogspot.dk/>, Retrieved on 08 Dec. 16

potentially released even with the final source code, was only then published when the work was really completed because usually the development process itself was closed. However, as [B5] saw on Linux Kernel, the development style of open-source became more like a ‘bazaar’ type of activity. This means that it was developed incrementally and evolved rapidly with the mantra “*release early, release often*” [O28].

Moreover, another characteristic of software such as Kernel is being developed by a large, geographically dispersed community of experts in their respective fields for example in networking, audio and video [O28]. The communication between these developers happens on various virtual channels such as mailing-lists and IRCs which are publically accessible and due to the written form searchable too [O28]. And this is also what distinguishes FLOSS from the traditional style of development and communication. On the one hand firms creating proprietary software naturally follow a development process which is hidden, hierarchical with top-down decision-making and centers on assigned tasks to individuals. But on the other hand, in the FLOSS communities a meritocracy (all decisions are publically and openly discussed and “*contributions are judged transparently on the basis of their merits*”), egalitarianism (everybody can contribute anytime) and self-organization (people finding their projects and the way they contribute in the development process on their own) is leading the way forward [38] [O28].

Such an *open collaboration*, a term Riehle et al. (2009; 2016a) propose, makes therefore open-source software being a distinctly unique artefact which has prompted new research in it, from different angles such as the nature of collaboration between actors or types of relationships between contributing organizations and larger community [31] [20]. The research is in early stages as noted by many scientists but already based on widely known examples such as above-mentioned Kernel, studies suggest that common assumptions of how the software should be developed, having worse quality than the proprietary one and non-existing support are in many case unjustified and false [O14] [1] [14] [29] [26] [37].

[79] [92] [93] [14] [9] [32] further provide a detailed look into many of FLOSS characteristics which however are beyond the scope to describe and explain in this paper. Therefore, following two subsections examine selected and important features of free and non-free computer programs with regard to business model and software licensing. Finally, in chapter 2.3.3, a Table 1 is provided summarizing contrasts with additional elements from literature while also benefits of FLOSS are discussed.

### 2.3.1 Business Models

Traditional proprietary software has a very ‘intuitive’ business model. If the end-user wants to use a certain product – due to the fact that the development is kept closed and unless it can be downloaded free of charge – he has to pay a license fee to the vendor because the software is not freely distributed and the source code is not available either [O28]. However, with the open-source – if it is on the Internet and everybody can download and use it gratis – a question arises how can a company create a sustainable business model?

In the above-mentioned (very simplified) situation, there are just two players – the vendor and the user – which interact with each other. Yet, according to [O23], in the open-source community two additional actors exist. Firstly, there are independent contributors who develop FLOSS in their free time and secondly there are different user and developer foundations which provide a basic infrastructure and governance processes for multiple software projects. Hence, the demander of software can look for FLOSS not just at businesses but such foundations too. In comparison to the non-free software environment, vendors employing own developers with no outside contributors have to act to a degree as a foundation themselves in order to foster their proprietary ecosystem e.g. by providing an access to the API.

As reported by [O20], three main types of revenue streams from open-source software can be distinguished. At first, there is a model where a foundation produces FLOSS and multiple firms are building commercial products around it [O23]. Looking at Hadoop<sup>7</sup>, while Apache’s version is free to all, versions from companies such as Hortonworks or MapR are not. In fact, they are additionally bundling many commercial add-ons and offering training and support services that enhance the basic functionality of the FLOSS product. This is firm’s way to differentiate in the marketplace even though they all draw from the same upstream version of open-source Hadoop.

The other business model is a ‘singular’ one where an open-source project has usually one vendor which commercially supports and develops it further. For example, this applies to the Automattic’s WordPress CMS. The business model here is to provide ‘best binaries’ to all in order to create the community-driven engagement. Subsequently, if third-party wants to have premium plugins, legal warranty and superior maintenance, it can purchase these services separately.

Last but not least, another example of how a company producing open-source product can create a revenue stream is RStudio Inc., which develops an IDE for R programming language<sup>8</sup>. Consistent with what [35] describe, it offers its server version of IDE with two types of licenses – the first

---

<sup>7</sup> “Apache Hadoop”, <https://hadoop.apache.org/>, Retrieved on 08 Dec. 16

<sup>8</sup> “RStudio”, <https://www.rstudio.com/products/rstudio/>, Retrieved on 08 Dec. 16



edition is available under the open-source terms of AGPL v3 license while the other is a commercial version which has additional features such as enhanced security, administrative dashboard and priority support for a price of \$9,995 per year. Indeed, here the company chose a hybrid strategy as it is differentiating its product between types of use by withholding certain features in the open source version. The commercial one is clearly aimed at corporate environments where for example single-sign-on and encryption are necessary.

These hybrid business models such as *dual-licensing* above have been previously studied and described in the literature [53]. While [21] and [35] give a rather broader overview how to set up a business-oriented open-source company and what are its potential revenue streams, [47] studied different hybrid business models of IBM, Apple and Sun and how they evolved their proprietary systems for the new era of open-source. [5] later conducted a study among 275 Italian open-source firms where they confirmed that those companies mainly choose hybrid types to “*profit from both the two production paradigms*”. This is also validated in a study by *The 451 Group* [O31] and again by [4]. Later [46] discussed how JBoss J2EE software has adapted its business model in several phrases from providing purely training services over the consulting, mission-critical support and growth through an ecosystem with partners.

At such, due to inherently richer structure including having four different actors, FLOSS allows companies to differentiate themselves in wider sense, and subsequently this impacts the choice of the business model as well. Indeed, [35] have investigated various business forms and found out that license of the software, developer communities and unique features of product and the market play a significant role in the choice of the business approach for the open-source company. These in fact usually pursue multiple of them at the same time.

As the industry and research shows, because the FLOSS lacks license fees for the right to use such programs, businesses have been moving towards service-oriented models for example in a form of ‘Software-As-A-Service’ and offering clients enterprise support, training and consulting services [20] [21] [O18]. Not so long ago, a new trend has emerged too when companies both producing proprietary (e.g. Adobe’s Creative Cloud) and open-source software (e.g. Red Hat Enterprise Linux) have started to use subscription-based pricing [35] [13]. Therefore, as can be observed, being an open-source player doesn’t mean to have a static business model [20]. On the contrary, these companies have to constantly adapt their revenue stream in order to satisfy their clients and OSS communities.

### 2.3.2 Intellectual property and software licensing

Because software – being a virtual intangible good – can be easily copied, the law offers three types

of protection – the copyright, trademark and patents [O28]. The first one is very much relevant to the IT industry because it gives the right to author to exclude somebody from reproduction and modification of the medium he created. At such a source code is usually automatically copyrighted and a software license is needed to give the licensee a permission to use it [O28].

The most common proprietary license contract between the end-user and manufacturer is so-called *end-user license agreement*, an example of such is Microsoft Windows EULA [O3]. This spells exactly what is permitted to do with such software and what are restrictions in its use.

Unfortunately, for many proprietary programs these licenses are not even publically available and can be additionally negotiated between the parties themselves [B1]. In the example mentioned above, however, if the user doesn't agree to certain parts of the license during the installation process of the operating system, then he is simply not allowed to use this computer program. Furthermore, any kinds of reverse engineering are not permitted and the user is also not allowed to redistribute or resell such software. In an accordance with these and many other limitations, software licensing (a right to use such computer programs) become *the* way of having a sustainable revenue stream for companies that commercialize proprietary software.

In the FLOSS community, software licensing is a legal field in itself as “*OSS license are surrounded by a lot of confusion*” because developers (who use them) are not legal experts [42]. Whereas in the proprietary products the situation is fairly simple due to explicitly written and signed contracts between parties, in the open-source environment (where in a base case no contract and signature is needed), the choice of the right license for the company “*can strongly influence the return on software investment*” and correspondingly its growth [13] [O31].

Open-source software licenses are public documents accepted by the OSI and grant “*the right to receive, display, perform, modify and redistribute some piece of software free of charge*” [O28].

Nonetheless, besides giving certain rights, open-source licenses can put obligations too, for example always requiring attribution and inclusion of the license document in the proprietary source code. Unfortunately, there are many different licenses and their different versions and hence ‘mixing’ multiple distinct licenses can lead to legal errors, with severe financial consequences.

In principle, two families exist: the reciprocal (‘copyleft’, ‘viral’) and permissive one, see Figure 2. The later ones, for example the MIT or Apache v2.0, allow such computer programs to be included and redistributed in the proprietary software. Additionally, by being business friendly, it allows enterprises to take e.g. Apache's Hadoop, modify it and commercialize on the market without ever open-sourcing modifications which have been applied to it. While at the beginning of the open-

source movement, the trend was to have more restrictive licenses (described below), today it is the opposite as many statistics of open-source projects show [O10] [O28] [O6] [O9].

The reciprocal licenses are far more restrictive than the other type because they place specific conditions on the use of such software in other programs. Especially, they intend “*to keep software from becoming proprietary*” and to be even included in such applications [O24]. The GNU General Public License family is the most famous example of “*explicitly require[ing] that derivative works be distributed under the terms of the GPL License and also that derivative works may only be permitted to be distributed under the terms of [this] license*” [B1]. In practice, any kind of modifications to the GPL software must be publically available (‘copyleft’ principle) and GPL software cannot be included in the software “*that does not use the same [viral] license*”, with some exceptions such as LGPL v3.0 [42]. Consequently, developers argue that these copyleft licenses impose a freedom on software which is as bad and “*evil*” because once these licenses infect a part of the software everything else has to be open-sourced as well [O25]. If this is not done, and later found out, costly litigations can take place [O16].

### 2.3.3 Benefits of FLOSS software

To conclude this first chapter, in my research statement I have asked what are the advantages and disadvantages of FLOSS compared to their proprietary counterparts. My goal was also to outline relevant characteristics of both types and compare them. As result, I present in Table 1 how free and non-free computer programs can be described, with regard for example to their communities, security and development style or a business model of the vendor.

From a point of view of the enterprise, which has still to adopt (or migrate to) such an open-source software and wants to keep adequate security and quality experienced from the proprietary products, it is important to extensively evaluate individual advantages and disadvantages of such move, as presented for example by [27]. Therefore, I assessed and additionally added (where applicable) “plus”, “minus” and “neutral” signs to articulate whether a characteristic or a feature could be beneficial to such company or not. And even though, benefits of open-source software (slightly) prevail both in my examination as well as in other literature studies, each company has to evaluate FLOSS advantages based on its own activities, needs and relevant factors [37] [26].













For example, significant determinant for a management decision about FLOSS can be costs savings [O1]. These, usually in a form of licensing rights, can be indeed substantial, however, a total costs of ownership (TCO) of such software can vary significantly (be both lower and even exceed of the proprietary one due to additional services in the support, consulting and training for the employees and need for proprietary components) [27]. With regard to the various studies conducted on this







topic by authors such as [41] or [28] and (marketing) reports published by companies e.g. [O26], managers have to critically assess their objectivity as each camp may lobby its own preference(s) [37] [14].

Nonetheless, authors such as [B7], [O12], [37] or [O1] agree that open-source software allows companies to leverage agility, flexibility and speed of the development which is found in large and important community projects such Debian OS and LibreOffice. With adoption of these free programs companies benefit by gaining a greater customization ability and due to open-source code they are allowed to hire developers to implement company-specific, closed-source, modifications. Because using such free/libre and open-source software is gratis, firms can quicker evaluate software's possibilities by starting small and increasingly growing its adoption in internal applications ('trialability') [O1]. Once they later require commercial, mission-critical, support they can freely choose usually from several providers, thus avoiding complex negotiation processes with a single company. Furthermore, they also avoid vendor lock-in as the commercial open-source software such as Red Hat Enterprise Linux is built upon free-software and offers community maintained versions as well (e.g. Fedora) – in case of a downgrade.

Finally, as suggested by [37] and second by [26], "*open-source model appears [to be] most viable for generic context-independent applications*" such as office suits, web browsers or content-management systems. Authors continue by saying that if on the other hand company highly relies on the custom-made software fulfilling industry-specific requirements, the use of open-source can be only partially beneficial – mainly in terms of permissively-licensed technologies that do not require open-sourcing such distinct systems. Overall, they conclude that "*there is no clear-cut dominance of open-source over proprietary software or vice versa*" [37].

For the organization that wants to adopt such an OSS – in principle an IS innovation [2] – a careful consideration of various types of factors mainly technological (software security, quality, reliability and stability) but also business ones (above-mentioned total costs of ownership, credibility and availability of vendors offering commercial support, responsibility for legal liability) have to be considered [23] [O1] [O11] [22].

Features and characteristics	Free software	Non-free software
Primary actors in the ecosystem	Vendors (which employ own developers) End-users Independent developers/volunteers Software Foundations	Vendors End-users
Source Code 	Full public access, either under permissive or copyleft licensing 	Usually kept secret, with restrictions in how to use the final program. No rights to copy, modify or redistribute.
Standards [44] 	Will implement open, industry wide, standards but unless closed standards are provided at least in a binary form, compatibility with closed standards may not exist (or be easily developed) 	Will implement proprietary standard (e.g. “docx”) and can (partially at least) implement open ones as well (e.g. “odt”).
Development style	“bazaar” type of activity Decisions discussed in public – accessible to anybody anywhere. Hence processes and style are transparent, meritocratic, egalitarianistic, self-organizing	“cathedral” type Decisions are internally discussed, without outside influence. Imposed processes and tasks, status more relevant than merits.
Coordination & Communication 	Open style, most of it is in public domain. Developers choose what they want to work at themselves, but can lead to duplicate efforts, lack of proper UX/UI and documentation. 	Hierarchical, usually very formal and top-down coordination to avoid overlap  Developer get their tasks assigned
Providing contributions & further development of software 	Potentially from all involved players. Judged based on their quality/merit. Anybody can do it if following rules (if any exists). 	No outside contributions are allowed – only the vendor develops it further.
Frequency of new program releases 	“release early, release often” paradigm (nightly builds) 	Depends, but usually there is a time distance in months/weeks between them
Costs for the user, i.e. cost for the right to use the software [37]. 	Gratis for download and thus attractive for him, albeit can come with some restrictions/limitations in use 	Unless it is freeware or shareware, users pay a price to offset costs of the R&D and development. The price

	due to specific license or a feature set. TCO can exceed the one of non-free SW + -	differs based on many factors that are set by the vendor.
Feedback from the user 	Because bug systems/mailling lists and other tools are public, constant flow of feedback in form of reporting issues and suggestions. Can however overwhelm developers and distract them from work. 	E.g. depending on the size can be limited and very fragmented (e.g. on Reddit, Facebook, Twitter, privately per email etc.). Thus, usually intensive small group meetings with users are conducted.
Number and dedication of developers [18] [17] 	Depending on project's size/popularity can be a large motivated group. Yet, overall less dedication especially if not employed to work on it full-time. 	Limited by firm's resources. But given that it is their daily job, they are (should be) fully dedicated to it.
Incentive for developers [39] [3] [18]	Recognition, respect, tournament for the most and best contributions → immediate rewards. Gaining experience, skills for future employment → delayed rewards.	No competition between developers but monetary incentive to deliver the work.
Business Model & Commercialization for software vendor	Dual licensing (commercial and community edition) Offering complementary plugins/extensions or services Commercial support, consulting, training	Perpetual licensing (e.g. per socket, per core or user)  Subscription-based  Commercial support, consulting, training
Customer support provided by either software category 	Online and free from community of users and developers. Documentation, manuals, forums and chats exist where user can seek free help. Separately, a commercial variant can be purchased too. + -	Usually with a purchase of such software some form of ongoing support is provided by the vendor with email/phone as defined in the contract or SLA. Free community engagement if software is widely used (e.g. MS Office) and vice-versa.
Intellectual Property Rights [14] 	Copyright (permissive vs. copyleft licenses)  (If possible attempt to avoid) software patents	Copyright can be negotiated but is usually protected by contractual agreement (under certain conditions seeing source code can be allowed).






	Trademarks Customer can be sued if found to infringe IP rights (see e.g. GPL enforcement) 	Software Patents Trademarks Company makes sure that their customers will not be sued.
Software foundations [O28]	Unique to the FLOSS – user (e.g. Kuali) and developer (e.g. Eclipse) ones. Promote & develop various projects and cover e.g. infrastructure costs	Role of foundation is absorbed by the vendor, yet consortia and standardization bodies e.g. ISO can be established too.
Software testing 	Free and public community effort 	Closed test suites and quality assurance processes
Software quality [37] 	Inconclusive. It varies based on size, importance, popularity of project and many other factors. But it can be comparable or even exceed the one of the closed-source. + -	
Software security [40] [32] [34] 	Issues are much easier exploited because of open-source code but they can also much quicker fixed. Because of “many eyes” in the community, customer can perceive such software as more secure. + -	Usually perceived as less secure (e.g. Adobe Flash, MS Windows) even though it is developed by a dedicated team of full-time developers. The customer has to trust the vendor of not having any backdoors inserted.

Table 1 shows a self-made characterization of free and non-free software. “Plus” sign means advantage, “minus” sign is disadvantage and “plus-minus” refers to both depending on perspective for the company (essentially neither positive nor negative). The “target” icon on the left shows whether this characteristic can be or is relevant to the organization. Only those are evaluated, which are deemed to be relevant for the adoption of FLOSS in the company.

### 3 Barriers of FLOSS adoption

Following a conclusion that free software can be “*very viable alternative*” [15] [73] [78] for some companies and thus be indeed beneficial, in this chapter I review and organize (based on the modified TOE framework) specific barriers of adoption that organizations in the public and private sector are and might be facing. Build upon knowledge of different authors and their studies, I also discuss the current integration of open-source software in the healthcare sector.

#### 3.1 Literature review

As a first step towards my goals, it is necessary to gather relevant and subject-specific information for this study. Thus, at the beginning I commence by conducting an exploration of the available literature in both primary and secondary sources.

##### CHOICE OF LITERATURE

My first step leads me to search iteratively Scopus, <http://library.au.dk> and <http://flosshub.org> databases where I look for important books (chapters), journal articles and conference proceedings with relation to FLOSS barriers and adoption in organizations. Furthermore, I complement this with similar search queries that include keywords such as “OSS”, “open (-) source software”, “free software”, “FOSS”, “FLOSS”, “adoption”, “inhibit(tors)”, “barrier(s)”, “factor(s)”, “challenge(s)”, “integration”, “organization(s)” and “company(ies)” in Google Scholar database for articles, (working) papers and reports from different even unreliable internet sources. By combining these two processes, I gather a variety of documents – though because of search limitations, particularly in the Google Scholar, there is always a chance of missing some important papers [79]. However, this is later mitigated by examining literature which researchers quote in their studies.

At this stage, furthermore, through searching such articles (naturally) in all cases I limit myself to English written ones while additionally filter out those which have been written before year 1985<sup>9</sup> as one can consider it, due to Richard Stallman’s creation of FSF, as a starting point of the free/libre and open-source software movement and its research. Even though aforementioned keywords have been found in these works, each document is also skimmed for its information content and abstracts are read to filter out those which are beyond the scope of this paper due to my delimitations and objectives. For example, scientists discussing topics related to the inner-source are excluded.

Being passed this first test – a practical screen – reports, coming from different sources such as highly and less prestigious journals as well as from other web resources, are subsequently included

---

<sup>9</sup> Here it is also relevant to note that academic research into the open-source software has really took off after 2000 since only a handful of important journal articles e.g. [21] have been written before that year. In this study the vast majority of papers come after the millennium [67].



into my research resulting into 33 articles (phase 1, see Figure 3) on various open-source subjects such as case studies about adoption of FLOSS or barriers of acceptance by developers and users [58]. In addition to diverse sources, these articles are both theoretical as well as empirical (a mixture of quantitative and qualitative research) and are set in multidisciplinary academic fields such as information systems research, management and organizational studies, social sciences or for example open-source software with relation to the medicine (which is used later in chapter 3.3.1) [79].

Once this inclusion step has been accomplished, a classification of various topics that authors try to tackle is conducted. Thus, papers are again scanned in more detail and put into several categories ('folders') according to their relevancy: general 'barriers and factors of FLOSS adoption', specific ones in 'healthcare', 'education', 'public' and 'private' sector and similar.

Moreover, the tactic I used here for gathering these research papers – in essence a sample of them – is what other scientists refer to as a 'snowball' effect [87]. This is the "*most widely employed method of sampling in qualitative research*" that is used across "*various disciplines [in] the social sciences*" [87]. In my case, documents are not only scanned for their topics and used methods but also for other important citations these authors have referenced e.g. in their own 'background' sections and throughout the article (this is so-called 'backward search') [58] [61]. Indeed, if a paper is found to be relevant to my purposes due to being cited in an article which is subsequently read and further examined, it is added to the database in order to further increase relevancy of my literature findings. Hence, above-mentioned 33 initial articles are now enriched by adding extra 30 studies to my sample through such 'snowballing' of additional references (phase 2) [87].

Unfortunately, however, in part due to 'snowballing' effect my research is not fully reproducible. For one, an article such as [56], a part of "*grey literature*", has not been published in peer-reviewed journal and is not indexed in the Scopus, ScienceDirect or Google Scholar databases either [58]. Being found online and quality of which I consider to be good, it is nevertheless appropriate to my field of investigations. Nonetheless, with five exceptions [36] [50] [57] [69] [71], in order to attempt to legitimize my non-systematic, arguably subjective choice of literature and increase its quality I look for articles with a citation count (from Scopus, ScienceDirect or Google Scholar) greater than 3 and coming from serious sources such as peer-reviewed journals, conference proceeding and related. Furthermore, because I consider some documents too general and otherwise unfit for my mission such as [56], they are excluded from this review as well (phase 3). A result of this quality appraisal, from overall 63 documents, I review and very closely read only 44 of them.

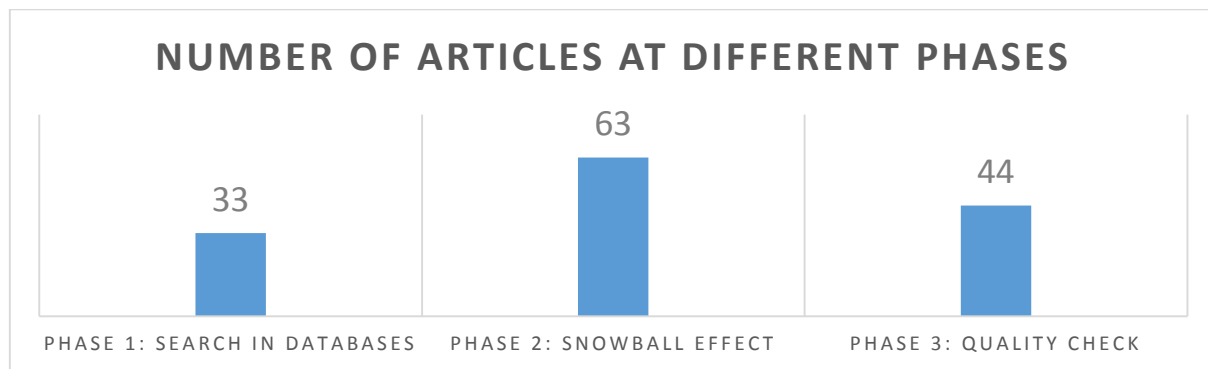


Figure 3 shows different phases of gathering articles that are reviewed.

To construct and increase validity of this research, in addition to asking a question ‘Is a study design clearly explained and are their findings sound and justifiable?’ [79], at every step of the subsequent literature review a check across various authors and their studies is performed [45]. In essence, I attempt to “*search for convergence among multiple and different sources of information [i.e. author’s studies] to form themes or categories*” of adoption barriers [83].

To sum up, my literature review for this chapter is initially based on 44 documents and to follow up on what has been described in the methodology section 1.4, I now continue to explain how this qualitative inductive study is carried out [68].

## THE PROCESS

First, a database in Microsoft Word of chosen literature containing a title of the work, respective authors and citation count is established, see Appendix 9. Consequently, this is supported by two graphics – one showing a number of articles per year and another one what type of sector or an angle authors have looked at (e.g. aforementioned healthcare or IT industry), see Figure 4 and Figure 5.

Next, during close readings of all articles that have been included into my review, four steps are performed: (I) I highlight relevant parts with relation to my purposes with a yellow color, (II)

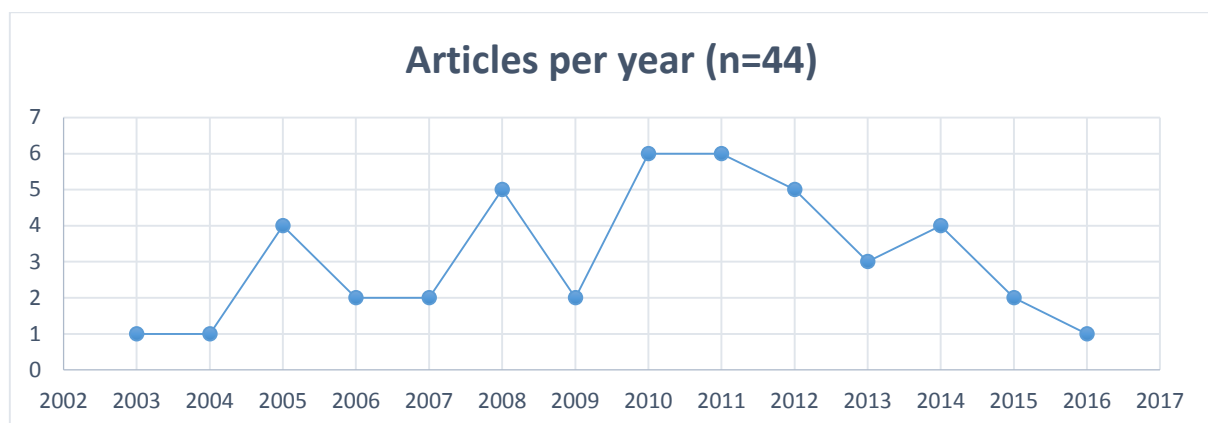


Figure 4 displays number of documents per year.

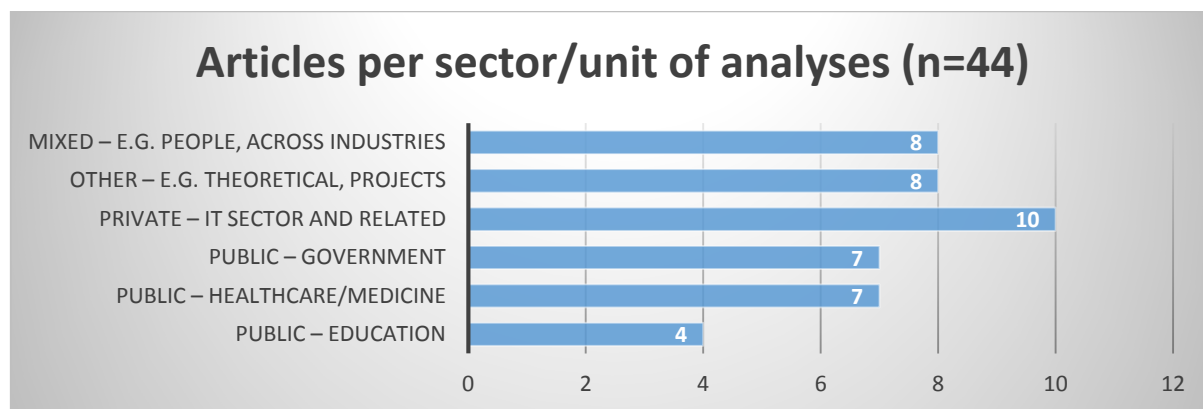


Figure 5 illustrates different sectors that are included in this review.

summarize the whole paper in the database and (III) I analyze and interpret findings of authors, for myself. Lastly (IV) in order to extract adoption barriers, where necessary, I re-read articles multiple times. In fact, this fourth step is a critically important for any literature review as by analyzing specific content segments, it allows me to create labels for key themes, meaning challenges of open-source software adoption in both public and private organizations [68].

Given that my primary goal is to identify and develop these barriers for the subsequent use in the TOE framework, as described by [61], it is recommended to “*compile a concept matrix*” – what I call an ‘inhibitor matrix’. By conducting an *open coding*, during which a text segment covering a key theme from “*each article [is] assigned to one of the broad categories derived from the literature review*”, an Excel file with the information about mentioned hurdles from each paper is created, see an extract of such in Figure 9 (the file is provided as well) [68] [85].

Therefore, to create such a core outcome of this inductive analysis, while iteratively studying each document and analyzing its content, I try to identify barriers, challenges and inhibitors that authors both mention explicitly as well as implicitly in their narrative. These parts of text, phrases and words are then assigned a meaningful description for their context (a label) and naturally there might be links between them (e.g. financial barriers that may encapsulate cost for switching, costs of commercial support etc.) which, however, at the first step I do not try to establish or otherwise structure (i.e. at first a non-hierarchical list of codes is created). When I am not able to assign a code using my existing coding scheme, I create a new category as I see it fit [85]. As a result, after such an initial run, I list over 50 different individual barriers at various levels, both high-level (e.g. risk factor) and low-level (e.g. documentation issues).

However, because of working alone, it should be also noted here that this subjective labeling of text is not supported by an independent parallel review of the same documents. If it has been indeed the case, this literature study would gain an additional trustworthiness by being possible to conduct

further “*peer debriefings and stakeholder checks*” [68].

After all articles are carefully read, summarized in the database and labeled accordingly, because of labels’ quantity, naturally the next step is to reduce their overlap and overall redundancy [68].

Hence, for each adoption challenge I try to link it to other obstacles which might be in a relationship, to establish a plausible hierarchy. The goal is now to iteratively decrease such a large number as much as possible and derive primary categories that are able to encompass those from the initial extraction of the raw data (i.e. while some are combined others are either reframed or renamed). Subsequently, this small number of (high-level) categories ought to capture all individual challenges that have been found during reviewing 44 academic papers [68].

As suggested in chapter one and in order to reduce this number of adoption barriers, I proceed now to examining TOE framework which forms my foundation for mapping over 50 individual challenges to three (plus one) major categories.

### 3.2 Technology-Organization-Environment(-Individual) framework

After screening and extracting data from the selected articles, my next step is to aggregate, organize and discuss these adoption barriers of open-source software [58]. Having both qualitative and quantitative studies in my review, I choose to proceed with their qualitative synthesis by explanation and interpretation [58]. Therefore, I am focusing on findings of such papers trying to “*find common concepts or themes*”, “*keeping in mind the context in which the conclusions were made*” [79].

As described in [69], technology, organization and environmental framework, developed by [59], “*is a fundamental approach to investigating a firm context that influences the process by which it adopts, implements, and diffuses technological innovations*”. Being able to combine with related theories such as institutional one and having a “*solid theoretical basis, consistent empirical support, and the potential of application to IS adoption*”, TOE examines three contexts by which an organization implements a technological innovation which FLOSS is an example of [60] [69].

The first one, technological aspects, includes both internal and external technologies that are available to and used by the firm [60] [69]. The next one refers to organization’s size, complexity of its structure or quality of human resources. The last one, environment, describes an area and a broader context in which industry, governments or competitors influence firm’s operations [60] [69]. Even though the framework covers well above-mentioned elements, individual factors such as for example rejection of use or user perceptions are insufficiently described by this theory as documented in studies [82] [65] [74] [71]. As a result, what many researchers do is to include a new

dimension to the theory of OSS organizational adoption, namely the *individual* factor – thus resulting into TOEI/ITOE abbreviation [19] [27] [50]. Consequently, I follow this approach as well and extend the TOE framework by adding this additional element.

Hence, building on the theoretical underpinnings of the above theory and my inhibitor matrix, I now proceed to map aforementioned over 50 individual barriers into 4 large dimensions. Each of them consists of several smaller categories that overall are supposed to explain FLOSS inhibitors that I was able to observe. As a result of the inductive analysis and iterative creation of high-level categories (e.g. non-functional technological barriers)<sup>10</sup>, in Figure 6 and Figure 8 I present challenges being mapped into four elements according to the TOEI scheme. This brings the reader an insight into what kind of factors, collected across 44 unique studies, organizations face when (considering) integrating open-source software into their development processes and applications.

It goes without saying that many of these challenges are naturally interconnected with each other. For example, the existing preferential purchasing agreements at the government level for the proprietary software can result into a having a technical vendor lock-in. Thus if an organization wants switch to open-source alternatives, it first has to deal with procurement policies, legal and political issues. As shown in the studies by [78] [89] [33] [19] [75], this then significantly inhibits companies to even consider FLOSS. Based on this outcome, in the next subsections I explore both top-level and individual challenges in a more detailed fashion [68] [58].

### 3.2.1 Technological inhibitors

Across studies, I observe five high-level categories of technological challenges that make FLOSS adoption significantly harder. Firstly, there are the *non-functional* and *functional* barriers. In addition to that, *project's reputation*, *open-source selection process* itself and *existing IT systems and infrastructure* contribute to free software being rather a dream and very long-term aspiration for many corporations.

The largest category of non-functional challenges ranging from the performance over the maintainability of the open-source software encompasses 12 individual barriers. The single most frequently mentioned inhibitor is the technological vendor lock-in. Because organizations widely use popular proprietary formats (e.g. psd) which FLOSS can sometimes only partially support, the move to the alternative computer programs has a wide spectrum of work-related implications impacting employees' productivity. Therefore, such a non-interoperability between two types of

---

<sup>10</sup> When assigning individual challenges to categories, due to different authors considering e.g. vendor lock-in either organizational or technological barrier, I use my judgement to assign it where it fits best – in part based on other studies.

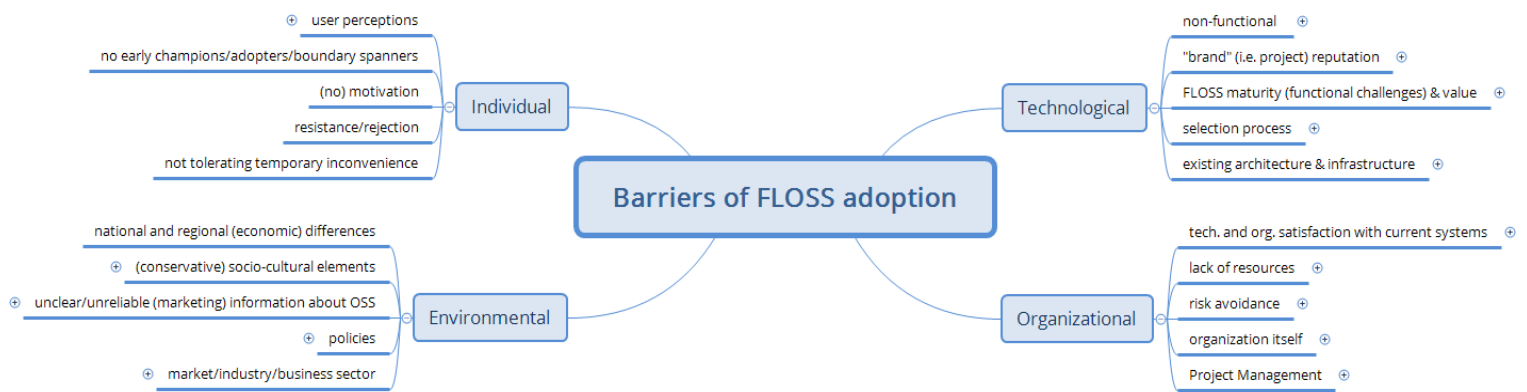


Figure 6 summarizes individual barriers from the scientific literature into 4 primary dimensions and 19 secondary factors. An expanded look is shown in Figure 8 in appendix.

applications, for example to even open an office document properly, leads to rejection of open-source solutions in the first place as they are non-conformable to the commonly exchanged closed standards [50] [27] [19]. At the governmental level to deal with such situations, [89] [53] for instance suggest different policies one of which is to require proprietary software to support open standards or apply a principle of “*explain [the use of proprietary software] and commit [for plan to use OSS]*” [88].

Another pressing issue firms have mentioned is the security and privacy of the open-source solutions in general [53] [77] [40] [81]. This becomes important in mission-critical sectors such as in finance, healthcare or for public bodies and their security agencies. The challenge is strongly related with the quality of software per se as well – a long and pressing research question both in the practice and academia [37] [26]. Indeed, proprietary programs are sometimes seen to have a higher quality because of full-time, dedicated developers [82] [73] [53]. However, as known from widespread examples of Adobe Flash Player and Microsoft Windows OS, the fact that computer programs are closed-source does not increase their quality and security – underscored by [37]. And this applies vice-versa too where because of having source code in the open, it does not automatically mean that software is superior to the proprietary one due to ‘many eyes’ from around the world who could fix its flaws.

The factor of reliability and confidence (i.e. uncertainty) in the software is also frequently mentioned as an adoption barrier [76] [27] [57] [50]. Due to the fact that developers are in many cases not being paid for their contributions in the open-source<sup>11</sup>, their voluntary effort can stop at any time and so the bug fixes. Therefore, lack of roadmaps and clarity in the future (and

<sup>11</sup> It should be noted that this is slowly changing. Today, many large companies such as IBM or Intel are paying independent developers to contribute in the open-source programs, e.g. commonly observed in the Linux Kernel.

“*questionable longevity*”) also does not encourage companies to adopt it because if they integrate such FLOSS which is later being abandoned, the maintenance costs for bug fixing and further development of additional features increases significantly [72] [27] [7] [44]. Overall, such project’s reputation plays an important role as the more popular the ‘brand’ is (or project being backed by a company with strong brand name [76]), the more users and developers it can attract and thus further increase its trustworthiness and reduce likelihood of becoming abandoned [50] [75].

Importantly, at this stage it is relevant to note that many of these afore- and below mentioned challenges need to be viewed in the specific organizational context because they are and can be in many cases double-sided. Indeed, the reliability of the open-source software is usually seen to be an advantage due to the increased flexibility and access to the source code [50] [27] [81]. Yet, for example a lack of reliable information about FLOSS, which I consider to be an environmental factor, is something that prevents companies to examine it more [33].

Speaking of open-source maturity, which together with the quality “*varies considerably*”, studies further mention that free software often doesn’t have certain functionalities and features when compared to the proprietary (commercial) counterparts [80] [30] [27] [25] [51]. Moreover, in order to be recognized and integrated into the company, as reported by [82] and [72], FLOSS needs to fit organizational operations and have a tangible value (i.e. be relevant) to them too [27].

Other researchers found selection challenges where due to software availability (including its discovery problems) [7], variety (e.g. which fork to pick) [44] and ownership questions [27], choosing a right open-source program is much more complicated than the proprietary one. In fact, the process which can function well for commercial off-the-shelf software may not apply at OSS at all due to different factors and characteristics that need to be considered [7].

The situation is also further complicated by the lack of availability and insufficient (commercial) support by the community and third-party companies [7] [27] [36]. In the inhibitor matrix, this is likewise the most cited technological reason which prohibits companies to adopt open-source software for their systems. And there is one arguably simple explanation: due to existing IT architecture, used systems and perceived challenges of integrating any kind of new technological innovation in the enterprise context, companies do want to have somebody with an expertise who can help them to overcome initial teething problems [30] [44]. And a lack of commercial open-source vendors thus inhibits the use of free software [82].

As observed in the literature, technical barriers play a critical part in integration of FLOSS in corporate environments. If the issues of technical compatibility and availability of commercial

support would be solved, many firms both in the public and private sector would be encouraged to try using OSS – even though there are other challenges to overcome as well.

### 3.2.2 Organizational barriers

Within organizational barriers, companies talk about lack of *resources* in order to switch and current *satisfaction with the existing systems*. Additionally, there are *organizational barriers* such as firm's size and lack of managerial support and vision. Lastly, when enterprises decide to proceed with the transition, there is insufficient *project management* as a result of which FLOSS integration either fails or is otherwise seen as unsuccessful.

The most significant barriers are missing financial and human resources. Firstly, because companies usually already have proprietary products and possess small IT budgets, they need to consider switching costs that include both hidden as well as sunk expenses (e.g. for licenses) [76] [33] [82] [30] [81]. Even though the price for the right to use FLOSS does not exist, firms may need “*to hire programmers to supplement their IT staff*” – thus total costs of ownership have to include this and other items such as enterprise support and employee training, see section 2.3.3 [72] [78] [76] [71]. Furthermore, particularly in developing countries proprietary products are seen to be more prestige and of higher quality. Thus low acquisition expenditures may actually discourage free software adoption, even though it is generally a strong facilitating factor [50] [51]. Interestingly, as documented in the hospital setting, successful integration of FLOSS can also lead to losing existing financial support (e.g. from grants) which otherwise would be used for further OSS development and surrounding activities [77] [33].

Secondly, companies report lack of internal availability of knowledgeable and skilled IT personnel and external accessibility of vendor support being a significant inhibitor for FLOSS adoption too [48] [51] [33] [76]. Indeed, given that many employees are used working with the proprietary software such as MS Office from home or study times, they need to be provided with the adequate training which only further complicates the transition and increases time and costs [43] [15] [95]. Here, many of the individual factors also arise for examples users' resistance, fear and demotivation towards free software due to feeling being deskilled and undervalued [81] [75] [19] [71].

Given that there is a tendency to focus on organization's core competences and missions, IT systems and their development and support are often outsourced. Thus negotiations with vendors are required which only makes migration time consuming and harder [81]. Hence because of all of this, companies rather devote their limited resources to the existing proprietary IS rather than trying something 'new' [82]. Naturally, managers prefer to avoid the (potential) failures at all costs as the risk of systems not functioning and providing satisfactory features is simply not worth it [76] [84].



Plus, with no legal accountability (unless company can find or already has enterprise support), the burden lies on the organization itself [2] [27] [81] [19].

Technological and organizational ‘satisfaction’ with existing systems also plays a critical role in FLOSS non-adoption. Where there is no user demand, there are no reasons to implement it either [72] [27] [82]. Furthermore, if the company accesses proprietary software through existing purchasing agreements for low or no costs, there is no need in considering switching to alternatives in the first place, *nota-bene* when these (mandatory) agreements are signed at the higher (e.g. state) level and cannot be influenced [19] [25]. In developing nations, albeit being very strong facilitator of OSS adoption, a high level of piracy inhibits corporate transition to low cost solutions too [69] [75].

Organization size was mentioned to be another barrier [74] [78] [84]. If the organization is small, it may lack a whole IT department and skilled employees who can support FLOSS integration and programs. On the contrary, the larger the organization is, the more probable open-source adoption can be successfully accomplished [81]. Further, as many studies show, particularly lack of managerial support, its awareness of alternatives and organizational structure is also a burden for corporate change [6] [69] [19] [89].

Lastly, there is a category of issues which is related to the project management itself. When the company has mandatorily decided to proceed with the migration, such projects are often unsuccessfully perceived due to lack of, a time rush, lack of detailed planning, training, pilot trials and success cases at other organizations and overall insufficient project governance [82] [6] [27]. Therefore, as shown on case study of [15], the best strategy is when employees themselves, bottom up, voluntarily adopt and demand open-source software – as a result such change is less painful and employees are more motivated and satisfied with the system.

As can be seen, free software is not cost free [88]. It takes its price in the organizational changes and employee morale, and therefore faces a resistance from those affected [6]. Thus, in order to implement non-proprietary computer programs, while staff has to be retrained (as they usually will not have an experience with OSS), managers have to be on board and support such an enterprise adoption due to required time, all sorts of costs and risks it brings [76] [25] [95].

### 3.2.3 Environmental challenges

The last major category of integration inhibitors is related with the environment in which firms operate. In fact, studies report that while *economic and country differences* (e.g. in terms of infrastructure) play only a minor role in FLOSS non-adoption [53] [74] [76], aspects of various *policies and laws* e.g. at IT level and *structure and conditions of the industry* are critical factors for

successful use of free programs. Moreover, in order to transition to them, *cultural issues* need to be taken care of, necessary information be available and *public discourse* about OSS should be rather positive and balanced.

Indeed, one of the most important environmental challenges are policies at different levels which play a critical role in FLOSS acceptance by the employers. For example, at the governmental level, they report that in many cases procurement models are discriminatory, nontransparent and not flexible enough for free software, thus hindering its market expansion [88] [49] [81]. In addition, political and public pressures and the lack of the governmental support or even a position toward open-source technology significantly inhibits any actions that enterprises undertake – especially relevant in the sectors supported by the public funds such as in the education and healthcare [89] [36] [33] [11] [64] [74]. As reported in the latter case by [81], politics (and the changes in the power structures) is “*a critical factor*” that decides “*how OSS would be used in the future, even before getting to the technology [and other] portion[s] of [open-source] adoption*”.

Additionally, there are also issues presented by the fact that companies require to have IT systems that are in compliance with business regulations (e.g. in the finance – privacy, security, standards etc.), which usually cannot be done without a corporate backing [7] [78] [81]. As briefly outlined in the section 3.2.2, a lack proper IT policies (in terms of e.g. planning, future directions and visions) and governance processes in general do not encourage adoption of innovative technology such as open-source as well [78]. Not to mention the fact that managers sometimes (have to) impose their top-down decisions at the whole organization without a broader support or an agreement [15].

Another barriers corporation both in the public and in the private sectors have mentioned is what I name ‘socio-cultural elements’. For example, [77] have found out (in the hospital setting in French speaking Mali) that concept of elder is there critical for any kind of investments as in their society people of higher age carry larger decision power than those of younger age. Therefore, if these elders are not entirely convinced and on board, the project will fail because other employees are reluctant to follow instructions without having a ‘blessing’ from older people. Similarly, this applies to the Hofstede's cultural power distance [74]. There, the higher the distance between different employees is, the less likely firms will adopt a novelty such as OSS. This is simply because “*the hierarchical distance between technical employees and top managers is wide*” and enterprises “*are reluctant to changes (...) because new things may threaten the existing power structure*” [74]. Furthermore, localization of open-source products has been also found to be a barrier due to e.g. in Africa not everybody being able to speak and understand English which on the other hand is the main language in the computer science and software industry [53] [55].

Next, it should be noted that market itself, its structure and conditions play a factor in decision about free software migration too [82] [88]. [49] say that “*type of industry matters*” because those with strong relation to the IT (e.g. communications) are naturally more keen to adopt new technology. Similarly, this is confirmed by [27] and [82] who add that statements such as “*other nearby firms had rejected*” FLOSS significantly influence a final decision about adoption as well. With relation to that – if the industry or an organizational culture is rather of a conservative nature, this characteristic does not support choice of free software either [19] [33]. As shown on the example of [77] again and confirmed by a study of [53], local needs and requirements – be it in the language localization or adaptability to the market – have to be appropriately addressed through engaging with local (or regional) communities and commercial vendors. Indeed, here openness of the free software strongly encourages its further development, customization and cooperative behavior, especially relevant to the provincial public bodies [88].

Lastly, what I have identified are also challenges related to the unreliable or otherwise unclear and invisible information on the side of open-source projects and commercial vendors [36] [33] [27] [82]. As [78] mention, such a lack of a marketing effort and consequently information goes hand in hand with a poor interpretability of a public discourse. Indeed, the information about OSS should be more reliable, consistent, easy to find and of high quality [78]. Furthermore, due to lack of people’s and organizational awareness [15] [19] [27] [30], companies have to invest more time and resources into educating their future customers. Therefore, certainly the use of FLOSS technology in the academia, which is already advanced, helps to promote it to young professionals who get to know about alternatives, its benefits and drawbacks, to proprietary software early on.

Depending on the sector, clear policies and legal frameworks have to be created and put into action while at the same time information about open-source software needs to improve – both qualitatively and quantitatively. Nonetheless, environmental inhibitors are not the last category of issues as all three above-mentioned do not cover another important group of challenges which individuals themselves can influence and change.

#### 3.2.4 Individual factors

Even though my study focuses on organizations, employees always hold a key to the successful adoption of new innovations, particularly in the IT field. Therefore, as suggested by the technology acceptance model (TAM), they need to perceive FLOSS to be both useful and easy to use – thus allowing them to further maximize their productivity [50] [2] [65]. However, if end-users from the beginning negatively look at the FLOSS transitioning initiative and have a ‘proprietary mindset’ due to never working with it, they naturally see a possible change from a widely used to a niche and

‘cheap’ software as being demotivating for their careers while at the same time also being de-skilled because of not having a higher value to the company who would opt to pay for the proprietary programs [81] [19] [15] [71]. Thus, what consequently occur are organizational resistance, fear and wholesale individual rejection of using such open-source software.

Moreover, looking from another angle, companies who would like to consider FLOSS often lack an early adopter, an “*individual(s) who connect their organization with external knowledge and can bring the organization in contact with new innovations*”, which at best should be a top-manager promoting and “*support[ing] its introduction*” in the company [19] [75] [69] [27] [89]. Having such a boundary spanner is particularly relevant because, as already mentioned before, individuals and as a result organization too lack personal awareness of alternatives to the non-proprietary software [71]. Therefore, the main task of these champions is to motivate users to try the free software, which consequently as shown, encourages open-source broader adoption [24].

Last but not least, another (less) frequently mentioned challenge is the fact that employees do not tolerate any (temporary) inconvenience which results from technical updates or a migration that FLOSS switching would ultimately bring [19]. Simplified, workers just want to use their programs in a productive matter, anytime. And any disturbance caused by adapting to new technological environment and computer programs is not in their primary interests.

To conclude, even though being its own category, individuals constantly interact and are in relationship with all three other dimensions. In fact, as described in above sections, many of the challenges which inhibit open-source adoption go back to people as it is them who cannot speak English, lack knowledge of PC/IT skills to start and support transitioning effort and are not aware and able to select appropriate free computer programs which their employers would benefit from. Not to speak about convincing organization and managers of OSS benefits in general.

### 3.3 FLOSS in Organizations

Now I want to explore a general level of open-source adoption in companies. The technological ones like Google [O38] and Facebook [O39] not only extensive use FLOSS in their products but also together with fellow competitors like IBM<sup>12</sup> heavily contribute back to the community e.g. by making their software innovations open-source or providing a commercial support to other enterprises. Therefore, adoption at these companies is incomparably faster and larger than for example in public bodies or strictly regulated sectors such as healthcare where both compliance and often inflexible and discriminatory procurement policies are playing a major role in free software

---

<sup>12</sup> “IBM Open Source at GitHub”, 08 Dec. 16, <https://ibm.github.io/>

non-consideration.

In 2012 [85] conducted a study of 2840 articles on open-source software between 1988 and 2009. In it, authors have investigated whether FLOSS has been already institutionalized, i.e. whether open-source computer programs are “*considered appropriate, even legitimate, in a given context and (...) acquired the status of norm in thought and social action*” – basically taken-for-granted. Their “*results allow [them] to state that OSS had been widely adopted and diffused by 2009, but only for specific types of applications or in organizations with particular profiles*” [85].

Indeed, due to better fitness in the technologically-driven domains, for the back-end applications such operating systems or generally those supporting software development, open-source has “*reached a critical mass of adopters*” in that year and has only increased it ever since [85] [86] [51]. On the other hand, for front-end (‘business’ type of) applications such as business intelligence or content management tools (e.g. in finance), the diffusion of FLOSS has been slower and less successful [86]. Notably for example in a city of Munich, where authorities first transitioned to free software and now are again considering switching back to closed-source model [11] [O40]. Nonetheless, for early adopters mainly among SMEs, the “*institutionalization process had [already] begun*” and grown significantly in recent years documented by the observation that reduction in development/acquisition costs is no longer a prime reason for their participation in such open-source community [85] [O41].

As argued by [51] in 2012, “*OSS has long passed the market introduction stage but has not yet reached the maturity stage*” (albeit disagreed by [77]) which is particularly seen in the public institutions that mention such a technological maturity of FLOSS as one of barriers for its non-adoption [30] [88] [89] [91] [45] [90] [76]. A survey conducted in 2015 among IT professionals in various industries however strengthens the argumentation that open-source is quickly becoming a more mainstream topic for all companies and is behind many innovations both in the IT as well as in the business sector [O41] [9]. In fact, its adoption has also risen and based on the reviewed literature in this study, one has to conclude similarly to the authors that “*the institutionalization of OSS [in firms] has been partly achieved*” [85] [86] [O28]. This especially applies at larger organizations associated with IT and knowledge intensive work [51].

Going back to gathering and analyzing my literature studies, researchers have put a significant effort to study two major sectors. Besides exploration and applicability of FLOSS in the private companies (and specifically with strong connections to the IT sector or having large in-house development), the two areas frequently being investigated for potentially adopting OSS and its principles are healthcare and public institutions. Indeed, what can be observed in my review is a relative broad

representation of various units of analysis (e.g. students, software firms) with specific depth in the aforementioned three of them, see Figure 5. Thus, in the following I want to take a closer look on one of them and see to what degree challenges in the previous chapters can be found here (and if at all) as well.

### 3.3.1 Healthcare sector

Having studies of FLOSS adoption in healthcare setting from Canada [33] [78], USA [81], Mali [77] and Ireland [19] [15], I first proceed to provide a general perspective on the subject.

Healthcare domain has unique characteristics which make it harder to embrace innovation resulting into significantly “*lagging behind in terms of adoption of modern ICT tools and infrastructure*” compared to other sectors [80]. Firstly, it is because of its complexity and (usually) fully outsourced development and assistance for its IT systems [80] [81]. Secondly, any delivered applications from vendors have to be in compliance with necessary national and international regulations (e.g. HL7 certification), standards and be professionally supported for the hospital end-users. Last but not least and similarly to other industries, software vendors must bear a legal accountability for their products which only adds to difficulties hospitals face when considering integration of free software [80].

FLOSS in the healthcare has seen a considerable growth in the last 5 to 10 years [80]. While a two decades ago there were only a handful of commercial health IS (covering use cases for patients’ administration and management of health facility) available on the market, “*now more than a hundred (...) open-source applications*” for various medical domains can be downloaded and used gratis [77] [80]. Yet, despite industry’s growth, today FLOSS “*is still under-adopted by [organizations] and under-exploited by technology providers*” [78]. And even though existing research encourages “*adoption and use of OSS (...) because of [its] potential to both enhance healthcare delivery and lower software acquisition costs*”, it remains “*a poorly understood phenomenon*” by both the industry, scientists and doctors themselves [81]. As a result of all of this, “*the global impact of [open-source software in healthcare setting] is still very limited and no figures on the penetration and usage (...) are available*” [80].

Nonetheless, due to budget shortcomings in the developed parts of the world, many hospitals are increasingly interested in FLOSS which promises them to break away from vendor lock-in while at the same time reduce costs for the medical (e.g. electronic health records) and decision supporting systems [81]. In the context of developing nations in Africa and Asia, open-source technology further pledges to improve computerization of healthcare facilities and thus avoiding being excluded and disconnected from the development in the world [77]. Because in many cases medical IT systems do not exist there and the proprietary ones simply cannot be implemented due excessive

acquisition costs, the possibilities of open-source adoption – seen from the outside – are large and more importantly they can be easier to accomplish.

[80] conducted a literature overview of free software in healthcare environment and confirmed that even though there is some information available about large variety of stable and mature FLOSS applications (e.g. US Veterans Health Administration VistA software, GNU Health), the adoption varies by sectors and (unsurprisingly even) within the regions [75]. On the one hand, in the medical research and academia, FLOSS “*has always been used to some extent*” [80]. But on the other hand, while the North America is far ahead in using such software for healthcare delivery, in Europe the deployment is significantly lower due to fragmented policies of many nations [80]. An exception to that is the UK that has made a considerable progress and “*has (...) the most active and vibrant [OSS] community*” on the continent [80].

In their research Karopka et al. (2014) found out that OSS adoption in this domain is no different from other sectors when it comes to integration barriers [80]. Although both absence of licensing costs and (arguably and usually disputed [78]) lower total costs of ownership are seen to be key supporting factors for open-source consideration, hospitals have issues with a lack of professional support and legal liability for the such ‘alternative’ applications [80].

[33] in 2008 and [78] in 2013 conducted all-in-all 31 interviews with CIOs, managers and IT experts in Quebec’s healthcare organizations. Authors then confirmed several important challenges that influence FLOSS adoption. For one (I) hospitals lack internal IT resources and expertise in terms of staff and budget not only to implement it but also to asses (II) reliability of small amount of information available about open-source products and services offered. In addition to that, (III) internal and external political pressures (e.g. resistance to have in-house IT development from ministries and general population in part due to power changes) and (IV) conservative nature of the industry are resulting into not considering non-proprietary and non-company-backed alternatives [33] [78]. The authors further acknowledge that even though some of those barriers can be defeated, political pressure and lack of information about third-party commercial OSS offerings on the market are indeed harder to solve.

Moreover, because of usually smaller IT departments (if any), these hospitals have to further consider hidden costs e.g. when hiring consultants having necessary IT and OSS skills [78]. Additionally, due to incompatibility with technological standards, existing architecture and the fact that governmental and IT policies do not encourage such an adoption (e.g. because of preferential agreements with proprietary vendors), hospitals are in essence left with a decision to stick to the existing software even though they are dissatisfied with current proprietary companies [78] [80].

Thus, authors conclude that an external environment, medical organizations themselves and characteristics of OSS contribute to the non-adoption of such software in the industry [78].

Because of these challenges, it is no wonder that 23 out of 30 hospitals in the USA have “*indicated that they have not adopted any type of OSS*” [81]. Nonetheless, authors also find out that these would adopt both general purpose and domain-specific products, but the latter ones to lesser extent [81]. Indeed, Linux based operating systems and well known applications such as Apache server or MySQL database are very likely to be already integrated into proprietary appliances by the commercial vendors. Therefore, coordination with these companies is critical towards (any) adoption of OSS in hospitals as it is them that embrace, provide and maintain their products with open-source components [81]. Furthermore, organizational size is an important adoption factor as well, especially if large hospitals have “*in-house technical staff with experience in software development*” [81]. However, contrary to other studies [77] [15] [19] [78], these authors have identified financial factors not to be a “*core concern for IT managers when deciding to adopt OSS*” – instead it is the quality, security and legal accountability [81].

What has been described above is a perspective in developed countries. On the other hand, in developing nations e.g. in Africa in many cases IS are not operated at all and therefore FLOSS-based ones – that can be used gratis – are usually the only way going forward. [77] report a pilot project in Mali where open-source hospital IS, developed initially in France, has been successfully implemented. Even though they document positive attitudes and user experiences with the system, they have also encouraged a different set of barriers: (I) language localization and need for adaptability to country’s realities (e.g. for billings), (II) cultural traditions (e.g. opinion of elders is of high significance), (III) training to update employees with “*basic knowledge of how to use computer*” and last but not least stability and further development of the used FLOSS-based system [77].

To summarize, use of free software in the medical setting can significantly impact quality of delivered healthcare. While in developed countries it is about spending control and optimization of existing infrastructure and processes, in quickly developing nations FLOSS-based IS are how countries can advance themselves and improve lives for their citizens.

## 4 Discussion

In this section I want to take a critical look on my findings with regard to how this literature study was performed in this first place. In fact, because of these limitations, my conclusions and outcomes in section 3.2 should be looked in light of these study design constrains [62].



As already mention in the chapter 1 and 2, my review suffers from being unreliable and therefore other authors cannot replicate my findings to the full extent. Doing such a study for the first time, my lesson learned is that it is critically important of working very systematically throughout the whole research process. Particularly, it is of paramount relevancy to follow corresponding research protocol in the right order – outlined right at the beginning of my work to the smallest possible detail [58]. Additionally, it is unwise trying to accomplish multiple things at the same time and hence it is worth writing the report only when either the whole or the majority of the study has been already concluded.

Next, by working alone certain biases are harder to militate against. For example, by engaging in the FLOSS developer community I have established a particular view and gained experiences that might skew my perceived significance of specific open-source adoption barriers which otherwise would be much more important for the organization. Not to speak about the possibility of overlooking some challenges while extracting information from articles due to not considering it (as much) relevant and clear.

A vital consequence of me being (and to a degree seeing myself as) a FLOSS advocate is the so-called pro-innovation bias which talks about “*technology innovation [that] is universally welcomed and perceived (...) beneficial by all stakeholders*” [15]. Indeed, OSS is not the ‘holy grail’ and depending on the situation and environment it might not be favorable to all companies and all settings. To mitigate such a possible influence on the study, a more cautionary opinion of a fellow researcher could bring a new angle on how to look at inhibiting factors for integrating open-source software in applications.

And this does not only relate to biases, but also to the literature review process itself. As already discussed in section 3.1, it is very much beneficial to increase a trustworthiness of the study by having a second coder to not only to discuss preliminary results and steps at various stages of the study, but also to “*create a second set of categories from the raw text*” [68]. As a result, by combining both inhibiting matrices, researchers are able to overcome common problems of having different views and understandings of the same articles and extract *the* most relevant information.

Initially this study come from the assumption that companies who want to integrate it face a lot of challenges. And indeed, according to the various post-millennium studies I am able to examine, in industries such as healthcare or in public bodies this proves to be a fact. However, for 2017 it should be noted that the vast majority of technological companies, even those which were ‘born’ in the closed-source model, “*have passed the stage of rejection and denial of FLOSS and, instead, have turned to open-source as a key part of their software development strategies, drawing on its*

*technical quality, low cost and favourable licensing terms*” [57] [O37] [51].

Therefore, it is important to distinguish between different sectors of the economy where free computer programs are being adopted as clearly in the ICT sector the question is not if to adopt it but rather where, what and to what degree [88]. Notably in the healthcare, majority of papers talk about software development being outsourced to external vendors. Hence, much of the concentration on open-source software adoption should be aimed at these software houses as they ultimately, being fully feature-wise and legally responsible, are capable of bringing such open principles in the innovation, knowledge and source code to the customer [79]. Last but not least, another limitation of this research is that it is one-sided and does not consider and contrast the facilitating factors for open-source adoption too.

## RESISTANCE TO ADOPTION

On an entirely differently note, I want to talk briefly about opposition to IS implementation, which FLOSS can be an example of. This new software paradigm, which faces a struggle to be integrated into company’s landscape and working environment, might be also analyzed through a theory of resistance to IS while applying three different perspectives as described by [94].

First, in my context, the *people-determined view* looks at developers and end-users of applications who might have large psychological differences in integrating and using FLOSS for their work-related tasks. Thus, as also suggested in reviewed studies and underscored by [95] on the example of ERP system, educating and persuading employees by showing them benefits of open-source technology needs to happen (all of which lowers their resistance) [94]. Particularly, this is very relevant to the fourth dimension of adoption barriers where individuals are being demotivated and fearful of the new and possibly incoming change. Hence, right communication and managers’ framing of such innovation becomes critical in order to successfully adopt OSS in the organization.

What [94] further describes is the *system-determined view* which means that users might resist such technology due to gaps in it, for example due to missing features, bad(/old) UX/UI or lack of similar functionality they already know. Being in a relationship with the technological challenges, all this however can be avoided by engaging and communicating with users e.g. already during the design phase or based on the feedback provided in a pilot testing where developers can iteratively modify the software in the subsequent updates – before a full corporate wide roll-out [94] [95].

Lastly, the *interaction theory* tells that the opposition to the (new) IT system or computer program can be a product of the interaction between the setting & context (e.g. conservative sector and environment), types of actors (and e.g. their demotivation, individual beliefs and past experiences)

and the new system/technology itself (e.g. less performant, less secure etc.) – all of which has been shown to be the important while integrating FLOSS. As again suggested, users might feel that there will a power shift in the organization – therefore changing technology will negatively influence them by taking away certain tasks, functions and responsibilities thus diminishing their importance in the enterprise. This theory also the most appropriate to further analyze, on a meta-level, why organizations (and employees) have not adopted FLOSS technology to a greater extent, i.e. is the ‘devil’ rather in the company and surrounding environment or in the technology itself [94].

Based on the acquired knowledge, reviewed literature and relationship with the resistance and its views, I can observe a following order in which companies might have to approach FLOSS adoption. Given the opposition coming from different areas and actors, first companies need to solve issues related with the environmental factors such as politics and legal frameworks/policies which have the most outside influence on the final implementation decision. Subsequently, firms need to tackle both technological and organizational issues at roughly the same time. Firstly, they as well as individuals have to see greater benefits behind such an integration as people are affected (i.e. trying to answer what is ‘the bossiness case’ as not just technological advantages are always relevant). Particularly, these individuals have to be convinced and be on board about the need to such enterprise and technological change because it is them who ultimately decide to use such software for their tasks. Secondly, barriers associated with IT need to be examined because for example if the software is not compatible or offering similar kinds of features to the proprietary counterpart, the adoption will not be perceived well by the whole company as it is affecting its productivity.

## 5 Implications

My literature study helps practitioners to understand what kinds of risks they have to prepare for and take care of in order to successfully integrate open-source software into their environments. Particularly, my work has organized challenges mentioned across 44 studies (and more depending on the required quality) into four thematic areas, each consisting of a number of summary factors that inhibit spreading of free software in enterprises.

Additionally, this examination of barriers is especially important for managers because while large number of studies concentrates on exploring, identifying and confirming only certain challenges and establishing relationships between them, my research – with a handful of others such as [44] [27] – provides a thorough look on various issues companies and organizations from different industries, regions and angles are facing when considering OSS in real-live scenarios.

For researchers this study adds an additional value in terms of having a starting point which they can further use to investigate and analyze. For example, albeit being my non-goal, this review does not work with companies and their experiences directly. Specifically, in Danish and other Nordic countries' context this might be slightly different from what researchers in other regions of this planet have found. Moreover, my study does not discuss aforementioned outcomes in action as I do not seek enterprise access to perform a confirmatory study of my findings. Similarly, I also do not contribute to the knowledge of how to mitigate such a great number of risks that companies face. Last but not least, an interesting research questions would be to explore relationships between these factors too and study how multiple organizations (e.g. in a consortium), which have to reach a decision on proprietary or open-source software, influence each other within multiple different contexts [B8]. Certainly these are the areas where more research can be conducted and my study helps in laying the groundwork.

To conclude, I want to mention that I have approached my objective from a (very) negative perspective. Certainly, barriers are very important to understand due to inhibiting corporations and public bodies to advance their 20<sup>th</sup> century, proprietary and inflexible IS. However, I ask myself another question: what if barriers are not the key issue and it is rather a question of open-source advantages that are not clearly communicated and explained to decision makers who base their understanding on wrong perceptions and misinformation. Given that studies confirm that there is a lack of reliable information, I believe that this is an additional area that needs to be empirically investigated as well, supported with suggestions of how to improve managers' knowledge.

Overall, my findings shed new lights on OSS adoption barriers and this can help leaders from various departments to make more informed decisions. Hence, this work should help both practitioners and researchers have an overview and an insight into what elements are prohibiting broader spreading of the free/libre and open-source software – even though it is considered today by many as an initiative of a strategical significance [14] [53] [86] [27]. Subsequently, executives should use this information while navigating their organizations towards greater OSS adoption.

## 6 Conclusion

To summarize, at the beginning I set to review literature which talks about different integration problems companies face when considering free and open-source software in their corporate environments. Once the research subject has been introduced, by conducting an inductive qualitative study, I was able to find over 50 different barriers that with help of slightly modified TOE(I) framework I have categorized and structured, resulting into having 4 dimension with 19 major categories of inhibiting challenges.

While in the technological dimension, barriers such as vendor lock-in, technical maturity and lack of external support were the most mentioned ones, at the organizational level companies lack financial and human resources and do not have a managerial support and awareness which prevents them to undertake a risk of adopting open-source principles and technology in their IT systems. On the environmental side it is cultural elements, policies & legal frameworks and lack of consistent information that further hinder FLOSS integration. Lastly, individual factors such as employees being (and feeling) demotivated, resistant to a change and lacking an interest of becoming a champion of the new technology do not contribute to the open-source considerations either.

Albeit, as I have shown in the second chapter, open-source software has several important characteristics (e.g. non-existing licensing costs or increased flexibility) that allow companies to consider such an alternative in the first place, many organizations are stuck with the proprietary computer programs and systems because they simply do not have a capacity (and a larger reason) to implement it. Indeed, even though costs for acquisition are low, the total costs of ownership are (rather and remain) unclear and thus coupled with the technical, environmental, organizational and individual issues, the benefits of a FLOSS transition do not seem to overweight potential disadvantages and risks – at least in minds of many decision makers. These, particularly in the non-IT sector, would like to consider it however from the inhibiting reasons mentioned in chapter 3 including on the example of the healthcare industry firms see FLOSS rather problematic.

Concluding, not surprisingly, many issues that have been presented in this study are not specific to the OSS only but to the adoption of any kind of technological innovation in general. As an example, all companies prefer to have somebody who can help them (be it in-house or outside) with a support and implementation of a new technology or a software product. Similarly, non-functional challenges can be applied at any software program independently whether it is proprietary or open-source. Therefore, given that FLOSS has unlimited trialability, companies need to start small and over the time – if it proves to be useful both from the technological and organizational perspective – increase its use keeping in mind a need for sufficient communication and training if it impacts end-users' productivity.

To conclude my report, on a more personal note, by conducting my first literature review I have learned tremendously in how to approach such complex academic research and will certainly benefit from the acquired knowledge on FLOSS barriers when trying to spread its use in private and public corporations.

## 7 References

### PAPERS AND JOURNAL ARTICLES

[1]	Anthony, D., Smith, S. W., & Williamson, T. (2007). The quality of open source production: Zealots and good samaritans in the case of Wikipedia. <i>Rationality and Society</i> .
[2]	Bhadauria, V. S., Mahapatra, R., & Manzar, R. (2009). -An Exploratory Study. <i>AMCIS 2009 Proceedings</i> , 114.
[3]	Bonaccorsi, A., & Rossi, C. (2006). Comparing motivations of individual programmers and firms to take part in the open source movement: From community to business. <i>Knowledge, Technology &amp; Policy</i> , 18(4), 40-64.
[4]	Bonaccorsi, A., Piscitello, L., Merito, M., & Rossi, C. (2006, June). How is it possible to profit from innovation in the absence of any appropriability? In <i>IFIP International Conference on Open Source Systems</i> (pp. 333-334). Springer US.
[5]	Bonaccorsi, A., Rossi Lamastra, C., & Giannangeli, S. (2004). Adaptive Entry Strategies under Dominant Standards-Hybrid Business Models in the Open Source Software Industry. <i>Available at SSRN 519842</i> .
[6]	Brink, D., Roos, L., Weller, J., & Van Belle, J. P. (2006, June). Critical success factors for migrating to OSS-on-the-desktop: common themes across three South African case studies. In <i>IFIP International Conference on Open Source Systems</i> (pp. 287-293). Springer US.
[7]	Chen, W., Li, J., Ma, J., Conradi, R., Ji, J., & Liu, C. (2008). An empirical study on software development with open source components in the Chinese software industry. <i>Software Process: Improvement and Practice</i> , 13(1), 89-100.
[8]	Comino, S. and Manenti, F. M. (2004), Free/Open Source vs Closed Source Software: Public Policies in the Software Market (7/2004). <a href="http://dx.doi.org/10.2139/ssrn.469741">http://dx.doi.org/10.2139/ssrn.469741</a>
[9]	Crowston, K., Wei, K., Howison, J., & Wiggins, A. (2012). Free/Libre open-source software development: What we know and what we do not know. <i>ACM Computing Surveys (CSUR)</i> , 44(2), 7.
[10]	Derntl, M. (2014). Basics of research paper writing and publishing. <i>International Journal of Technology Enhanced Learning</i> , 6(2), 105-123. <a href="http://dbis.rwth-aachen.de/~derntl/papers/misc/paperwriting.pdf">http://dbis.rwth-aachen.de/~derntl/papers/misc/paperwriting.pdf</a>
[11]	Dobusch, L. (2008, September). Migration discourse structures: Escaping Microsoft's desktop path. In <i>IFIP International Conference on Open Source Systems</i> (pp. 223-235). Springer US.
[12]	Donaldson, D. R., & Yakel, E. (2013). Secondary adoption of technology standards: The case of PREMIS. <i>Archival Science</i> , 13(1), 55-83.
[13]	Ferrante, D. (2006). Software Licensing Models: What's Out There? <i>IT Professional</i> , 8(6), 24-29.
[14]	Fitzgerald, B. (2006). The transformation of open source software. <i>Mis Quarterly</i> , 587-598.
[15]	Fitzgerald, B. (2011). Open source software adoption: anatomy of success and failure. <i>Multi-Disciplinary Advancement in Open Source Software and Processes</i> , 1-23.
[16]	Fitzgerald, B., & Kenny, T. (2003). Open source software the trenches: Lessons from a large-scale OSS implementation. <i>ICIS 2003 Proceedings</i> , 27.
[17]	Fuggetta, A. (2003). Open source software—an evaluation. <i>Journal of Systems and Software</i> , 66(1), 77-90.
[18]	Ghosh, R. A., Glott, R., Krieger, B., & Robles, G. (2002). Free/libre and open source software: Survey and study.
[19]	Glynn, E., Fitzgerald, B., & Exton, C. (2005, November). Commercial adoption of open source

	software: an empirical study. In 2005 International Symposium on Empirical Software Engineering, 2005. (pp. 10-pp). IEEE.
[20]	Hauge, Ø., Ayala, C., & Conradi, R. (2010). Adoption of open source software in software-intensive organizations—A systematic literature review. <i>Information and Software Technology</i> , 52(11), 1133-1154.
[21]	Hecker, F. (1999). Setting up shop: The business of open-source software. <i>IEEE software</i> , 16(1), 45.
[22]	Johnston, K., Begg, S., & Tanner, M. (2013). Exploring the factors influencing the adoption of Open Source Software in Western Cape schools. <i>International Journal of Education and Development using Information and Communication Technology</i> , 9(2), 64.
[23]	Kagiri, T. M., He, Y., & Henglin, S. (2013). Factors limiting OSS adoption. <i>Oulun, Finland: University of OULU</i> .
[24]	Li, Y., Tan, C. H., Xu, H., & Teo, H. H. (2011). Open source software adoption: motivations of adopters and amotivations of non-adopters. <i>ACM SIGMIS Database</i> , 42(2), 76-94.
[25]	Macredie, R. D., & Mijinyawa, K. (2011). A theory-grounded framework of Open Source Software adoption in SMEs. <i>European Journal of Information Systems</i> , 20(2), 237-250.
[26]	Mishra, B., Prasad, A., & Raghunathan, S. (2002). Quality and profits under open source versus closed source. <i>ICIS 2002 Proceedings</i> , 32.
[27]	Morgan, L., & Finnegan, P. (2010). Open innovation in secondary software firms: an exploration of managers' perceptions of open source software. <i>ACM SIGMIS Database</i> , 41(1), 76-95.
[28]	Moyle, K. (2004). Total cost of ownership and open source software. <i>Department of Education and Children's Services, Adelaide, Australia. Retrieved February, 25, 2007</i> .
[29]	Munga, N., Fogwill, T., & Williams, Q. (2009, October). The adoption of open source software in business models: a Red Hat and IBM case study. In <i>Proceedings of the 2009 Annual Research Conference of the South African Institute of Computer Scientists and Information Technologists</i> (pp. 112-121). ACM.
[30]	Nagy, D., Yassin, A. M., & Bhattacharjee, A. (2010). Organizational adoption of open source software: barriers and remedies. <i>Communications of the ACM</i> , 53(3), 148-151.
[31]	Nakakoji, K., Yamada, K., & Giaccardi, E. (2005, December). Understanding the nature of collaboration in open-source software development. In <i>12th Asia-Pacific Software Engineering Conference (APSEC'05)</i> (pp. 8-pp). IEEE.
[32]	Pankaja, N., & Mukund Raj, P. K. (2013). Proprietary software versus open source software for education. <i>American Journal of Engineering Research</i> , 2(7), 124-130.
[33]	Paré, G., Wybo, M. D., & Delannoy, C. (2009). Barriers to open source software adoption in Quebec's health care organizations. <i>Journal of medical systems</i> , 33(1), 1-7.
[34]	Paulson, J. W., Succi, G., & Eberlein, A. (2004). An empirical study of open-source and closed-source software products. <i>IEEE Transactions on Software Engineering</i> , 30(4), 246-256.
[35]	Perr, J., Appleyard, M. M., & Patrick, P. (2010). Open for business: emerging business models in open source software. <i>International Journal of Technology Management</i> , 52(3/4), 432-456.
[36]	Poba-Nzaou, P., & Uwizemungu, S. (2013). Barriers to Mission-Critical Open Source Software Adoption by Organizations: A Provider Perspective.
[37]	Raghunathan, S., Prasad, A., Mishra, B. K., & Chang, H. (2005). Open source versus closed source: software quality in monopoly and competitive markets. <i>IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans</i> , 35(6), 903-918.
[38]	Riehle, D., Ellenberger, J., Menahem, T., Mikhailovski, B., Natchetoi, Y., Naveh, B., & Odenwald, T. (2009). Open collaboration within corporations using software forges. <i>IEEE software</i> , 26(2), 52-

	58.
[39]	Sacks, M. (2015). Competition between open source and proprietary software: Strategies for survival. <i>Journal of Management Information Systems</i> , 32(3), 268-295.
[40]	Schryen, G., & Kadura, R. (2009, March). Open source vs. closed source software: towards measuring security. In <i>Proceedings of the 2009 ACM symposium on Applied Computing</i> (pp. 2016-2023). ACM.
[41]	Shaikh, M., & Cornford, T. (2011). Total Cost of Ownership of Open Source Software. <i>A report for the UK Cabinet Office supported by OpenForum Europe</i> , 60.
[42]	Singh, P. V., & Phelps, C. (2010). Determinants of open source software license choice: A social influence perspective.
[43]	Sohn, S. Y., & Mok, M. S. (2008). A strategic analysis for successful open source software utilization based on a structural equation model. <i>Journal of Systems and Software</i> , 81(6), 1014-1024.
[44]	Stol, K. J., & Ali Babar, M. (2010, May). Challenges in using open source software in product development: a review of the literature. In <i>Proceedings of the 3rd International Workshop on Emerging Trends in Free/Libre/Open Source Software Research and Development</i> (pp. 17-22). ACM.
[45]	Stol, K. J., Babar, M. A., Avgeriou, P., & Fitzgerald, B. (2011). A comparative study of challenges in integrating Open Source Software and Inner Source Software. <i>Information and Software Technology</i> , 53(12), 1319-1336.
[46]	Watson, R. T., Wynn, D., & Boudreau, M. C. (2005). JBoss: The evolution of professional open source software. <i>MIS Quarterly Executive</i> , 4(3), 329-341.
[47]	West, J. (2003). How open is open enough?: Melding proprietary and open source platform strategies. <i>Research policy</i> , 32(7), 1259-1285.
[48]	Li, Y., Tan, C. H., & Yang, X. (2013). It is all about what we have: A discriminant analysis of organizations' decision to adopt open source software. <i>Decision Support Systems</i> , 56, 56-62.
[49]	Thakur, D. (2012). A limited revolution—The distributional consequences of Open Source Software in North America. <i>Technological Forecasting and Social Change</i> , 79(2), 244-251.
[50]	Tome, L., Allan, K., Meadows, A., & Nyemba-Mudenda, M. (2014). Barriers to open source ERP adoption in South Africa. <i>The African Journal of Information Systems</i> , 6(2), 1.
[51]	Spinellis, D., & Giannikas, V. (2012). Organizational adoption of open source software. <i>Journal of Systems and Software</i> , 85(3), 666-682.
[52]	Kwon, T. H., & Zmud, R. W. (1987, April). Unifying the fragmented models of information systems implementation. In <i>Critical issues in information systems research</i> (pp. 227-251).
[53]	Yildirim, N., & Ansal, H. (2011). Foresighting FLOSS (free/libre/open source software) from a developing country perspective: The case of Turkey. <i>Technovation</i> , 31(12), 666-678.
[54]	Gallivan, M. J. (2001). Organizational adoption and assimilation of complex technological innovations: development and application of a new framework. <i>ACM Sigmis Database</i> , 32(3), 51-85.
[55]	Bhatt, P., Ahmad, A. J., & Roomi, M. A. (2016). Social innovation with open source software: User engagement and development challenges in India. <i>Technovation</i> .
[56]	Moon, Nathan W. and Baker, Paul M.A. (2006). Adoption and Use of Open Source Software: Preliminary Literature Review. Retrieved on 08 Dec. 16 online at <a href="http://www.redhat.com/f/pdf/ossi-literature-review.pdf">http://www.redhat.com/f/pdf/ossi-literature-review.pdf</a>
[57]	Ramanathan, L., and Sundaresan K. (2015). An empirical investigation into the adoption of open source software in Information Technology outsourcing organizations. <i>Journal of Systems and</i>



	<i>Information Technology</i> 17.2 (2015): 167-192.
[58]	Okoli, C., & Schabram, K. (2010). A guide to conducting a systematic literature review of information systems research. <i>Sprouts Work. Pap. Inf. Syst</i> , 10, 26.
[59]	Depietro, R., Wiarda, E., & Fleischer, M. (1990). The context for change: Organization, technology and environment. <i>The processes of technological innovation</i> , 199(0), 151-175.
[60]	Oliveira, T., & Martins, M. F. (2010, September). Information technology adoption models at firm level: review of literature. In <i>European Conference on Information Management and Evaluation</i> (p. 312). Academic Conferences International Limited.
[61]	Webster, J., & Watson, R. T. (2002). Analyzing the past to prepare for the future: Writing a literature review. <i>MIS quarterly</i> , xiii-xxiii.
[62]	Midha, V., & Palvia, P. (2012). Factors affecting the success of Open Source Software. <i>Journal of Systems and Software</i> , 85(4), 895-905.
[63]	Russo, B., Zuliani, P., & Succi, G. (2003, May). Toward an empirical assessment of the benefits of open source software. In <i>Taking Stock of the Bazaar, Proceedings of 3rd Workshop on Open Source Software Engineering, International Conference on Software Engineering</i> (pp. 117-120). Portland, Oregon.
[64]	Ozel, B., Jovanovic, U., Oba, B., & van Leeuwen, M. (2007, June). Perceptions on F/OSS adoption. In <i>IFIP International Conference on Open Source Systems</i> (pp. 319-324). Springer US.
[65]	Gwebu, K. L., & Wang, J. (2010). Seeing eye to eye? An exploratory study of free open source software users' perceptions. <i>Journal of systems and software</i> , 83(11), 2287-2296.
[66]	Holck, J., Larsen, M. H., & Pedersen, M. K. (2004, September). Identifying business barriers and enablers for the adoption of open source software. In <i>Proceedings of the 13th International Conference on Information Systems Development, Vilnius, Lithuania</i> .
[67]	Holck, J., Larsen, M. H., & Pedersen, M. K. (2005, February). Managerial and technical barriers to the adoption of open source software. In <i>International Conference on COTS-Based Software Systems</i> (pp. 289-300). Springer Berlin Heidelberg.
[68]	Thomas, D. R. (2006). A general inductive approach for analyzing qualitative evaluation data. <i>American journal of evaluation</i> , 27(2), 237-246.
[69]	Gichira, C. M., Kahonge, A. M., & Miriti, E. K. (2012). Adoption of Open Source Software by Organizations-A Framework for Kenya. <i>International Journal of Computer Applications</i> , 59(7).
[70]	Chamili, K., Jusoh, Y. Y., Yahaya, J. H., & Pa, N. C. (2012). Selection Criteria for Open Source Software Adoption in Malaysia. <i>International Journal of Advancements in Computing Technology</i> Volume 4, Issue 21, 2012, Pages 278-287
[71]	Kisanjara, S., & Tossy, T. (2014). Investigating Factors Influencing the Adoption and Use of Free and Open Source Software (FOSS) in Tanzanian Higher Learning Institutions: Towards an Individual-Technology-Organizational-Environmental (ITOE) Framework. <i>International Journal of Research In Business And Technology</i> , 5(2), 645-653.
[72]	Williams van Rooij, S. (2007). Perceptions of open source versus commercial software: Is higher education still on the fence? <i>Journal of Research on Technology in Education</i> , 39(4), 433-453.
[73]	van Rooij, S. W. (2009). Adopting open-source software applications in US higher education: A cross-disciplinary review of the literature. <i>Review of Educational Research</i> , 79(2), 682-701.
[74]	Qu, W. G., Yang, Z., & Wang, Z. (2011). Multi-level framework of open source software adoption. <i>Journal of Business Research</i> , 64(9), 997-1003.
[75]	Laila, U., & Bukhari, S. F. A. (2010). Open Source Software (OSS) Adoption Framework for Local Environment and its Comparison. In <i>Innovations in Computing Sciences and Software Engineering</i> (pp. 13-16). Springer Netherlands.

[76]	Miralles, F., Sieber, S., & Valor, J. (2006). An exploratory framework for assessing open source software adoption. <i>Systèmes d'Information et Management</i> , 11(1), 85.
[77]	Bagayoko, C. O., Dufour, J. C., Chaacho, S., Bouhaddou, O., & Fieschi, M. (2010). Open source challenges for hospital information system (HIS) in developing countries: a pilot project in Mali. <i>BMC medical informatics and decision making</i> , 10(1), 1.
[78]	Marsan, J., & Paré, G. (2013). Antecedents of open source software adoption in health care organizations: A qualitative survey of experts in Canada. <i>International journal of medical informatics</i> , 82(8), 731-741.
[79]	Årdal, C., Alstadsæter, A., & Røttingen, J. A. (2011). Common characteristics of open source software development and applicability for drug discovery: a systematic review. <i>Health Research Policy and Systems</i> , 9(1), 1.
[80]	Karopka, T., Schmuhl, H., & Demski, H. (2014). Free/Libre open source software in health care: a review. <i>Healthcare informatics research</i> , 20(1), 11-22.
[81]	Munoz-Cornejo, G., Seaman, C. B., & Koru, A. G. (2008). An Empirical Investigation into the Adoption of Open Source Software in Hospitals. <i>International Journal of Healthcare Information Systems and Informatics (IJHISI)</i> , 3(3), 16-37. doi:10.4018/jhisi.2008070102
[82]	Goode, S. (2005). Something for nothing: management rejection of open source software in Australia's top firms. <i>Information &amp; Management</i> , 42(5), 669-681.
[83]	Golafshani, N. (2003). Understanding Reliability and Validity in Qualitative Research. <i>The Qualitative Report</i> , 8(4), 597-606. Retrieved from <a href="http://nsuworks.nova.edu/tqr/vol8/iss4/6">http://nsuworks.nova.edu/tqr/vol8/iss4/6</a>
[84]	Li, Y., Tan, C. H., Teo, H. H., & Siow, A. (2005, December). A Human Capital Perspective of Organizational Intention to Adopt Open Source Software. In <i>ICIS</i> .
[85]	Marsan, J., Paré, G., & Wybo, M. D. (2012). Has open source software been institutionalized in organizations or not? <i>Information and Software Technology</i> , 54(12), 1308-1316.
[86]	Marsan, J., Paré, G., & Beaudry, A. (2012). Adoption of open source software in organizations: A socio-cognitive perspective. <i>The Journal of Strategic Information Systems</i> , 21(4), 257-273.
[87]	Noy, Chaim (2008). Sampling knowledge: the hermeneutics of snowball sampling in qualitative research. <i>International Journal of Social Research Methodology</i> 11 (2008), 4, pp. 327-344. <a href="http://dx.doi.org/10.1080/13645570701401305">http://dx.doi.org/10.1080/13645570701401305</a>
[88]	Bouras, C., Filopoulos, A., Kokkinos, V., Michalopoulos, S., Papadopoulos, D., & Tseliou, G. (2014). Policy recommendations for public administrators on free and open source software usage. <i>Telematics and Informatics</i> , 31(2), 237-252.
[89]	van Loon, A., & Toshkov, D. (2015). Adopting open source software in public administration: The importance of boundary spanners and political commitment. <i>Government Information Quarterly</i> , 32(2), 207-215.
[90]	Deng, J., Seifert, T., & Vogel, S. (2003, May). Towards a Product Model of Open Source Software in a Commercial Environment. In <i>3rd Workshop on Open Source Software Engineering</i> (pp. 31-38).
[91]	Shaikh, M. (2016). Negotiating open source software adoption in the UK public sector. <i>Government Information Quarterly</i> , 33(1), 115-132.
[92]	Hippel, E. V., & Krogh, G. V. (2003). Open source software and the "private-collective" innovation model: Issues for organization science. <i>Organization science</i> , 14(2), 209-223.
[93]	Lakhani, K. R., & Von Hippel, E. (2003). How open source software works: "free" user-to-user assistance. <i>Research policy</i> , 32(6), 923-943.
[94]	Markus, M. L. (1983). Power, politics, and MIS implementation. <i>Communications of the ACM</i> , 26(6), 430-444.

[95]	Amoako-Gyampah, K., & Salam, A. F. (2004). An extension of the technology acceptance model in an ERP implementation environment. <i>Information &amp; Management</i> , 41(6), 731-745.
------	--

## BOOKS:

[B1]	Laurent, A. M. S. (2004). <i>Understanding open source and free software licensing</i> . O'Reilly Media, Inc. <a href="http://www.oreilly.com/openbook/osfreesoft/book/">http://www.oreilly.com/openbook/osfreesoft/book/</a>
[B2]	Editors DiBona, C., Ockman, S. and Stone, M. (1999). <i>Open sources: Voices from the open source revolution</i> . O'Reilly Media, Inc. Chapters written by various authors. <a href="http://www.oreilly.com/openbook/opensources/book/">http://www.oreilly.com/openbook/opensources/book/</a>
[B3]	Goldman, R., & Gabriel, R. P. (2005). <i>Innovation happens elsewhere: Open source as business strategy</i> . Morgan Kaufmann. <a href="http://dreamsongs.com/IHE/IHE.html">http://dreamsongs.com/IHE/IHE.html</a>
[B4]	Fogel, K. (2005). <i>Producing open source software: How to run a successful free software project</i> . O'Reilly Media, Inc. <a href="http://producingoss.com/en/index.html">http://producingoss.com/en/index.html</a>
[B5]	Raymond, E. (1999). The cathedral and the bazaar. <i>Knowledge, Technology &amp; Policy</i> , 12(3), 23-49. <a href="http://www.catb.org/esr/writings/cathedral-bazaar/">http://www.catb.org/esr/writings/cathedral-bazaar/</a>
[B6]	Williams, S. (2002). Free as in Freedom (2.0)-Richard Stallman and the Free Software Revolution. <i>Boston: The Free Software Foundation</i> . <a href="https://archive.org/stream/faif-2.0">https://archive.org/stream/faif-2.0</a>
[B7]	Woods, D., & Guliani, G. (2005). <i>Open Source for the Enterprise: Managing risks, reaping rewards</i> . <a href="http://shop.oreilly.com/product/9780596101190.do">http://shop.oreilly.com/product/9780596101190.do</a>
[B8]	Baker, J. (2012). The technology–organization–environment framework. In <i>Information systems theory</i> (pp. 231-245). Springer New York.

## OTHER SOURCES

[O1]	“Benefits of Using Open Source Software.” Retrieved on 08 Dec. 16 <a href="http://open-source.gbdirect.co.uk/migration/benefit.html">http://open-source.gbdirect.co.uk/migration/benefit.html</a>
[O2]	“Gartner Survey Reveals More than Half of Respondents Have Adopted Open-Source Software Solutions as Part of IT Strategy.” Gartner, 8 Feb. 2011, <a href="https://www.gartner.com/newsroom/id/1541414">https://www.gartner.com/newsroom/id/1541414</a>
[O3]	“Microsoft software license terms - Windows operating system”, Retrieved on 08 Dec. 16, <a href="https://www.microsoft.com/en-us/Useterms/OEM/Windows/10/UseTerms_OEM_Windows_10_English.htm">https://www.microsoft.com/en-us/Useterms/OEM/Windows/10/UseTerms_OEM_Windows_10_English.htm</a>
[O4]	“Open Source Software Key Driver for Innovation at Global Companies, Wipro-Oxford Economics Survey Shows.” Wipro-Oxford Economics, 27 July 2015, <a href="http://www.wipro.com/newsroom/press-releases/open-source-software-key-driver-for-innovation-at-global-companies-wipro-oxford-economics-survey-shows/">http://www.wipro.com/newsroom/press-releases/open-source-software-key-driver-for-innovation-at-global-companies-wipro-oxford-economics-survey-shows/</a> . Based on a study from Oxford Economics here <a href="https://www.oxfordeconomics.com/recent-releases/the-open-source-era">https://www.oxfordeconomics.com/recent-releases/the-open-source-era</a>
[O5]	“Open Source Software.” Australian Department of Finance, 10 Feb. 2014. Retrieved on 08 Dec. 16. <a href="https://www.finance.gov.au/policy-guides-procurement/open-source-software/">https://www.finance.gov.au/policy-guides-procurement/open-source-software/</a>
[O6]	“Top Open Source Software Licenses   Black Duck.” <i>Black Duck Software</i> . Retrieved on 08 Dec. 16, <a href="https://www.blackducksoftware.com/top-open-source-licenses">https://www.blackducksoftware.com/top-open-source-licenses</a>
[O7]	Amrit, Chintan (2013). “EYR4 Blog #8: Innovation Dynamics of Open Source Software.”

	<i>SURF Blog</i> , 31 Oct. 2013, <a href="https://blog.surf.nl/en/eyr4-blog-8-innovation-dynamics-of-open-source-software/">https://blog.surf.nl/en/eyr4-blog-8-innovation-dynamics-of-open-source-software/</a>
[O8]	Andreessen, Marc (2011). "Marc Andreessen on Why Software Is Eating the World." <i>WSJ</i> , 19 Aug. 2011. Web. 29 Oct. 2016. <a href="http://www.wsj.com/articles/SB10001424053111903480904576512250915629460">http://www.wsj.com/articles/SB10001424053111903480904576512250915629460</a>
[O9]	Aslett, M. (2011). <i>The trend towards permissive licensing</i> . 06.06.2011. <a href="https://blogs.the451group.com/opensource/2011/06/06/the-trend-towards-permissive-licensing/">https://blogs.the451group.com/opensource/2011/06/06/the-trend-towards-permissive-licensing/</a>
[O10]	Balter, Ben. "Open Source License Usage on GitHub.com." <i>GitHub</i> . 09 Mar. 2015. <a href="https://github.com/blog/1964-open-source-license-usage-on-github-com">https://github.com/blog/1964-open-source-license-usage-on-github-com</a>
[O11]	Buffett, Brian (2014). "Factors Influencing Open Source Software Ad Option in Public Sector National and International Statistical Organisations." 17 Feb. 2014. Retrieved on 08 Dec. 16 <a href="https://www.unece.org/fileadmin/DAM/stats/documents/ece/ces/ge.50/2014/Topic_1_UNE_SCO.pdf">https://www.unece.org/fileadmin/DAM/stats/documents/ece/ces/ge.50/2014/Topic_1_UNE_SCO.pdf</a>
[O12]	Congdon, Lee (2015). "8 Advantages of Using Open Source in the Enterprise." <i>The Enterprisers Project</i> . 3 Feb. 2015. <a href="https://enterpriseproject.com/article/2015/1/top-advantages-open-source-offers-over-proprietary-solutions">https://enterpriseproject.com/article/2015/1/top-advantages-open-source-offers-over-proprietary-solutions</a>
[O13]	Corbet, Jonathan (2015). "Development Activity in LibreOffice and OpenOffice." <i>LWN.net</i> , 25 Mar. 2015, <a href="https://lwn.net/Articles/637735/">https://lwn.net/Articles/637735/</a>
[O14]	Coverity, Inc. "Coverity Scan Report Finds Open Source Software Quality Outpaces Proprietary Code for the First Time - Synopsis." Published 2013. Retrieved on 08 Dec. 16. <a href="http://www.coverity.com/press-releases/coverity-scan-report-finds-open-source-software-quality-outpaces-proprietary-code-for-the-first-time/">http://www.coverity.com/press-releases/coverity-scan-report-finds-open-source-software-quality-outpaces-proprietary-code-for-the-first-time/</a>
[O15]	Devin, Leonard, and Rick Clough (2016). " <i>How GE Exorcised the Ghost of Jack Welch to Become a 124-Year-Old Startup</i> ." Bloomberg, 17 Mar. 2016, <a href="https://www.bloomberg.com/news/articles/2016-03-17/how-ge-exorcised-the-ghost-of-jack-welch-to-become-a-124-year-old-startup">https://www.bloomberg.com/news/articles/2016-03-17/how-ge-exorcised-the-ghost-of-jack-welch-to-become-a-124-year-old-startup</a>
[O16]	Edge, Jake (2016). "On the Boundaries of GPL Enforcement." <i>LWN.net</i> . 20 July 2016. <a href="https://lwn.net/Articles/694890/">https://lwn.net/Articles/694890/</a>
[O17]	Haack, Phil (2016). "Source Available vs Open Source vs Free Software." 25 July 2006. <a href="http://haacked.com/archive/2006/07/26/CodeAvailableVsOpenSourceVsFreeSoftware.aspx/">http://haacked.com/archive/2006/07/26/CodeAvailableVsOpenSourceVsFreeSoftware.aspx/</a>
[O18]	Hall, Andrew (2015). Open-Source Business Models: Making money by giving it away. Linux Collaboration Summit 20 <sup>th</sup> of February 2015. Retrieved on 08 Dec. 16 <a href="http://events.linuxfoundation.org/sites/events/files/slides/lfcs15_hall.pdf">http://events.linuxfoundation.org/sites/events/files/slides/lfcs15_hall.pdf</a>
[O19]	Kwayisi, Michael (2012). "Bill Gates's Betrayal of the Hacker Ethic." 4 Dec. 2012, <a href="http://www.kwayisi.org/writings/betrayal.html">www.kwayisi.org/writings/betrayal.html</a>
[O20]	Langham, Matthew (2012). "The Business of Open Source." Oss-Watch.ac.uk, 9 Sept. 2012, <a href="http://oss-watch.ac.uk/resources/businessofopensource">http://oss-watch.ac.uk/resources/businessofopensource</a>
[O21]	Loftus, Tom (2016). "CIO Stats: Worldwide IT Spending to Rise in 2017, Gartner Says." <i>WSJ</i> . 24 Oct. 2016. Web. 29 Oct. 2016. <a href="http://blogs.wsj.com/cio/2016/10/24/cio-stats-worldwide-it-spending-to-rise-in-2017-gartner-says/">http://blogs.wsj.com/cio/2016/10/24/cio-stats-worldwide-it-spending-to-rise-in-2017-gartner-says/</a>
[O22]	Lyons, Ken (2014). "Distinguish Between Primary and Secondary Sources." University of California, Santa Cruz, 14 Aug. 2014, <a href="http://guides.library.ucsc.edu/c.php?g=119713&amp;p=780816">http://guides.library.ucsc.edu/c.php?g=119713&amp;p=780816</a>
[O23]	Mickos, Marten (2013). "MySQL CEO on 2 Open Source Business Models." Heavybit

	Industries, 15 Oct. 2013, <a href="http://www.heavybit.com/library/video/mysql-ceo-on-2-open-source-business-models/">www.heavybit.com/library/video/mysql-ceo-on-2-open-source-business-models/</a>
[O24]	Montague, B. (2013). Why you should use a BSD style license for your Open Source Project. Retrieved on 08 Dec. 16. Available from <a href="https://www.freebsd.org/doc/en/articles/bsd-gpl/index.html">https://www.freebsd.org/doc/en/articles/bsd-gpl/index.html</a>
[O25]	Morgan, Dewi (2006). "GPL Is Evil, Viral and Parasitic. Public Domain Is Good." 19 Nov. 2006. <a href="http://blog.greggman.com/blog/time_for_the_gpl_to_die">http://blog.greggman.com/blog/time_for_the_gpl_to_die</a>
[O26]	Optimus Information White Paper (2015, June). "Open-Source vs. Proprietary Software Pros and Cons". Retrieved on 08 Dec. 16 <a href="http://www.optimusinfo.com/downloads/white-paper/open-source-vs-proprietary-software-pros-and-cons.pdf">http://www.optimusinfo.com/downloads/white-paper/open-source-vs-proprietary-software-pros-and-cons.pdf</a>
[O27]	Petechuk, David (2016). "Bill Gates 1955— Biography." Reference For Business, Retrieved on 08 Dec. 16, <a href="http://www.referenceforbusiness.com/biography/f-l/gates-bill-1955.html">www.referenceforbusiness.com/biography/f-l/gates-bill-1955.html</a>
[O28]	Riehle, Dirk (2016a). Lecture slides and notes to the course "Free/Libre and Open-Source Software" held in Summer term 2016 at FAU. Not publically available <a href="https://osr.cs.fau.de/teaching/ss2016/floss/">https://osr.cs.fau.de/teaching/ss2016/floss/</a>
[O29]	Stallman, Richard (2016a). "Why Open Source Misses the Point of Free Software." Gnu.org, <a href="http://www.gnu.org/philosophy/open-source-misses-the-point.html">www.gnu.org/philosophy/open-source-misses-the-point.html</a> . Retrieved 08 Dec. 16
[O30]	Stallman, Richard (2016b). "Proprietary Software Is Often Malware." Gnu.org, <a href="https://www.gnu.org/proprietary/proprietary.html">https://www.gnu.org/proprietary/proprietary.html</a> Retrieved 02 Nov. 2016.
[O31]	The 451 Group. "Open Source Is Not a Business Model.", Oct. 2008. Retrieved on 06 Nov. 2016. <a href="https://blogs.the451group.com/opensource/2008/10/13/open-source-is-not-a-business-model/">https://blogs.the451group.com/opensource/2008/10/13/open-source-is-not-a-business-model/</a> , Report available as of 08 Dec. 16: <a href="http://www.shenlanguage.org/Download/osnbm.pdf">http://www.shenlanguage.org/Download/osnbm.pdf</a>
[O32]	User: apsilers. "Why Was Open Source as a Term Created, Although Free Software Was Already Established?" Opensource - StackExchange, 25 June 2015, <a href="https://opensource.stackexchange.com/questions/348/why-was-open-source-as-a-term-created-although-free-software-was-already-establ">https://opensource.stackexchange.com/questions/348/why-was-open-source-as-a-term-created-although-free-software-was-already-establ</a>
[O33]	Various Authors. "Using Open Source Software." <i>UK Government Service Design Manual</i> . Retrieved on 08 Dec. 16. <a href="https://www.gov.uk/service-manual/making-software/open-source.html">https://www.gov.uk/service-manual/making-software/open-source.html</a>
[O34]	Vaughan-Nichols, Steven J. "Red Hat Becomes First \$2b Open-source Company." <i>ZDNet</i> . 22 Mar. 2016. <a href="http://www.zdnet.com/article/red-hat-becomes-first-2b-open-source-company/">http://www.zdnet.com/article/red-hat-becomes-first-2b-open-source-company/</a>
[O35]	Wurster, Laurie (2011). "Open Source Software Hits a Strategic Tipping Point." <i>Harvard Business Review</i> , 9 Mar. 2011, <a href="https://hbr.org/2011/03/open-source-software-hits-a-st">https://hbr.org/2011/03/open-source-software-hits-a-st</a>
[O36]	J. T. S. Moore (Director). (2001). <i>Revolution OS</i> [Video file]. Retrieved 08 Dec. 16, from <a href="https://www.youtube.com/watch?v=jw8K460vx1c">https://www.youtube.com/watch?v=jw8K460vx1c</a>
[O37]	Microsoft. "Third Party Disclosures." <i>Third Party Disclosures</i> . Web. 08 Dec. 16. <a href="https://thirdpartysource.microsoft.com">https://thirdpartysource.microsoft.com</a>
[O38]	Kerner, Sean M. (2016). "How Google Does Open Source." 26 Aug. 2016. <a href="http://www.datamation.com/open-source/how-google-does-open-source.html">http://www.datamation.com/open-source/how-google-does-open-source.html</a>
[O39]	Pearce, James. "2013: A Year of Open Source at Facebook.", 20 Dec. 2013. <a href="https://code.facebook.com/posts/604847252884576/2013-a-year-of-open-source-at-facebook/">https://code.facebook.com/posts/604847252884576/2013-a-year-of-open-source-at-facebook/</a>
[O40]	Heath, Nick. (2016). "Open-source Pioneer Munich Debates Report That Suggests Abandoning Linux for Windows 10." <i>TechRepublic</i> . 09 Nov. 2016. <a href="http://www.techrepublic.com/article/open-source-pioneer-munich-debates-report-that-">http://www.techrepublic.com/article/open-source-pioneer-munich-debates-report-that-</a>

	<a href="#">suggests-abandoning-linux-for-windows-10/</a>
[O41]	Vaughan-Nichols, Steven J. (2015). "It's an Open-source World: 78 Percent of Companies Run Open-source Software." <i>ZDNet</i> . 16 Apr. 2015. <a href="http://www.zdnet.com/article/its-an-open-source-world-78-percent-of-companies-run-open-source-software/">http://www.zdnet.com/article/its-an-open-source-world-78-percent-of-companies-run-open-source-software/</a> . Results of more recent is available here: <a href="http://www.northbridge.com/2016-future-open-source-survey-results">http://www.northbridge.com/2016-future-open-source-survey-results</a> , Retrieved on 08 Dec. 16

## 8 Abbreviations

API	Application programming interface
COTS	Commercial off-the-shelf (software)
CMS	Content management system
EULA	End User Licence Agreement
FLOSS	Free/Libre and open-source software
FOSS	Free and open-source software
FSF	Free Software Foundation
GPL	GNU General Public License
IDE	Integrated development environment
IRC	Instant Relay Chat
IS	Information Systems
NA	Not available
OSI	Open Source Initiative
OSS	Open-source software
R&D	Research and development
SLA	Service Level Agreement
SME(s)	Small and medium-sized enterprises
SW	Software
TCO	Total Costs of Ownership
UX/UI	User Experience / User Interface

## 9 Appendix

### PROTOCOL

#### Protocol for IS Development and Implementation in Business Context

##### STEPS TO CONDUCT IN ORDER TO HAND-IN THE PAPER

[CONTINUOUSLY UPDATED]

1. Read "overall" papers such as [9] and [20] to get an overview of the FLOSS topic
2. (after changing the topic with Nikolaus) Clarify final Research Questions and plan how to write this paper
  - a. Read articles how to write literature review [58] [68]
3. Write chapter One
  - a. Outline ideas & structure
  - b. Establish methodology for the introductory part (i.e. here just for chapter 2) and outline how second part will be done
  - c. Make necessary graphic(s)
  - d. Check for style & grammar AND appropriate context & meaning & logic & objective which must be clearly explained
4. Think how to write chapter Two (as detailed as possible)
5. Gather necessary sources [albeit most probably unsystematically] from Google Scholar, flosshub and books!
6. Write chapter Two
  - a. Explain a brief history of FLOSS development (FSF, OSI) and proprietary software (Gates @ MSFT)
  - b. Describe it further according to business model(s) and software licensing & IPR
  - c. Create table that both characterises and differentiates software with relevant ("target") pluses and minuses [for adoption firms]
7. Think what is necessary to do before chapter three + how it will be structured
8. Write & Outline methodology → as detailed as possible because it will be my guideline ← follow [58/68]

9. At same time, start gathering relevant adoption literature which talks about barriers and factors (because usually seen together)
  - a. [Theoretically from 1985 (FSF) but practically from >2000]
  - b. Create MS Word database and Excel file that will capture all the "things"
  - c. Go over all sources and categorize & put them into it [again 58/68]
  - d. Make them iteratively small(er) number, directly in XMind
  - e. Make Table/Graphic – in the middle "barriers for adoption"
10. Once done, back to writing:
  - a. Update established methodology and revise all text for clarity & meaning & logic ("red-line") – from chapter 0 until 3.1
  - b. Explain (updated version of) TOE(I) framework
  - c. Go (subjectively) over some most important barriers and write 0.5/1 paragraph per each → Chapter 3.2 ← try to link them (a bit) as well
  - d. (Chapter 3.3) Explain major specifics/challenges/current adoption specifically of healthcare industry
11. Finish up & hurry up with discussion (mainly limitations but also see "resistance and TAM presentations"), implications and conclusion (short this one)
12. Make Excel and Word look nicer and append them into the document

Figure 7 is how my research protocol looks like at the very end.



# FLOSS ADOPTION BARRIERS

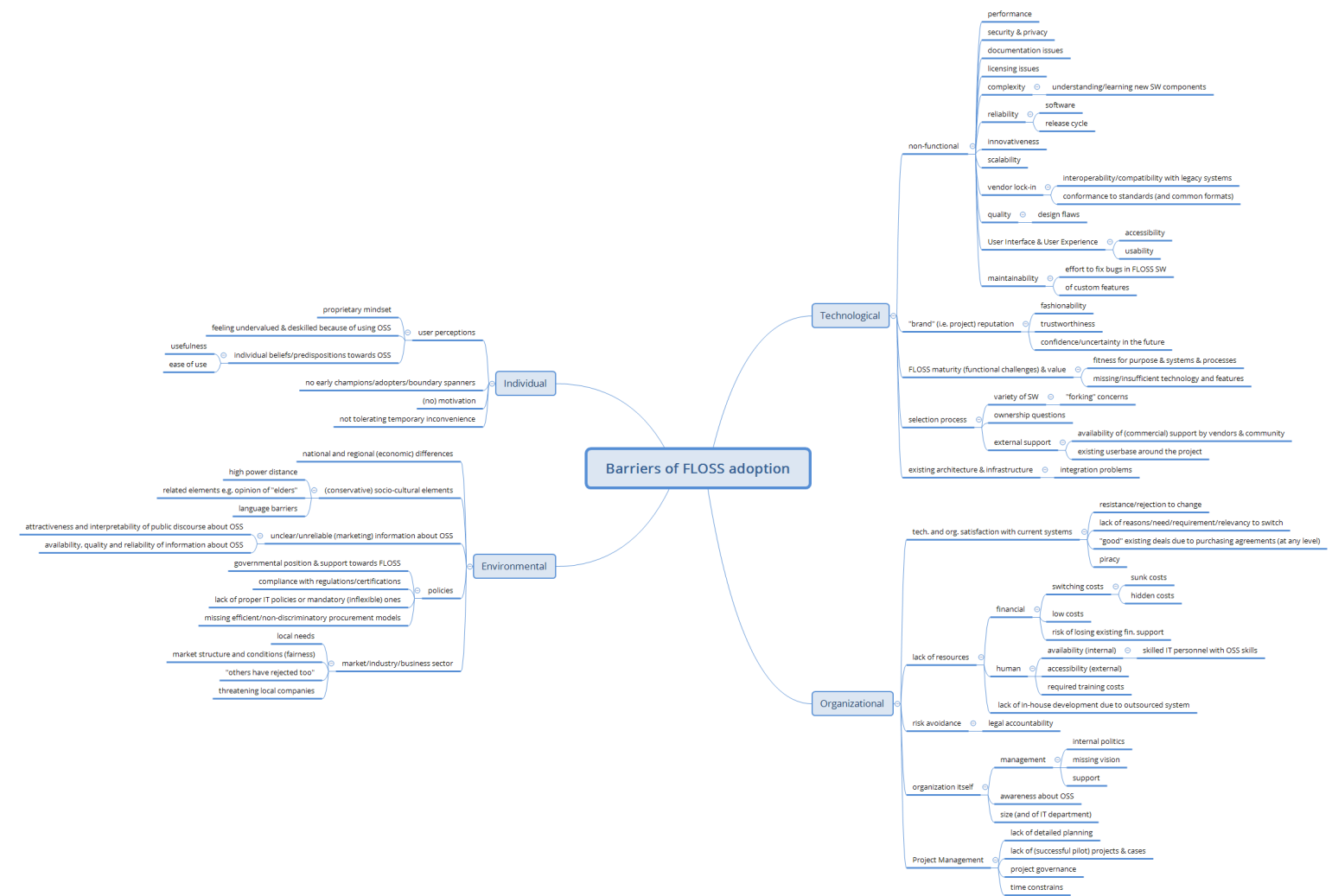


Figure 8 provides an expanded and detailed look on categorization of FLOSS integration challenges.



## INHIBITOR MATRIX

Literature Number								Organizational												
	existing architecture		Brand/Reputation					Lack of Ressources								satisfaction				
	existing user base & community	integration problems	Existing architecture/infrastructure	trustworthiness	confidence/uncertainty	brand reputation	fashionability	switching Costs/TCO/sunk costs	hidden costs	low costs	IT/financial resources/budget limited	lack of financial support and risk of losing existing funds	Availability/accessibility of human capital/skilled personnel (with OSS)	training costs of employees	development of systems is outsourced - dependency on inhouse	satisfaction with current system	"Good licensing deals"/existing purchasing agreements	Lack of reasons/resistance to switch	no need/requirement/relevancy	piracy
[72]								x					x	x		x				
[71]			x	x	x			x					x	x				x		
[73]								x					x					x		
[75]			x			x							x	x			x	x		x
[76]			x		x	x		x						x				x		
[74]								x					x							
[77]		x										x	x	x				x		
[78]		x	x					x					x							
[80]	x												x							
[81]		x									x		x		x		x			
[82]		x	x				x	x			x		x	x			x	x	x	

Figure 9 shows here only an excerpt from the Excel sheet. The full matrix is provided in the supplement. It should be noted that in many cases confusing and improper labels of individual barriers (e.g. "switching costs/TCO/sunk costs") have been taken into account later when creating a final mind map seen in Figure 8. Therefore this is sheet needs to be considered as "very raw" version of my inhibiting categories.

Article Citation count	Source/ Conference	Public/Private Sector and type of study	Summary/Conclusion	Adoption barriers identified/cited	Adoption factors identified/cited
<p>“Factors Influencing Adoption of Open Source Software – An Exploratory Study”</p> <p>Bhadauria et al. (2009) [2]</p> <p>3 (Google Scholar)</p>	<p>Proceedings of the Fifteenth Americas Conference on IS</p>	<p>2 US based firms, one in SW solutions space and another in telecommunications industry.</p> <p>Explorative Empirical Qualitative</p>	<p>Through multi-site case study and framework by Kwon &amp; Zmud (1987) [52], interviews with 12 informants are conducted to find critical factors that influence OSS adoption.</p>	<p>Similarly to [52]: Technological support (availability, reliability, scalability → not in mission-critical systems); Structural (deployment affects); Organizational operating environment and business strategy; Individual (tech. users comfortable with functionality); Task related (suitability for low-end apps and non-updatable environment)</p>	<p>Costs savings, “factors, including, technological attributes, network externalities, organizational capabilities, vendor lock-in, and influence of the user community”, enhanced trialability, easy-of-use, human capital</p>
<p>“Critical success factors for migrating to OSS-on-the-desktop: common themes across three South African case studies”</p> <p>Brink et al. (2006) [6]</p> <p>8 (Google Scholar)</p>	<p>IFIP International Conference on Open Source Systems</p>	<p>Government, private industry, educational institution in S. Africa</p> <p>Exploratory Empirical Qualitative</p>	<p>Using a case study approach, 3 organizations in South Africa that have migrated from non-free to free desktop environments are analyzed to uncover success factors.</p>	<p>-</p>	<p>Gaining competitive advantage with better financial performance by avoiding fines from The Software Alliance; support of top management; user awareness and communication; detailed planning/analysis/ testing; training; (pilot projects and) partial migration; support</p>
<p>“An empirical study on software development with open source components in the Chinese software industry”</p> <p>Chen et al. (2008) [7]</p> <p>28 (Google Scholar)</p>	<p>(Journal) Software Process: Improvement and Practice</p>	<p>Chinese software companies</p> <p>Explanatory Empirical Quantitative &amp; Qualitative</p>	<p>Based on 47 projects in 43 surveyed companies, authors investigate how Chinese companies decide to use, integrate and maintain FLOSS-based systems. They aim is to establish guidelines to ease FLOSS-based development.</p>	<p>Learning OSS components; technical issues (bug fixes needed); lifecycle costs (selection, maintaining, upgrades etc.); requirements compliance – legal/IP exposure (and avoidance of it); adaptation and configuration;</p>	<p>Modifiability, low-license costs, standardized function and architecture</p>
<p>“Migration discourse structures: Escaping Microsoft’s desktop path”</p> <p>Dobusch (2008) [11]</p> <p>8 (Google Scholar)</p>	<p>IFIP International Conference on Open Source Systems</p>	<p>Regional government in Germany</p> <p>Theoretical &amp; Empirical Explanatory Qualitative</p>	<p>Munich’s migration to FLOSS ended up into complete restructuring of IT organization. Uses interviews and structuration theory with discourse analysis to explain why and how this could happen.</p>	<p>- software monopoly &amp; lock-in - not necessarily economic or technological reasons for switching, → “always political”</p>	<p>Lower purchasing costs, end of vendor lock-in, cross-platform</p>
<p>“Open source software adoption: anatomy of success and failure”</p>	<p>Chapter in a book “Multi-Disciplinary</p>	<p>Irish public hospital</p>	<p>Discuss a conducted longitudinal single-site case study of two OSS deployments</p>	<p>Knowledge deficit &amp; awareness Mandatory/voluntary use</p>	<p>Cuts in IT budget</p>

Fitzgerald (2011) [15]  37 (Google Scholar)	Advancement in Open Source Software and Processes”	Exploratory Empirical (but theory heavy) Qualitative	(desktop office suit and email application) in the organization (secondary adoption).	No initial trial & missing training and support Negative image “downplay the observability of the differences between” OSS	
“Open source software adoption: motivations of adopters and amotivations of non-adopters”  Li et al. (2011) [24]  19 (Google Scholar)	ACM SIGMIS Database: the DATABASE for Advances in IS	476 undergraduate students in Asian setting  Mixed methods (theoretically-grounded)	Using Self-Determination theory, a theoretical model to identify individual motivational factors is constructed and subsequently assessed with a survey.	FLOSS Vendors need to look at: -Users awareness of OSS -OSS usefulness for users -Differences in expectations between developers and users (understand users’ needs better)	institutional pressures to adopt OSS; source code availability; avoiding vendor lock-in; costs; ease of use; individual intrinsic & extrinsic motivation factors and amotivation (beliefs in capacity, effort and strategy)
“A discriminant analysis of organizations' decision to adopt open source software”  Li et al. (2013) [48]  3 (ScienceDirect)	(Journal) Technological Forecasting and Social Change	FLOSS adopting and non-adopting firms in China  Exploratory Empirical (with a “theoretical angle”) Quantitative	Based on a survey of 215 companies (CIOs), a discriminant analysis of firm’s adoption behaviors was conducted.	Perceived uncertainty (reliability) in (technical) service and support leading to high switching costs	Internal and external human capital as a critical factor to adopt FLOSS
“Open innovation in secondary software firms: an exploration of managers' perceptions of open source software”  Morgan & Finnegan (2010) [27]  45 (Google Scholar)	ACM SIGMIS Database: the DATABASE for Advances in IS	Secondary software sector in Europe  Theoretical & Empirical Exploratory Quantitative	A field study interviewing 13 managers in companies investigates how managers perceives benefits and drawbacks of FLOSS.	16 factors categorized in 4 categories (technological, organizational, environmental and individual) that may affect FLOSS adoption  10 benefits and 11 drawbacks are identified too	
“Organizational adoption of open source software: barriers and remedies”  Nagy et al. (2010) [30]  71 (Google Scholar)	Communications of the ACM	NA	Introduce and discuss barriers and offer potential remedies for them.	5 categories: knowledge barrier, legacy integration, forking, sunk costs and technological immaturity	-

<p>“Barriers to open source software adoption in Quebec’s health care organizations”</p> <p>Paré et al. (2008) [33]</p> <p>38 (Google Scholar)</p>	<p>Journal of medical systems</p>	<p>Healthcare in Canada</p> <p>Empirical Exploratory Qualitative</p>	<p>15 CIO interviews are conducted from the Canadian healthcare organizations to identify main barriers to their adoption of FLOSS.</p>	<p>At first 11 challenges was identified. These were discussed with interviewers resulting into 7 main ones: lack of resources &amp; <u>expertise</u>, responsible 3<sup>rd</sup> party and information sharing policies; <u>politics</u>; reliability issues; conservative nature of healthcare and finally hidden costs for maintenance and support</p>	-
<p>“Barriers to Mission-Critical Open Source Software Adoption by Organizations: A Provider Perspective”</p> <p>Poba-Nzaou &amp; Uwizyemungu (2013) [36]</p> <p>0 (Scopus)</p> <p>Keyword: TOE</p>	<p>Proceedings of the Nineteenth Americas Conference on IS</p>	<p>Software industry in Canada and France</p> <p>Empirical Exploratory Qualitative</p>	<p>Conducted a Delphi study of 29 experts in FRA and CAN to research barriers in adoption of mission-critical OSS.</p>	<p>Identify 3 categories (environmental, organizational, OSS specific) with 19 challenges</p>	-
<p>“A strategic analysis for successful open source software utilization based on a structural equation model”</p> <p>Sohn &amp; Mok (2008) [43]</p> <p>35 (Google Search)</p>	<p>Journal of Systems and Software</p>	<p>77 IT companies in Korea</p> <p>Explanatory Theoretical at first, later carried out a mixed empirical study with surveys and interviews</p>	<p>Develop a structural equation model to investigate relationships between quality factors and OSS utilization. Furthermore, they want to find out what affects OSS utilization and suggest “utilization” index.</p>	<p>Developers sharing their knowledge. For increase in utilization, legal protection of IP must be ensured.</p>	<p>Following quality factors (ISO/IEC 9126) are found to be relevant: “functionality, efficiency, and sharing have significant influences on OSS utilization directly, while portability, reliability, and maintainability influence OSS utilization indirectly.” Flexibility, portability</p>
<p>“Organizational adoption of open source software”</p> <p>Spinellis &amp; Giannikas (2012) [51]</p> <p>40 (Google Scholar)</p>	<p>Journal of Systems and Software</p>	<p>US Fortune 1000 companies</p> <p>Explanatory Empirical Quantitative</p>	<p>Through a scrapping 278 million web server logs, examine organizational factors and behaviors when adopting OSS.</p>	<p>Switching costs, loyalty, level of trust, risks, knowledge barriers, integration with legacy systems, forking, sunk costs and tech immaturity [30]. Categorize their research questions into technological, organizational and individual factors.</p>	<p>Lower costs/TCO, software features, portability, avoidance commercial license management, customizability, quicker deployment, company’s growth stage</p>

<p>“Challenges in using open source software in product development: a review of the literature”</p> <p>Stol &amp; Ali Babar (2010) [44]</p> <p>17 (Google Scholar)</p>	<p>Proceedings of the 3rd International Workshop on Emerging Trends in Free/Libre/Open Source Software Research and Development</p>	<p>NA</p> <p>Explorative literature review</p>	<p>Present and discuss in detail 21 challenges of adoption that may arise, based on reporting in 17 studies.</p>	<p>5 categories with 29 individual challenges: product selection; documentation; community, support and maintenance; integration and architecture; migration and usage; legal and business</p>	-
<p>“A limited revolution—The distributional consequences of Open Source Software in North America”</p> <p>Thakur (2012) [49]</p> <p>11 (Google Scholar)</p>	<p>(Journal) Technological Forecasting and Social Change</p>	<p>USA and Canada (NA)</p> <p>Explorative Empirical Qualitative</p>	<p>By conducting interviews with people from academy, industry, NGOs and government, author examines distribution of FLOSS benefits and costs in the society.</p>	<p>Firm’s industry and size; skills requirements of OSS; Restrictive gov. policies and laws (e.g. DMCA, software patents, market structure); Legal/procurement bias towards non-free software</p>	<p>Reduced costs, security, efficiency</p>
<p>“Barriers to open source ERP adoption in South Africa”</p> <p>Tome et al. (2014) [50]</p> <p>2 (Google Scholar)</p> <p>Keyword: TOE</p>	<p>The African Journal of Information Systems</p>	<p>Various organizations in South Africa</p> <p>Explorative Mixed methods (de- and inductive &amp; qual. and quant.)</p>	<p>By means of a survey with 158 responses, investigated adoption of FLOSS ERP systems in South African context.</p>	<p>22 barriers, used in questionnaire were identified and grouped into individual, organizational, technological and environmental category. Further, it showed that generally accepted challenges with FLOSS might not relate to the FLOSS ERP due to unique application type.</p>	-
<p>“Social innovation with open source software: User engagement and development challenges in India”</p> <p>Bhatt et al. (2016) [55]</p> <p>3 (Google Scholar)</p>	<p>(Journal) Technovation</p>	<p>Social enterprise in India</p> <p>Explorative Empirical Qualitative &amp; inductive methods of analysis</p>	<p>A longitudinal single-case study of a social enterprise, 9 employees and its FOSS product offered to e.g. farmers in India.</p>	<p>socio-cultural barriers and issues surrounding literacy and language; lack of engagement</p>	<p>Cost savings; increased productivity; trialability; experimentation; customization; free distribution; quicker response to market demands; “desire for [country’s] independence, a drive for security and autonomy and a means to address IP rights enforcement”</p>
<p>“An empirical investigation into the adoption of open source software in Information Technology outsourcing organizations”</p>	<p>Journal of Systems and Information Technology</p>	<p>482 employees in Indian IT services firms (working for</p>	<p>Develop a conceptual model based on TOE framework to identify factors and to what extent they influence adoption of OSS in global IT outsourcing firms</p>	<p>Reliability, legal concern, software costs, management support (most important), OSS support availability and software vendor</p>	-

Ramanathan & Sundaresan (2015) [57]  1 (Scopus)		clients in various sectors)  Explanatory Theoretical and Empirical validation Quantitative	serviced by Indian IT vendors. Using a survey, they found out that 84% use to some degree OSS.	relationship are relevant barriers for OSS adoption by IT outsourcing organizations.  License concerns and IT outsourcing strategies are not significant for OSS adoption.																											
“Factors affecting the success of Open Source Software”  Midha & Palvia (2012) [62]  47 (Google Scholar)	Journal of Systems and Software	283 OSS projects  Explanatory Empirical Quantitative	By using 3-year long longitudinal analysis of 283 OSS projects examining their popularity and developer activity, authors try – based on Cue Utilization Theory – understand impact of success, intrinsic and extrinsic, factors.	OSS Design flaws  <table><tr><td colspan="2">Table 3 Path coefficients for all the models.</td></tr><tr><td>Structural path</td><td>Conclusion</td></tr><tr><td colspan="2"><hr/></td></tr><tr><td>H1: Technical success → Market success</td><td>Not supported</td></tr><tr><td>H2a: License type → Market success</td><td>Supported at t1</td></tr><tr><td>H2b: License type → Technical success</td><td>Supported at t3, t4</td></tr><tr><td>H3a: User base → Market success</td><td>Supported</td></tr><tr><td>H3b: Developer base → Market success</td><td>Mixed results</td></tr><tr><td>H3c: Developer base → Technical success</td><td>Mixed results</td></tr><tr><td>H4: Language translations → Market success</td><td>Supported</td></tr><tr><td>H5: Responsibility assignment → Technical success</td><td>Supported</td></tr><tr><td>H6: Complexity → Technical success</td><td>Supported</td></tr><tr><td>H7: Modularity → Technical success</td><td>Supported</td></tr></table>	Table 3 Path coefficients for all the models.		Structural path	Conclusion	<hr/>		H1: Technical success → Market success	Not supported	H2a: License type → Market success	Supported at t1	H2b: License type → Technical success	Supported at t3, t4	H3a: User base → Market success	Supported	H3b: Developer base → Market success	Mixed results	H3c: Developer base → Technical success	Mixed results	H4: Language translations → Market success	Supported	H5: Responsibility assignment → Technical success	Supported	H6: Complexity → Technical success	Supported	H7: Modularity → Technical success	Supported	“complexity, modularity and responsibility assignment can be adjusted to enhance developer activity. More importantly, these can be adjusted in the current version to improve technical success and project popularity in future versions” “project administrators should select restrictive licenses, such as GPL, when starting a project to attract more contribution from developers”
Table 3 Path coefficients for all the models.																															
Structural path	Conclusion																														
<hr/>																															
H1: Technical success → Market success	Not supported																														
H2a: License type → Market success	Supported at t1																														
H2b: License type → Technical success	Supported at t3, t4																														
H3a: User base → Market success	Supported																														
H3b: Developer base → Market success	Mixed results																														
H3c: Developer base → Technical success	Mixed results																														
H4: Language translations → Market success	Supported																														
H5: Responsibility assignment → Technical success	Supported																														
H6: Complexity → Technical success	Supported																														
H7: Modularity → Technical success	Supported																														
“A theory-grounded framework of Open Source Software adoption in SMEs”  Macredie & Mijinyawa (2011) [25]  35 (Google Scholar)	European Journal of Information Systems	10 SMEs in IT sector in UK  Theoretical, and later empirical Explorative & Explanatory Qualitative	Present theory-grounded framework, based on decomposed theory of planned behavior, which allows to explore factors influencing OSS adoption (what and why). Later authors evaluate it on the 10 UK SMEs in IT sector	Human capital Financial constrains for switching	License cost-saving; lack of drivers; functionality; support community; web media; innovativeness; capital investment; internet infrastructure																										

<p>“Toward an empirical assessment of the benefits of open source software”</p> <p>Russo et al. (2003) [63]</p> <p>13 (Google Scholar)</p>	<p>International Conference on Software Engineering</p>	<p>“several” small public bodies in Italy</p> <p>Empirical (case based)</p> <p>Explorative</p>	<p>With focus on OpenOffice suit, authors report a (successful) deployment of it in 10 townships</p>	<p>Costs of transition, interoperability and integration, costs &amp; time for training employees; hostility to use/change resulting into reduced productivity of people; lack of user experience</p>	-
<p>“Perceptions on F/OSS adoption”</p> <p>Ozel et al. (2007) [64]</p> <p>5 (Google Scholar)</p>	<p>IFIP International Conference on Open Source Systems</p>	<p>Public administrators from 13 European countries</p> <p>Empirical</p> <p>Explorative</p> <p>Quantitative &amp; qualitative</p>	<p>Conduct a survey to capture perceptions of FOSS adoption. Conduct statistical tests and discuss adoption in public bodies among users familiar with FOSS concept.</p>	<p>Quality of FOSS and its user friendliness; politics; resistance to change; compatibility (docs. vs. odt)</p>	-
<p>“Seeing eye to eye? An exploratory study of free open source software users’ perceptions”</p> <p>Gwebu &amp; Wang (2010) [65]</p> <p>18 (Google Scholar)</p>	<p>Journal of systems and software</p>	<p>168 responses from FOSS community users and students completed survey in the USA</p> <p>Theoretical, later empirical</p> <p>Exploratory</p> <p>Quantitative</p>	<p>Develop a typology for classification of FOSS individual users (and their perceptions which influence adoption) into market segments. While focusing on user perceptions, use empirical survey to assess findings, concluding that different users’ groups perceive FOSS differently.</p>	<p>Perceived usefulness, ease of use, risks, compatibility; learning costs, concerns over reliable software and SW updates by vendors/community</p>	-
<p>“Managerial and technical barriers to the adoption of Open Source Software”</p> <p>Holck &amp; Pedersen (2005) [67]</p> <p>44 (Google Scholar)</p>	<p>International Conference on COTS-Based Software Systems</p>	<p>Danish Hospital</p> <p>Theoretical</p> <p>Exploratory</p>	<p>Authors “develop and discuss the hypothesis that a major barrier may be the ‘customer’s’ uncertainty and unfamiliarity with OSS vendor relationships”. Attempt to answer what are the barriers and how they affect managerial business decisions. Examine a case in Danish hospital.</p>	<p>Unfamiliarity and uncertainty with vendor/FLOSS accountability</p> <p>Existing software architecture</p> <p>Lack of reliable procurement models that includes legal, technical, corporate policy and business elements</p>	-

<p>“Identifying business barriers and enablers for the adoption of open source software”</p> <p>Holck &amp; Pedersen (2004) [66]</p> <p>20 (Google Scholar)</p>	<p>Proceedings of the 13th International Conference on Information Systems Development</p>	<p>Danish educational institution</p> <p>Theoretical Exploratory</p>	<p>Authors examine decision making challenges for managers when they are confronted with OSS adoption and try to answer why OSS is not more widespread in private and public organizations.</p>	<p>Similarly to [67]: Lack of support, compatibility with proprietary technology</p>	-
<p>“Adoption of Open Source Software by Organizations-A Framework for Kenya”</p> <p>Gichira et al. (2012) [69]</p> <p>0 (Google Scholar)</p>	<p>International Journal of Computer Applications</p>	<p>55 individuals in IT industry in Kenya</p> <p>Empirical Qualitative Exploratory</p>	<p>After identifying and organizing 24 adoption challenges using TOE framework, authors explore the extent of OSS adoption in organizations and conduct survey with interviews among employees in IT sector.</p>	<p>These were found to be relevant: Org.: costs, innovation, early-adopters Tech.: perceived reliability &amp; security &amp; scalability, functionality Env.: lack of widespread use, OSS awareness and piracy</p> <p>in addition to: fashionability, interoperability, lack of management support, quality</p>	-
<p>“Commercial adoption of open source software: an empirical study”</p> <p>Glynn et al. (2005) [19]</p> <p>89 (Google Scholar)</p>	<p>International Symposium on Empirical Software Engineering</p>	<p>111 responses from 350 different organizations</p> <p>Theoretical, later empirical Quantitative Explanatory</p>	<p>Based on innovation adoption theory, authors derive a framework consisting of environment, organizational, individual and technological factors for adoption. Then it was validated on a single case and a survey was constructed to find out that software houses and firms in communications are far head in OSS adoption.</p>	<p>These are found significant: Individual: staff resistance, perception of being undervalued &amp; deskilled if using OSS and unwilling to tolerate “teething problems” Technological: current architecture is stable &amp; coherent, Organizational: already favorable arrangements with vendors, hard to switch e.g. due to (commercial) maintenance Environmental: no other success story</p>	<p>Technological benefits outweigh costs (e.g. open source code); availability of OSS-literate personnel (e.g. from university); top management and individual (“champion”) support for OSS; benefits from community sharing; limited budget;</p>



<p>“Selection Criteria for Open Source Software Adoption in Malaysia”</p> <p>Chamili et al. (2012) [70]</p> <p>4 (Google Scholar)</p>	<p>International Journal of Advancements in Computing Technology</p>	<p>Public bodies in Malaysia</p> <p>Theoretical, later empirical</p> <p>Exploratory</p> <p>Qualitative</p>	<p>Along three dimensions (system, information and service quality) of the Delone and McLean IS success model authors suggest selection criteria (characteristics) for adoption in public organizations. A survey is conducted among IT/management users in public bodies in Malaysia.</p>	<p>System (software) quality: reliability, usability, performance efficiency, functionality</p> <p>Information (code) quality: maintainability, security</p> <p>Service (user support) quality: support &amp; documentation, reliability, community, competence &amp; credibility, communication</p>	-
<p>“Perceptions of open source versus commercial software: Is higher education still on the fence”</p> <p>Rooij (2007) [72]</p> <p>35 (Google Scholar)</p>	<p>Journal of Research on Technology in Education</p>	<p>20 individuals from higher educational institutions in US</p> <p>Exploratory</p> <p>Empirical</p> <p>Qualitative</p>	<p>Conduct 20 interviews with Chief Information and Chief Academic Officers to investigate characteristics of OSS and interest in moving to OSS technology. Moreover, ask what would they be willing to pay for open-source solution.</p>	<p>The most important are lack of support, security and quality issues</p>	<p>Avoidance of vendor price increases (cost perception); controlling own destiny; flexibility for needs;</p>
<p>“Investigating Factors Influencing the Adoption and Use of FOSS in Tanzanian Higher Learning Institutions: Towards an ITOE Framework”</p> <p>Kisanjara &amp; Tossy (2014) [71]</p> <p>0 (Google Scholar)</p>	<p>International Journal of Research In Business And Technology</p>	<p>Public and private higher educational institutions in Tanzania</p> <p>Explorative</p> <p>Empirical</p> <p>Quantitative</p>	<p>With a survey of 560 individuals, investigate factors influencing adoption and use of FOSS and propose Individual-Technological-Organizational-Environmental (ITOE) framework.</p>	<p>Individual: low confidence; lack of technical support, capacity to implement and awareness; staff resistance</p> <p>Technological: missing/bad infrastructure or lack of supporting technology</p> <p>Organization: TCO; proper IS/IT strategy</p> <p>Environmental: complexity; unfit for purpose</p>	<p>- staff awareness</p> <p>- code modifications; low costs – licensing &amp; scalability;</p> <p>trustworthiness</p> <p>-fulfillment of standards/policies</p> <p>-collaboration and knowledge sharing</p>

<p>“Adopting open-source software applications in US higher education: A cross-disciplinary review of the literature”</p> <p>Rooij (2009) [73]</p> <p>46 (Google Scholar)</p>	<p>Review of Educational Research</p>	<p>higher educational institutions in US</p> <p>Explorative Empirical Qualitative</p>	<p>Investigate key drivers of FOSS adoption from a literate review of 58 articles in the higher institution environment.</p>	<p>Costs for implementation of FOSS, with skilled IT personal and (!) users of these systems; Vendor support for Learning Management Systems; potential risks stemming from security, training, use, maintenance; Fit for the pedagogy/learning context</p>	<p>Adoption driven by desire to have social and philosophical benefits, software development benefits, security and risk management benefits, software adoption life cycle benefits, and total cost of ownership benefits</p>
<p>“Multi-level framework of open source software adoption”</p> <p>Qu et al. (2011) [74]</p> <p>24 (Google Scholar)</p>	<p>Journal of Business Research</p>	<p>NA (data from 3 sources)</p> <p>Empirical Quantitative Exploratory</p>	<p>Develop and test multi-level framework specifying country-level, firm-level and cross-level factors in firm’s OSS adoption. Firms in countries with high power distance unlikely to adopt OSS. Firms in less-developed countries, more likely.</p>	<p>Country with high-power distance; having more economic resources</p>	<p>Individualism orientation of country; countries with strong IT-competences (both are not statistically supported by data); country’s uncertainty avoidance orientation</p>
<p>“Open Source Software (OSS) Adoption Framework for Local Environment and its Comparison”</p> <p>Laila &amp; Bukhari (2010) [75]</p> <p>3 (Google Scholar)</p>	<p>Innovations in Computing Sciences and Software Engineering</p>	<p>Diverse firm with stable IT in Pakistan</p> <p>Theoretical, later empirical Explorative Qualitative &amp; Quantitative</p>	<p>First provide a framework for adoption of OSS in the Pakistani context and compare it to other methods developed by researchers from advanced countries. Then, an empirical survey was conducted with 30 responses which derived order of importance.</p>	<p>1. Intrinsic: existing stable IT infrastructure, lack of skilled staff and supporting organizations 2. External: lack of awareness, existing purchasing agreements, 3. Technological: training 4. Individual: resistance to change, brand reputation, being undervalued</p>	<p>- Governmental support, avoiding piracy, open standards - low costs/TCO, top management support, organization’s size -individual importance to many -tech. benefits, functionality, dissatisfaction with existing prop. system</p>
<p>“An exploratory framework for assessing open source software adoption”</p> <p>Miralles et al. (2006) [76]</p> <p>17 (Google Scholar)</p>	<p>Systèmes d'Information et Management</p>	<p>11 nation and multinational companies and their CIOs [including Spanish context]</p> <p>Empirical Exploratory Qualitative</p>	<p>Using configurational typology approach, 11 cases (through CIOs interviews) are analysed to present framework which tries to examine under which cases OSS adoption may unfold. Also present company groupings and their adoption cases when they might consider OSS.</p>	<p>Based on 11 interviews with CIOs, following are found to be relevant: Risk aversion to non-proven system and having a quantity of different platforms; Costs for switching; user-non-interest; organizational constrains</p>	<p>Mainly TCO, technological attributes and vendor lock-in but also reputation and network externalities; “user-community effects”</p>

<p>“Open source challenges for hospital information system (HIS) in developing countries: a pilot project in Mali”</p> <p>Bagayoko et al. (2010) [77]</p> <p>29 (Google Scholar)</p>	<p>BMC medical informatics and decision making</p>	<p>Hospital in Mali (Africa)</p> <p>Explorative Empirical Quantitative</p>	<p>A case study presents a pilot project that implements OSS hospital information system in Mali. Authors share their results using a questionnaire asking 13 users of HIS to report their experiences.</p>	<p>Security risks; cultural aspects; IT literacy; (developer) maintenance and support</p>	<p>Quality medical information; Costs of proprietary system</p>
<p>“Antecedents of open source software adoption in health care organizations: A qualitative survey of experts in Canada”</p> <p>Marsan &amp; Paré (2013) [78]</p> <p>12 (Google Scholar)</p>	<p>International journal of medical informatics</p>	<p>18 experts in Quebec (CAN) health and social service sector 10 IT suppliers in province</p> <p>Explorative Theoretical at first, later empirical Qualitative</p>	<p>Asking what factors influence OSS adoption in healthcare setting, authors through interviews develop a research model for investigating antecedents of OSS adoption.</p>	<p>Costs for switching, (in)compatibility with organizational needs, lack of expertise and IT (size of teams) people with skills required for the switch, maintenance and support, standardization (hl7), external expertise, political clarity, difficulty in finding right information</p>	<p>TCO still less than with proprietary, attractiveness of topic and public discourse</p>
<p>“Common characteristics of open source software development and applicability for drug discovery: a systematic review”</p> <p>Årdal et al. (2011) [79]</p> <p>14 (Google Scholar)</p>	<p>Health Research Policy and Systems</p>	<p>Literature study of 47 papers</p> <p>Explorative</p>	<p>Analyse existing OSS research through a systematic review of OSS characteristics and focus specifically on their applicability in the drug discovery (in all phases).</p>	<p>Barriers for adoption of OSS in drug discovery projects: Attracting participations (physical assets are necessary, financial returns); Management of volunteers (PMs, milestones, funding/salaries) and quality; Legal: IPR protection; Physical contains (knowledge sharing similar to OSS principles while labs require physical goods); Business/incentive model for scientists</p>	<p>Allows research to be quicker performed with reduced labor costs (due to community) and avoidance of duplicate effort.</p>

<p>“Free/Libre open source software in health care: a review”</p> <p>Karopka et al. (2014) [80]</p> <p>11 (Google Scholar)</p>	<p>Healthcare informatics research</p>	<p>Literature review (healthcare)</p> <p>Explorative</p>	<p>Investigate current adoption of FLOSS in healthcare setting.</p>	<p>Professional support and lack of legal liability and accountability; FLOSS governance; software documentation; “support the clinical workflows”; required in-house skills; adaptability to local needs</p>	<p>Dissatisfaction with proprietary vendors; lower acquisition costs; utilization of large community</p>
<p>“An Empirical Investigation into the Adoption of Open Source Software in Hospitals”</p> <p>Munoz-Cornejo et al. (2008) [81]</p> <p>20 (Google Scholar)</p>	<p>International Journal of Healthcare Information Systems and Informatics</p>	<p>Managers involved with IT in hospitals in US in 3 areas</p> <p>Explorative Empirical Mixed-methods using grounded perspective</p>	<p>Explore current adoption of OSS (along with factors influencing it) in hospitals and try to answer what are benefits and disadvantages of OSS in this domain. A survey (30) and interviews (5) are conducted.</p>	<p>Vendor support, complexity and required expertise of domain, lack of skilled IT personnel and fear of being de-skilled, lack of mature OSS, management support, negative user perceptions and missing OSS champion, favorable existing agreements, (perceived) lack of security &amp; quality &amp; accountability &amp; privacy protection/legislation and policy</p>	<p>Limited financial resource, functionality and user experience, top management support, pressure to upgrade IT systems, government support, reduction of bugs and development costs, no more vendor lock-in, adherence to standards</p>
<p>“Something for nothing: management rejection of open source software in Australia’s top firms”</p> <p>Goode (2005) [82]</p> <p>160 (Google Scholar)</p>	<p>Information &amp; Management</p>	<p>Australian top-publically listed firms</p> <p>Explorative Empirical Qualitative</p>	<p>Through a survey of 500 ASX firms and their CTOs &amp; CIOs (and 108 responses), investigate why they reject open-source.</p>	<p>Environmental: insufficient market acceptance, lack of policy, fashion effects Organizational: infrastructure problems, lack of resources &amp; support &amp; strategic planning User: personal rejection/fear/resistance, lack of skills System: security, complexity, relevancy to operations; system or training costs</p>	-
<p>“A Human Capital Perspective of Organizational Intention to Adopt Open Source Software”</p> <p>Li et al. (2005) [84]</p> <p>28 (Google Scholar)</p>	<p>International Conference on Information Systems 2005 Proceeding</p>	<p>Organizations in Singapore</p> <p>Empirical Exploratory Qualitative</p>	<p>Through a survey of 81 CIOs, take a human capital perspective on adoption OSS in organizations. Confirm that if firm has enough internal skilled people, adoption intention will greatly increase.</p>	<p>Human capital (accessibility and availability), switching costs, organizational/IT department size</p>	-

<p>“Foresighting FLOSS (free/libre/open source software) from a developing country perspective: The case of Turkey”</p> <p>Yildirim &amp; Ansal (2011) [53]</p> <p>12 (Google Scholar)</p>	Technovation	<p>Private and public bodies in Turkey</p> <p>Exploratory Empirical Qualitative</p>	<p>Examine strategic factors, through a technology foresight &amp; Delphi survey and panel (n=112), that affect deployment of FLOSS in Turkey. Conduct a SWOT analysis to identify strengths and weaknesses of FLOSS adoption in the country. Suggestions are developed.</p>	<p>Switching costs, training costs, sharing knowledge and availability of human skills, stability and security, immature communities</p>	<p>Lower hardware and labor costs, flexibility &amp; local needs, decreases piracy, technological independence, social and ethical benefits</p>
<p>“Policy recommendations for public administrators on free and open source software usage”</p> <p>Bouras et al. (2014) [88]</p> <p>8 (Google Scholar)</p>	Telematics and Informatics	European public organizations	Provides 25 policy recommendations/ framework on adoption of FLOSS in public organizations.	<p>Politics, policy issues, governmental support, market fairness (non-discrimination), existing architecture, procurement, enabler of interoperability, compliance</p>	<p>Costs effects (with licensing), re-usable SW, customization, data openness, redistribution rights</p>
<p>“Adopting open source software in public administration: The importance of boundary spanners and political commitment”</p> <p>van Loon &amp; Toshkov (2015) [89]</p> <p>3 (Google Scholar)</p>	Government Information Quarterly	<p>Dutch local administrations</p> <p>Explorative Empirical Quantitative</p>	<p>Attempt, with a survey among 65 Dutch municipalities, to reveal factors that explain differential success in diffusion of OSS.</p>	<p>Lack of early adopters (“boundary spanners”) and activists, political commitment, switching costs, standards</p>	<p>No license costs, flexibility</p>

Table 1 displays all included articles (phase 3) and their summaries (‘database’).

Year	Number of articles
2003	1
2004	1
2005	4

2006	2
2007	2
2008	5
2009	2
2010	6
2011	6
2012	5
2013	3
2014	4
2015	2
2016	1

Industry/Sector (unit of analysis)	Count	#
Public – education	4	[66, 72, 71, 73]
Public – healthcare/medicine	7	[15, 33, 67, 77, 78, 80, 81]
Public – government	7	[11, 63, 64, 70, 53, 88, 89]
Private – IT sector and related	10	[2, 7,27, 36, 43, 57, 25, 69, 75, 55]
Other – e.g. theoretical, projects	8	[30, 44, 62, 74, 79, 84, 65, 24]
Mixed – e.g. people, across industries,	8	[6, 51, 49, 50, 19, 76, 82, 48]