CHALMERS

Prerequisites for a Systemized Approach to Intellectual Property Rights in Open Source

Managing the Risks and Challenges around Open Source Software in the Information and Communications Technology Sector

Master of Science Thesis in the Master Degree Programme, Entrepreneurship and Business Design

KEVIN GOLESTAN

Department of Technology Management and Economics
Division of Management of Organizational Renewal and Entrepreneurship - MORE
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2014
Report No. E2014:089

REPORT NO. E2014:089

Prerequisites for a Systemized Approach to Intellectual Property Rights in Open Source

Managing the Risks and Challenges around Open Source Software in the Information and Communications Technology Sector

KEVIN GOLESTAN

Department of Technology Management and Economics
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2014

Prerequisites for a Systemized Approach to IPR in Open Source
Managing the Risks and Challenges around Open Source Software in the Information and Communications Technology Sector
KEVIN GOLESTAN

Technical report no E2014:089
Department of Technology Management and Economics
Chalmers University of Technology
SE-412 96 Gothenburg
Sweden
Telephone + 46 (0)31-772 1000

Chalmers
Gothenburg, Sweden 2014

ABSTRACT

Today, many leading ICT companies make substantial profits from patent licensing operations. Therefore, a company's licensing operations has become one of the most relevant factors determining if the company will be successful in its work with intellectual property. In a business setting that is constantly evolving, it is important for all companies engaged in licensing activities to access and identify the risks that are associated with these types of operations.

In the highly competitive environment companies in the ICT sector operate today; the effective implementation, development and use of products with Open Source code is becoming more and more relevant for success. However, many companies are faced with complex challenges that require proper evaluation strategies and management in relation to Open Source. These complex challenges are currently being highlighted in the industry as the use of Open Source increases and the revenues and importance of patents and licensing operations is becoming central in many firms.

This report presents a framework that has been created by conducting a case study which has significant implications on the management of these difficulties. The case study was performed in order to provide insight and understanding in how the members of the Open Source collaboration platform "Code Aurora Forum" benefit from their involvement and also how the platform has been designed to benefit its members the most. The case study is of most interest since it is designed and mainly administrated by Qualcomm, a company in the ICT industry known for its successful IPR & Open Source management strategy.

The report presents two alternatives for an Open Source management structure at an ICT company. The first alternative is the creation of a setup that emulates Code Aurora Forum and the second alternative is to join an existing organization that is able to fulfill the specified goal.

Keywords: Open Source, Code Aurora Forum, Intellectual Property, Software Development, ICT

ACKNOWLEDGEMENTS

This Master Thesis has been written as the last part of my education at the Intellectual Capital Management (ICM) track at the Master's program Entrepreneurship and Business Design at Chalmers School of Entrepreneurship.

I would like to thank my academic supervisor Bowman Heiden for his help with the thesis during the final months. Also, I would like to thank Ulf Petrusson and all the other teachers at the ICM program for two very interesting and rewarding years. I would also like to thank all my classmates from both the ICM track, and also from the Chalmers School of Entrepreneurship as a whole, for two really great years together.

Lastly, I would like to thank my friends and family that have been supporting me during my whole education; so a special thanks goes to Hami Golestan, Homa Bahmanpour, Ahmad Golestan, Amanda Dahlquist, Sami Bayke and the Circus.

TABLE OF CONTENTS

NOMENCLATURE

ICT	Information and Communications Technology
IP	Intellectual Property
OSS	Open Source Software
IPR	Intellectual Property Rights
CLA	Contributor License Agreement
AOSP	Android Open Source Project
KBB	Knowledge-Based Business
EPC	European Patent Convention
EPO	European Patent Office
WIPO	World Intellectual Patent Organization
FOSS	Free and Open Source Software

1 INTRODUCTION

This chapter presents the thesis and its background, explaining the purpose and states the research questions that will be answered. In the end of this chapter, the scope and outline of the thesis is set.

1.1 BACKGROUND

Today, many leading ICT companies make a great share of their profits from patent licensing operations. Therefore, a company's licensing operations has become one of the most relevant factors determining if the company will be successful in its work with IP. In a business setting that is constantly evolving, it is important for all companies engaged in licensing activities to access and identify the risks that are associated with these types of operations.

In the highly competitive environment companies in the ICT sector operate today, the effective implementation, development and use of products with Open Source code is becoming more and more relevant for success. Collaborative and community-driven software development is becoming increasingly important as it helps shorten time to market and to create software with less flaws, higher reliability and greater features.

However, many companies are faced with complex challenges that require proper evaluation strategies and management in relation to Open Source. If a company fails to properly manage their work with Open Source software (OSS), they could risk losing the proprietary rights to important software code and as a consequence of this also lose any connected licensing income.

These complex challenges are currently being highlighted in the industry as the use of Open Source increases and the revenues and importance of patents and licensing operations is becoming central in many firms.

1.2 PURPOSE

The purpose of the report is to create a framework from which ICT companies can develop a systemized approach to IPR in Open Source. As general guidelines are scarce and as many companies are struggling with the management of these issues, this report will highlight the risks and challenges and present a framework from

which these obstacles can be tackled. The framework will be created by deconstructing a case study which has significant implications on the management of these difficulties. The framework will then be the basis for the recommendations and guidelines put forth in the end of this report.

1.3 RESEARCH QUESTION

The overall research question of the report is:

How can ICT companies engage in the development and use of Open Source and avoid unintentional exposure of relating IPR?

As a company in the ICT sector, it has become virtually impossible to completely avoid involvement with Open Source in one way or another. To be able to embrace the opportunities presented by the Open Source community, companies must develop a management strategy allowing them to engage in Open Source activities without risking their ordinary IP business in an unintended negative way. As ICT companies differ in many ways, including IP licensing strategy, there is no simple general guideline that will fit all. This question aims to be answered in a way that will present different ways to engage in Open Source activities, depending on company and its IP licensing strategy.

The report will present the identified risks and challenges associated with Open Source and IPR's and end up with recommendations and guidelines for the management of Open Source in an ICT company.

In order to answer this, the overall research question has been divided into three sub-questions:

What are the risks and challenges of Open Source relating to IP licensing business?

When investigating how ICT companies engage in the development and use of Open Source, it is important to understand the risks and challenges associated with doing so. Many of these risks and challenges are not obvious and complex and could have potentially devastating results for a company's whole business if not avoided. The risks and challenges of the use of Open Source in the ICT sector will be identified and the different business and legal implications of them will be analyzed.

How can these risks and challenges be avoided?

As the risks and challenges with Open Source are identified, they will be further analyzed in order to understand what sort of management strategy that is needed to make sure that Open Source is handled in an effective way. This sub-question will be answered by looking at how successful ICT companies handle Open Source today and how they have structured their Open Source management. This will be done by looking at a case study of an Open Source Management setup which has been thoroughly designed in all aspects to avoid the identified risks and challenges.

What are the key elements to consider when developing an Open Source strategy?

Derived from the identified risks and challenges, the key elements that are important to incorporate when developing and implementing an Open Source management strategy will be thoroughly described. These key elements will be proposed solutions to the main obstacles that ICT companies come across in their work with Open Source.

1.4 SCOPE/DELIMITATIONS

As the use of Open Source and IPR differs between different industries, this research will be limited to the ICT sector. For the purpose of this report, an ICT company will be the general term for companies in the ICT sector that are involved with R&D, IP licensing, and other commercial activities of both software and hardware. With the differences that exist in the management of Open Source and IPR in different industries and companies it is not the purpose of this report to create a general framework for all but instead to focus on this type of companies.

The research will mainly focus on the business implications resulting from the contents of the most commonly used Open Source licenses. These licenses can be seen as institutional structures for creating new information and new innovations. As some legal background in needed in order to understand the clauses in the licenses, the report will present the basic legal foundation for understanding these contractual structures. However, the pure legal implications of the licenses are not in the main focus of this research. For the purpose of this report, a most commonly used license is defined as a license that is used by more than 1% of the total Open

Source projects. Also, all legal facts presented in this report will be based on Swedish law unless otherwise stated.

This research report will present a case study of collaborative Open Source management. The case study has been chosen as it displays a currently effective strategy for Open Source management in a patent strong ICT company.

The report will not separate Free and Open Source Software (FOSS) from Open Source Software (OSS) as the differences that exist in the philosophy and values behind the different definitions are not in the main interest of this research. Therefore, the term Open Source software will be used as a general term to include both FOSS and OSS.

1.5 THESIS OUTLINE

The report is constructed of seven chapters. This introduction chapter begins by introducing the reader to the report and set the frame for the research. The next chapter will present the background that is needed to understand the contents of the report and also explain the basics of software development and the two main strategies in corporate software development. The third chapter will provide a general overview of how the research was conducted. The 4th chapter of this report presents the underlying theory on top of which the research is based upon. The conducted case study will be presented in chapter 5 before the results are presented in chapter 6. The report ends by presenting the main conclusions in chapter 7.

2 BACKGROUND

In order to put the research into context, this chapter will present the necessary background needed to understand the conducted research. Also, the basic ideas with Intellectual Property (IP) and Intellectual Property rights (IPR) will be presented in relation to software.

2.1 THE KNOWLEDGE ECONOMY

In the current paradigm shift from industrial-based focus to knowledge-based focus within companies; it is often easy to rely on the existing definitions of the firm, market and competitive advantage that have existed since the industrial revolution. As we are moving towards a knowledge-based economy, the definition and activities of a firm are being altered.

The creation of a firm has historically been perceived as a way to limit risks and to create a setting in which multiple individuals can achieve benefit from working as a team. The economist Harold Demsetz defines the existence of firms as "institutions for producing goods and services because they can create conditions under which multiple individuals can integrate their specialist knowledge" and he also states that firms are formed when "the sum of what individuals can produce as a team is greater than the sum than they can produce alone" (Arnold, 2008).

The traditional resource-based view of the firm perceives the firm as a set of resources and capabilities which have the main aim of maximizing its value through the deployment of developed products and services while at the same time developing the firm's resources and capabilities for future purposes (Grant, 1996). By stating this one could say that in the industrial-based view, the firm's main activities lie in the management of its material value chain by ensuring that it has an optimum setting when for its product and factors of production such as the right geographical location, the right suppliers and distributors, the best raw material price etc. In this view, most firms take a closed and proprietary focus to ensure that it is successful in its business activities.

In the knowledge-based view, these activities are still in focus but the main focus is switched to managing an intellectual value chain where more collaboration with

different actors is involved and to manage a whole network of different layers when it comes to its products and services. The different layers could for example be illustrated as components in a technology product where each layer has its own network infrastructure involving different types of collaborations and agreements with different actors. The knowledge-based business (KBB) focuses on managing its intellectual assets in forms of know-how, relationships, inventions etc. and exploring them with the help of new business models and not only through the production of physical goods.

When it comes to the definition of the market, the shift has gone from a point where a market was mainly considered a place for trading of goods, where a seller could engage in supplying of produced goods, to in this new setting also be perceived as a network with different layers consisting of different actors and customers. The industrial-based market was in general focused on the selling of goods at a marginal cost which exceeded the cost of manufacturing or at prices based on the current supply and demand, whereas now the transactions are increasingly becoming license based and priced at a level which is according to the perceived value of the product or service, instead of the marginal cost of creation.

New business models are constantly arising as a consequence of the current transition to a knowledge-based economy and each new business model also brings a new type of market. New challenges arise with these markets as the transacted resources are in terms of knowledge. The fact that transacted knowledge is non-reversible and that its value is contextual creates difficulties in the construction of these new markets and the difficulties must be managed in order for any KBB to be successful.

The competitive advantage of a firm is often described as the activities that enable the firm to create more economic value than its competitors. This can be achieved either by a value advantage where one offers a higher perceived value at the same cost of ones rivals, or a cost advantage where one offer the same perceived value as ones rivals, but at a lower cost. In the industrial-based business this competitive advantage was determined by the closest actors operating in a similar context of differentiation and cost strategies (Heiden & Petrusson, 2009). In an industrial-based business, a competitive advantage can therefore be achieved through having good geographical

location of factories, optimization of supply chain management, negotiating better price on raw material and components etc.

In the knowledge economy, competitive advantage is instead determined by which actors that are best able to recognize the potential strategic options and manage the opportunities and threats presented by the intellectual value chain. So one could say that the focus has shifted to the intangible resources and capabilities and how they are managed and exploited in the intellectual value chain in order for the KBB to perform well in its activities.

2.2 INTELLECTUAL PROPERTY RIGHTS AND SOFTWARE

In the knowledge economy, most businesses are highly dependent on information and computer technology. As technological enhancements are progressing at a higher rate than ever, more and more computer-related inventions are being used in all fields of technology. Therefore, it is not hard to understand that the importance of software is ever increasing. As software is becoming more and more crucial for the success of many businesses, it is also important to understand how software can be protected from infringement.

Traditionally, the most common way to protect any type of software has been through copyright. In the modern society, this type of protection may fall short. The biggest difference between copyright and patents is that copyright covers the details of expression of a work and not any ideas. On the opposite, patents cover ideas and the possible use of those ideas.

2.2.1 LEGAL SITUATION

As previously stated, the purpose of this research is not to focus on the pure legal aspects of the researched area. However, in order to understand the contractual structures that are the underlying fundaments for the legal protection of software, one must look at the current legislation regarding the protection of computer programs.

In 1991, the first EU directive concerning the legal protection of computer programs was launched. The European Union saw a clear problem in the disharmonized protection of computer programs in the different member states and therefore

released this directive to the member states in order to achieve a more harmonized legislation. This harmonization was seen as an important step to ensure the further development of the computer program industry. Since software markets almost by default are international, the differences in the legislations at national level were thought to disturb a proper functioning of the markets. Hence, the incentives for the harmonization of the laws regarding computer programs were economic by their nature. (Council of the European Union, 1991)

The aim of the directive was that computer programs should be protected under the same copyright laws as literary and artistic works in every member state. Moreover, the scope of protection was intended to become uniform. The council directive in 1991 has since been replaced by a similar directive from 2009, which only includes minor changes compared to its predecessor. Since the directive from the European Union recommends that computer programs should be protected under copyright law, one must look into what incentives there are to patent software.

According to (Lemley, 1995), copyright has been considered to provide weaker protection than a patent due to certain limitations on the rights of copyrights owner to protect their works. Lemley particularly points out that a copyright owner may only claim the means used to express the idea, rather than claiming the idea itself.

In some ways, the patenting of Open Source components works similar to the creation of patent portfolios in individual firms; as they are used in the same way to create IP fences around the technology incorporated in the Open Source software. This statement has also been strengthened by the reports that suggest that pure Open Source firms are increasingly acquiring own patents. (Mann, 2005)

Many firms that are involved in the Open Source development are motivated by "value-chain" returns, i.e. they contribute at one point of the value chain while at the same time gaining profit, competency etc. at another. This corporate model is dependent on the role of patent in the core areas in which those firms are focusing their proprietary efforts on. Without the use of software patents, the Open Source community will risk losing the contributions and efforts of these corporations. (Mann, 2005)

2.2.2 COPYRIGHT

The copyright of a work arises immediately upon termination of the creative process and lasts for 70 years after the author's death. By this one could say that as soon as a work is created, it is copyrighted; and it is so without any requirements for registration. To get copyright protection, there are no novelty, quality or quantity requirements. Also, the copyright only protects the form that the work has been expressed and not the intellectual essence of the work. (Levin, 2011)

For a creation to be considered a work and be protected by copyright law, it must be of original character. A work is considered to have original character if it is made by a human being and created in such a way that it is a result of a personal, creative effort. (Levin, 2011)

Generally, copyright covers most forms of literary works and artistic works and is always conferred to a physical personal. There is however an important exemption for computer programs where the copyright is conferred to the employer, unless there is an agreement stating otherwise (Levin, 2011). The same exemption applies in USA under the principle of "works made for hire" (United States Copyright Office, 2012).

Copyright is divided in two sets of rights; moral rights and economic rights. The moral rights demand that the name of the author is stated when the work is made available to the public and they also place certain restrictions on how a work can be changed or how it can be made available to the public. The moral rights can be waived by the author, but not transferred in full (Levin, 2011).

The economic rights give the author an exclusive right to control the work. These rights give the author the exclusive rights to produce copies of the work and to make the work available to the public. The interesting aspect of the economic rights is that they can be transferred. The author can choose to transfer or license his or her rights freely to another person or entity (Levin, 2011).

Another special exemption regarding copyright and computer software concerns the exhaustion of copyright. Generally, when a work or a copy of a work has been transferred with the consent of its author within the European Economic Area, the copyright is exhausted. However, with software there is a special exemption where this does not include a right to make copies of the work. This exemption prevents the

software from being copied and transferred without the consent of its author (Levin, 2011).

2.2.3 PATENTS

In contrast to copyright, a patent must be applied for in each country where the inventor wishes protection. In order to obtain a patent in Europe, the invention must meet several criteria and be considered a patentable subject matter. The invention must be novel, capable of industrial application and also involve an inventive step (be non-obvious for a person skilled in the art) (European Patent Office, 2014).

In the United States, it has been practice to grant software patents since they were first applied for in the end of the 20th century, with the main requirement that the invention should be useful (World Intellectual Property Organization). However, under the European Patent Convention (EPC), a computer program "as such" is not a patentable invention. This means that in Europe, a software patent may only be granted if it is considered to have an inventive step, a "further technical effect". According to the European Patent Office, this "further technical effect" must meet at least one of the criteria:

1) Controls a technical process
2) Governs the operation of a technical device
3) Causes the software to solve a technical problem

By this, the conclusion that can be drawn is that it is possible to obtain software patents in Europe, although the requirements for being granted such are considerably stricter than in the United States.

2.2.4 PATENT LICENSES

A patent gives its owner an exclusive right to exploit the invention stated in the patent. However, in order for a patent to be profitable the owner must engage in profitable activities. To make profit from a patent, one can; sell the patent, license usage rights, or simply commercialize the patented invention (Hermansson, 2013).

In the ICT industry, many actors have great royalty incomes from licensing patents. Many patent owners choose to license the rights to make, use, or sell the patented

invention to other licensors. By licensing a patent, the ownership is retained while the owner will earn royalty income from any sales. Actors in the ICT industry are viewing licensing more and more as one of their core business operations. With the increasing use of computerized technology in all fields and the increasing number of patented inventions, licensing of software patents is more profitable than ever.

It is the licensor that sets the scope of the license agreement. It can be made exclusively to one company, or non-exclusively to several companies. The licensor can also use other limitations to limit the scope of the licensing agreement, such as (Hermansson, 2013):

- Field of use limitations (narrow vs. broad)
- Geographical limitations
- Royalties, minimum payments, milestones
- Sublicense rights

Many ICT companies choose to license patents to each other, so called cross-licensing. This allows them to make use of each other's patented inventions without risking being sued for infringement. This cross-licensing is a main reason to why many major ICT companies are actively filing for patents and striving to have a strong portfolio of patents.

2.3 RECENT DEVELOPMENT IN THE ICT INDUSTRY

The ICT sector is a knowledge-intensive industry where the major technical breakthroughs made in the last few decades have led to a complex and increasingly challenging environment to conduct business in. The challenging developments can be seen as just over the past ten years, the pace of technical innovation has intensified to the point where product life cycles are getting short to the extent that the marketing life cycle of many devices, such as smartphones, from their creation to their marketing and market retirement, can be down to six months.

These recent developments directly relate to the new challenges arising in the management of the ever-increasing IPR complexity. One single device such as a smartphone usually consists of parts from several different manufacturers, which mean that each smartphone is covered by patents from multiple companies. This is a

strong contributing factor to the increasing IPR complexity fueled by the pace of innovation and increasing complexity of devices.

With these short product life cycles and complex IPR situation it is challenging for companies in their work with IPR's as when a patent has been granted, or when a competitor infringes on patented technology, the development and innovation could have already gone so far that the product or technology is already obsolete.

The current situation with increasing competition in most areas with ICT combined with the increasing IPR complexity puts pressure on companies, and especially their R&D departments, to keep innovating to maintain a competitive advantage. These mentioned trends are mostly affecting ICT companies with strong patent portfolios and licensing operations, and therefore demand proper IPR strategy and management.

3 METHODOLOGY

In this chapter, the overall methodology of the research will be presented. The chapter includes a description of the different steps that were taken to conduct this research. The chapter starts by introducing the reader to an overview of the research methodology. After, the case study methodology and analysis methodology is presented before the chapter ends with the used methodology for assuring validity and reliability of the research.

3.1 RESEARCH METHODOLOGY OVERVIEW

The research has involved a literature review that was intended to provide the author with knowledge in the area and to provide background theory for the analytical parts of the thesis. The theory presented in the next chapter is the necessary theoretical parts needed to understand the context the research.

However, although many academics have written about the area, there are no clear cut academic sources that can act as a theoretical framework for the main area of the research. Most existing theory have focused primarily focused on understanding the incentives of individual actors and their contributions to the Open Source community and have not gone into discussing the pure business implications with the same approach that is discussed in this report.

Therefore, the research has involved an inductive approach where the author has inductively created own theory based on existing sources (Bryman & Bell, 2011). This theory has then been used to conduct the case study which is presented further down in the report.

As previously stated, the research aims to answer the question: "*How can ICT companies engage in the development and use of Open Source and avoid unintentional exposure of relating IPR?"* In order to do so, the research will be performed using an inductive method where data and knowledge is gathered to build a theory (Bryman & Bell, 2011).

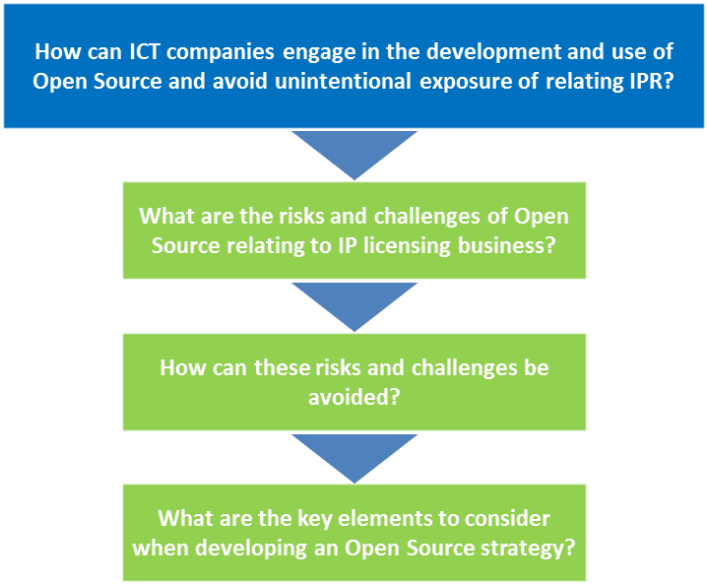

Figure 1. Illustration showing the research questions of the thesis

In order to answer the main research questions, the research has been divided into three stages based on the three sub-questions. The findings of the research performed for answering the first sub-question "*What are the risks and challenges of Open Source relating to IP licensing business?"* can be found in chapter 4.

The second sub-question "*How can these risks and challenges be avoided?"* is answered in chapter 5, Case Study, where the performed case study is presented. This sub-question is answered in the case study as the study represents a unique example of how a specific ICT company has chosen to structure it IPR & Open Source management strategy to overcome the identified risks and challenges.

The third and last sub-question is "*What are the key elements to consider when developing an Open Source strategy?"* and is answered in chapter 6 where the results of the case study analysis is broken down and used in order to create a general framework for the development of an Open Source strategy at an ICT company.

The research will be designed as a qualitative case study, which is a method, used for studying complex phenomena within their contexts. This implies that the phenomena being studied, in this case the Open Source management in Code Aurora Forum, can be subject to more thorough analyses, which implies that deeper and more far-reaching conclusions can be derived (Bryman & Bell, 2011).

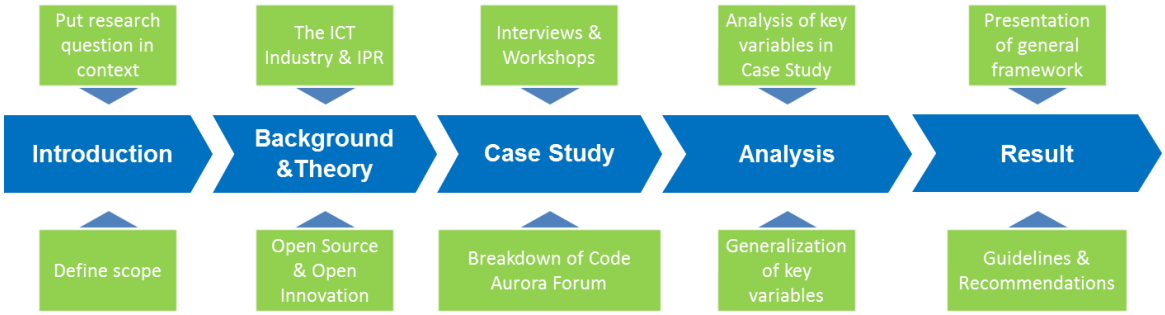

Figure 2. Research Methodology Overview

An overview of the methodology for this research project is captured in figure 5 above. The starting point is to define the scope of the project and to put the research question in the right context. The theory part of the research should include the necessary background and theory used to understand the research context (Golestan, Granberg, Hult, & Svensson, 2013).

The next step is to evaluate, through a case study, how a current Open Source management setup tackles the identified obstacles. The findings will then be broken down to be able to analyze how the setup of the case study can be generalized. In the end, the results will be presented along with a concluding discussion.

3.2 CASE STUDY

The case study was conducted as an ideographic qualitative study of a single case (Bryman & Bell, 2011). The case study was performed in order to provide insight and understanding in how the members of the Open Source collaboration platform "Code Aurora Forum" benefit from their involvement and also how the platform has been designed to benefit its members the most. The case study is of most interest since it is designed and mainly administrated by Qualcomm, a company in the ICT industry known for its successful IPR & Open Source management strategy.

The study started by looking into the history of Code Aurora Forum; when it was founded, by whom and with what purpose. The next parameters that were researched were the current members of the platform, what type of actors they are and how the members benefit from a membership in Code Aurora Forum. The operating and legal structure of the platform was then reviewed in detail in order to provide a deeper understanding of the purpose of the organization.

Additionally, interviews and workshops were held throughout the case study and served as a way to strengthen the internal validity of the performed research as the theoretical ideas developed during the research could be agreed upon or rejected among experts in the field.

3.3 ANALYSIS

The research involved a hermeneutic qualitative analysis which consists of an analysis of the risks and challenges in Open Source & IPR in the ICT industry (Bryman & Bell, 2011). The analysis was conducted using the information found in the Open Source licenses that are most commonly used today.

The idea behind the hermeneutic approach is to find the meaning of the written licenses from its authors' perspective (Bryman & Bell, 2011). This is essential due to the changes in the ICT industry that have occurred since the writing of most of the licenses; therefore, the contents of the licenses must be analyzed in the context they were written in.

Since the analysis is made with public documents, the authenticity and credibility of them are relatively unquestionable.

3.4 QUALITY EVALUATION

Due to the qualitative nature of the performed research, many of the typical measurements that are used to evaluate the quality in terms of reliability and validity are not suitable. The reliability and validity of the research is particularly at issue with quantitative research, and not qualitative research as in this case. Therefore, the quality of the research will be alternate criteria that are more suitable for evaluating qualitative research (Bryman & Bell, 2011).

The criteria that will be used to evaluate the qualitative research are trustworthiness and authenticity. These alternative criteria for evaluating research have been suggested by several authors as an alternative to reliability and validity. Trustworthiness is made up of four criteria; credibility, transferability, dependability, and conformability (Bryman & Bell, 2011).

The credibility of the research can be discussed in regards to the performed case study and analysis and whether the drawn conclusions are accurate, objective and consistent. The interpretation of the contents of the case study is one of the main research points of this study and can therefore be questioned due to the author's subjective interpretation (Bryman & Bell, 2011).

Transferability parallels external validity and is a criterion that is used to evaluate the degree to which a study can be replicated in another context or even in the same context, at another time (Bryman & Bell, 2011). The results and recommendations that are made based on the performed research have been generalized to fit companies operating in the ICT industry. As large differences exist between companies in the ICT industry, the generalized results will not fit all companies and business models, but attempts have been made to present the results and conclusions in a way that will be applicable to all companies within the scope of the research.

The dependability of the research entails that complete records are kept of the research so that auditing is possible. The dependability aspect of this research paper is believed to be quite high since most of the research has been conducted using information that was publicly available at the time of the research (Bryman & Bell, 2011).

The confirmability of the report is of high essence since the guidelines and recommendations are based on the observations and conclusions made by the author. Although a single author has conducted the research, the confirmability of the report is believed to be quite high. This is due to the fact that the findings and results from the conducted research has been reviewed with and confirmed by experts in the field (Bryman & Bell, 2011).

4 THEORY

The theory chapter presents the underlying theory that the thesis is built upon. It starts by explaining the concepts of open innovation before moving into software development and then to the developed framework "OSLO - Open Source Licenses Overview". The chapter ends with an overview of the patent granting clauses in Open Source licenses.

4.1 OPEN INNOVATION

Open Innovation is a term that was originally promoted by Henry Chesbrough, professor and professor at the Haas Business School, UC Berkeley. In broad terms, Open Innovation can be described as looking outside the company boarders in order to find new ways to innovate. By involving external parties in the R&D activities, companies will be able to identify and hopefully solve challenging problems simultaneously in order to achieve a more efficient innovation strategy. In the current economy, knowledge can be easily spread and acquired and companies do not longer have to rely entirely on their own research when searching for new opportunities for innovation. (Chesbrough, 2003)

Open Source is a great example of Open Innovation. By working with Open Source, companies will also be able to leverage on other developers knowledge and expertise. Open Source software development gives access to the source code in order to involve others in the development, a sort of collaborative effort. Open Source is an inclusive method rather than exclusive method of working as it is a way of including the users and competitors in the development, in theory gaining the benefits of their knowledge and expertise.

"Open innovation is the use of purposive inflows and outflows of knowledge to accelerate internal innovation, and expand the markets for external use of innovation, respectively. [This paradigm] assumes that firms can and should use external ideas as well as internal ideas, and internal and external paths to market, as they look to advance their technology." (Chesbrough, 2003).

When working with Open Innovation, it is important that companies have a clear R&D strategy and clear IPR management in order to gain all the benefits that come with

working with Open Innovation as an R&D model. The challenges that arise in working with Open Innovation is mainly to implement and IPR management strategy that will allow the company to acquire and exploit intellectual property while at the same time be transparent and work with collaborative methods. One of the keys to thrive in this ecosystem and to benefit from the work with Open Innovation is to have clear guidelines and platforms that help to stipulate the creation and exploitation of valuable assets (Petrusson, 2005).

4.2 Software Development

This chapter will present an overview of the two main business strategies in software development. These strategies will be reviewed and explained in detail as the analysis and results of the conducted research will be reliant on these software development strategies.

A software program generally consists of two types of code, namely source code and object code. In order to modify a software program such as updating or adding new features, a programmer must be able to access the source code. Accessibility to the source code can be seen as one of the major differences between proprietary software and Open Source software. As the name suggests, Open Source software allows the user/programmer to access the source code and make any changes whereas proprietary software does not allow access to the source code at all.

Proprietary software is usually very restrictive when it comes to what rights it gives the purchaser. In the general case with proprietary software, all rights are reserved to the author of the software, except a license to the purchaser allowing the software to be used on a computer. By restricting the rights of the user, the developers are preventing the source code from being accessed by anyone but themselves; thereby retaining the exclusive right to modify the software.

4.2.1 Open source software

Open Source software (OSS) is software that is distributed under a license that enables any person or organization to share, view and modify the source code. OSS has since its upcoming transformed from oddity to an important element for all companies working with any type of software. The history of Open Source software

development goes back to 1985 when Richard Stallman founded the non-profit organization Free Software Foundation (FSF) who actively promoted that it should be a universal freedom to create, modify and distribute any computer software. It is also FSF who are the creators of the GNU project who wrote the GNU licenses, some of the most commonly used Open Source licenses.

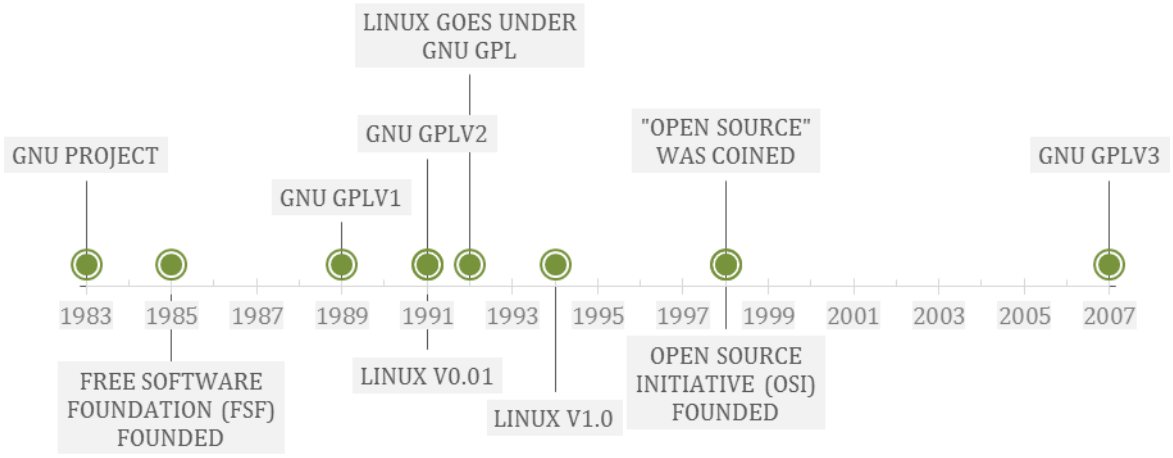

Figure 3. Brief history of Open Source

In 1992, the operating system Linux was released under GNU General Public License (GNU GPL). In the beginning of 1998, the American company Netscape, announced that it would release the source code of its popular web browser, Navigator. In reaction to this, a conference was held in Palo Alto, California where the term "Open Source" was coined. (Open Source Iniative, 2012) That same year, the Open Source Initiative (OSI) was founded in California. OSI are a community-recognized body for reviewing and approving licenses according to the Open Source Definition, which can be found in Appendix 3.

OSS is becoming more and more important to consider in all ICT companies and is becoming widely used even at companies that are known for distributing proprietary software.

4.2.2 The Open Source transition

This chapter will look into why the use and development of Open Source software has become increasingly attractive for ICT companies and also what factors that motivate these companies to engage in Open Source activities.

The Open Source software development is currently in somewhat of a transition. From previously mostly being preached by the founder of Free Software Foundation (FSF), Richard Stallman, it has now become one of the most important cornerstones in many major corporations. Richard Stallman and the FSF had the intention of developing free software, i.e. software that can be obtained, modified and used with no cost.

OSI has a more business approach to Open Source than the FSF and they believe that with the use of Open Source, the quality and many other features of the software and code will be better. This in contrast to FSF's philosophical view of Open Source, as they promote Open Source not for the potential business benefits but for the sake of user freedom, and the freedom of the code.

The most important distinction in the views of OSI and FSF is that FSF promote "free software" while OSI promote "Open Source software". Although the difference is mostly philosophical, these different words also convey different ideas. FSF promotes "free software" both in terms of free as in no cost but also free as in freedom. OSI on the other hand focus more on the openness of the code, i.e. the freedom of the developers and users.

Open Source is becoming more business oriented with many companies investing into the Open Source ecosystem and building parts of their business around the Open Source ecosystem. Therefore it is important to understand the whole ecosystem and what type of risks and challenges that exist in relation to IP for companies that are engaging in the use of Open Source.

According to (Tirole & Lerner, 2002), the interest in Open Source software development has mainly been spurred by three factors:

- The rapid diffusion of Open Source software which has led to the situation where a number of Open Source products now dominate their respective product category.
- The significant capital investments that have been made by major corporations.

- The new organization structure where the collaborative nature of Open Source software development has been praised in the industry.

The use of Open Source as an Open innovation business strategy can help release the potential of IP within the firm that is not creating value. In addition to this, corporations may benefit from situating their technologies outside the firm, while at the same time maintaining an ongoing corporate involvement (Gallagher & West, 2006).

ICT companies are also to a larger degree transforming internal development project into externally visible Open Source projects. One of the ways where this creates value is through the role of complements. A company may for example choose to open up and expose its IP for a certain technology to accelerate market adoption for that technology in order to increase the demand for products and services related to that certain technology (Gallagher & West, 2006).

According to (Gallagher & West, 2006), some of the benefits of doing this is:

- by releasing the IP related to a certain technology, the likelihood of it becoming a de facto standard increases.
- it also attracts improvements, complements and interest from external parties that make the technology even more attractive.
- the innovative technology and its complements enable the sale of related products.

Under this new paradigm, a firm's internal innovation is supplemented by systematic scanning for external knowledge, with firms maximizing the returns that accrue from both internal and external sources (Gallagher & West, 2006).

This type of open innovation business strategy requires firms to realign their innovation strategy to extend beyond the boundaries of the firm (Chesbrough, 2003). At the same time, mechanisms are needed for embracing and utilizing the value from both open and closed innovation. This type of open innovation strategy entails three main challenges (Gallagher & West, 2006):

• *maximization*. In order to maximize the returns from internal innovation, firms need to have a broad range of business activities. The most obvious one is feeding the company's product pipeline, but it is also important to work with IP licensing, patent pooling and also working with IP in order to attract attention to certain technologies.

• *incorporation*. If the firm wants to be able to gain benefits from external knowledge, it is important that it has the right tools and methods to identify and incorporate the relevant knowledge in its innovation activities.

• *motivation*. As most firms want to be able to gain benefits from external knowledge, it is important that these external sources of innovation keep flowing in. This issue is a relevant challenge for firms in the Open Source community. IP must be contributed in order for there to be a stream of external innovation, but to what extent will firms expose IP to their competitors?

4.2.3 TECHNICAL FEATURES OF OPEN SOURCE SOFTWARE

An article written by Katherine Noyes, writer at the magazine PCWorld, presents the five most interesting characteristics of Open Source for corporate entities as the following (Noyes, 2010):

1. *Quality*

   The quality aspect of OSS is something that has been debated during recent years. The proponents mean that since so many people are downloading, testing, and deploying an Open Source project, the code quality is usually on par or better than proprietary software.

2. *Security*

   Some prefer OSS as they consider it more secure and stable. The argument behind this is that since anyone can freely access and view the source code of the software, it is more likely that any bugs are spotted and reported or corrected. This is something that is also stated by "Linus' Law", named for Linus Torvalds, the creator of Linux. According to that maxim, "Given enough eyeballs, all bugs are shallow." What that means is that the more people who can see and test a set of code, the more likely any flaws will be caught and fixed quickly.

On the other hand, the opponents of Open Source software claim that since OSS has not been developed in a controlled environment nor reviewed or validated before use, it is more vulnerable to security breaches such as bugs and viruses.

3. Cost

The cost aspect is one of the most important factors for ICT companies choosing to work with OSS. As it can be obtained and used with little to no upfront costs, companies can save time, and thereby money, by adapting OSS as a type of modular innovation. By developing themselves as well as using available OSS components they avoid "reinventing the wheel". As communities of volunteers do the development of OSS project, it is possible to cut down on internal development expenses.

4. Customizability

Since the source code for OSS is freely available, it is possible to adapt the functionality to fit the company's and their customers' needs. This customizability allows the company to have more control over the software and thereby making sure that it has the right functionality for different purposes.

5. Longevity

Many companies prefer OSS due to the fact that it is considered more long-term that proprietary software. When a commercial software company goes out of business you lose all your support, bug fixes, security patches, and possibility of future versions. This is not the case with OSS. Since the source code is freely accessible, it can still be modified and updated even though the original creators have stopped working on it.

Another reason to why OSS is attractive for ICT companies is that by using and developing OSS, companies will be able to include other developers into the development of their software. Because even though companies strive to employ the most skilled developers, they are still missing out on many talented developers that want to contribute. With the use of OSS, companies will be able to include these external community developers in the development of their software.

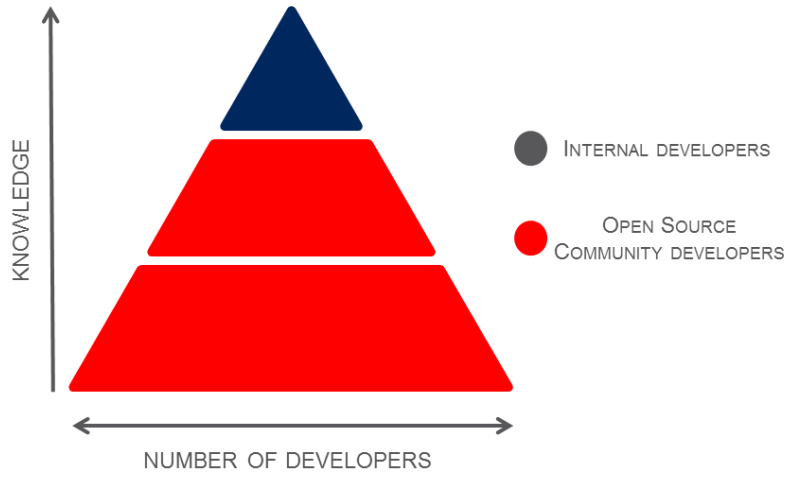

Figure 4. Illustrating how companies will include many talented developers by implementing OSS

To conclude, the acceptance of Open Source among enterprises is increasing. The high level of customizability brought by Open Source allows companies to cut development costs as it allows the IT departments to easily adapt the source code to create, change and remove software functions. This is the main driver to why Open Source is becoming increasingly popular at the enterprise level.

4.2.4 Open Source software licenses

As previously mentioned, the implementation and use of OSS is not completely risk-free when you are an ICT company involved in licensing of IPR. This chapter will present the most common Open Source licenses that will be examined and compared using the identified variables that can be found in the licenses.

When looking into the most commonly used Open Source licenses, one can identify that the clauses in the licenses demand certain things from a licensee. The licensee is either required to, free to, strongly discouraged to, or forbidden to take certain actions according to each license. In order to be able to identify the risks and challenges that exist for ICT companies when becoming a licensee to an Open Source license, one must look at each of the variables and define what implications they might have for an ICT company involved in licensing of IPR's. Each license includes different variables and therefore no license is exactly like another. To see a detailed overview of which variable that is included in each license, see Appendix 2.

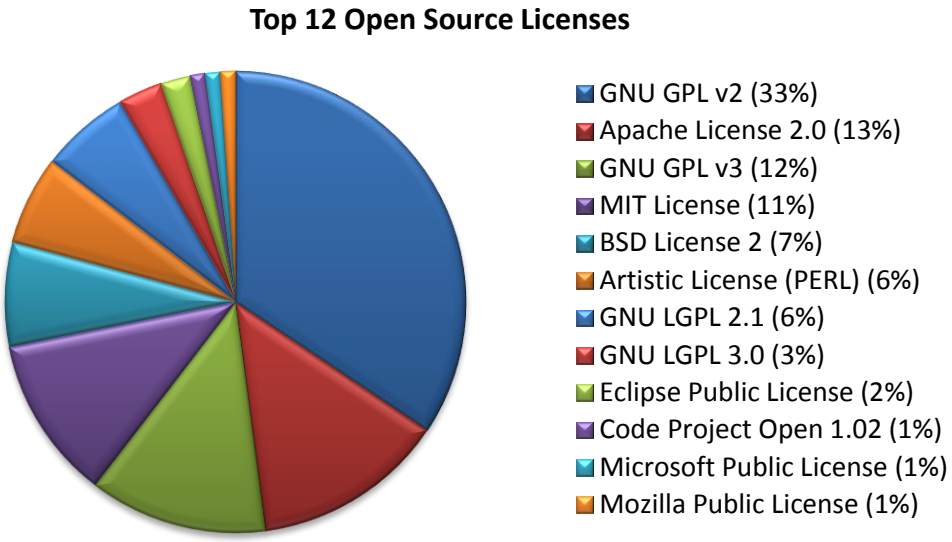

Figure 5. Overview of the Top Open Source licenses based on market adoption

4.2.5 VARIABLES

The different variables that exist in the different licenses will be presented here along with a short description of how they can be interpreted.

Depending on which variables that are included in a license, there are generally two types of licenses; permissive and copyleft/reciprocal/restrictive licenses. The permissive licenses contain much fewer restrictions on how the software can be redistributed. This means that under permissive licenses, code may be modified and put under another Open Source license or even kept as proprietary code.

Copyleft licenses on the other hand contain more restrictions. They are meant to ensure the continued "freedom" downstream and make sure that no code under a copyleft license is distributed under another license or combined with software under other licenses. However, with copyleft licenses there is a span; weak copyleft licenses have milder restrictions and allow that additions are put under another license whilst strong copyleft licenses contain stricter restrictions that does not allow additions to be put under another license.

Assuming a licensee contributes to and distributes the work, the licensee might be ***required* to**:

- **State changes**:

Any significant changes made to the code must be stated if that code is distributed. This variable could be good as the code will become better documented but it does however require more documentation time.

- **Provide license and copyright notice:**
  A copy of the license and copyright notice must be included with the distribution.
- **Disclose source:**
  This explicitly states that the source code must be made available when distributing the software. Comes down to the question whether the company should go Open Source or not. One must be willing to share if in Open Source
- **Grant patent license:**
  This variable explicitly states that any patent rights connected to the work must be granted along with the work. Requires thought as one will give licenses for all patents to code surrounding the project. If the project is not well defined, this poses a risk that it penetrates a company's core business.

Practically all mentioned top licenses have the next four variables that will be shown below. Only exception is Code Open Project (not OSI approved) which only allows commercial usage if the Open Source code is part of a larger package.

Assuming a licensee contributes to and distributes the work, the licensee might be **free to**:

- **Modify**:
  States that the code may be freely modified
- **Use commercially:**
  States that the software and any derivatives of the software may be used for commercial purposes
- **Distribute:**
  The software may be distributed.
- **Use privately:**
  The software may be modified with any distribution.

Assuming a licensee contributes to and distributes the work; the licensee might be **strongly discouraged to**:

- **Litigate:**
  Does not forbid litigation against licensees or contributors but strongly discourages it. If one licensee files a lawsuit against another licensee/contributor in relation to the work (including a cross-claim or counterclaim in a lawsuit) they would lose some or all patent licenses granted under the license or lose the entire license.

Assuming a licensee contributes to and distributes the work, the licensee might be ***forbidden* to**:

- **Hold the licensor liable**:
  This license contains a disclaimer of warranty, this does not prevent anyone to offer such a warranty should they wish to do so.
- **Sublicense:**
  This license is not possible to sublicense to 3rd parties not included in the license
- **Use of others trademark:**
  Implicitly true for all licenses but this variable explicitly states that the license my not make use of the names, logos, or trademarks of any of the contributors.

4.2.6 OSLO – OPEN SOURCE LICENSE OVERVIEW

In order to provide more clarity into the similarities and differences in the different Open Source licenses, this research has included the creation of an Open Source license overview, hereafter referred to as OSLO. As the research in the area is quite scarce, there has not been found any overview presenting the top Open Source licenses together with their included variables. Therefore, as a part of the research, OSLO has been created to act as a framework from which the licenses and their variables can be analyzed and compared by.

The top Open Source licenses and their included variables that are presented in the previous subchapters are illustrated on OSLO, which can be found in Appendix 2. To illustrate the variables more clearly, they have been divided in two different categories, “required to” and “allowed to”. As the name suggests, the variables that are in the first category are the ones that, if included in the license, the licensee is required to follow. In the same way, the rest of the variables in the “allowed to”-category are the variables that explain what each licensee is allowed to do according to each of the

licenses. Except for providing clarity in the specific contents of each license, OSLO also enables a quick overview of the similarities and differences between the licenses.

4.3 PATENT GRANTING IN TOP OPEN SOURCE LICENSES

The variables presented in this chapter are the key elements that need to be identified and understood in order to understand what risks and challenges that exist for an ICT company involved in Open Source activities. For an ICT company engaged in patent licensing, some of the mentioned variables are quite undesirable. As shown in Appendix 2, all of the most common Open Source licenses contain either an explicit or an implicit patent granting clause. This could be a problem as any contribution or distribution activities involving Open Source could mean that the company unintentionally grants a patent license to any patents surrounding the code or the project to all recipient of the code.

Another of the variables is regarding the right to litigate against other licensee or contributor without losing any rights connected to the Open Source project. As shown in Appendix 2, most of the most common Open Source licenses strongly discourage litigation against other licensees or contributors. If one licensee files a lawsuit against another licensee or contributor in relation to the work (including a cross-claim or counterclaim in a lawsuit) they would lose some or all patent licenses granted under the license or lose the entire license.

The variable regarding patent granting is the biggest challenge for ICT companies who wish to contribute to Open Source projects. It can be seen as the biggest risk for any company with a strong patent portfolio who wishes to contribute to Open Source as any minor contribution of code could mean granting of surrounding patents leading to a potential loss of income. In order to understand how a company can be involved in Open Source activities without have to unintentionally grant any patent licenses connected to the code, one must look at how the patent granting clause in each license is constructed and what type of activities that trigger the clause.

To avoid granting any unintentional patent licenses it is important to consider the specific terms under which each license requires a party to grant such licenses. Most licenses state that the “contributor” should grant such licenses, however they do differ

in their definition of the “contributor”. In table 1, an attempt to systemize this information has been made for the most common Open Source licenses.

It can be concluded that although most licenses that explicitly grants patents to downstream licenses state that it is the “contributor” that grants licenses to patents, the definition given in the different licenses for a “contributor” varies to a large degree.

The different definition of “contributor” means that there are different actions other than direct contributions of the code that will “trigger” the patent granting clause in the different licenses. In short, these are the actions of modifying the code, owning the code as of being the assigned copyright holder, and distributing the code, with or without modification of the same. It is however important to keep in mind that all these “triggers” require the code to actually be distributed for a patent license to be granted to the recipients of the code.

LICENSE	*PARTY GRANTING A PATENT LICENSE*	*PARTY DEFINED AS THE*
GNU GPLv2	*No explicit patent granting clause*	***N/A***
Apache License 2.0	*Contributor*	***Copyright holder***
GNU GPLv3	*Contributor*	***Copyright holder, Distributor***
MIT License	*No explicit patent granting clause*	***N/A***
BSD License	*No explicit patent granting clause*	***N/A***
GNU LGPL 2.1	*No explicit patent granting clause*	***N/A***
Artistic License 2.0 (PERL)	*Copyright holder*	***Copyright holder (Distribution required)***
GNU LGPL 3.0	*Contributor*	***Copyright holder, Distributor***
Eclipse Public License 1.0	*Contributor*	***Distributor***
Code Project Open 1.02	*Author*	***Distributor***
Mozilla Public License 2.0	*Contributor*	***Copyright holder, Modifier***
Microsoft Public License	*Contributor*	***Distribution, Modifier***

Table 1. Defining which parties that are considered contributors according to the top 12 Open Source licenses

There are 4 licenses in table 1 above that contain no explicit patent granting clause and at best it could be said that they contain an implicit patent granting. Since this grant is implicit rather than explicit it is hard to say what the trigger for such implicit granting would be. The "trigger" might be modification of the code, or it might be implied that the act of distributing unmodified code means that the distributor have accepted the implicit patent grant for the entirety of the code and grants patent licenses to all patents that might be required to make full use of the software distributed.

When looking into these licenses and considering when a party is forced to grant a patent license we have to consider the whole journey of the code. This journey can be summarized by four key elements; origin, modification, implementation and distribution.

Figure 6. Four key elements in Open Source management

4.3.1 Origin

The first consideration in this setting is the origin of the code. Depending on the origin of the code there are different alternatives for how one can handle Open Source code without risking to expose relating patents. If the code has been acquired, the copyright claims of the code must be examined. There are two ways of acquiring the code, either the code is acquired with a transfer of the copyright or a license is taken including a right to use the code.

If only a license to the code is taken, the licensee will not be the copyright holder and will therefore not fall as a contributor under those licenses that define the copyright holder as a contributor. If the code is entirely or partly developed internally and released under an Open Source license, the entity will be the copyright holder and therefore fall as a contributor under certain licenses (see table 1).

The origin of the code is due to these mentioned issues, of outmost importance. If the code is developed internally from first to last line, it can be put under license of choice. If however, the code is acquired, one must look at how the code has become acquired and which license the code is licensed under.

4.3.2 Modification

The modification of code is the next key parameter. This is important since, under many top Open Source licenses (see table 1), one will fall in as a contributor when code under those licenses has been modified in any way. This means that a company that wants to avoid unintentional patent granting in those Open Source licenses, must find a ways to modify and customize the code without having to grant patent licenses.

Therefore it is important to consider the modification, and which actor that will be performing it. This modification can be done internally, externally, or trough other forums.

4.3.3 IMPLEMENTATION

Implementation of the Open Source code into products, whether hardware or software products, is the next of the four key elements that must be considered in Open Source management in ICT companies. For most companies involved in work with Open Source, the most essential thing they must be able to do is to the developed code into their products. The implementation of this code can however be an issue as this could trigger the patent license clause in many licenses. However, in most licenses, the patent granting clause is triggered first when the product is distributed.

4.3.4 DISTRIBUTION

The distribution of a product which has Open Source code included is the last key element that must be considered. Distribution is one of the activities that trigger a patent granting clause in certain licenses. In some licenses it is sufficient that Open Source code is distributed, without being modified in any way, for the distributor to fall in as a contributor and therefore activate a patent granting clause. For other licenses, the distributor must have some copyright to the code to be considered a contributor and thus being forced to grant patent licenses.

5 CASE STUDY – CODE AURORA FORUM

This chapter will present a case study that will go into the details of how one collaborative entity, with constellation of different corporate entities, manage the work with Open Source and Open Source contributions. The solutions to the previously identified risks and challenges will be highlighted in order to provide a detailed insight into the construction and operations of the different settings.

This chapter will go through the history, members, structure and goal of Code Aurora Forum, how the global semiconductor company Qualcomm has restructured its operations to facilitate Code Aurora Forum, and how Code Aurora Forum distributes its contributions to the Android platform.

Code Aurora Forum (CAF) is a platform initially set up by the Qualcomm subsidiary Qualcomm Innovation Center and the Intel subsidiary Wind River. Wind River has since left the organization and another Qualcomm subsidiary Qualcomm Atheros have since joined. Apart from Motorola Mobility joining Qualcomm there have been few notable additions to the members of Code Aurora Forum since its foundation.

The purpose of Code Aurora Forum is for its members to be able to contribute improvements to various Open Source projects, primarily the Android Open Source Project (AOSP). However, the AOSP contributor license agreement (Google Inc., 2014) contains certain clauses that grants downstream recipients of code any necessary patent license to use that code; therefore, certain steps have been taken by Qualcomm to find a way around this.

In short, these steps involve reorganizing the corporate structure at Qualcomm and placing all important IP under Qualcomm Inc., setting up Code Aurora Forum as an own legal entity that Qualcomm have no formal control over, and to use Code Aurora Forum as a platform from which they can neutrally contribute code to AOSP without falling within the AOSP CLA's definition of a contributor.

5.1 BRIEF HISTORY

CAF was founded as a "California non-profit mutual benefit corporation" on April 28th 2009 (Wysk, 2014) by Qualcomm Innovation Center Inc. (wholly-owned subsidiary

to Qualcomm Inc.) and Wind River (wholly-owned subsidiary to Intel Corp.). At that time, there were 5 members beyond the two founding companies. Sometime in between mid-2011 and mid-2012, Wind River ended their membership in CAF and there is no public information about exactly when they ended their membership or why they chose to do so.

On August 29th 2012, the Linux Foundation announced that CAF and the Linux Foundation will join forces and that CAF will be a Linux Foundation Collaboration Project. Due to this, CAF did not need to exist as a separate corporation and therefore on March 7th 2013, CAF handed in a certificate of dissolution.

5.2 GOAL

According to Code Aurora Forum website (Code Aurora Forum - About Us, 2014), the goal of CAF is to:

"…provide the tested code needed to bring innovative, performance optimized, Open Source based products to market and also serves as a staging area for code that is submitted to the upstream projects. CAF welcomes the participation of projects for multiple architectures. QuIC, as one of the members of CAF, provides support for Qualcomm HW via the code it contributes to CAF. CAF also mirrors key upstream projects for use by the community."

However, when one looks more closely into the structure it becomes evident that this organization is formed to help Qualcomm separate their R&D from their IP operations, to create a sort of "safe harbor", giving more freedom to operate in the area of Open Source. This layer of security has the purpose of allowing the members of CAF to contribute to Open Source projects, without being too restricted by their IP licensing operations. In order to make this possible, the structure and legal character of the organization has been thoroughly designed to fulfill this purpose.

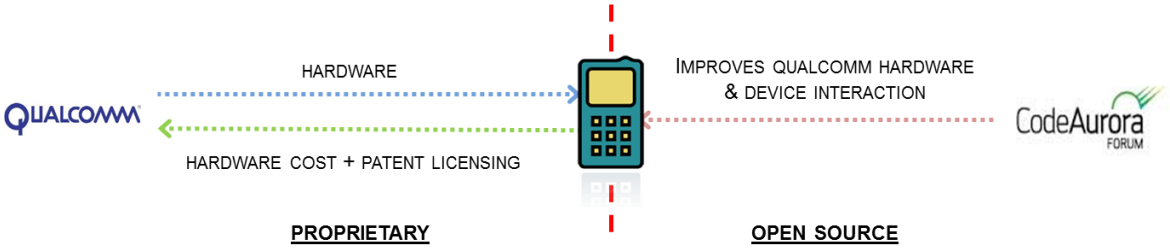

Figure 7. Illustration showing Qualcomm's goal with CAF

To summarize Qualcomm's goal with CAF it is to create a setup that enables Qualcomm to provide improvements to the interaction between their hardware and the Android Open Source Project, without risking to expose their IP portfolio.

5.3 QUALCOMM CORPORATE STRUCTURE

In order for CAF to function purposefully, Qualcomm has made a corporate reorganization in order to add another layer of security in this defensive setup. As Qualcomm wants to ensure that they do not risk the IP portfolio in this setup, they have completed a corporate reorganization effective as of October, 2012. (Qualcomm Inc., 2012) What they have done is an isolation of the parent company Qualcomm Inc. from all other operations of the company. As shown in figure 8 below, Qualcomm Inc. only consists of Qualcomm Technology Licensing (QTL) and the corporate groups at Qualcomm. QTL owns almost all the essential IPR in the company and that IPR stands for over essentially all of the total IP licensing income. (Qualcomm Inc., 2012)

Figure 8. Qualcomm's new corporate structure.

All other business operations as well as the company's engineering, research, and development functions are now operated by the subsidiary Qualcomm Technologies Inc. (QTI) and its direct and indirect subsidiaries.

The next step in the structure is the subsidiaries of QTI. Two of those are Qualcomm Innovation Center (QuIC) and Qualcomm Atheros, both members of Code Aurora Forum. They are wholly-owned subsidiaries to QTI which in turn make them wholly-owned by Qualcomm Inc.

The legal relationship between Qualcomm Inc. and its subsidiaries concerning IP is that QTI and its subsidiaries have a license to all of Qualcomm Inc. patents, but no right to sublicense or grant any rights to any patents owned by Qualcomm Inc. The following is a direct quote from Qualcomm patent policy statement (Code Aurora Forum - Legal Information, 2014):

"Neither QTI nor any of its subsidiaries has any right, power or authority to grant any licenses or other rights under or to any patents owned by QUALCOMM Incorporated."

5.4 Code Aurora Forum operating structure

The CAF has a clear goal as to allow its members to contribute to Open Source projects and at the same time shield them from being forced to grant any patent rights to Android. In order to achieve this goal, CAF has been setup with an operating structure which makes this possible. The following figure shows a simplified overview of the operations at CAF.

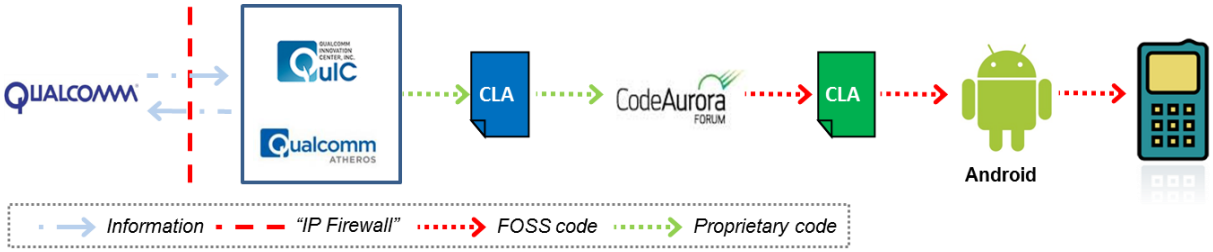

Figure 9. CAF operating and code contribution structure

As previously stated, Qualcomm Inc. owns all important IP. Qualcomm Inc. then licenses this IP to its subsidiaries, among them QuIC and Qualcomm Atheros. None of the subsidiaries have any right to sublicense any of the IP, thus creating an "IP firewall".

How CAF then operates is that QuIC, Qualcomm Atheros and other CAF members write code to be implemented into the Android Open Source Project. All copyrights to this code are assigned, through the CAF CLA, to the neutral CAF which is likely mainly controlled by Qualcomm.

CAF then submits this code to the Android Open Source Project. Because of the way CAF is set up as an independent separate entity over which Qualcomm have no formal control, no patents are licensed to either the Android Open Source Project or any downstream recipients.

The Android Open Source Project then distributes the code to original equipment manufacturer (OEM) that implements the code into their devices that use the Android operating system and Qualcomm hardware. The OEM's then sells their products to consumers with the CAF code, and consumers get access to CAF improvements without Qualcomm having granted any patents licenses to any of the recipients of the Open Source code.

This setting involves many links in the process from the actual writing of the code until it ends up in the hands of consumers. The design of these links is what enables Qualcomm's goal with CAF.

5.5 FINANCIALS

The financial data of CAF shows that CAF does not pay any salaries. The total revenues, which come from membership fees, have been around $120.000-$180.000 during the 4 active years. Each year, 40-80% of this revenue has been spent and declared as fees for services (non-employees). These fees include payments to management, legal and accounting. The rest of the expenses include smaller amounts for advertising, promotion, travel, conferences, conventions and meetings. (Citizen Audit, 2013)

In the tax forms provided by CAF to the IRS in 2013, it is observable that $550.000 of the total assets of $617.080 was transferred to the Linux Foundation with the following comment (Citizen Audit, 2013):

"The assets were transferred pursuant to a written asset transfer agreement. Upon the transfer, CAF ceased to conduct business except for the purpose of dissolving itself."

As visible in figure 9, QuIC, Qualcomm Atheros (and the other members of CAF) contribute code to CAF via a contributor license agreement (CLA). In turn, CAF contributes this code to the Android Open Source Project (AOSP) via Android's CLA.

5.6 MEMBERS

There are currently 12 corporate members in CAF, including the founding member QuIC. (Code Aurora Forum - Community, 2014) The members are generally actors working with Open Source related activities such as development and implementation of different Open Source projects. The most well-known actor, except for the Qualcomm subsidiaries Qualcomm Innovation Center and Qualcomm Atheros, is Motorola Mobility. Little information is available about Motorola Mobility's membership as there have been no press releases announcing that Motorola joined CAF, neither from Motorola nor CAF or the Linux Foundation. It can be estimated that Motorola Mobility joined sometime after CAF joined the Linux Foundation as a collaborative project in August of 2012 since they are not listed as a member in the press release that announced this event. (Linux Foundation, 2012)

Partly because of this lack of information it is hard to guess why Motorola, who at the time was owned by Google, was motivated to join CAF. It is however quite clear that Motorola did not wish to draw any attention to the fact that they had joined CAF. From the very limited information that is public about Motorola's motivation to join one can only speculate and make educated guesses on the basis of what is known.

What is known is that Google bought Motorola Mobility in August of 2011, paying US$ 12.5 billion. Google CEO Larry Page stated on Google's official blog that the acquisition of Motorola Mobility was a strategic move to strengthen Google's patent portfolio to better defend the Android platform, which at the time had been the subject of numerous patent infringement lawsuits. (Larry Page, CEO Google Inc., 2011) The fact that Google's acquisition of Motorola Mobility was primarily motivated by Motorola's patent portfolio was made clear in early 2014 when Google announced the sale of Motorola Mobility to Lenovo for US$2.91 billion. A sale in which Google kept all but 2.000 of Motorola Mobility's over 17.000 patents.

With this background it can be concluded that Motorola Mobility joined CAF after they were acquired by Google and that Google have made a substantial investment in acquiring Motorola Mobility's patent portfolio. As the portfolio was acquired for defensive purposes, Google was likely not interested in risking exhausting any of the patents through contribution to Android Open Source Project. This would make them useless for cross-licensing and for purposes of counter-suing other entities for patent infringement, thus the need to use CAF to safely release contributions into the Android Open Source Project.

Another factor might be that Motorola's two latest smartphones, the Moto G and Moto X, both use Qualcomm Snapdragon chipsets. The lack of information about Motorola Mobility joining CAF is interesting and noteworthy in and of itself. CAF is set up with an underlying purpose of avoiding the granting of patent licenses to downstream recipients. This is something that, if raised to the surface, might not go so well with Google's public image of how they handle patents. So it is understandable that they try to "fly under the radar" in this case and draw minimal attention to CAF and Motorola's participation.

5.7 UMBRELLA ORGANIZATION – LINUX FOUNDATION

The Linux Foundation is a non-profit technology consortium founded in 2007 by the merger of Open Source Development Labs and the Free Standards Group. Its mission is to foster the growth of Linux, act as a neutral spokesperson, and also to manage the "Linux" trademark. They also sponsor the work of Linus Torvalds (chief architect and project coordinator of the Linux kernel) and other key kernel developers allowing them to remain independent while working fulltime on maintaining and developing the Linux kernel. The Linux Foundation also advertises the fact that there are public relations and community relations benefits to projects being hosted by the Linux Foundation and thus being affiliated with the Linux Foundation brand provides a seal of approval from the Open Source community. (Linux Foundation)

CAF joined the Linux Foundation in late 2012. (Linux Foundation, 2012) CAF is officially a collaborative project hosted by the Linux Foundation. Other collaborative projects hosted by the Linux Foundation are projects such as MeeGo, Open Daylight and Tizen. The Linux foundation provides these collaborative projects with support such as public relations support, technical infrastructure, legal framework, and an organizational framework. These could undoubtedly be provided by Qualcomm Incorporated or any of its subsidiaries; however Qualcomm or its subsidiaries could not receive the Linux Foundation's seal of approval without joining.

5.8 CONTRIBUTOR LICENSE AGREEMENT

In the CAF setup there are two Contributors License Agreements (CLAs) to consider. First in the chain shown in figure 9 is the CAF CLA and then there is the Android CLA. The Android CLA needs to be considered first since that is what in many ways will govern how Qualcomm hade designed CAF and the CAF CLA.

For CAF or rather for Qualcomm there are a few things in the Android CLA that are worrying for their business model, first and foremost clause 3. (Google Inc., 2014) This grants all recipients of the code a patent license for those patent claims that read upon the contributions made by the contributor, and those claims that reads upon a combination of the contributions and the previously existing contributions.

For Qualcomm this would be problematic since this likely would expose a large part of their patent portfolio. Qualcomm have partly solved this problem by distancing themselves from the contributions made to Android by using CAF to submit all

contributions to Android and in the CAF CLA clause 2 assign all rights to the contributions to CAF. (Grunbaum, 2011) This means that it is the legal entity CAF that makes all the contributions to Android and that they own all the necessary copyright to the contributions but none of the patents that could possibly read on the contributions made by them.

However this situation has been somewhat addressed in the Android CLA that states in the definition of "You" (the contributor) that:

"For legal entities, the entity making a Contribution and all other entities that control, are controlled by, or are under common control with that entity are considered to be a single Contributor. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity."

If this is broken down into its individual parts it is becomes obvious why CAF is set up the way it is. First of all it can be concluded that for the purposes of the Android CLA it would likely not matter if Qualcomm's subsidiaries (in this case QuIC and Qualcomm Atheros) lack the right to sublicense Qualcomm's patents, therefore they cannot contribute directly to Android and CAF is needed as an intermediary. What then becomes important is that CAF is not controlled in any of the three ways described in the Android CLA above. It is almost certain that CAF does not fall into the second definition of "control". The first definition of control remains problematic for Qualcomm since outside observers could suspect that they have some indirect way of controlling CAF. Also the definition of beneficial ownership is somewhat unclear in the third definition. Both of these elements become more complicated since the android CLA has not defined any jurisdiction for the CLA. This means it would likely be up to any court where disputes might end up to according to their legal tradition define "indirect control" and "beneficial ownership".

5.9 Board and legal setting

By setting up Code Aurora Forum as a Mutual Benefit Non-Profit organization there are no shares to control and thus the only way to fall under the Android CLA definition of control would be to control the board of Code Aurora Forum. To avoid this

Qualcomm have ensured that they never have a majority of the representatives on the board.

The bylaws stipulate that Qualcomm Innovation Center as a founding member is entitled to appoint one board member in a board consisting of at least three board members. (Grunbaum, 2011) One can conclude that at least three board members is required in order for Qualcomm not to have control since having only two members in the board will would mean that Qualcomm will have control according to Android's CLA. This board member serves as long as that founding member is part of the organization.

Having only one member of the board means that Qualcomm will not have a voting majority, and even though it is unclear who any of the other members of the board are, it can be concluded that if there are three board members, Qualcomm Atheros will not have one of these as a representative since that would give Qualcomm formal control.

When CAF was founded Wind River was one of the founding members and thus had one representative on the board as well. Since they have left the organization, their seat on the board is now filled by another member even though it is unknown who currently holds this seat. Wind River's departure likely leaves a gap for Qualcomm since it is likely that the two founding members had interests that aligned closely with each other. So Qualcomm is either in a worse position regarding the actual control of Code Aurora Forum or have found another reliable partner among Code Aurora Forums current members.

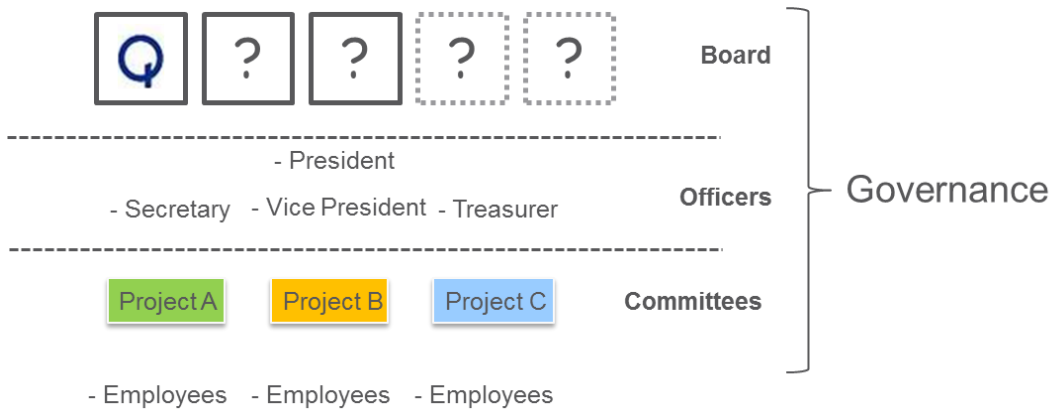

Figure 10. The governance structure of Code Aurora Forum

The governance structure of CAF is visible in the figure above. As previously mentioned, the board consists of QuIC and at least two other CAF members, but most likely not Qualcomm Atheros. Under the board there are a set of officers who manage several projects. Each project has its own focus, employees and project administrator. Currently, there are 31 active projects run by 50 project administrators. Of these 50 project administrators, 36 of them or 72% are employed by either QuIC or Qualcomm Atheros (Appendix 1). This gives a strong indication of which company that is influencing the direction of the projects at CAF and the forum as a whole.

5.10 DISTRIBUTION

All of the time spent by Qualcomm engineers writing code under CAF serves the purpose of improving the performance of Qualcomm products for consumers. So far, this case study has explained how Qualcomm have managed to have Qualcomm engineers writing this code and submitting it via CAF to the Android Open Source Project; left is the matter of distribution.

Using Android through CAF for distribution has several benefits for Qualcomm. First and most important, they avoid becoming a distributor and therefore do not count as a distributor according to those licenses that consider the distributor of the code as a contributor (see table 1). Second, they are able to utilize the existing and well established infrastructure for software updates that Android has, thus ensuring maximum spread of their improvements with minimal effort. Third, by using Android for distribution, another layer of separation is also added between Qualcomm and the finished code.

6 RESULT

This chapter will present the results from the analysis along with guidelines and recommendations. These guidelines and recommendations can serve as a general framework for the future work with Open Source and IPR in the ICT sector.

The chapter is intended to provide detailed guidelines and recommendations for how an Open Source management setup can be designed in an ICT company. The information will be based on information gathered about other Open Source in general and the conducted case study.

This document will go through two suggested alternatives for an Open Source management structure at an ICT company. These alternatives will present details on how the operating structure of the two alternatives could be designed. The contents of these alternatives is mainly focused on the design of the four key elements in Open Source management; origin, modification, implementation, and distribution.

In order to simplify the presentation of the contents of the two alternatives, a fictive ICT company named “Company A” and its wholly-owned subsidiary, “Subsidiary A”, will be used. Also, to make this setup more adapted to the business model of most product oriented ICT companies, the presented framework along with the recommendations and guidelines will be designed for a setting where Company A is directly involved in the distribution of Open Source products.

The first alternative and the one that will be explored in most detail is the creation of a setup that emulates Code Aurora Forum. The second alternative is to join an existing organization that is able to fulfill the specified goal.

6.1 STRATEGY AND GOAL

The goal of this Open Source management setup is to create a setting in which an ICT company can engage in Open Source activities without being forced to unintentionally grant any connected patent licenses.

Before starting to look into which type of Open Source management structure that is suitable, a strategy must be developed. In order to develop a strategy for how Open Source Software should be managed; a clearly formulated goal must be stated. The

purpose of the Open Source management setup presented in this report is to explain how the following goal can be achieved:

A setup where Company A can acquire, develop, modify, implement, and distribute Open Source code without risking to unintentionally expose its patent portfolio.

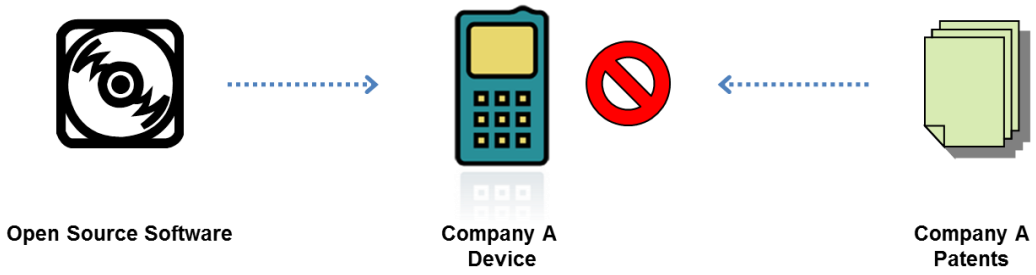

Figure 11. Illustrating the goal of the Open Source Management at Company A

This goal can be achieved by using alternative strategies and methods. An overview of the different alternatives will be presented in this chapter. In order to achieve this goal, one must first identify the risks and challenges that exist in the management of Open Source.

The identified risks and challenges with Open Source will represent different obstacles that must be overcome in order to successfully develop an effective Open Source strategy. However, with the differences that exist in Open Source licenses, the proposed solutions will differ depending on the license.

6.2 Alternative 1: Emulated Code Aurora Forum

The first alternative is to create a setup that emulates Code Aurora Forum (CAF) which was founded by Qualcomm Innovation Center (QuIC) and Intel subsidiary Wind River. CAF has an underlying goal which is to allow its members to contribute to Open Source projects without granting patent licenses. The members mainly contribute to the Android Open Source Project and the setup at CAF enables them to do so without granting any connected patent licenses to any recipients along with the contribution. If the same type of management model is to be used at another ICT company, the key elements of CAF must be reconstructed.

6.2.1 EMULATED STRUCTURE

As previously mentioned, CAF has an underlying goal with its Open Source management structure. In order to achieve this goal, CAF has been setup with an operating structure which makes this possible. When looking at how the structure of a setting similar to CAF must be constructed, one must consider if it is possible to emulate CAF and reap the same benefits.

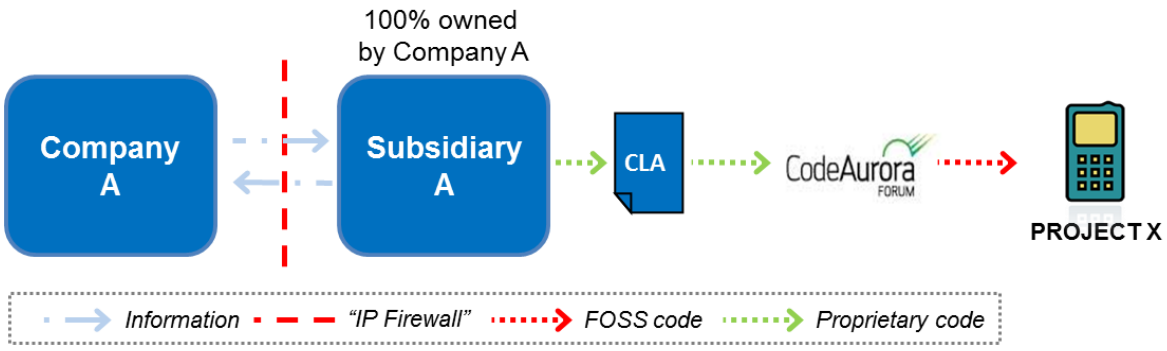

Figure 12. Emulation of the operating structure of Code Aurora Forum

The figure above shows an illustration of the operating structure of the hypothetical forum Random Open Source Alliance (ROSA) if the operating structure of Code Aurora Forum was emulated. What the figure shows is the parent company, Company A; separated with an "IP firewall" from the wholly owned subsidiary, Subsidiary A. Subsidiary A is in turn a member of ROSA along with several other companies with aligning interests. The last step in the chain, Project X, illustrates the distribution step.

Company A owns all important IP which is licensed to its subsidiary; Subsidiary A. Subsidiary A does not have any right to sublicense any of the IP, thus creating an "IP firewall". The validity and strength of this "IP firewall" can be questioned and is therefore the first critical issue that must be solved in order to achieve a functioning structure. This issue, Issue 1, will be addressed in the next chapter.

What happens next is that Subsidiary A and other members of ROSA write code to be implemented into Company A's products. All copyrights to the code are assigned, through the ROSA Contributors License Agreement (CLA), to the "neutral" ROSA. The issue at hand in this part of the process is if the code that is assigned to ROSA is licensed under an Open Source license. In such case there is a possibility that patent licenses will be part of what is transferred to ROSA and downstream recipients if the

"IP firewall" solution is not sufficient to prevent this. This issue will also be addressed in the next chapter.

ROSA implements this code into a Company A device which is then also distributed by ROSA using existing distribution channels. Because of the way ROSA is set up as an independent separate entity over which Company A have no formal control, no patents are licensed to any downstream recipients. The customers then get access to the device without Company A having granted any patent licenses. However, in the case of Company A, they are in charge of their own distribution as there are few or no existing distribution channels that fit the purpose. Therefore, the distribution aspect must be solved using other methods. This issue, Issue 2, will be addressed further down in this chapter.

6.2.2 ISSUES

The previously presented setup can be seen as a direct copy of the operating structure of Code Aurora Forum. However, as mentioned, there are two critical issues that must be solved in order for this setup to fulfill the set goal;

(1) "IP firewall" and the modification and contribution of code to ROSA and

(2) The distribution of the Open Source code.

Issue 1: "IP firewall" and the modification and contribution of code to ROSA

The separation of the parent company and the subsidiary regarding IP licensing can be seen as an extra layer of security. What Qualcomm Inc. has done is that they have given its subsidiaries rights to license but not to sub-license its patents. However, it can be questioned if this separation is enough for the parent company not to be seen as a contributor.

The copyright arises instantly when code is written or modified. Therefore, as soon as a developer at Subsidiary A would write a piece of code under an Open Source license, all rights demanded by that license will automatically be connected. This would mean that as soon as a piece of code is modified under an Open Source license that defines modification of code as a contribution; all connected patents should be granted to all recipients downstream. If Company A is to make a similar separation from Subsidiary

A as Qualcomm Inc. have made from its subsidiaries regarding patent licenses, it is still questionable whether this separation holds up legally due to the ownership structure of the parent company and the subsidiaries.

The code developed internally by Subsidiary A needs to be proprietary and not licensed under any Open Source license to avoid granting any patent licenses. In the circumstance that the new code builds on previously existing Open Source code as may often be the case, only the new code should be assigned to the newly formed industry collaboration platform ROSA as a single module. It will then be up to ROSA to integrate this new proprietary code with the old Open Source code; thus creating a new contribution that can be licensed under an Open Source license.

Figure 13. Modification and contribution of code from Subsidiary A to ROSA.

Issue 2: Distribution

In the case with CAF, there is already an existing distribution channel in which neither Qualcomm Inc. nor any of its subsidiaries need to intervene. As previously stated, this is generally not the case for most ICT companies, as the distribution is a part of their activities. This can, as previously mentioned, be an issue as certain licenses consider the distributor as a contributor; thus forcing the distributor to grant patent licenses to any recipients downstream. This issue can be worked around using one of two methods which will be further described on the next page. Both methods use a setup where the device is distributed "naked" without any Open Source software, i.e. either with only proprietary software or without any software at all.

1. *Customer Implementation*

The first method can be described as a form of self-service where Company A ships the device free from software or only including proprietary software. The customer is then able to download and implement the Open Source code themselves. The Open Source code will be made available for download by ROSA. With his workaround the Open Source code is implemented into the device without any distribution activity from Company A.

This method could also be angled as a value adding service where the customer will get devices customized specifically for their needs and with software upgrades only a few clicks away. By allowing the customers to download and implement the Open Source software developed through ROSA themselves, Company A could develop a type of "pay as you go" subscription where the customer can add or remove services online through their customer accounts on ROSA servers; services which then Company A would charge for.

2. *Subsidiary A Implementation Team*

The second method is having an implementation team employed at Subsidiary A, the subsidiary of Company A. The distribution of Open Source code would work in a way where the device is shipped from Company A, free from Open Source code, to the customer. The customer then either pays for, or receives the included service, of a Subsidiary A implementation team who head out to the customer for the implementation of Open Source code into the device. The Subsidiary A implementation team can then download the Open Source code from ROSA servers and implement them into the device. Once again, the Open Source code is implemented into the device without any distribution activity from Company A.

3. *Company A Implementation Team*

Company A could be involved in the distribution without triggering any patent clauses only if the implementation of the Open Source code is made at the site of the customer. This would mean that an implementation team from Company A will head out to the site of the customer after the hardware has been distributed "naked" from Company

A. At the site of the customer they are then free to download the software from ROSA servers and implement it on the device without triggering any patent clauses.

6.2.3 Proposed operating structure

To summarize the raised issues and solutions presented in the report, this chapter will present a proposed solution for an Open Source management setup at an ICT company. The figure presented below shows the proposed operating structure of a future Open Source management setup at Company A. What is suggested is that Company A creates an "IP firewall" as an extra layer of security where they create a wholly-owned subsidiary, Subsidiary A, which handles all Open Source related activities for the company. The relationship between Company A and Subsidiary A will be restricted regarding intellectual property as Subsidiary A will only have a right to license but not to sub-license any of Company A's patents. Subsidiary A will be involved in writing and modifying code for Company A purposes. The written code will be proprietary and will not be licensed under any Open Source license or merged with any Open Source code until the next step.

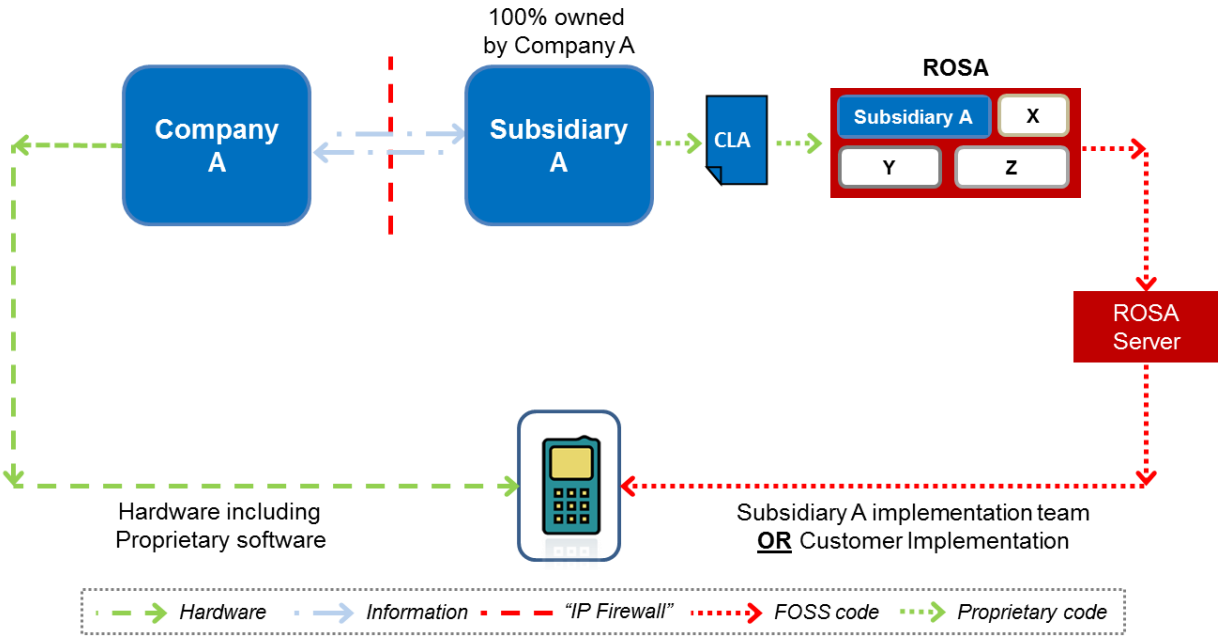

Figure 14. Proposed operating structure of an Open Source management setup using an emulated Code Aurora Forum structure.

The next step involves contribution of the proprietary code to ROSA, a separate entity. At ROSA, the contributed proprietary code will be merged with Open Source code.

Subsidiary A will be founding members of ROSA over which neither Company A nor Subsidiary A has no formal control (less than 50% of the seats in the board). All contributions to ROSA are regulated by contributor license agreements which transfer all economic rights connected to the code, from Subsidiary A and other contributing members, to ROSA.

The final step involves the distribution of the merged code at ROSA, which is now licensed under an Open Source license, to Company A devices. The hardware is distributed as usual by Company A but without software or only with proprietary software. The newly merged Open Source code is then uploaded to ROSA servers and distributed to the Company A device in either of three ways as described in detail in chapter 7.2.2;

1. *The Open Source code is downloaded and implemented by the customer.*
2. *The Open Source code is downloaded and implemented by a Subsidiary A implementation team at the site of the customer.*
3. *The Open Source code is downloaded and implemented by a Company A implementation team at the site of the customer.*6

7.2.4 Legal structure

CAF is organized with a setting in which Qualcomm have no formal control. This enables them to use CAF for the purpose of submitting code to the Android Open Source Project. If they had formal control or ownership of CAF they would be considered the same legal entity according to the Android CLA. This would mean that they, according to the terms in the Android CLA, would have to grant patent licenses downstream to patents belonging to the same legal entity as CAF.

ROSA would also have to be set up in such a way as to avoid being controlled by either Company A or any of its subsidiaries, including Subsidiary A. If one is to emulate the CAF structure one would setup a collaborative project through the Linux foundation. A number of different solutions, such as creating a separate non-profit entity, are of course possible.

If ROSA is to adopt the same legal and board setting as Code Aurora Forum, company A will have a permanent seat on the board via its member, Subsidiary A. This is good

since this would give Company A shared informal control over the steering of the operations, without legally having any formal control of the entity.

6.2.4 MEMBERS

When looking into which actors that should be members of a future ROSA, there are two main paths that could be followed. One could either attempt to "fly under the radar" and go largely unnoticed by the Open Source community or "shout from the rooftops". The two approaches will require different rosters of members.

The first approach would likely benefit from smaller members that would draw far less attention to themselves and the fact that they are collaborating with Company A on Open Source matters. The members could be companies that have existing relations to Company A and other "friendly" smaller Open Source implementation companies that have no direct link to Company A.

The biggest advantage of such an organization would be the anonymity it would operate under. Another advantage would be the control position which Company A would benefit from as the largest player in the organization. The biggest disadvantage of such an organization is the fact that it cannot go unnoticed forever, sooner or later it will grab the attention of the community at large thus negating the benefit of a keeping a low profile. The large discrepancy between the members in terms of size compared to Company A is also something that might draw attention and get the community to question the legitimacy of the organization.

As previously stated, the second solution could be an organization with more prominent members. The members could with benefit include Company A's direct competitors that have aligning interests. The benefit of such an organization would be the potential PR value of creating an open platform for collaboration with many large actors and stakeholders.

The drawbacks are that such an organization would likely not go unnoticed like CAF has done, and with that exposure the likelihood of some entity questioning the validity of the setup increases. Also by inviting other members that are not in a "dependent" position to Company A such as its subcontractors, there would be a corresponding loss of control. However by choosing members whose interests closely align with

Company A's, it is likely that such loss of control could be accepted since the predictability remains high.

6.2.5 Umbrella Organization

CAF is hosted by the Linux foundation as a collaborative project. To CAF the greatest benefit of this arrangement is access to The Linux Foundations non-profit umbrella and a "neutral" flag to wave. It is likely more a matter of branding than a necessity for them to join with The Linux Foundation. CAF could by an independent observer be seen as somewhat "fishy", this is something that is offset by the seal of approval acquired through the Linux Foundation.

The need for ROSA to be hosted under an umbrella organization and The Linux Foundation in particular should be judged from the perspective of the members list; if ROSA opts for a quiet approach it would likely draw more attention to the project if it was part of The Linux Foundation's collaborative projects, but this might be offset by the benefits of association with the Linux Foundation's brand. The larger more influential members are opted for there is less need for the positive effects of the branding of Linux Foundation. However, with many larger members there might be a larger need for a neutral party to host the project to avoid conflict of interests that might arise from one company hosting the project. Regardless of choice in members, ROSA could always benefit from the Linux Foundation's experience in hosting the similar setup CAF after which ROSA is modeled.

6.2.6 Contributor License Agreements

The Contributors License Agreement between ROSA and its members and contributors (including Subsidiary A) should clearly regulate that all copyright is assigned to ROSA. Apart from this there are few clauses apart from standard clauses that absolutely have to be in the CLA for it to meet the goal set out for the organization.

6.3 Alternative 2: Join an existing Open Source forum

The second available alternative for an Open Source management setup is to join an existing Open Source collaboration forum, such as Code Aurora Forum, instead of creating an own similar setup. However, a membership in any of those forums would in Company A's case mainly solve patent granting issues in two of the four key elements, namely origin and modification.

The implementation and distribution elements would remain unsolved as a CAF only solves all four key elements when there are existing distribution channels. As previously stated, Company A will be distributing the Open Source products directly to its customers. Therefore, by joining an existing Open Source collaboration platform, Company A would have to take charge of implementation and distribution themselves as CAF does not have any existing distribution channels for Company A's products. Due to this aspect, the patent granting issues of implementation and distribution of Open Source code remain. However, the implementation and distribution aspects can be solved using one of the three methods described in the emulated Code Aurora Forum setup in chapter 6.2.2, Issue 2. The pros and cons of creating a new forum versus joining an existing are illustrated in the table on the next page.

Founding member of ROSA	Join existing Open Source forum
+ As a founding member, Company A will be able to influence the design of ROSA to fit its needs	+ Company A can join immediately. No resources has to be allocated in to the design and implementation of a new forum
+ As a founding member, Company A has the possibility of acquiring a permanent seat in the board of ROSA	+ Pre-established structures such as public relations support, technical infrastructure, legal framework, and an organizational framework
- Requires extensive resources during the design and implementation phases of ROSA	- The current distribution channels might not fit the business model of Company A
- Not as simple as joining an existing forum with pre-established structures	- Company A will has less influence compared to a founding member

Table 2. Comparison of the pros and cons of creating a new forum versus joining an existing.

6.3.1 Proposed Operating Structure

This setup is in most parts very similar to Alternative 1 described in chapter 6.2. The main difference is that Company A, or its subsidiary for that matter, would not have to engage in the design, construction and development of the industry collaboration platform ROSA. To join an existing forum instead of engaging in the creation of a new can be more suitable.

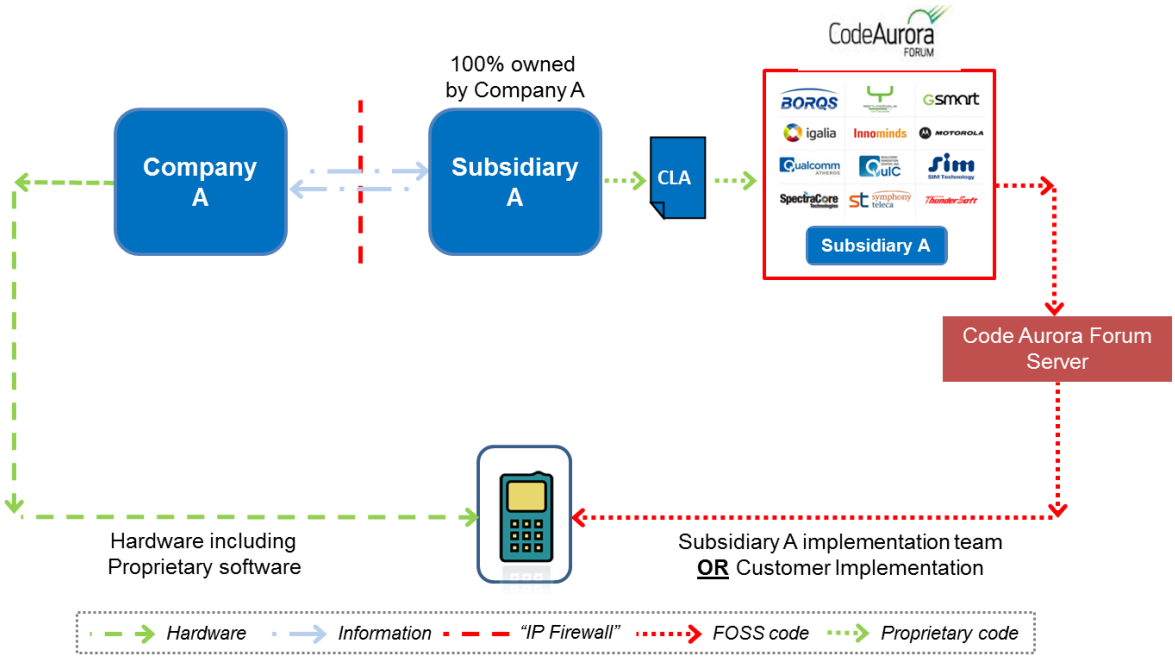

Figure 15. Proposed operating structure of an Open Source management setup using existing collaboration platforms.

Using this explained setup helps Company A with the development of Open Source code in an existing forum. The distribution and implementation aspect will be solved as their customers or a Subsidiary A implementation team can basically download the Open Source software from forum servers and implement the upgrades at the site of the customer.

7 CONCLUSION

The use of Open Source and software patents is increasing both in terms of quantity and importance. Most ICT companies engage heavily in work with Open Source and it is practically impossible to run a successful ICT company today without encountering Open Source in one way or another. As many of the major ICT companies have strong patent portfolios it can be seen as a risky move to get involved in the work with Open Source without having a proper strategy involving how to interact with Open Source and how it will affect the company's IPR activities.

The many different Open Source licenses that exist propose a complex situation as it complicates the creation of one IPR strategy for Open Source projects, independent of the license. This thesis has therefore had the aim of presenting a framework which will be applicable to all the clauses in all of the most commonly used Open Source licenses.

In the analysis section, the conclusion was that the main threat for ICT companies is the patent granting clause in the Open Source licenses. As the clause was differently constructed in the different licenses, a mapping was made of the different actions that trigger the clause. The main activities that trigger the clause were identified as acquiring, modifying, implementing and distributing Open Source code. If these activities are made with knowledge about the licenses and the meaning of their contents, a company may be able to perform these activities without having to unintentionally grant any patent licenses.

The creation of Code Aurora Forum and the restructure of their corporate structure is an excellent strategy for Qualcomm when adapting to the new challenges that have are constantly arising in the work with IPR and Open Source. As Qualcomm are one of the leading actors in licensing in the ICT sector, it is of great importance for them to be on the leading edge of IPR management and Open Source as well. What Qualcomm has achieved with their setup is a so far successful way of making sure that they will be able to contribute to Open Source projects without risking to unintentionally expose any of their patents.

By looking into the details of the newly incorporated strategy at Qualcomm, this research has aimed to identify the key elements behind the thinking. With the help of the breakdown of Code Aurora Forum, the key elements and the reasoning behind them have been identified and generalized in order to provide a general framework for all ICT companies interested in strengthening their IPR management in relation to IPR.

The created framework is based on the conclusions that have been made from the case study of Code Aurora Forum. The suggested framework demands resources to implement in a company as it demands changes throughout the whole company. The are two main alternatives presented in the results which should act as a general framework for how the IPR strategy can be setup in relation to IPR at an ICT company .

The first alternative is an emulated Code Aurora Forum where the key elements of that setup have been generalized to fit any ICT company. Even though the framework should act as a complete recommendation for construction of a IPR strategy in relation to Open Source, there are still a few issues that have to be considered and further investigated before the solution can be considered fully functional.

The second alternative that is presented is to join an existing Open Source collaboration platform where the company will be able to contribute through, and thereby avoid any unintentional patent granting. However, a membership in any of those forums would in mainly solve patent granting issues in two of the four key elements, origin and modification.

LIST OF REFERENCES

Arnold, R. A. (2008). *Economics.* South-Western College Publishing.

Bryman, A., & Bell, E. (2011). *Business Research Methods 3rd Ed.* Oxford University Press.

Chesbrough, H. W. (2003, March). Open Innovation. *Open Innovation: The New Imperative for Creating and Profiting from Technology*. United States of America: Harvard Business Review Press.

Citizen Audit. (2013). *Citizen Audit*. Retrieved June 2, 2014, from http://citizenaudit.org/270418657/

Code Aurora Forum - About Us. (2014). *Code Aurora Forum - About us*. Retrieved June 2, 2014, from https://www.codeaurora.org/about-us

Code Aurora Forum - Community. (2014). *Code Aurora Forum - Community*. Retrieved June 3, 2014, from https://www.codeaurora.org/about-us/community

Code Aurora Forum - Legal Information. (2014). *Legal Information*. Retrieved June 2, 2014, from https://www.codeaurora.org/legal-information

Council of the European Union. (1991). *Council Directive 91/250/EEC of 14 May 1991 on the legal protection of computer programs.* Brussels: Council of the European Union.

European Parliament, Council of the European Union. (2009). *Directive 2009/24/EC of the European Parliament and of the Council of 23 April 2009 on the legal protection of computer programs.* Strasbourg: European Parliament.

European Patent Office. (2014, April 8). *European Patent Office - How to apply for a European patent*. Retrieved June 3, 2014, from http://www.epo.org/applying/basics.html

Gallagher, S., & West, J. (2006, Feb 19). *Challenges of open innovation: the paradox of firm investment in open-source software.* Organization and Management, Department of. San José: Faculty Publications.

Golestan, K., Granberg, J., Hult, J., & Svensson, G. (2013). *Swedish Government Spending on R&D - A Waste of Taxpayers' Money?* Gothenburg.

Google Inc. (2014). *Corporate Contributor License Agreement*. Retrieved June 2, 2014, from http://source.android.com/source/cla-corporate.html

Grant, R. M. (1996). *Toward A Knowledge-Based Theory of the Firm.* John Wiley & Sons.

Grunbaum, D. (2011). *Open Source Development in the Telecom Industry - A study of intellectual property rights in the android open source project.* Gothenburg.

Heiden, B., & Petrusson, U. (2009). Assets, Property, and Capital in a Globalized Intellectual Value Chain. *From Assets to Profits*, 275-292.

Hermansson, C. (2013, November 27). Licensing pt 1 - Advanced Intellectual Capital Management. Gothenburg, Sweden: University of Gothenburg.

Larry Page, CEO Google Inc. (2011, August 15). *Official Google Blog*. Retrieved June 2, 2014, from http://googleblog.blogspot.com/2011/08/supercharging-android-google-to-acquire.html

Lemley, M. A. (1995). Convergence in the law of software copyright? *HIgh Technology Law Journal, 10:1*.

Levin, M. (2011). *Lärobok i immaterialrätt.* Stockholm: Nordstedts Juridik AB.

Linux Foundation. (2012, August 29). *Linux Foundation to Host Code Aurora Forum*. Retrieved June 2, 2014, from http://www.linuxfoundation.org/news-media/announcements/2012/08/linux-foundation-host-code-aurora-forum

Linux Foundation. (n.d.). *Becoming a Linux Foundation Collaborative Project.* Retrieved June 2, 2014, from http://collabprojects.linuxfoundation.org/sites/collabprojects/files/lf_collaborative_projects_brochure.pdf

Mann, R. J. (2005). *The Commercialization of Open Source Software: Do Property Rights Still Matter?* Austin, Texas: University of Texas at Austin, School of Law .

Noyes, K. (2010, November 5). *PCWorld*. Retrieved June 3, 2014, from PCWorld - 10 Reasons Open Source is Good for Business: http://www.pcworld.com/article/209891/10_reasons_open_source_is_good_for_business.html

Open Source Iniative. (2012, Sep 1). *History of the OSI*. Retrieved August 3, 2014, from Open Source Initiative: http://opensource.org/history

Petrusson, U. (2005). *Intellectual Property & Entrepreneurship.* Gothenburg: Center for Intellectual Property Studies.

Qualcomm Inc. (2012, October 1). *Qualcomm's New Corporate Structure*. Retrieved Junw 2, 2014, from http://www.qualcomm.com/about/businesses/structure

Tirole, J., & Lerner, J. (2002, June). The Simple Economics of Open Source. *Journal of Industrial Economics, 52*.

Ullberg, E. (2012, October). Copyright Basics. Gothenburg, Sweden.

United States Copyright Office. (2012, September 1). *Works Made For Hire*. Retrieved July 28, 2014, from United States Copyright Office: http://www.copyright.gov/circs/circ09.pdf

World Intellectual Property Organization. (n.d.). *WIPO - Patenting Software* . Retrieved June 3, 2014, from http://www.wipo.int/sme/en/documents/software_patents_fulltext.html

Wysk. (2014, May 27). *Wysk*. Retrieved June 2, 2014, from http://www.wysk.com/index/california/fremont/7q9nkpf/code-aurora-forum/profile

APPENDICES

Appendix 1 - Code Aurora Forum Project Administrators

Project	Name	Company
Admin, Hisense 8x30 QRD Android	Wang Keqiang	HiSense
Admin, Hisense 8x30 QRD Android	Pavel Potoplyak	HiSense
Admin, MQ	Jim Van Peursem	Motorola
Admin, MQ	Chris Fries	Motorola
Admin, MQ	Connie Zhao	Motorola
Admin, 2net Wireless Healthcare Hub	Alfred Tan	Qualcomm
Admin, APQ-Linux	Rahul Anand	Qualcomm
Admin, APQ-Linux	Gene Marsh	Qualcomm
Admin, CRIU	Christopher Covington	Qualcomm
Admin, CRIU	Alan Booker	Qualcomm
Admin, Chromium Browser for Snapdragon	Rob Walker	Qualcomm
Admin, Chromium Browser for Snapdragon	Tom Zakrajsek	Qualcomm
Admin, Femto Linux Project	Kumar Visweswara	Qualcomm
Admin, Gobi Project	Alfred Tan	Qualcomm
Admin, LLVM	Sundeep Kushwaha	Qualcomm
Admin, MSM Linux Bluetooth	Mallikarjuna GB	Qualcomm
Admin, MSM WLAN	Jeff Johnson	Qualcomm
Admin, MSM WLAN	Prakash Dhavali	Qualcomm
Admin, QRD Android	Jun Ji	Qualcomm
Admin, QSDK	Mathieu Olivari	Qualcomm
Admin, QSDK	Matthew McClintock	Qualcomm
Admin, QWRAP Project	Viren Umrigar	Qualcomm
Admin, QWRAP Project	Mao Feng	Qualcomm
Admin, Snapdragon Performance Visualizer Project	Alan Booker	Qualcomm
Admin, Ubicom	Duraid Musleh	Qualcomm
Admin, Ubicom	Bob Amstadt	Qualcomm
Admin, WiFiRouters	Vijay Kumar Peshkar	Qualcomm
Admin, WiFiRouters	Kris Muthusamy	Qualcomm
Admin, WiFiRouters	Lance Zimmerman	Qualcomm
Admin, X Window System Graphics for MSM	Michael Street	Qualcomm
Admin, Chromium Browser for Snapdragon	Sagar Shah	QuIC
Admin, Firefox OS for MSM	Michael Vines	QuIC
Admin, Hexagon	Jim Rowan	QuIC
Admin, Hexagon MiniVM	Richard Kuo	QuIC
Admin, IMath Library	Anshu Dasgupta	QuIC
Admin, Linux-MSM	Bryan Huntsman	QuIC
Admin, Linux-MSM	David Brown	QuIC
Admin, Snapdragon Developer Platform	Kiran Chitriki Rudramuni	QuIC
Admin, Vellamo Open	Enrico Ros	QuIC
Admin, Vellamo Open	Robert Nance	QuIC
Admin, WebTech	Shyama Mondal	QuIC
Admin, SIMCOM QRD Android Project	Hongqiao Zhang	SIM

Admin, SpectraCore 8064 Open Embedded Linux BSP	Ashwani Arya	SpectraCore
Admin, SpectraCore 8064 Open Embedded Linux BSP	Hiep Huynh	SpectraCore
Admin, Thundersoft	Chaingun	Thundersoft
Admin, Thundersoft	Barry Zhang	Thundersoft
Admin, Thundersoft	ThunderSoft	Thundersoft
Admin, Android for MSM	James Melvin	Unknown
Admin, Android for MSM	Ramesh Garimella	Unknown
Admin, LLVM	Anshu Das Gupta	Unknown
Admin, MSM Linux Bluetooth	Sunny Kapdi	Unknown
Admin, Open Embedded for MSM	Brian Daugherty	Unknown

Company	***Project Admins***	***Percentage***	
Qualcomm	25	50%	72%
QuIC	11	22%	
Unknown	5	10%	
Thundersoft	3	6%	
Motorola	3	6%	
HiSense	2	4%	
SIM	1	2%	
Total	**50**	**100%**	

Source: Codeauroraforum.org

Appendix 2 – Open Source Licenses Overview (OSLO)

Open Source Licenses Overview (OSLO)

License Name	Usage (2014-04-02)	License Type	*Required to*					*Allowed to*							
			State Changes	Grant Patent Licenses	Provide License	Provide Copyright Notice	Disclose Source	Modify	Use Commer-cially	Use Privately	Distribute	Hold the Licensor Liable	Use Others Trademark	Sublicense	Litigate Without Patent Retaliation
GNU GPL v2	**33%**	**Strong Copyleft**	✓	! 1	✓	✓	✓	✓	✓	✓	✓	✗	✓	✗	! 2
Apache License 2.0	**13%**	**Permissive**	✓	✓	✓	✓	✗	✓	✓	✓	✓	✗	✗	✓	✗
GNU GPL v3	**12%**	**Strong Copyleft**	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗	! 3	✗	✗
MIT License	**11%**	**Permissive**	✗	! 4	✓	✓	✗	✓	✓	✓	✓	✗	✓	✓	✓
BSD License (3-Cluase)	**7%**	**Permissive**	✗	! 5	✓	✓	✗	✓	✓	✓	✓	✗	✗	✓	✓
GNU LGPL 2.1	**6%**	**Weak Copyleft**	✓	! 6	✓	✓	! 7	✓	✓	✓	✓	✗	✓	✗	! 8
Artistic License 2.0 (PERL)	**6%**	**Hybrid**	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗	✗	✓	✗
GNU LGPL 3.0	**3%**	**Weak Copyleft**	✓	✓	✓	✓	! 9	✓	✓	✓	✓	✗	! 10	✗	✗
Eclipse Public License 1.0	**2%**	**Hybrid**	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗	✓	✓	✗
Code Project Open 1.02	**1%**	**Strong Copyleft**	✓	✓	✓	✓	✗	✓	! 11	✓	✓	✗	✓	✗	✗
Mozilla Public License 2.0	**1%**	**Weak Copyleft**	✗	✓	✓	✓	✓	✓	✓	✓	✓	✗	✗	✓	✗
Microsoft Public License	**1%**	**Hybrid**	✗	✓	✓	✓	✗	✓	✓	✓	✓	✗	✗	✓	✗
CDDL	**<1%**	**Permissive**	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗	✗	✓	✓
Erlang Public License	**<1%**	**Permissive**	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗	✗	✓	✓

License Type Definition

Permissive	**Hybrid**
Minimal restrictions on how the software/code can be redistributed and combined with other licenses	Elements of both Permissive licenses and Copyleft licenses depending on the specific situation
Weak Copyleft	**Strong Copyleft**
Milder restrictions. May not be distributed under other licences. Additions *may* be under *other* licenses	Strict restrictions. May not be distributed under other licenses. Additions *must* be under the *same* license

GNU Classpath Exemption

Please not that using the GNU Classpath Exemption will not change anything other than the classification of the license from a strong copy left to a weak copyleft, and would thous closely resamble the LGPL licensees.

Explanation of Special Provisions

1 Contains an implied patent license	**2** Contains an implied "patent retaliation"
3 Allows use but contains provision for revoking that permission	**4** Uncertainty whether a patent license is implied or not
5 Uncertainty whether a patent license is implied or not	**6** Contains an implied patent license
7 In LGPL only the source for the library (not the entire program) must be made avilable	**8** Contains an implied "patent retaliation"
9 In LGPL only the source for the library (not the entire program) must be made avilable	**10** Allows use but contains provision for revoking that permission
11 Only when distributed as part of a larger commercial software package	

✓	**YES**
✗	**NO**
!	**SPECIAL PROVISION**

Appendix 3 – The Open Sourc Definition

Introduction

Open Source doesn't just mean access to the source code. The distribution terms of open-source software must comply with the following criteria:

1. Free Redistribution

The license shall not restrict any party from selling or giving away the software as a component of an aggregate software distribution containing programs from several different sources. The license shall not require a royalty or other fee for such sale.

2. Source Code

The program must include source code, and must allow distribution in source code as well as compiled form. Where some form of a product is not distributed with source code, there must be a well-publicized means of obtaining the source code for no more than a reasonable reproduction cost preferably, downloading via the Internet without charge. The source code must be the preferred form in which a programmer would modify the program. Deliberately obfuscated source code is not allowed. Intermediate forms such as the output of a preprocessor or translator are not allowed.

3. Derived Works

The license must allow modifications and derived works, and must allow them to be distributed under the same terms as the license of the original software.

4. Integrity of the Author's Source Code

The license may restrict source-code from being distributed in modified form ***only*** if the license allows the distribution of "patch files" with the source code for the purpose of modifying the program at build time. The license must explicitly permit distribution of software built from modified source code. The license may require derived works to carry a different name or version number from the original software.

5. No Discrimination Against Persons or Groups

The license must not discriminate against any person or group of persons.

6. No Discrimination Against Fields of Endeavor

The license must not restrict anyone from making use of the program in a specific field of endeavor. For example, it may not restrict the program from being used in a business, or from being used for genetic research.

7. Distribution of License

The rights attached to the program must apply to all to whom the program is redistributed without the need for execution of an additional license by those parties.

8. License Must Not Be Specific to a Product

The rights attached to the program must not depend on the program's being part of a particular software distribution. If the program is extracted from that distribution and used or distributed within the terms of the program's license, all parties to whom the program is redistributed should have the same rights as those that are granted in conjunction with the original software distribution.

9. License Must Not Restrict Other Software

The license must not place restrictions on other software that is distributed along with the licensed software. For example, the license must not insist that all other programs distributed on the same medium must be open-source software.

10. License Must Be Technology-Neutral

No provision of the license may be predicated on any individual technology or style of interface.