THE NATIONAL UNIVERSITY OF ADVANCED LEGAL STUDIES, KOCHI

DISSERTATION SUBMITTED IN PARTIAL FULFILLMENT OF
THE REQUIREMENT FOR AWARD OF DEGREE IN

MASTER OF LAW in
INTERNATIONAL TRADE LAW

On the topic

LEGAL ISSUES AND CHALLENGES IN OPEN SOURCE LICENSING UNDER IPR LAWS IN INDIA- A CRITICAL ANALYSIS

Under the Guidance and Supervision of

Prof. Dr. MINI. S

National University of Advanced Legal Studies, Kochi

Submitted by:

VYDEHI P

REGISTRATION NO. LM0223016

BATCH: (2023-2024)

JUNE 2024

CERTIFICATE

This is to certify that Ms. **VYDEHI P (Reg. No.LM0223016)** has prepared and submitted the dissertation titled **" ISSUES AND CHALLENGES IN OPEN SOURCE LICENSING UNDER IPR LAWS IN INDIA- A CRITICAL ANALYSIS** in partial fulfilment of the requirement for the award of the Degree of Master of Laws in International Trade Law, to the National University of Advanced Legal Studies, Kochi, under my guidance and supervision. It is also affirmed that the dissertation she submitted is original, bona fide, and genuine.

Date: 20th June, 2024
Place: ERNAKULAM

Prof. Dr. MINI. S
Guide and Supervisor
NUALS, KOCHI

THE NATIONAL UNIVERSITY OF ADVANCED LEGAL STUDIES,

KALAMASSERY, KOCHI – 683 503, Kerala, India

CERTIFICATE ON PLAGIARISM CHECK

NAME OF THE CANDIDATE	VYDEHI P
TITLE OF THE DISSERTATION	ISSUES AND CHALLENGES IN OPEN SOURCE LICENSING UNDER IPR LAWS IN INDIA- A CRITICAL ANALYSIS
NAME OF THE SUPERVISOR	DR. MINI. S
SIMILAR CONTENT (%) IDENTIFIED	8%
ACCEPTABLE MAXIMUM LIMIT	10%
SOFTWARE USED	GRAMMARLY
DATE OF VERIFICATION	19th June 2024

DECLARATION

I, VYDEHI P (LM0223016), pursuing Master in International Trade Law, do hereby declare that the Dissertation titled 'Issues and Challenges in Open Source Licensing under IPR laws in India- a critical analysis', submitted for the award of L.L.M Degree in the National University of Advanced Legal Studies, Kochi, during the academic year 2023-2024, is my original, bonafide and legitimate research work, carried out under the guidance and supervision of Dr. Mini. S. This work has not formed the basis for the award of any degree, diploma, or fellowship either in this university or other similar institutions of higher learning.

Date: 20.06.2024

Place: Ernakulam

VYDEHI P

Register No. LM0223016

ACKNOWLEDGEMENT

I would like to express my deepest gratitude to everyone who supported and encouraged me throughout the course of this dissertation. Working on this dissertation had been very challenging and equally interesting. This paper nevertheless, is the result of the pertinent efforts and combination of many a people around me.

First and foremost, I am proudly grateful to my Guide, **DR. MINI. S**, whose expertise, guidance, and encouragement were invaluable throughout this study. Your insightful feedback and constant support have significantly shaped my work. I had thoroughly enjoyed every single stage of this research work.

I express my sincere thanks to the Vice Chancellor **Hon'ble Justice (Retd.) Sri. SIRI JAGAN** for his constant support. I would like to convey my gratitude to all the teachers and staffs in NUALS, KOCHI without whose help and support, it would have been impossible for me to complete my dissertation work. I would also like to thank to all Library Staffs for their timely assistance to carry out the workhand.

Finally, I owe a great debt to my family and friends for their unwavering support and patience throughout this journey. Their encouragement and understanding were vital in helping me stay focused and motivated. On a personal note, I am deeply thankful to my friend Adv. R Goutham Krishnan for his guidance and support and encouragement for timely completing my work.

VYDEHI P
REG NO: LMO223016

TABLE OF CONTENTS

SL. No.	CONTENTS	PAGE No.
1	INTRODUCTION	1-2
1.2	STATEMENT OF PROBLEM	2
1.3	OBJECTIVES OF STUDY	2
1.4	RESEARCH QUESTIONS	3
1.5	HYPOTHESIS	3
1.6	RESEARCH METHODOLOGY	3
1.7	SCOPE OF STUDY	4
1.8	LITERATURE REVIEW	4-6
1.9	CHAPTERIZATION	6-8
	OPEN SOURCE LICESNSING: EVOLUTION, TYPES	
2	DEFINITION	9

2.1	**OPEN SOURCE SOFTWARE**	**9-10**
2.2	**OPEN SOURCE SOFTWARE TYPES**	**10-20**
2.3	**OPEN SOURCE LICENSING**	**20-21**
2.3.1	**HISTORY OF OPEN SOURCE LICENSING**	**21-24**
2.3.2	**TYPES OF OPEN SOURCE LICENSING**	**24-26**
2.3.2.1	**COPYLEFT LICENSING TYPES**	**26-34**
2.3.2.2	**PERMISSIVE LICENSING TYPES**	**34-38**
	DEVELOPMENT AND PRESENT STATUS OF OPEN SOURCE LICENSING	
3	**INTRODUCTION**	**39**
3.1	**EVOLUTION OF OPEN SOURCE LICENSING**	**39-42**
3.2	**CURRENT STATUS OF OSS IN INTERNATIONAL LEVEL**	**42-49**
3.3	**CURRENT STATUS OF OSS IN INDIA**	**49-53**
	LEGAL ISSUES AND CHALLENGES FACED BY OPEN SOURCE LICENSING	

4	**INTRODUCTION**	**54**
4.1	**ENFORCEABILITY OF OSS IN CONTRACT**	**55-64**
4.2	**LEGAL ISSUES IN RELATION WITH IPR**	**64-84**
4.3	**ISSUES AND CHALLENGES OF OSS IN INDIA**	**84-98**
	CONCLUSION AND SUGGESTIONS	
5	**CONCLUSION**	**99-102**
5.1	**SUGGESTIONS**	**102-105**
6	**BIBLIOGRAPHY**	**106-112**
7	**APPENDIX**	**113**

LIST OF ABBREVIATIONS

1. **AGPL**- Affero General Public License
2. **AFL**- Academic Free License
3. **BOSS**- Bharat Operating System Solution
4. **BSD**- Berkeley Software Distribution
5. **CARP**- Common Address Redundancy Protocols
6. **CDA-** Content Delivery Application
7. **CMA**- Content Management Application
8. **CMS**- Content Management System
9. **CNET**- Computer Network
10. **DCM**- Distribution Channel Management
11. **DMS**-Document Management Systems
12. **ETL**-Extract Transform and Load
13. **EPL-** Eclipse Public License
14. **ESR**- Firefox Extended Support Release
15. **FLOSS**- Free/Libre Open Source Software
16. **FSF**- Free Software Foundation
17. **FOSS**- Free and Open Source Software
18. **GUI-** Graphical Users Interfaces
19. **GIMP**- GNU Image Manipulation Program
20. **GNOME**-GNU Network Object Model Environment
21. **GNU**-Gnu's Not Unix
22. **GNU GPL**- GNU General Public License

23. **HTTP**-Hypertext Transfer Protocol
24. **ICT**- Information and Communication Technology
25. **IDE**- Integrated Development Environment
26. **IEEE**-Institute of Electrical and Electronics Engineers
27. **IPO**- Initial Public Offering
28. **IPR**- Intellectual Property Rights
29. **JRE**- Java Runtime Environment
30. **JVM-** Java Virtual Machine
31. **KDE**-K Desktop Environment
32. **LGPL**- Lesser General Public License
33. **ML**- Machine Learning
34. **MLA**- Master License Agreement
35. **MIT**- Massachusetts Institute of Technology
36. **MPL**- Mozilla Public License
37. **MS Office**- Microsoft Office
38. **NASA**-National Aeronautics and Space Administration
39. **NROER**- National Repository of Open Educational Resources
40. **Open API**- Open Application programming Interface
41. **OS**- Operating System
42. **OSI**- Open Source Initiative
43. **OSS**- Open Source Licensing
44. **PDF**-Portable Document Format
45. **PNG**-Portable Network Graphics
46. **RDBMS**- Relational Database Management System
47. **SCTP**-Stream Control Transmission Protocol

48. **SQL**- Structured Query Language
49. **SVG**-Scalable Vector Graphics
50. **SVN**- Subversion
51. **TCO**- Total Cost of Ownership
52. **TPC-DS**- Transaction Processing Performance Council
53. **TRIPS**- Agreement on Trade- Related Aspects
54. **UETA**- Uniform Electronic Transaction Act
55. **UCITA**- Uniform Computer Information Transaction
56. **UNIX**- Uniplexed Information Computing System
57. **UNIVAC**- Universal Automatic Computer
58. **VAR**-Value-Added Reseller
59. **VCS**- Version Control System

LIST OF CASES

1. Bobbs-Merrill Co. v. Strauss et al., 210 U.S. 339 (1908)
2. Donoghue v. Stevenson, (1932) A.C. 562
3. Thornton v. Shoe Lane Parking Ltd., (1971) 1 All ER 686CA
4. Nirmala v. Tamil Nadu Electricity Board, AIR 1984 Mad. 201
5. VICOM/Computer-related invention (T 0208/84), [1987] 2 EPOR 74 (EPO Technical Board of Appeal 1986)
6. State of MP v. Asha Devi, AIR 1989 MP. 93
7. Merrill Lynch's Application, [1989] RPC 561 (UK Patents Court)
8. Gale's Application, [1991] RPC 305 (UK Patents Court)
9. Feist Publications Inc. v. Rural Telephone Service Co. Inc., 499 U.S. 40 (1991)
10. Fujitsu Ltd's Application, [1997] RPC 608 (UK Patents Court)
11. Register.com v. Verio, Inc., 356 F.3d 393 (2nd Cir. 2004)
12. CCH Canadian Ltd. v. Law Society of Upper Canada, 2004 (1) SCR 339
13. Caldera Sys. Inc. v. Int'l Bus. Machs. Corp., (D. Utah 2003) (No. 03-CV-0294)
14. Cadence Design System Inc. v. Avant! Corp., 1999 U.S. App. LEXIS 18302
15. Versata Software Inc. v. Ameriprise's Financial Inc. & Ors, May 3, 2013 (the "Texas case")
16. Microsoft Corporation v. Vishal Mehta on 7 November, 2013
17. Eastern Book Co. v. D.B. Modak, (2008) 1 SCC 1
18. Tata Consultancy Services v. State of Andhra Pradesh, 271 ITR 401
19. Campus Eai India Pvt. Ltd. vs. Neeraj Tiwari & Ors on 29 March, 2019
20. Google LLC v. Oracle America, Inc., 141 S. Ct. 1183 (2021)

CHAPTER 1

1 INTRODUCTION

Software is a crucial component of the Information and Communication Technology (ICT) industry, encompassing electronic components used in various products such as computer hardware, telecommunications, equipment, and services. It has evolved significantly over time, supporting users' everyday needs and usage. The expansion of Free and Open Source Software (FOSS) globally over the past 20 years has been an encouraging trend. FOSS has become an indispensable part of daily life, with India benefiting from these public goods. The relationship between IPR and OSS is nuanced and often disputed. Open-source software is based on collaboration and unfettered exchange, but it may restrict its distribution and utilization. The concept of open-source licensing and development approaches have been challenging and transforming software development for decades.

Open-source licensing and development approaches are built on solid, traditional legal foundations, including copyright laws in the United States and other countries. The foundation of computer software is its source code, which is produced and given via open-source software (OSS) so that everyone can examine, check, alter, enhance, and develop it. The software industry is founded upon the protection of intellectual property, with trade secrets, copyrights, patents, and trademarks being the four categories of property rights that safeguard software. Copyrights protect an idea's appearance, while patents protect software's literal expression.

The existing intellectual property laws strike a middle ground, protecting various computer program elements separately. Trade secret law is the traditional method of protecting software, while patents shield software's technical representation. However, treating source code as trade secrets has disadvantages, such as poor security, user privacy, data security risks, and reduced development incentives.

Software patents seriously threaten open-source software, as even a minor infraction against a proprietary program would result in a stop in the open-source community. The issues that the open-source community faces are made worse by the fact that open-source programs are susceptible to patent surveillance and that software patents, on the one hand, incentivize clandestine infringement of open-source goods. Thus, in light of this new development, the study on "Legal issues and challenges in Open Source Licensing- a critical Analysis concerning Intellectual Property Laws in India" becomes relevant. Moreover, it is to be noted that at

present, due to advancement and greater emphasis on open source licensing, mainly in the field of electronic and software fields, the sphere of intellectual property rights faces many modern legal issues and challenges, such as questions regarding the originality of the product due to duplication, questions regarding extend of protection guaranteed to such open source licensed products under the existing global as well as national intellectual property laws and so on. It is in the context of this present scenario that this study becomes relevant.

1.2 STATEMENT OF PROBLEM

Open-source software emerged as an ethical revolt against traditional proprietary practices by software developers. It is protected by intellectual property laws, allowing programmers to copy, share, and alter software without restriction. The Indian IT software sector has grown significantly since the 1990s, highlighting the potential for artificially encouraging the expansion of Free and Open Source in transitioning nations.

India needs a comprehensive federal policy regarding using OSS or Public Shared Services (PSs). Some governments, like West Bengal and Kerala, have adopted aggressive strategies against using and disseminating PSs for political goals. The Indian Copyright Act of 1957 provides IPR protection for computer software, but open-source applications raise new legal issues. There are legal issues and challenges in regulation of open source licensing with respect to the national and international IPR legal frame work. Open Source Licensing has both positive and negative implications on the modern society. The present IPR laws are not sufficient enough to deal with the Open Source Licensing.

1.3 OBJECTIVES OF STUDIES

- To understand the evolution of open source licensing and the relevance of regulating open source licensing.
- To evaluate how open source licensing is addressed under IPR laws.
- To analyze the legal issues and challenges currently prevailing due to the lack of legislation governing open-source licensing.
- An attempt to put forward suggestions that address the issues in open-source licensing.

1.4 RESEARCH QUESTIONS

1. What is the current legal framework (international and national) regarding open-source licensing?

2. Does the current Indian legal framework address the regulation of open-source licensing?
3. Whether is there a legal lacuna in open-source licensing regulation?
4. What are the issues caused by the lack of legislation governing open-source licensing?
5. How can the law address lacunae in open-source licensing regulations?

1.5 HYPOTHESIS

Existing copyright, patent, and trademark laws are inadequate for open source software (OSS) development and distribution. Law enforcement agencies require specialized technology knowledge to control IPR violations. The Indian Information Technology Act 2000 and Copyright Act 1957 have gaps in OSS domain protection. A more nuanced legal approach is needed for OSS-related intellectual property.

The existing intellectual property laws and enforcement mechanisms must be equipped to effectively address the challenges posed by the open-source software paradigm and the Internet. Policymakers and lawmakers must revisit and revise the legal landscape to develop a more comprehensive and practical framework for protecting intellectual property rights in open-source software. India's IPR law framework must be revised to regulate Open Source Licensing.

1.6 RESEARCH METHODOLOGY

The research methodology of this paper is based on the Doctrinal Method of Research. A comprehensive study of both the primary and secondary available data is made. A lot of articles have been referred to. Apart from the published articles accessible through the remote access of NUALS Kochi, and also relied on web sources, databases, books, and law journals. The various primary study resources include enacted legislation, judicial precedents, etc. The secondary resources include works of various eminent authors' textbooks, law journals, newspapers, etc.

1.7 SCOPE OF THE STUDY

The study focuses on the legal challenges and issues involved in Open Source Licensing about the existing IPR laws. The laws and regulations concerning the OSS are dealt with under the Indian Copyright Act of 1957. However, the existing IPR laws cannot meet the requirements for dispute resolution about open source software, which is a significant drawback.

The study aims to understand the difficulty in the present IPR laws in the country in protecting open-source software, an emerging trend in the software industry. The main obstacles governments face in implementing and creating free software are outlined in this study. The study also helps to understand the existing open-source software and licensing worldwide. This study also deals with the central open source initiatives and current developments of OSS in India. The study will briefly explain the main threats to the development of FOSS. At the same time, the final section will examine the potential measures the community might implement to prevent this perceived threat.

1.8 LITERATURE REVIEW

1. St Laurent, A. M. (2004, August 16). Understanding Open Source and Free Software Licensing (1st ed.) "O'Reilly Media

Open-source licensing and development approaches have been challenging and transforming software development for decades. Although open-source licensing is often radical, it is built on solid, traditional legal foundations, including the rights granted by copyright under the law of the United States (and elsewhere) and how basic contract principles can alter and supersede those rights. With its explanation of the various licenses and how they affect commercial and non-commercial projects, this book aims to make those choices easier. It looks at how different license styles interact with multiple project types and how licenses can serve as a glue to bring people together.

2. Mikko Valimaki, (2005)The Rise of Open Source Licensing- A challenge to the use of Intellectual Property in the Software Industry, , Turre Publishing, A Division Of Turre Legal Ltd.

This book examines how the software industry's use of intellectual property has been challenged by open source. New software licensing techniques have entered the mainstream thanks to the rise of open-source software and the Internet's explosive growth. With creative copyright licensing techniques and fearless anti-patent laws, newcomers have temporarily put incumbents in the growing software industry to the test.

Open-source licensing has dramatically impacted the software business, as large corporations have used open-source licensing models since 1998. This change has raised the possibility of intellectual property infringement and contributed to the growth of "Internet businesses" in the software industry. Since open source promotes a more thorough examination of intellectual property rights while striking a balance with commercial regulation, it also impacts intellectual property rights legislation. The ramifications for the intellectual property system are studied

when many right holders in an industry choose not to pursue their fundamental rights. This raises concerns about whether the government should fully grant intellectual property rights.

3. Andrés Guadamuz González (June 2005), Legal Challenges To Open Source Licenses , Script –ed Volume 2, Issue 2, ,

The key legal issues facing open-source software that will be covered in this paper are software patents and SCO's litigation. The validity of these issues, their potential influence on the future of open-source software, and potential legal defense against them are all covered in this paper.

4. Raymond T. Nimmer(2007) Legal Issues In Open Source And Distribution Free Software, The Law of Computer Technology (1997, 2007 Supp)

These materials have been adapted from Chapter 11 in Raymond T. Nimmer's the Law of Computer Technology (1997, 2005 Supp.). It contains the difference between Open, Free, and Proprietary Software, license terms, mapping the legal context, ownership issues in Free and Open Source, license terms relating to Patent Rights, and other legal issues.

5. K D Raju (March 2007) Is the Future of Software Development in Open Source? Propriety vs. Open Source Software: A Cross Country Analysis, Journal of Intellectual Property Rights, Vol.12, no.2, Centre for International Legal Studies, Jawaharlal Nehru University, New Delhi.

Many argue that open-source software is "free" and free from intellectual property protection clutches. This paper argues that this notion is a myth when taking into account the percentage of proprietary software usage all over the world. Government policies and decisions to support or adopt one model or the other will significantly impact the software industry. This study substantiates that government neutrality promotes innovation and development rather than supporting a particular model through a cross-country analysis of Europe, Brazil, China, and India. The study shows that more governments are enforcing open-source laws and policies.
Software.

6. Josh Lerner and Jean Tirole (Apr. 2005), The Scope Of Open-Source Licensing
   Journal of Law, Economics, & Organization, Apr. 2005, Vol. 21, No. 1, pp. 20-56, Oxford University Press

This article first investigates the factors influencing the decision to use an open-source source. First, it emphasizes how the tastes of the developer community also affect the choice, in addition to the licensor's own. The paper then uses the Source Forge database, an assemblage of close to 40,000 open-source projects, to empirically analyze the factors influencing license choice. Projects targeted at developers are less likely to have restrictive licenses than those aimed at end users. Restrictive licenses are less common for projects whose primary language

is English and intended to run on commercially operating systems. Software developed in a corporate environment and projects like games designed to appeal to consumers are likelier to have restrictive licenses. Unrestricted license projects draw in more contributors. These results mostly agree with theoretical predictions.

7. Richard Kemp (2009)Current Developments In Open Source Software
   , Kemp Little LLP, London, UK, Elsevier Ltd.

Because there is a shortage of more established case law and a variety of licensing strategies, open-source software (OSS) has grown in popularity. The basic idea of 'copyleft' in the GPL, the most popular and radical open-source software license, is in its Article 2(b) provision. Lawyers face difficulties interpreting open source software (OSS), and the growing adoption of OSS in businesses underscores the significance of OSS governance. This article aims to provide an overview of current OSS challenges from a legal standpoint. The topic of OSS is becoming broader. It has been extensively covered in writing, most of it online. Tables and footnotes provide several links to some of the publicly accessible content.

1.9 CHAPTERIZATION

CHAPTER 1: INTRODUCTION

Chapter 1 contains the introductory part of the study. The chapter includes a statement of the problem, the study's objective, the research methodology used, and the research questions. The chapter explains the scope of the study and contains a specific review of the literature used for the study.

CHAPTER 2: OPEN SOURCE LICESNSING: EVOLUTION, TYPES

This chapter includes basic information about open-source licensing. Many intellectual property rights are present in today's society. Patents, copyrights, trademarks, and so on. All these rights provide maximum protection to the creator of the product or work they cover. The expression ‘ultimate protection’ means that these rights preserve the originality of the works, protect them from any duplication, and thereby recognize the creator's claim over the product or result. Thus, all these intellectual property rights can be called closed-ended rights. Copyright covers an extensive area of all these, from artistic works like literature to scientific works like software. However, with the present-day developments in the field of IPRs, particularly concerning software protection, relatively a new system of protection via licensing,

known as the Open Source Licensing system, is gaining international acceptance. This chapter explains various open-source software used worldwide today and their features. Further, the chapter gives ideas about open-source licensing and its evolution in national and international regimes.

CHAPTER 3: DEVELOPMENT AND PRESENT STATUS OF OPEN SOURCE LICENSING

Currently, two main types of software are based on their source code: closed and open. In previous chapters, it established the provenance of open-source software, explored the various types of open-source software and why licensing open-source software, and looked at different licenses in place for open-source software that exist today. The critical difference between one category and another is that the former is where developers have made software tamper-proof and duplication-resistant. In the latter, users can duplicate and change the software itself. The main problem with open-source software is that it needs more laws to protect it. As public disclosure does not protect open software, all currently known for their accommodation is Permission from the authors of the initial version, which is again of limited protective force. This section discusses the worldwide status of open-source software at the national level concerning the existing international and national IPR laws.

CHAPTER 4: CHALLENGES FACED BY OPEN SOURCE LICENSING

This chapter, briefly discuss the legal issues and challenges faced by OSS and the licensing of the same in India. Open Source Software (OSS) has changed technology by providing transparent, affordable, cooperative solutions. However, it also presents many legal difficulties, especially intellectual property rights (IPR). This chapter examines the legal nuances surrounding open-source software (OSS) and how various jurisdictions handle related concerns. It looks at global institutions' role in promoting a unified legal framework for open source software (OSS) and attempts made by various countries to unify IPR laws. Additionally, it looks at how different nations have modified their legal frameworks to allow for Open Source Software (OSS), emphasizing changes made to laws, rulings from courts, and policy changes. The legal response of India to IPR concerns relating to OSS is also evaluated critically in this chapter.

CHAPTER 5: CONCLUSION AND SUGGESTIONS

The last chapter provides a thorough overview of the study's endeavors and discoveries, resulting in particular suggestions for safeguarding Intellectual Property Rights (IPR) concerning Open-Source Software (OSS). It looks at the rules that now govern intellectual property rights in India, pointing out its shortcomings and the need for changes to meet the particular difficulties that OSS in cyberspace presents.

As society evolves, new intellectual property issues arise, necessitating novel dispute settlement methods. New legislation may be needed to address these challenges. Legislators, public, attorneys, and judges must be prepared to navigate the complexities of open-source software, the Internet, and information technology. This chapter provides crucial insights for adequate intellectual property protection in the digital era.

CHAPTER 2

OPEN SOURCE LICENSING, EVOLUTION AND TYPES

2 DEFINITION

Many intellectual property rights are present in today's society. Patents, copyrights, trademarks, and so on. All these rights provide maximum protection to the creator of the product or work they cover. By the expression ‘ultimate protection’ it is meant to say that these rights preserve the originality of the works, protect the same from any duplication, and thereby recognize the claim of the creator over the product or result. Thus, all these intellectual property rights can be referred to as closed ended rights. Of all these, copyright covers an extensive area, from artistic works like literature to scientific works like software. However, the present-day developments in regard to the field of IPRs, particularly concerning software protection, relatively a new system of protection via licensing known as the Open Source Licensing system is gaining International acceptance.

“Free software is a matter of liberty, not price. To understand the concept, you should think of 'free' as 'free speech,' not as ‘free beer’.” - Richard Stallman, the founder of the Free Software Foundation[1].

2.1 OPEN SOURCE SOFTWARE

The recent development of open-source software, which powers some of the most resilient and significant inventions of our day. Computer software that has its source code made available under a license that allows users to study, modify, and share it with anybody for any purpose is known as open-source software (OSS). This contrasts sharply with proprietary software, which is subject to tight copyright licenses and does not allow users to modify the source. In this context, "source code" refers to the easily comprehensible human-readable version of the program that end users may study, comprehend, and alter. The Open Source Initiative has defined what constitutes an "Open Source" license, and this definition is known as the Open Source definition. To verify that claims meet the Open Source Definition, the Open Source Initiative also certifies them as OSI Certified.

[1] Pal, S. (2021, August 16). History of Open Source Software (with an interactive timeline). Btw. https://www.btw.so/blog/history-of-open-source-software

The source code in the software that makes it available for anyone to see, alter, and improve is known as “open-source software”. The open source way" implies being willing to share information, working transparently with others (so that others can see and participate too), accepting failure as a chance to grow, and anticipating even enticing everyone else to follow pace.

It also involves committing to actively contribute to the betterment of the world, which is only achievable when everyone has access to how that world is created.

More broadly, the phrase "open source" also describes a community-based method of producing intellectual property (like software) through inclusivity, openness, transparency, and regular updates for the public. The term *open source* was introduced in 1998 to clarify that the software was not free but gave users more flexibility because the source code was readily available. Software licensing means that rights of use to computer programs are granted. Such rights may be presented in various ways, including single/straightforward rights of use with all others having the same privileges or exclusively so that the licensee is the only one who may lawfully exercise a particular right[2].

Programs with openly accessible source code that users are free to examine, alter, accept, and distribute for personal and professional use are called Open-Source Software (OSS).

Open-source software is distinct. The people who created it grant permission to anyone who wants to read, copy, modify, share, or learn from the code. Examples of open-source software are Libre Office and the GNU Image Manipulation Program.

2.2 OPEN SOURCE SOFTWARE TYPES

Open source software (OSS) is available as source code under a license that enables anyone to examine, modify, and share the program with anybody for any reason. Community-oriented development, quick prototyping, and the free flow of ideas are all made possible by this collaborative approach to development.

Different types of open-source software are available according to their purpose. These are classified under various categories.

1. Operating Systems**:-**

i. Linux – Linux, an open-source program, has gained worldwide popularity since its launch in the 1990s. It is primarily used on desktop computers and Android devices and

[2] Opensource.com. *What Is Open Source? | Opensource.Com*. https://opensource.com/resources/what-open-source. Accessed 21 Dec 2023

is highly customizable and free of charge. Linux is distributed under an open source license, allowing users to run the program, study its functionality, and modify it[3]. The Linux operating system consists of several components, including the bootloader, kernel, init system, daemons, and graphic server[4]. Users interact with the desktop environment, which includes various desktop settings like KDE, Xfce, Enlightenment, Pantheon, Cinnamon, Mate, and GNOME. Linux also offers thousands of high-quality software titles, including the Ubuntu Software Center, which simplifies and centralizes app installation. Popular Linux distributions include Linux Mint, Manjaro, Debain, Ubuntu, Antergos, Solus, Fedora, Elementary Os, and Openuse.

ii. BSD (Berkeley Software Distribution): - BSD is an open-source platform, one of the first operating systems based on UNIX, developed by Jordan Hubbard in 1978. It has a monolithic kernel, allowing programs to interact with hardware directly. BSD is stable and has no bugs. It supports networks like IPv6, IPsec, and SCTP, and promotes legacy protocols and CARP (Common Address Redundancy Protocols). BSD has three versions: Open, Free, Net, and Dragonfly. Open BSD focuses on security, while Free BSD focuses on performance and ease. Net BSD is portable and used by NASA for space missions.

2. Web Servers: -

a) Apache HTTP Server: - The Apache Server (httd), most popular web server launched in the year 1995[5]. It operates under the terms of the Apache License 2.0 and is developed and maintained by a vibrant community of developers under the auspices of Apache Software Foundation[6]. An open-source platform's primary goal is to provide a secure, efficient, and extensible server that adheres to the current HTTP standards. The Apache HTTP server project develops and maintains an open-source HTTP server for modern operating systems, including UNIX and Windows.Apache software offers security, authentication, performance, user features, and is open and extendable. It supports modern features like proxies, caching, and load balancing. Apache can host different websites and provide

[3] Opensource.com. *What Is Linux? | Opensource.Com*. https://opensource.com/resources/linux. Accessed 21 Dec 2023

[4] Ibid

[5] "What Is Apache? | Definition from TechTarget." *WhatIs*, https://www.techtarget.com/whatis/definition/Apache. Accessed 29 Dec 2023.

[6] *Welcome! - The Apache HTTP Server Project*. https://httpd.apache.org/. Accessed 29 Dec 2023.

dynamic content through Common Gateway Interface scripts. Projects include Apache flink, Apache JMeter, Apache Kafka, Apache 2.0 license, Apache Mesos, and Apache Open Office[7]. It can be easily integrated into other projects and expanded with new modules.

3. Web Browsers : -

   a) Mozilla Firefox: - A customized web browser and free open-source software. With a simple mouse click, thousands of plugins are available for usage. . According to CNET, Mozilla reshaped the technology industry and fanned the flames of open-source software that changed the way social networks and operating systems function[8]. In 2004, the Mozilla Foundation and Mozilla Corporation developed a free open-source software called Firefox[9].

      Firefox is a web browser used on iOS mobile devices, Windows, Mac, and Linux operating systems. Its home page is a Google search page, guided by the Mozilla Manifesto. Firefox is unique due to its privacy and open-source principles, including a pop-up blocker and plugins for developers to add new features. It is mainly used for online browsing but can be used for more purposes like modifying the browser, publishing versions, and uploading source code to a code repository. Different versions include Firefox Quantum, Firefox Nightly, Beta, Developer Edition, and Enterprise. Firefox Extended Support Release (ESR), or Firefox Enterprise- This enterprise version of Firefox lets organizations deploy the browser at scale. ESR updates once a year, unlike the standard Firefox browser, which updates more frequently[10].

   b) Chromium: - The goal of the open-source Chromium browser project is to provide a quicker, more stable, safer, and more reliable online experience for all users.

---

[7] "What Is Apache? | Definition from TechTarget." *WhatIs*, https://www.techtarget.com/whatis/definition/Apache. Accessed 29 Dec 2023.

[8] "20 Years Ago, Mozilla's Move to Open-Source Its Browser Was Radical. Now Even Microsoft's a Convert." *CNET*, https://www.cnet.com/culture/mozilla-open-source-firefox-move-helped-rewrite-tech-rules-anniversary/. Accessed 29 Dec 2023.

[9] "The Future of the Open Web — White Paper." *Mozilla*, https://www.mozilla.org/en-US/foundation/reimagine-open/. Accessed 29 Dec 2023.

[10] "What Is Firefox? | Definition from TechTarget." *WhatIs*, https://www.techtarget.com/whatis/definition/Firefox. Accessed 29 Dec 2023.

This website offers access to design papers, architectural summaries, testing data, and more resources to assist in building and working with the Chromium source code. Google primarily developed and maintained it. The chrome browser was developed in the year 2008 and introduced Chromium as a platform for developers to code their own software for free[11]. Developers use Chromium to test updates, learn how it functions, and pinpoint areas needing improvement. Bugs can also be reported using it by non-developers. Chromium doesn't collect user information or engage in invasive data gathering.

Because Chromium is a less accurate web browser than Chrome, it crashes more may display other unwanted behaviours. However, it works with Chrome browser extensions and provides a comparable user experience without requiring you to commit to Google's intrusive data collection. Non-developers use Chromium because it allows for a similar browsing experience without any overt connection to Google. Chrome is the primary web browser that uses the Chromium source code. However, several other browsers are also based on the same framework. These browsers modify the user experience by adding proprietary features and interfaces to the Chromium source code[12].

The most popular browsers that are built on Chromium are: -

- Opera
- Yandex
- Vivaldi
- Brave
- Epic [13]

4. Office suites: -
    a) Libre Office: - Libre Office is a community-driven software project produced by The Document Foundation, a non-profit organization. The most actively maintained OpenOffice.org replacement project is LibreOffice, free and open-source software initially built on OpenOffice.org (often known as Open Office)[14]. Users who share your belief in the ideals of Free Software and non-

[11] *Chromium*. https://www.chromium.org/Home/. Accessed 30 Dec 2023.

[12] "What Is Chromium Browser Used For?" *Lifewire*, https://www.lifewire.com/chromium-web-browser-4171288. Accessed 30 Dec 2023.

[13] Ibid

[14] *What Is LibreOffice? | LibreOffice - Free and Private Office Suite - Based on OpenOffice - Compatible with Microsoft*. https://www.libreoffice.org/discover/libreoffice/. Accessed 30 Dec 2023.

restrictive sharing of their work with the world are the ones who create LibreOffice. With its feature-rich tools and intuitive UI, unleash creativity and increase productivity. LibreOffice is the most adaptable Free and Open Source office suite available on the market thanks to a number of its applications: Word processing, spreadsheets, presentations, drawings, vector graphics, databases, and formula modification are all handled by the programs Writer, Calc, Impress, and Math[15].

b) Onlyoffice: -OnlyOffice Docs is a free tool that supports MS Office files and supports all standard formats. It allows users to build and edit complex objects, format texts professionally, and use third-party plugins for translation, thesaurus searching, text publishing, and more. It also allows users to create online forms and document templates, export files as PDFs, and offers document viewers. It supports over 30 Sync Share and CMS systems and can be integrated with open APIs for new applications. Other options include onlyoffice Workspace, which combines Docs with a native productivity platform. Free mobile editors for iOS and Android may be used on the device or in any compatible cloud storage[16].

5. Database Systems: -

a) MySQL**:** - One of the critical elements of several web applications and services is MySQL, an open-source relational database management system (RDBMS) that is extensively utilized. It is renowned for being user-friendly, scalable, and dependable. The acronym "SQL" stands for "Structured Query Language". The most widely used standard language for database access is SQL. You can utilize a language-specific API that conceals the SQL syntax, insert SQL statements into code written in another language, or input SQL directly (for example, to produce reports), depending on your programming environment[17]. High-profile applications like Facebook, Twitter, YouTube, Yahoo!, and many more utilize MySQL, which has emerged as the top database option for web-based applications. Oracle is the driving force behind MySQL innovation, providing

[15] *Who Are We? | LibreOffice - Free and Private Office Suite - Based on OpenOffice - Compatible with Microsoft*. https://www.libreoffice.org/about-us/who-are-we/. Accessed 30 Dec 2023.
[16] *Online Office Applications for Business*. https://www.onlyoffice.com/. Accessed 3 Jan 2024.
[17] *What Is MySQL?* https://www.oracle.com/in/mysql/what-is-mysql/. Accessed 3 Jan 2024.

enhanced features to support online, cloud, mobile, and embedded applications of the future. MySQL is a relational database that organizes data into distinct tables for maximum speed. It is a cost-effective and quicker cloud database solution that combines transactions, machine learning (ML) services, and real-time analytics into a single database. MySQL HeatWave is a fully managed database service that allows developers and data analysts to automate machine learning model creation, training, deployment, and explanation. MySQL is open-source software, used by major e-commerce platforms like Shopify, Uber, and Booking.com.

b) PostgreSQL: - PostgreSQL is an open-source, powerful object-relational database system that combines several capabilities to securely store and handle even the most complex data demands with the ability to utilize and expand the SQL language. PostgreSQL has been actively developed for almost 35 years, with its primary platform having its roots in the POSTGRES project at the University of California, Berkeley, which started in 1986[18]. The University of California, Berkeley's POSTGRES package served as the model for the object-relational database management system currently known as PostgreSQL.

PostgreSQL is a popular open-source relational database known for its design, dependability, data integrity, feature set, and extensibility. It is compatible with all major operating systems and has been ACID compliant since 2001. PostgreSQL offers numerous capabilities for developers, administrators, and data management. It is open-source, accessible, and extensible, allowing for new functions, data types, and code writing without recompiling the database. PostgreSQL has been applied to various research and production projects, including geographic information systems and educational aids.

6. Programming Languages: -

a) Python: - It is a common programming and scripted language used by custom software developers. The original version, which featured only a few built-in data types and rudimentary functionality, was released in 1991. Guido van Rossum designed it at the Netherlands National Research Institute for Mathematics and

[18] *Postgresql: About*. (n.d.). from https://www.postgresql.org/about/ ,Accessed on June 23, 2024

Computer Science. It was utilized for system programming, web development, software development, and mathematics[19].

The Python Software Foundation develops the open-source Python programming language. It was the most widely used language in 2021, according to IEEE. Its rapidly expanding machine-learning sector has drawn many new users in recent years. Because of its ease of use, most developers also prefer this open-source software. Making it more straightforward for developers to read and comprehend while also cutting down on the number of lines of code was the main goal in building it[20].

Python, introduced in 1994, is an object-oriented programming language with features like filter, lambda, and map functions. It is free and open-source, versatile, multi-platform, and has hundreds of libraries and frameworks. Python is easy to read, interpret, and has database system connectivity[21]. It can handle extensive data handling and sophisticated mathematical operations, and can be used for software production or quick prototyping. Newer features were included in Python 3.0 (2008 release), Python 2.0 (2000 release), and Python 1.5 (1997).

Python works on various platforms like Windows, Mac, Linux, and Raspberry Pi.

b) Java: - In 1995, Sun Microsystems introduced Java, a programming language and computer platform[22]. From modest origins, it has grown to power a significant portion of the modern digital world by offering a dependable foundation on which many services and applications are based. Additionally, Java is still necessary for cutting-edge, futuristic goods and online services. Java's grammar and principles are derived from the C and C++ programming languages. The Java Runtime Environment (JRE) includes the Java Plugin software. A few Java programs may be launched using specific browsers thanks to the JRE. Java Plugin software cannot be installed independently, as it is not a stand-alone program.

[19] "Welcome to Python.Org." *Python.Org*, , https://www.python.org/about/. Accessed on 4 Jan 2024

[20] "Introduction to Python." *GeeksforGeeks*, 6 Nov. 2015, https://www.geeksforgeeks.org/introduction-to-python/. Accessed on 4 Jan 2024

[21] "What Is Python? It's Uses and Applications." *GeeksforGeeks*, 4 June 2023, https://www.geeksforgeeks.org/what-is-python/. Accessed on 4 Jan 2024

[22] Java, what is Java? https://www.java.com/en/download/help/whatis_java.html (accessed on Jan 22, 2024).

The portability of Java is a crucial benefit while designing applications. It's relatively simple to transfer Java application code from a notebook computer to a mobile device after it's been developed.

The Java Virtual Machine (JVM), the Java API, and an entire development environment comprise the Java software platform. The Java bytecode is parsed and run (interpreted) by the JVM. The Java API includes a vast library collection, including online services, networking and security features, fundamental objects, and XML (Extensible Markup Language) production. The Java programming language and platform provide corporate software developers with a robust, tried-and-true technology[23].

7. Content Management System**:** -

   Software that makes it possible for anyone without technical expertise to produce, administer, and edit content on a website. A content management system (CMS) offers an intuitive user interface that lets people create and manage websites by taking care of the technical parts like building web pages, storing photos, and more. Users can concentrate on the more artistic aspects of managing sites this way[24]. The two main component of the CMS. First one is Content Management Application (CMA) which allows us to add and manage content on our site through a user interface and the second component is Content Delivery Application (CDA). This component is the backbend process that takes the content input in the CMA, stores it properly and makes it visible to your visitors.

   Examples for the CMS are- Word Press, Drupal, Joomla, Magento etc[25].

8. E-commerce Platforms**:** -

   Software that helps companies oversee sales, marketing, and management of their online storefronts is an e-commerce platform. Features that facilitate the following are frequently found in e-commerce platforms: payment methods, shopping carts, inventory management, content management, and checkout.

   A person can create and personalize your online business with e-commerce platforms. He can design his website's general style and appearance and build

[23] Ibid

[24] Guru99, 23 Best CMS Platforms (2024 Update), Guru99, https://www.guru99.com/best-cms.html (accessed on Jan 22, 2024).

[25] Newcomer, Colin. "21 Best CMS Software to Build a Website (And Manage Content Effectively)." *Kinsta®*, 20 July 2020, https://kinsta.com/blog/cms-software/. Accessed on 22 Jan 2024

product pages and payment methods. One can edit, add and organize the products within the platform which includes details of product description, prices, images and inventory levels. Shopify, Woo Commerce (for Word Press), Magento, Big Commerce, Wix, and Square Online are a few well-known e-commerce platforms. Every platform has advantages of its own and meets various company requirements. Think about things like cost, usability, scalability, and feature availability when selecting an e-commerce platform[26].

9. Graphics and design:-
    a) GIMP (GNU Image Manipulation Program):- GIMP is an image editor that works on multiple operating systems, including Windows, macOS, and GNU/Linux. It is open-source software that you are free to modify and share your modifications with others. GIMP is an excellent framework for manipulating images programmed in multiple languages, including C, C++, Perl, Python, Scheme, and others. GIMP has excellent colour management tools to guarantee accurate colour reproduction in both printed and digital media. It works best in workflows with other free programs like Inkscape and Scribus. Scheme, Python, Perl, and many other programming languages are integrated with GIMP to offer extensibility. The abundance of scripts and plugins developed by the community demonstrates the great degree of customization that results[27].
    b) Inkscape: - For GNU/Linux, Windows, and macOS, Inkscape is a free and open-source vector graphics editor[28]. With its extensive feature set, it is famous for its usage in both technical and artistic graphics, including flowcharting, diagramming, cartoons, clip art, logos, and typography. Unlike raster graphics, which are confined to a set number of pixels, vector graphics provide excellent printouts and renderings at any resolution. The primary file format used by Inkscape is the standardized SVG file format, which is compatible with a wide range of other programs, including web browsers. Several file formats, including SVG, AI, EPS, PDF, PS, and PNG, can be imported and exported using it. With its extensive feature set, easy-to-use interface, support for

[26] Ibid
[27] "GIMP." *GIMP*, https://www.gimp.org/. Accessed 22 Jan 2024.
[28] admin. "What Is Inkscape? All You Need to Know." *OnWorks*, 16 Dec. 2022, https://www.onworks.net/blog/what-is-inkscape/.

multiple languages, and flexible design, users can apply add-ons to personalize Inkscape's functionality further[29].

10. Development Tools: -

    a) Git**:** - A version control system (VCS) is a tool that tracks changes made to a project, allowing individuals and groups to work together. It provides a uniform view of the project, allowing each contributor to work independently. The most commonly used DVCS is Git, which allows developers to view their decisions, modifications, and project progress. Git allows collaboration across time zones and securely suggests changes to production code. It also enables professionals from different departments to coordinate on large-scale projects.
    b) Eclipse: - Eclipse is an integrated development environment (IDE) that offers a stable software development platform under the Apache license. Plugins enable Eclipse, an extensible integrated development environment (IDE) first created by the Eclipse Foundation, to support a wide range of programming languages. Eclipse is a versatile IDE that uses Java and can be customized to suit different development needs. It offers a suite of tools for software development, including compilers, debuggers, code editors, and project management. It also provides powerful debugging features, version control, branch management, and code collaboration. Eclipse also supports development, testing, deployment, and build automation.

       Eclipse is appropriate for desktop and web application development since it facilitates the creation of graphical user interfaces (GUIs) with the help of tools like Window Builder. Eclipse is a flexible option for developers working on several platforms because it is made to run on a variety of operating systems, such as Windows, macOS, and Linux[30].

There are other open source software such as Wire shark, which is for networking and security functions, open VPN, educational software such as Moodle, and audio video editing. Audacity is the open source software.

[29] Developers, Inkscape Website. *About | Inkscape*. https://inkscape.org/about/. Accessed 22 Jan 2024.
[30] Singh, R. (2023, December 13). What is Eclipse and use cases of Eclipse? *DevOpsSchool.Com*. https://www.devopsschool.com/blog/what-is-eclipse-and-use-cases-of-eclipse/ Accessed 22 Jan 2024

2.3 OPEN SOURCE LICENSING

The legal framework controlling open-source software's usage, distribution, and modification is known as open-source licensing. Software licenses that permit content to be used, changed, and shared are known as open-source licenses. They aid in the creation of free and open-source software, or FOSS. Intellectual property (IP) laws limit how creative works can be altered and shared. These extant legal frameworks are used oppositely by free and open-source licenses. They provide the receiver permission to use the program, view the source code, make changes, and share those changes. Hence, by granting receivers access to the program's source code, open-source licensing preserves the software's uniqueness while allowing them to alter and utilize it as they see fit.

Distributing the open source software has to follow specific criteria. It follows as:-

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 sources. The license shall not require a royalty or other fee for such sale.
2. Source Code: - Program distribution must be permitted in source code and compiled form, and source code must be included. If a product is not supplied in source code form, there should be a well-known way to access the source code for no more than the cost of a decent reproduction—preferably, free downloads via the Internet. The format a programmer would prefer to use to alter the software must be the source code. It is not permitted to obscure source code intentionally. Forms in between are prohibited, like the result of a translation or pre-processor.
3. Derived works: - The license must permit the distribution of derived works and modifications under the same conditions as the original software.
4. Integrity of The Author’s Source Code: - If the license permits the release of "patch files" containing the source code to alter the program at build time, then the license may prohibit sharing source code in modified form. The license must expressly permit the distribution of software created using altered source code. Derived works can need to use a different name or version number from the source program according to the licensing.
5. Distribution of License: - Any individual to whom the program is redistributed must be entitled to use it under the terms of the program without those parties having to execute an extra license.

6. License must not be specific to a product: - Program rights cannot be granted about the program's membership in a specific software distribution. Redistributing the program should offer all parties the same rights as it did with the original software distribution, provided that the program is extracted from that distribution and utilized or disseminated by the licensing terms.
7. The license must not restrict other software: - Restrictions on other software distributed with licensed software cannot be included in the license. The license should not, for instance, mandate that any other software delivered on the same media be open-source.
8. License must be technology- neutral: - No term of the license may be based on a particular interface design or technology.

2.3.1 HISTORY OF OPEN SOURCE LICENSING

The legal structure that controls the usage, distribution, and modification of open-source software is known as open-source licensing. Open source licensing has its roots in the early years of computers when the software was viewed as an exclusively intellectual product subject to copyright protection. The open-source movement, which gained popularity in the 1990s, aimed to reframe software development by prioritizing community involvement, open communication, and teamwork.

Richard Stallman's GNU General Public License (GPL), first published in 1989, was one of the first open-source licenses. The GPL was created to guarantee that software distributed under it would always be open-source and accessible and that any changes or additions would be governed by the same rules and regulations as the source code.

Numerous open-source licenses with unique terms and conditions have been created. These comprise, among others, the Mozilla Public License, the MIT License, the BSD License, and the Apache License. Developers must carefully assess which license is most suited for their project, as each has pros and downsides.

The historical evolution of open source licensing can be traced out by analysing the following time line:

In 1953, the A-2 system (an equivalent of today’s compilers) was released together with its source code, and customers were asked to send any improvements to UNIVAC (the Universal Automatic Computer). UNIVAC was a division under Remington Rand, Inc., an early

American business machine manufacturer. Remington Rand had acquired UNIVAC patents as well as its creators, J. Presper Eckert, Jr., and John Mauchly.

There were a few decades of OSS “silence” after the 1950s. Creating software was an expensive and exceedingly complex process. Giving it away for free was thus out of the question.

In 1983, however, Richard Stallman started to work on the GNU project, which was made up of rewrites of closed software he frequently used. GNU stands for "GNU’s Not Unix" and is pronounced as one syllable with a hard. In 1984, Stallman spearheaded the creation of the GNU, a free operating system that was made to counter closed systems. In 1985, he wrote the GNU Manifesto, asking for support in the development of the GNU operating system. He also founded the Free Software Foundation (FSF), a non-profit that was aimed at promoting freedom in computer use. By 1987, most of the essential components of the GNU operating system were complete. There was an assembler, editor, and various UNIX utilities like *grip* and *ls.* A C compiler was almost finalized. In the 1980s, Stallman also created the GNU General Public License (GPL). All the components of the GNU operating system were released under this license. Today, the GPL allows for the freedom to share and change all program versions, ensuring that they remain accessible to all users.

The 'free software' or open source movement made little headway until 1987 when Larry Wall created the Unix-based computer language Perl. The GNU GPL was first used to post this on Usenet, but the author later devised his "Artistic License." In 1991, Linus Torvalds, a 21-year-old undergraduate student from Finland, fulfilled his aim of developing an operating system resembling UNIX and named it Linux.

The Internet became increasingly significant in society after emerging from research facilities and academic institutions. The number of people connected to the Internet increased to approximately 100 million by 1997, and by 1998, the amount of traffic on the network had doubled. The Internet's foundational protocols and parts were open source and non-proprietary.

The open-source community embraced a collaborative form of software development in which programmers shared code, concepts, and time to produce software via online forums. The majority of software companies employ a closed-source strategy, which is in opposition to this approach. The for-profit corporation Netscape made the source code of its Communicator

browser available to the public, adapting terms and elements from pre-existing open-source agreements.

The rise of open-source software that is sold for a profit emphasizes how crucial it is to consider the terms and enforceability of open-source licenses. Open source software may become "closed" or "quasi-open" due to the appearance of commercialized versions, underscoring the need for improved licensing arrangements and enforcement[31].

Other salient events in the 1990s that significantly shaped OSS include:

- The publication of the Python interpreter source code (1991)
- The launch of the Apache HTTP Server (1995)
- The coining of the term “open source” (1998)
- The release of the Netscape browser’s source code (1998)
- The release of Open Office’s source code, the free software counterpart to Microsoft Office (2000)

The 20th century laid a study foundation for OSS. We’ll explore more recent open source developments a bit later on.

- The Free Software Foundation Europe is created to support free software in Europe (2001)
- Version 1.0 of Mozilla Firefox becomes available to the public (2004)
- Git, a version control system, was released in 2005 by Linus Torvalds, Linux’s creator.
- Google releases the Android mobile OS (2008)
- Block chain is built on lots of open source technologies and developers contribute to its code base (2008)
- Node.js comes out (2009)

[31] Steve H. Lee, Open Source Software Licensing (Pre-Publication Version as of Apr. 28, 1999), https://cyber.harvard.edu/openlaw/gpl.pdf (Accessed on Jan 23, 2024).

- Google Chrome is based on the Chromium project's open source code. Chrome was initially release lot of open source tools power IoT, including the Linux Kernel. IoT became a conglomeration of technologies in 2013
- Containers and container orchestration (Docker 2013 and Kubernetes 2014)
- The cloud (2016)
- Firefox Quantum (2017). Firefox's biggest update in 2017, saw it rise to become the third most used browser.

All these mentioned above are the various facets of how the open source licensing system are developed over a period of time into what we see at present.

2.3.2 TYPES OF OPEN SOURCE LICENSING

Programmers created what is now called "Open Source" licensing in part as a response to this distributor-driven copyright licensing paradigm. Programmers created what is now called "Open Source" licensing in part as a response to this distributor-driven copyright licensing paradigm. The term open source is misinterpreted by software developers, including beginners, to imply that the program is free to use, duplicate, alter, and distribute as one wants. This misinterpretation could result from mixing up open source with shareware or public domain, which are unrestricted in usage and modification without licensing or authorization. Open-source software is not always free and is typically subject to one of several kinds of open-source licenses.

Open-source licensing differs significantly from proprietary software licensing because users can reuse, share, modify, and even distribute open-source code. However, depending on the type of open source license in effect, open source software may be subject to different legal terms and restrictions, much like with proprietary software licensing. As a result, it's critical to abide by the conditions of open-source software licenses. Before using open-source software, one must also be aware of its additional dangers.

Although there are more than 80 different types of open-source licenses, they typically belong to one of two main categories: permissive and copyleft[32].

[32] *Open source licenses: Types and comparison*. (n.d.). Snyk., from https://snyk.io/learn/open-source-licenses/ Accessed on Jan 23, 2024

A. Copyleft Licensing: - The concept of copyleft ensures free and open software or other creative work for everyone. The main objective of the copyleft is to give users the freedom to use, modify, and distribute software. The term copyleft was first shown up in the year 1976 in a Tiny BASIC program written by Dr. Li- Chen Wang. Intellectual property can be made reusable and adaptable without limitations by employing the copyleft technique, but anything new created with the original asset must likewise be publicly available. This applies to software as well as artwork. According to copyleft licenses, software may be altered and redistributed by anybody, provided that derivative works maintain the same rights.

B. Permissive Licensing: - A permissive software license is a free software license that imposes very few limitations on the program's uses, modifications, and redistributions. Permissive licenses offer greater flexibility than copyleft licenses, which demand the reciprocal release of source code for changed versions. Developers are free to take the software released under a permissive license, add to or modify it as they see fit, and then share their updated version with others. This is a crucial feature if you want to develop exclusive software that you can market and keep a secret from rival developers. It is one of the main reasons for the popularity of the permissive license.

Copyleft licenses require derivative work of licensed software to be distributed under the same copyleft license, while permissive licenses allow developers to share source code of modifications. The Prior BSD license, the precursor to the first BSD license, was released in the late 1980s. Permissive licenses require users to include the original copyright notice and license language in any redistribution of licensed software.

2.3.2.1 COPYLEFT LICENSE TYPES

A. A GNU GPL(GNU General Public License) –

The GNU General Public License is a free, copyleft license for software and other kinds of works[33]. One of the foundation open source licenses is the General Public License, or GPL, published by the GNU. It is the recommended license for projects approved by the Free Software Foundation (FSF), which has made several contributions to open-source coding. Examples of these projects include the GNU C Compiler and the GNU Emacs Editor, among

[33] *The gnu general public license v3. 0—Gnu project—Free software foundation*. (n.d.)., from https://www.gnu.org/licenses/gpl-3.0.en.html accessed on June 22, 2024

hundreds more, including the GNU/Linux kernel. GNU GPL was first written by Richard Stallman in 1989. Developers and organizations use the GPL to prevent the software from becoming proprietary[34].

Most software and other valuable works have licenses that limit your ability to share and alter the works. In contrast, the goal of the GNU General Public License is to ensure that you have the freedom to distribute and modify any version of a program, ensuring that it stays free software for all of its users[35].

The copyright notice is followed by a preamble that forbids changes to the license itself: "Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed[36]." Although derivative works from the licensed code are allowed under the license, derivative licenses from the license itself are not.

Nobody should be restricted by the software they use. There are four freedoms that every user should have:

• The freedom to use the software for any purpose,
• The freedom to share the software with your friends and neighbours,
• The freedom to change the software to suit your needs, and
•The freedom to share the changes you make[37].

Software developers are permitted to release their works under the GNU GPL. The program will always be free software when they do, regardless of who distributes or modifies it. Refer to this as copyleft software because, unlike proprietary software, it uses copyright protections to guarantee user freedom rather than to impose restrictions on users.

The GNU General Public License aims to ensure that you have the freedom to distribute and alter free software, ensuring that it remains free for all users. The majority of the software

---

[34] *Gnu general public license | gplv3 explained*. (n.d.). Snyk., from https://snyk.io/learn/what-is-gpl-license-gplv3-explained/ Accessed on Jan 24 2024

[35] Suprantoe41

[36] St Laurent, A. M. (2004, August 16). *Understanding Open Source and Free Software Licensing* (1st ed.) "O'Reilly Media, Inc."http://books.google.ie/books?id=04jG7TTLujoC&printsec=frontcover&dq=Understanding+Open+Source+and+Free+Software+Licensing+By+Andrew+M.+St.+Laurent&hl=&cd=1&source=gbs_api

[37] Brett Smith, A Quick Guide to GPLv3, Free Software Foundation, Inc. 2007

developed by the Free Software Foundation is covered by this General Public License, as is any other software whose creators agree to use it. The GPL protects users' rights in two steps: firstly, by copyrighting the software, and secondly, it offers the users this license, which gives them legal permission to copy, distance, and or modify the software. It also wants to ensure that everyone knows this free software is provided without warranty for the safety of each author and their own. They want the recipients of modified software to be aware that it is not the original to prevent any issues caused by third parties from harming the original authors' reputations.

The three primary goals of the GPL are outlined in its preamble in an easy-to-read manner. The first and most crucial is maintaining the open software, meaning it can be shared and altered without the licensor's extra consent. It puts the licensee under a mirror-image restriction. Although the licensee can access the licensed work at no cost, they must distribute any derivative works with the same restrictions and limits as the licensed work. Making sure licensees are aware that software released under the license is provided "as is" and without warranty is the second goal of the GPL. The GPL is one of many licenses that have this goal. The third goal, which is a variation of the first, is that the licensed software must be free from patents that impose restrictions; if a patent applies to the program, it must be licensed concurrently with the code[38].

The GPL has been updated to protect its copyleft from being undermined by legal or technological developments. Three main threats protect users with the recent version. The most recent threats are[39]

- Tivoization**:** Several businesses have developed GPLed software-running devices of various types, and they have tampered with the hardware to allow them to alter the operating software while preventing you from doing so. A gadget is a general-purpose computer whose owner should control what it does if it can run any program. We refer to it as tivoization when a gadget prevents you from doing that.
- Legal restrictions on free software: Writing or distributing software capable of breaking DRM is illegal according to laws like the European Union Copyright Directive and the

[38] Supranote36

[39] Supranote37

Digital Millennium Copyright Act. Your rights under the GPL shouldn't be affected by these laws.

- Discriminatory patent agreements**:** As long as the software is obtained from a vendor paying Microsoft for the right to use it, Microsoft has said that it will not prosecute free software users for patent infringement. In the end, Microsoft is interfering with users' freedom by attempting to collect royalties for free software. No business ought to be able to carry out this.

It is not permitted to incorporate your program into proprietary programs using this General Public License. The GPL Version2 is the first version of the GNU GPL License and it get upgraded to GPL Version3. GPLv3 offers stronger copyleft safeguards than GPLv2, yet it still has the same main goal. The GPLv3's language is far more thorough when it comes to handling legal and technological modifications, such as international license exchange provisions. The main clauses that included in the GPL Version3 are:-

⇒ Regulations governing compatibility - When two distinct elements, each licensed under a different set of terms, come together to create a new work, this is referred to as licensing compatibility.

⇒ Digital rights management: As users resorted to legal laws (such as the DMCA) for technical protective measures, new clauses aim to limit GPL software modifications at will.

⇒ Explicit patent licensing: This new patent provision requires licensees to guarantee that all users receive the same benefits or that no one can profit from the program, protecting consumers from the possibility of only a small number of licensees benefiting from patent rights.

⇒ Source code exception for ASP - It makes clear that if users do not provide a copy to clients, they must reveal the source code in an ASP implementation of the GPL[40].

Conclusively, the GPL has significantly influenced the software sector by endorsing an ideology that prioritizes transparent, cooperative software development. It has made it possible for software to continue being an innovative and empowering tool and has guaranteed that everyone's rights to use, examine, alter, and distribute software are upheld.

[40] Supranote33

A. B GNU Lesser General Public License (LGPL)

The GNU Lesser General Public License is another license designed by the FSF to allow a specific class of programs typically subroutine libraries to be licensed under an FSF license but still be able to link with non-GPL software. Developers and businesses are no longer obliged to disclose the source code of their software components under a restrictive copyleft license; instead, they can use and integrate software components released under the LGPL into their own, even proprietary, software. Developers must, however, release their altered version of an LGPL-covered element under the same license if they change it. It is unnecessary to use the entire program to continue using the LGPL; only the derivative program's licensed portion is required. Shared components like libraries (.dll, .so, .jar, etc.) are commonly licensed under the LGPL. The choice of license makes a big difference: using the Lesser GPL permits the use of the library in proprietary programs; using the ordinary GPL for a library makes it available only for free programs.

However, the LGPL is considered a "weak copyleft" license when compared to the GPL and AGPL. This kind of license lies in the middle of permissive licenses like the MIT or BSD licenses and stringent copyleft licenses like the GPLs. However, because of its unique safe harbour for dynamic linking integration, the LGPL differs slightly from other weak copyleft agreements, such as the Mozilla Public License or Eclipse Public License[41]. The main application of the LGPL is to libraries. In practical terms, GNU- LGPL, or GNU Library General Public License, was the original name of this license when it was published in 1991. With the release of version 2.1 in 1999, the name was altered. The most recent version, 3.0, was made available by the Free Software Foundation in 2007, along with the most recent GNU GPL version (also v3)[42].

LGPL v2.1 and v3 are still in use today. Although v3.1 is the most recent version the Free Software Foundation advises adopting, v2.1 is still the most often used. The requirements for the two versions are similar. Nonetheless, LGPL version 3.0 is based on GPL version 3.0, and, in case the program is utilized as a component of a consumer device, users are required to

---

[41] *Open Source Software Licenses 101: The LGPL License - FOSSA*. (2022, October 26). Dependency Heaven. https://fossa.com/blog/open-source-software-licenses-101-lgpl-license/ or (*Open Source Software Licenses 101: The LGPL License - FOSSA*, 2022)

[42] Ibid

submit any installation information needed to update and reinstall the application. Furthermore, the LGPL v3 explicitly grants patent rights to the developers who created or contributed to the code. Thus, they forfeit their patent rights concerning any further software reuse.

When using code covered by the GNU LGPL (both LGPL 2.1 and LGPL 3), users must Provide a copy of the original copyright notice and the entire license language. When you share a derivative work built using the licensed library, you must make the source code available. As previously mentioned, any library derivatives must be licensed under the same or a later version of the GPL or LGPL[43].

A. C Mozilla Public License (MPL 1.1)

The Mozilla Foundation offers the Mozilla Public License (MPL), likewise regarded as a weak copyleft license. In contrast to the Eclipse Public License, this license is a file-based copyleft, meaning that code may be mixed with either proprietary or open-source code[44]. It is intended to work with the GNU General Public License (GPL), although it has some features customized for Mozilla project requirements. File-level copyleft, which mandates that changes made to files licensed under the MPL be made available under the MPL, is one of the MPL's unique features. It does not, however, apply this need to additional files or program components. Because it permits the combination of MPL-licensed code with code under other licenses, including proprietary licenses, without requiring the entire work to be open-sourced, the MPL is called a "weak copyleft" license. This makes it appropriate for initiatives that promote transparency and cooperation while maintaining a certain degree of commercial use flexibility. The Mozilla Public License (MPL) is a cross between the BSD and GPL licenses, allowing for the use of "Covered Code" in "Larger Works" and combining it with code licensed under another license[45]. The MPL is more like a corporate contract, with a detailed definition of "Contributor" and usage[46]. It differs from the GPL in

---

[43] Ibid

[44] Horcasitas, J. (2021, November 2). *Understanding Open-Source Software Licenses*. DigitalOcean. https://www.digitalocean.com/community/tutorials/understanding-open-source-software-licenses. accessed on Jan 29 2024

[45] Section 3.7

[46] "Contributor" means each entity that creates or contributes to the creation of Modifications, *Mozilla Public License, and version 2.0*. (i.e.). https://www.mozilla.org/en-US/MPL/2.0/accessed on Jan 29 2024

terms of the definition of "source code" and the distribution of source code. The BSD License permits this latter outcome, whereas the GPL forbids it. The MPL creates a compromise between the two licenses[47].

The MPL allows for the dissemination of source code in compressed or archived file format, as long as it can be unzipped with freely accessible software. This innovative solution to Netscape's challenges involves releasing a pre-existing open-source code base, establishing guidelines and restrictions to safeguard its intellectual property, and motivating developers to enhance and adapt it. Its definitions and areas of interest mirror its inception, differing from the more freeform development vision of the GPL. The MPL's focus on patent rights and limited transfer of those rights also aligns with Netscape's corporate roots and intention to restrict grant of rights while adhering to an open-source framework.

A. D Eclipse Public License (EPL)

The Eclipse Foundation maintains the weak copyleft open source Eclipse Public License (EPL). The EPL was launched in two versions: EPL-1.0, which debuted in 2004, and EPL-2.0, which came out in 2017[48]. IBM's Common Public License (CPL) was the foundation for the first edition of the Eclipse Public License, version 1.0. The CPL and EPL-1.0 differed in two main ways. Firstly, The Eclipse Foundation is the agreement steward for the EPL, and IBM is the agreement steward for the CPL. Secondly, The EPL removed the following sentence from the CPL that covered potential patent litigation: “If Recipient institutes patent litigation against a Contributor with respect to a patent applicable to software (including a cross-claim or counterclaim in a lawsuit), then any patent licenses granted by that Contributor to such Recipient under this Agreement shall terminate as of the date such litigation is filed[49].” The statement was taken down because it was seen as "overly broad" and a hindrance to "the continued growth of the Eclipse eco-system," according to the Eclipse Foundation website. The Eclipse Public License 2.0 was released by the Eclipse Foundation in

---

[47] Supranote36

[48] Supranote41

[49] Ibid

2017; the EPL-1.0 is no longer used. The Eclipse Public License (EPL) requires derivative works of code licensed under it to be licensed under the EPL. This requirement is not included in permissive licenses but is more limited than the GPL. The EPL requires a copy of the full license text and original copyright notice, making source code available when distributing a derivative, and licensing derivative works under the same or later version. Users must defend EPL contributors from legal damages involving their product[50]. The EPL allows users to modify code, provide source code, and add a Secondary License for compatibility with GPL v2. The EPL forbids utilizing contributors' names, trademarks, or logos and forbids holding contributors accountable for any legal problems or losses[51].

A. E GNU Affero General Public License (AGPL)-

The AGPL is a more limited edition of the GPL. It targets software specifically that operates on servers and offers network-based services. When utilizing AGPL-licensed code in a networked application, you must provide users interacting with your service via the network with access to the updated source code. The preamble of AGPL states that- The GNU Affero General Public License is a free, copyleft license for software and other kinds of works, specifically designed to ensure cooperation with the community in the case of network server software[52]. The GNU Affero General Public License ensures the community can access the revised source code. Users of a network server must have access to the source code of the changed version that is now operating on the server from the server's operator. As a result, the public gains access to the changed version's source code when they utilize it on an open server.

Similar objectives were intended to be achieved by an earlier license that Affero produced, known as the Affero General Public License. This license is not an Affero GPL version; Affero has issued an updated version of the Affero GPL that allows for

[50] Milinkovich, M. (n.d.). Eclipse Public License 1.0 (EPL) Frequently Asked Questions | the Eclipse Foundation. https://www.eclipse.org/legal/eplfaq.php?ref=fossa.com accessed on 3 Feb 2024

[51] Supranote41

[52] *GNU Affero General Public License - GNU Project - Free Software Foundation*. (n.d.). https://www.gnu.org/licenses/agpl-3.0.en.html accessed on 3 Feb 2024

relicensing[53]. The AGPL license code is used for commercial purposes and does not impose conditions on using the code in commercially sold software. It is permissible for users to alter or modify the code. Still, they must share the updated versions in source code form under the AGPL license if they publicly distribute the changes or modifications (such as over a server). Original code distributors may provide their license software warranty. The code cannot be sublicensed under the AGPL License.

MongoDB, a well-known NoSQL database application, is one corporation that has used AGPL previously. GNU AGPL was the license for all versions released before October 16, 2018. The company used its Server Side Public License for everything released after that[54].

2.3.2.2 **TYPES OF PERMISSIVE LICENSES** –

A. Massachusetts Institute of Technology (MIT) License: -

A renowned open-source software license, the MIT License is applied to programs like Node.js, Jquery, and Ruby on Rails. The MIT License, an open-source license, establishes requirements for using, modifying, and redistributing the licensed code and what you are not allowed to do. MIT license carries a few restrictions compared to copyleft licenses like GPLv3 and AGPL. In addition, compared to many other joint agreements, the MIT License is simpler and easier to understand. MIT License does not mandate that anyone who changes the original code must make those changes available under the same license. Even if you significantly alter the code, there is no obligation to reciprocate or "pay it forward." The licensee is free to keep his updated version private[55].

The MIT License is a copyright-based license that allows users to modify and use the software without requiring the original copyright notice. It operates after the copyright notice has been completed, making compliance easier. Unlike copyleft licenses, the MIT License does not require anyone to make changes under the same license, and users can use the software for

[53]Ibid

[54]Supranote40

[55]Supranote40

various purposes. Several versions of the License have been developed, including MIT-0, JSON license, and X11 license. MIT-0 eliminates attribution clauses, JSON license emphasizes good use, and X11 license is similar to the original MIT License[56]. The MIT License is known for its simplicity, encouraging open collaboration, and offering a compromise between open-source contribution and proprietary development. It is popular among developers and organizations concerned about legal liability.

Besides the benefits of an MIT license, it also has certain limitations, including no warranty. The developers or copyright holders make no promises regarding the software's operation, dependability, or fitness for a particular purpose. A limitation of liability provision is included in the license. This implies that if the software causes any harm or problems, the copyright holders are not liable. Any risks incurred by users while using the software are their responsibility. In contrast to certain other open-source licenses, the MIT License does not demand that derivative works be made available as open source. Although this may be advantageous to certain users, it can also be perceived as a constraint by those who want to guarantee that all modifications of their software stay open source[57].

B.B. The BSD License (Berkeley Software Distribution)

A set of permissive free software licenses known as the Berkeley Software Distribution (BSD) license places few limitations on the usage and redistribution of licensed software. The open-source community loves the BSD licenses because of their ease of use and compatibility with many other licenses. The BSD License has several comparable versions and is marginally more restrictive than the MIT License.

The BSD license allows users to use, modify, and distribute software with minimal restrictions. It requires a copy of the license and a disclaimer of liability. Developers and businesses use this license to ensure their creations are accessible while preserving software rights. For software development projects, the BSD license has various benefits: no patent protection, no need to reveal source code, no viral impact, flexibility, and increased adoption. It promotes cooperation, does not demand source code disclosure, and permits unrestricted free use,

[56] *What is mit license?* (n.d.)., from https://memgraph.com/blog/what-is-mit-license Accessed on 5 Feb, 2024

[57] Ibid

modification, and distribution. Specific iterations lack patent protection provisions, enabling developers to employ proprietary technologies without legal ramifications. Software projects like OpenBSD, FreeBSD, OpenCV, Python, LLVM, PostgreSQL, etc... Use BSD license. BSD licenses are of various versions. The original BSD license, the BSD 4 Clause License, was released in 1990[58]. The most notable distinction between the 4 clauses and all BSD license versions is the introduction of an "advertising clause" requiring the source of the code to be acknowledged in all advertising materials.

The 3-Clause BSD License, sometimes called the "New BSD License" or the "Modified BSD License," has an extra clause. William Hoskins, the director of UC Berkeley's Office of Technology Licensing, published the three-clause BSD License in 1999[59]. The third clause disagreed with the GNU GPL license and the advertising provision because the GPL license forbids adding new restrictions on top of those it currently imposes. Future iterations of the BSD and GNU GPL licenses would fix this conflict.

In line with the 2-Clause BSD, The two clauses from the 2-Clause BSD License remain. Clause of No Endorsement: Add a disclaimer that says that without express prior written consent, neither the project's name nor the names of its contributors may be used to support or advertise goods made using this software. A far more streamlined BSD license was made available in 1999[60]. The FreeBSD, or BSD 2 Clause, is widely used.

The BSD 3-clause License and FreeBSD differ in their non-endorsement clauses, with the latter adding a supplementary disclaimer. Both licenses are compatible with the GNU GPL and are given "as is" without guarantees. The BSD family of licenses was created before software patenting became popular in the US, so they lack an explicit patent license. The MIT License and Apache 2.0 are compatible with BSD 3 and 2 clause licenses, while BSD 3 is compatible with the GNU-GPL family. The BSD Zero Clause license eliminates the obligation to include copyright statements or license wording.

---

[58] Librarian, U. L. S. (n.d.). *Guides: Open licenses: creative commons and other options for sharing your work: bsd licenses*. Retrieved Feb 6, 2024, from https://pitt.libguides.com/openlicensing/BSD

[59] Ibid

[60] *What is BSD licenses? | Definition from TechTarget*. (n.d.). WhatIs. Retrieved Feb 22, 2024, from https://www.techtarget.com/whatis/definition/BSD-licenses

B.C Apache License, v1.1 and v2.0

The previously mentioned MIT and BSD licenses bear a striking resemblance to the Apache License. The Apache License, Version 1.1, precludes distribution and modification upon compliance with comparatively lax requirements, following the same pattern as the BSD License. The license was rewritten from the top down in version 2.0, initially released in 2004[61]. Although Version 1.1 is longer than the licenses covered previously in the chapter, it functions much similarly.

Redistributing and using in source and binary formats, whether altered or not, are accepted as long as the requirements listed below are satisfied:

1. The above copyright notice must be included in any redistributing of source code; this list of Terms and the disclaimer that follows

2. The copyright notice above, the requirements listed here, and the disclaimer below must be reproduced in the documentation and other materials included with any binary redistributions.

The first two restrictions, the copyright notice and the section outlining the distribution restrictions are essentially the same as those included in the BSD Licenses. The first two constraints, the copyright notice and the section outlining the distribution restrictions, are the same as those in the BSD License. This clause shields the author from potentially harmful associations with works derivative of the source code[62]. Clauses indicating the contributors to the code being distributed close the license. Since the user is not required to do anything, these are not technically included in the permit. This software was partially developed using public domain software at the University of Illinois at Urbana-Champaign's National Center for Supercomputing Applications.

The second iteration of the Apache License, v2.0, was published in January 2004. The Apache License, version 1.1, functions similarly to a BSD or MIT License and prohibits using the Apache name without authorization through a non-endorsement clause. However, version 2.0 is a more comprehensive and intricate license that delineates the rights given in greater depth. Specifically, v2.0 is different in that it covers the usage of other permits for derivative works

[61] Supranote41
[62] Ibid

and the patent rights granted under the license. Works derived from v2.0-licensed works. The most significant feature of v2.0 is probably the provision for "Contributions" to the licensed work produced with the explicit understanding that they will be covered by v2.0 and integrated into the licensed work[63].

Based on the BSD license, the GPL v2 becomes incompatible with the Apache License v2. According to the Free Software Foundation, the limitation in v2 prevents code from v2 from being combined with code licensed under the GPL v2 license. Nevertheless, the GPL v3 eliminates this mismatch by allowing a patent retaliation clause, which makes the two licenses compatible again.

B.D The Academic Free License

The Academic Free License and the Apache License, v1.1, are very similar in that they both prohibit assertions of the creator's endorsement of the work, demand attribution to the creator, make no warranties, and allow distribution of the original work and derivative works only under specific restrictions. Four additional clauses, two of which deal with patent law and the other two with choice of law and shifting of attorneys costs, are added to the Academic Free License and are not included in the Apache or BSD Licenses. Lawrence E. Rosen designed the Academic Free License (AFL), a permissive free software license. It seeks to balance the interests of developers and users by offering a transparent and legally binding structure for software distribution and use. The AFL is appropriate for commercial applications and academic and research projects due to its simplicity, flexibility, and compatibility with other open-source licenses. Some of its salient characteristics are permissiveness, express grant of rights, patent grant, legal clarity, attribution requirements, compatibility with different licenses, warranty Disclaimer, and Limitation of Liability. Clarity and legal precision, broad use, attribution requirements, patent protections, and interoperability with other licenses are some of the advantages of the AFL. With the interests of consumers and developers protected, it fosters innovation and teamwork[64].

---

[63] *The Apache License (v2) - An Overview*. (2012, May 14). http://oss-watch.ac.uk/resources/apache2 accessed on 22 Feb 2024

[64] *Academic Free License v. 3.0*. (2023, June 27). Open Source Initiative. https://opensource.org/license/afl-3-0-php

CHAPTER 3
DEVELOPMENT AND PRESENT STATUS OF OPEN SOURCE LICENSING.

3 INTRODUCTION

At this time, there are two main types of software based on their source code: closed and open software. In previous chapters we have established the provenance of open-source software, explored the various types of open-source software and why we license open-source software and have looked at different licenses in place for open-source software that exist today. The key difference between one category and another is that the former is where developers have made software tamper-proof and duplication resistant and in the latter users have the freedom to duplicate and change the software itself. The main problem with open-source software is that it just needs more law to protect it. As public disclosure does not protect open software, all that is currently known for their accommodation is a Permission from the authors of the initial version, which is again of limited protective force. This section discuss about current world wide status of open-source software and national level with respect to the existing international and national IPR laws.

3.1 EVOLUTION OF OPEN SOURCE LICENSING

OpenBSD founder and project leader Theo de Raadt removed a security software package called IPFilter [written by Darren Reed] after its author changed its license". —Stephen Shankland[65].

Ensuring the interoperability of FOSS components in a product requires tracking the evolution of licensing. This is especially important if a component's licensing changes and prevents distribution. This necessitates carefully reviewing all included source files and binaries, which could impact how the product is utilized and integrated. The evolution of licenses has been

[65] CNET News, 2001/05/30, (IEEE - the World's Largest Technical Professional Organization Dedicated to Advancing Technology for the Benefit of Humanity. n.d.)

damaging because open-source software has made it easier for contributors to include code that infringes intellectual property rights and makes it more difficult to audit the whole code base.

Several things influence the development of licensing. Copyright holders want licenses to, on the one hand, take into account their particular needs and, on the other, adapt to the current legal landscape. For example, the IBM Public License, Apple Public License, and Netscape Public License were created to satisfy the requirements of their businesses. However, consumers want licenses to adapt to their requirements, and the usual way to do this is to remove restrictions. For instance, the more liberal 3-clause and 2-clause versions of the BSD license replaced the original 4-clause BSD license. Over time, some licenses become increasingly restrictive. For instance, hardware locks and digital rights management were removed from the General Public License (GPL) version 2, resulting in the GPL v3[66]. In other cases, licenses are altered by external factors. For example, Mozilla's license evolved from NPL to MPL v1.1 due to the open-source community's disagreement with numerous Netscape Public License (NPL) terms. As seen by its evolution, the organization that controls the rights to Mozilla, the Mozilla Foundation, was eager to answer customer complaints.

Five Examples of How the Evolution of Licensing Affects Software Usage:-

Case 1: - OpenBSD IPFilter replacement.

The author of the OpenBSD-compatible firewall software IPFilter added a new line to each file's license declaration in 2001, which became part of the IPFilter license. While OpenBSD developers perceived this statement as a new requirement that violated the license's terms, the author stated that it defined its requirements. OpenBSD developers decided to use a new implementation based on OpenBSD instead of IPFilter[67]. If the license for the FOSS system is altered, users might no longer be able to reuse the software.

Case 2: - Java

Java had license problems in November 2006, restricting its availability in Linux distributions. Java 5.0 is released under the GPL v2 with the CLASSPATH exception by Sun Microsystems in partnership with the Free Software Foundation (FSF). With this modification, Java could now be updated and modified without Sun's intervention. Additionally, Java programs may

---

[66] Supranote65

[67] D. Hartmeier. Design and Performance of the OpenBSD Stateful Packet Filter. www.benzedrine.cx/pf-paper.html . Accessed Sept. 2023.

now be issued under any license as long as they meet the requirements for the CLASSPATH exemption[68].

Case 3: - Mono

Novell developed the Mono framework to enable the .Net API across many operating systems. Mono developers modified the license from GPL v2 to MIT/X11, which permits its use with systems licensed under any commercial or FOSS license. This modification might expand the FOSS system's contributor community[69].

Case 4:- QT

QT is a library of GUI widgets, originally developed by Toltec, bought by Nokia in 2008. QT was first released under a non-open source but free license, called the FreeQT License, and a commercial license. The foundation of KDE, the desktop environment for Unix-based systems, was QT. Richard Stallman was among those who opposed the foundation of a significant open-source system, which was a non-open-source library. QT v2.0 was released under a new Q Public License to address these problems. Despite the Open Source Initiative's approval, the FSF determined that the Q Public License was incompatible with the GPL[70]. The Free Software Foundation (FSF) claimed that using QT violates their license because many KDE applications are licensed under the GPL (although the KDE project disagreed). As a result, the Harmony project was launched to develop a QT-free alternative to KDE, while the GNOME project was founded as a substitute QT with a GPL-licensed version. Toltec modified the QT v3 license to the GPL v2. The Harmony Project was discontinued because it was deemed unnecessary. Following Nokia's acquisition of Toltec, QT v4.6 was released under a dual LGPL v2.1 and GPL v3 license.

A competing system may be abandoned if the license for a free and open-source software system is changed to one more permissive.

Case 5: - MySQL.

MySQL AB switched from LGPL v2.1 to GPL v2 licensing for its client libraries in 2004. The purpose of this modification was to stop businesses from using proprietary product libraries without purchasing a commercial license. Unfortunately, it also had unforeseen consequences:

---

[68] D. M. Germ´an and J. M. Gonz´alez-Barahona. An empirical study of the reuse of software licensed under the GNU General Public License. In *Proceedings of the International Open Source Systems Conference (OSS'09)*, pages 185–198. Springer, 2009.

[69] http://mono-project.com

[70] *Open source development | open source license | qt*. (n.d.). Retrieved May 28, 2024, from https://www.qt.io/download-open-source

PHP is licensed under a different license than GPL v2, so PHP computers could no longer connect to MySQL. MySQL added the MySQL FOSS License Exception to the GPL v2 to address this issue[71].

A FOSS system's authorized users may have unexpected or unwanted effects if its license is changed.

Therefore, license modifications and their possible repercussions should be known to developers and their organizations. Numerous developers are encouraged to contribute to FOSS development, and these developers may purposefully or unintentionally alter license declarations. As a result, a method for examining modifications to source code licensing is required.

3.2 CURRENT STATUS OF OSS IN INTERNATIONAL LEVEL

Over the past ten years, Open Source Software (often known as "OSS") has become increasingly popular in the software industry. From the 1970s[72], its roots in early OSS in US academics broke into the mainstream in the 1990s, and it has continued to gain traction in the 2000s. You may quickly obtain OSS modules from websites such as sourceforge.com. The current generational shift in the software industry is best illustrated by the announcement in July 2009 that Google is developing its System Chrome browser into an entire operating system[73]. This shift from the traditional "software as a license" on the PC at home or in the server room at the office to remote, service-based computing that embraces these Internet-enabled tools is similar to that of OSS. These techniques include virtualization, software-oriented architecture, cloud computing, and software as a service (SaaS).

This change is another catalyst for OSS, as is the beginning of unfavourable economic conditions: The introduction of unfavourable economic conditions has added to the appeal of OSS by increasing competitive pressure to innovate and cut costs. The notion that the "eco-community" (or "bazaar") approach to software development is at least as innovative as more structured "cathedrals" and that OSS benefits from lower costs than traditional proprietary software has also become more acute as a result. This has also led to "tipping point" acceptance, which holds that OSS has finally reached a tipping point. In its November 2008 study of 300

---

[71] D. M. Germ´an and J. M. Gonz´alez-Barahona. An empirical study of the reuse of software licensed under the GNU General Public License. In *Proceedings of the International Open Source Systems Conference (OSS'09)*, pages 185–198. Springer, 2009.

[72] Kemp, R. (2009). Current developments in Open Source Software. *Computer Law and Security Report/Computer Law & Security Report*, *25*(6), 569–582. https://doi.org/10.1016/j.clsr.2009.09.009

[73] Ibid

OSS users in the private sector, IT research firm Gartner discovered that 85% of them were utilizing OSS and that the rest Year, 15%, all had plans to do so within the and in the public sector, HMG has committed to accelerating use of OSS.

The OSS model is famous for various reasons, and as indicated at the outset, it now has a unique set of advantages. A community of programmers that strive to continuously improve the code they work with and offer bug patches and additional features without charge often forms the basis of an OSS ecosystem for many projects. Large open-source projects typically use strict peer review processes to organize all contributions into a stable, consistent, and cohesive final output. Therefore, code used in an open-source software project (OSS) can be at least as high-quality as software developed for profit. Since the source code is easily accessible, OSS can be modified to function with newly released hardware. Both consumers and developers know that the obsolescence of the original hardware platform will not cause the Open Source Software (OSS) product to become obsolete, unlike proprietary software that might do so if maintaining a version for the outdated platform becomes unprofitable. The length of the product cycle, which is the interval between the beginning of the design and the initial client availability, is being shortened by competitive forces. Pre-made open-source software components (OSS) expedite development and free up internal resources to create higher-level software that gives a competitive edge, especially for routine, lower-level tasks. This trend, becoming more apparent in consumer electronics, applies to various industry sectors where technology transforms business. Finally, OSS can be introduced to emerging nations to help create a local software economy. As a result, governments in countries like South Africa, Brazil, Russia, India, and China are increasingly adopting OSS.

There are hundreds of open source software licenses (OSS) in use, ranging from the invasive "copyleft" GPL to brief licenses with almost no specified terms. These licenses differ significantly in length, clarity, intent, and legal effect. Identifying the OSS in question and the license terms under which it is made available are solid places to start. From there, you can determine whether the license has any additional terms. A top-tier OSS service provider offers a frequently updated table with the twenty most common OSS licenses in use and an estimate of their popularity.

Development communities for open source software (OSS) have become a strong contender for replacing commercial knowledge-based collaboration projects. OSS projects have expanded quickly in recent years, with some directly competing with proprietary or "closed" software alternatives. The Apache Web server, which processes 70% of all Internet requests

for Web pages, and the Linux operating system are two of the most well-known open-source software projects.

OSS is envisioned as the driving force behind creating communities that support open-source software and information freedom and closing the digital gap.

Over the past few years, open-source software (OSS) has drawn a lot of attention because, in addition to constant technological innovation, free and open-source projects are distinguished by creative coordination, solid interpersonal ties, and a flat organizational structure.

The technological revolution assisted emerging nations in achieving more significant economic development in a globalized economy. Over the past two decades, software, in particular, and information technology, in general, have been crucial to expanding contemporary economies. The rate of change has quickened almost exponentially, continuously driving the world economy's expansion. OSS is currently being promoted by governments everywhere for a variety of reasons.

The flexibility and openness of OSS fosters innovation and offers a unique chance to close the technology divide at reasonable prices. In recent years, OSS projects have drawn government organizations that operate inside walls and private partners.

OSS has become an intriguing phenomenon with notable effects in the software industry. Implementing OSS offers developing countries a unique opportunity to join the global ICT mainstream. It’s evolving development process, ideology, and knowledge standards present potential for governments and the private and public sectors[74].

Additionally, attempts have been made to comprehend the use of OSS in government sectors. Government incentives (direct subsidies) and export subsidies for software development and exporting are very high. Another significant inducement for PSs and OSS is the government procurement policies. Tax benefits and government-funded infrastructure projects are examples of indirect incentives.

A nation's economic activities are what primarily determine its level of development.

Many governments are using information technology as a tool for economic growth and employment creation. The government's policies are the only ones that determine how OSS and PSs are used and promoted inside the government and throughout the nation. In many developing countries, including most developed nations like the US and European countries,

[74] **Development and use of open source software in India.** / Sarma, Meera.
ICTs in developing countries. ed. / Bidit Dey; Karim Sorour; Raffaele Filieri. Basingstoke: Macmillan, 2015. p. 129-138.

OSS and PSs have co-existed for a long time. Debroy and Morris say, "Economic growth and development is often an elusive goal.[75]"

"Free software" for office applications first appeared in the United States in the 1980s. Technology adoption affects the industry primarily because of its expense and room for advancement. Many developing nations think that having proprietary software (PS) and open-source software (OSS) together promotes economic prosperity. Governments play a critical role in advancing technology through domestic laws and policies. Some governments promote open source software (OSS) for political purposes, disregarding the economy's and technology's demands. For example 2003, Brazil decided to use OSS compulsorily in all municipal governments. Some governments decide primarily for political reasons instead of analysing technological and economic needs[76].

The highest level of OSS is found in Brazil (12.9%), Kenya (12.3%), and Russia (12.8%)[77].

Turkey and India use OSS and PR in parallel. The cross-country analysis reveals increased use and promotion of OSS in developing countries and developed countries like Europe[78].

For instance, in September 2001, the European Parliament passed a resolution urging the European Commission and Member States to support software projects whose source code is available to the public. Throughout Europe, some governments have developed official strategies for adopting OSS. This involves thinking about passing laws requiring open-source software in government applications or, at the very least, giving them careful consideration as a viable substitute for proprietary software. This trend has been most pronounced in Europe, especially in France and Germany, among other industrialized nations. The primary EU tool for funding research in Europe is the Sixth Framework Program (FP6). 2003–2006, the project will distribute $17.5 billion, or 3.9% of the Union's overall budget (2001)[79].

A law about the usage of open standards and the accessibility of source code for government software was suggested by the French Parliament.

---

[75] Bibek Debroy and Julian Morris, *Open to Development: Open Source Software and Economic Development,* Rajiv Gandhi Foundation, http://www.rgfindia.com/rgf2/text/oss.pdf., 2.06.24

[76] Kd, R. (2007). Is the Future of Software Development in Open Source? Proprietary vs. Open Source Software: A Cross Country Analysis. *Journal of Intellectual Property Rights*, *12*(2). http://nopr.niscair.res.in/bitstream/123456789/267/1/JIPR%2012%282%29%20%282007%29%20199-211.pdf

[77] Ibid

[78] Ibid

[79] The work program is available at: ftp://ftp.cordis.lu/pub/ist/docs/wp2003-04_final.pdf, 05.02.2007, Kd, R. (2007). Is the Future of Software Development in Open Source? Proprietary vs. Open

A measure under progress in Italy requires all government departments to prioritize open-source software (OSS). At the same time, a bill in Spain mandates regional governments to support and encourage the use of open-source technologies. The Extremadura district administration in Spain implemented a strategy in April 2002 to migrate all government, commercial, and residential computer systems to Linux and Open Source Software (OSS) programs.

The UK government has established a policy to take into account both proprietary and open-source solutions when procuring IT; to use products that support open standards and specifications in all future IT development; to investigate the possibility of using open source software (OSS) as the default exploitation route for government-funded research and development software; and to consider obtaining full rights to bespoke and customized software code for proprietary software it procures. According to the EU-sponsored FLOSS survey, OSS is used by 43.7% of German businesses and 31.5% of British companies.

In Europe, it is clear that several governments are promoting OSS[80]. According to a European Internet monitoring outfit, Firefox's market share has increased dramatically in Europe during the past several months. The percentage of people using Firefox was 22.6% in Finland, 21.46% in Germany, 14.9% in Poland, 12.4% in France, and 10.7% in the United Kingdom. Europe accounts for about thirty percent of the world's IT services market[81].

Any endeavour to be sustainable over the long run must be developed with capital and profits. Two years after its hugely successful initial public offering (IPO) in December 1999, the American open-source company VA Linux Systems laid off about 25% of its workforce.

One of the principal backers of OSS is the European Union. The government should refrain from requiring specific products or technology to make purchases or provide policy support. Allow the customer to select the product that best suits his needs and work. The e-Europe program, launched in December 1999, has contributed to a rise in computer literacy throughout Europe, which will benefit the market for computers and software going forward.

Because security is a problem, China encourages independent software development. The Chinese Academy of Sciences received funding from the Ministry of Information Industry in 2000 to market an operating system created by Red Flag Linux. The Beijing Software Industry Productivity Center was founded to enhance regional GNU/Linux distributions. Microsoft and the Chinese government reached a deal in 2002 that limited the usage of Windows source code.

---

[80]Source Software: A Cross Country Analysis. *Journal of Intellectual Property Rights, 12*(2). http://nopr.niscair.res.in/bitstream/123456789/267/1/JIPR%2012%282%29%20%282007%29%20199-211.pdf

[81] Ibid

Nonetheless, the absence of e-governance, user inexperience, and government rules impacting PC and software sales mean the Chinese software business is still in its infancy. Microsoft's profits in China are less than 5% of what they would have been if customers had paid for the software in newly sold PCs due to the high levels of piracy and counterfeiting.

Today, the software industry generates $300 billion worldwide yearly[82]. In addition to its critical role in infrastructure support, software development, security, and research, open-source software (OSS) has been shown to negatively affect sensitive and security-focused groups' abilities and increase the Department of Defences' susceptibility to cyber- attacks. Open source software (OSS) is still thriving, but it must now contend with the hyper-clouds, who want to turn it into a commodity or impose a new wave of disruption[83].

The first "closed source software" to be disrupted by open source software (OSS) started with databases like MySQL and operating systems like Linux, where the developer community was a significant contributor to the code base. After that, it was just a matter of time until its use exploded across industries. It finally got a substantial lift from the tech behemoths of the 2000s, like Microsoft and JFrog clients Oracle and Salesforce.

Because OSS encourages organic user adoption, it lowers customer acquisition costs and is well-positioned to encourage more excellent natural conversion. Another is a change in how code is contributed: rather than solely driven by the community, development is mainly driven by corporations.

Many vendors actively participating in the developer community show that open-source software (OSS) is now a fundamental component of every major organization's software strategy. From the vendor's perspective, this is particularly evident in the traditional sales cycle due to the growing involvement of developers in the software purchasing process. To achieve higher developer utilization (users and downloads) and a stronger value-market fit (commercial value, price), indicative of a more natural conversion to commercial value-add, this evolution has led vendors to identify better product-market fits[84].

Businesses from all industries are implementing OSS to support innovation and expansion.

---

[82] Kd, R. (2007). Is the Future of Software Development in Open Source? Proprietary vs. Open Source Software: A Cross Country Analysis. *Journal of Intellectual Property Rights*, *12*(2). http://nopr.niscair.res.in/bitstream/123456789/267/1/JIPR%2012%282%29%20%282007%29%20199-211.pdf

[83] Ibid

[84] Cohen, O. (2023, August 17). 7 Takeaways on the state and future of OSS disruption. *Forbes*. https://www.forbes.com/sites/forbesbusinessdevelopmentcouncil/2023/08/16/7-takeaways-on-the-state-and-future-of-oss-disruption/?sh=54fbf5b66dba

It should be clear by now how commonplace OSS has grown among tech-savvy companies and software vendors. The tech behemoths have aggressively acquired necessary assets and actively participated in the developer community. With these behemoths, the adage "the rich keep getting richer" is undoubtedly true. Perhaps none is more well-known than IBM, which offers products and services today, including StackRox, JBoss, MetaMatrix, Amentra, Identyx, Qumranet, Makara, and Gluster.

In recent years, OSS enterprises have been valued at premiums.

Any M&A transaction is primarily motivated by the acquiring company's belief that there is still more value to be extracted from the target company. In recent years, OSS companies have been valued at extremely high levels, which has continued. The most notable transactions involving OSS companies occurred in 2018 and 2019 when JFrog customers RedHat, GitHub, and Mulesoft were acquired[85].

There are several open-source revenue models.

For many years, OSS providers have been experimenting with various revenue streams. More recently, these have come together to form a "business model playbook," reflecting the most well-known and successful of these strategies: professional services add-ons. Using an OSS tool to encourage the use of a nearby commercial tool, developing an open-core model with proprietary features available as paid extensions, using cloud and SaaS hosting, and developing API marketplaces with pre-integrated solutions are a few noteworthy methods[86].

The most significant danger facing today's commercial OSS players is the hyper-clouds.

Utilizing computing and managed cloud services is the primary way hyper-cloud businesses are growing, and OSS is a significant stimulant for activity in their larger ecosystems. Consider software firms like Confluent, MongoDB, Elastic, and Redis, which are JFrog clients and rely on OSS usage as a component of a broader monetization strategy.

Let's now consider AWS, Azure, and Google Cloud, which are in an excellent position to profit because the OSS offerings of the previously mentioned companies are hosted on their clouds. They have started to service-wrap these OSS projects and provide them as OSS-aaS, profiting (or perhaps even stealing?) from this monetization. The most significant danger we are currently facing is this commercial OSS player today without a meaningful path to monetization or commercial conversion, their sustainability hangs in the balance[87].

---

[85] Ibid
[86] Ibid
[87] Suprantote84

3.3 CURRENT STATUS OF OSS IN INDIA

Over the years, proprietary software has dominated India's information technology (IT) environment, impacting many technical advancements. Open source software (OSS) is a beneficial alternative to proprietary software that many government organizations and software enterprises use. The spread of open source software (OSS) in India has resulted in diverse technological discourses and the production of available resources to all, generated by communities for the betterment of society.

One of the most astounding technological advancements of the past two decades has been the explosive global expansion of Free and Open Source Software (FOSS). Nowadays, FOSS powers most digital experiences; in India, FOSS powers over 85% of the Internet. FOSS is essential for large organizations like the State Bank of India, IRCTC, and the courts to expand their operations and offer millions of people quick and effective digital services. FOSS provides organizations access to a worldwide talent pool and the tools necessary to produce scalable, safe, and dependable software, democratizing technology and fostering rapid innovation. Promoting free and open-source software is in India's best interests as it will contribute to the country's scientific independence and technology.

The initial attempt of the government of India to promote open source has mainly involved adopting Linux-based operating systems and open document formats[88]. However, it failed because governments couldn't build better consumer products than corporations or open-source communities.

Current Situation of FOSS Developers: a significant portion of this ecosystem comprises Indian developers. India ranked third globally in 2021, after the US (13.5 million) and China (7.6 million), with over 7.2 million of its 73 million users coming from the country[89]. However, the rate of growth of the Indian developer base is higher than that of China and the US, with estimates of close to 40% in 2020–21. By 2023, GitHub anticipates that 10 million Indian developers will use its platform[90]. The fact that millions of Indian coders are involved in the global open-source ecosystem is encouraging and could provide India with a competitive edge in high-tech geopolitics.

---

[88] *The Open-Source Mission for India*. (n.d.). Drishti IAS. https://www.drishtiias.com/daily-news-editorials/the-open-source-mission-for-india, accessed on 30 May 2024

[89] Supra note84

[90] Ibid

By 2025, OSS is anticipated to generate $350-400 billion in revenue and contribute to cost reduction, increased security, and improved business software dependability[91]. The Information Technology Bill mandates that government agencies enhance the security and reliability of business software and give preference to open-source software (OSS) alternatives when making procurement decisions.

When OSS is adopted by private citizens, governmental agencies, commercial enterprises, and academic institutions, its impact is realized. Governmental agencies in India have established guidelines for creating and applying Open Source Software. India's software industry is expanding, which indicates that the country is doing well in the international software market and that additional efforts are needed to maximize software development potential. The primary driver behind adopting open source software in India is decreased expenses.

These include (a) licensing fees, license acquisition, and software acquisition costs. When commercial software is updated to newer versions, additional fees are associated with license upgrades. (b) Service charges are the expenses related to obtaining internal and external assistance for software upkeep. This expense is especially noteworthy for businesses purchasing and utilizing specialized software for the first time. (c) The distribution expenses are incurred when software is distributed within the company since specific licenses restrict software distribution[92].

The area where open-source software (OSS) is quite advantageous, as it allows for easy sharing without copyright restrictions. Performance optimization is another crucial component. Organizations that employ open-source software (OSS) do so for ease of operation and upkeep. Scalability is an additional consideration. Organizations utilize open-source software (OSS) to expand their operations, which is made possible by the software's easy distribution and lack of copyright. Organizations are drawn to Open Source Software (OSS) for its security characteristics, especially desktop usage[93]. Lastly, another element that encourages organizations in emerging economies like India to use OSS is the need for vendor lock-in or the idea that they are not restricted to working with a single vendor. At the 'micro-level,' grassroots OSS groups could be created; for example, the "IT@School" project in Kerala suggests that adopting OSS is feasible and productive[94]. The project involved training teachers

[91] India embraces open-source software to boost tech industry amid layoffs - India News News (wionews.com)

[92] *Open source development | open source license | qt*. (n.d.). Accessed on May 28, 2024, from https://www.qt.io/download-open-source

[93] De, R. (2009). Economic impact of free and open source software: A study in India. *Interop, Mumbai, October*, 7–9.

[94] Supra note 92 page at 133-134

and students to make small developer groups use OSS and develop OSS further. The project resulted in education benefits apart from cost savings.

Open source software (OSS) can create value in the software industry by improving IT skills and encouraging its adoption in the private sector. This could result in acquiring fundamental skills and raising the stature of the Indian IT sector. On a larger scale, state localization, policy frameworks, advocacy, and the availability of IT skills can all be used to introduce open-source software. Government agencies must consider OSS-based technologies as feasible substitutes when making software acquisition and licensing requests.

The IT bill aims to encourage the use of open-source software in all facets of business and government. It was introduced in the Indian parliament in late 2022[95]. The measure promotes open standards for data interchange and mandates that government entities prefer open-source software solutions when making procurement decisions. Given that India's economy is among the top five in the world, OSS has plenty of prospects.

Organizations worldwide have embraced creative alternative methods to optimize costs by investigating avenues of Open-Source Software (OSS) in accordance with the Ministry of Electronics and Information Technology (MeitY) of the Indian Government's 2021 policy titled "Adoption of Open-Source Software for Government of India[96]."

Large and small businesses realize that open-source platforms may meet most operational requirements with dependability. Enterprises like SUSE, headquartered in Germany, invest significantly in providing new technology and solutions to Indian enterprises.

Manu Dhir, General Manager of SUSE India, emphasizes the need for India to promote an open-source economy by incentivizing developers and organizations to develop a sustainable, home-grown OSS innovation. He told WION, "India must now promote an open-source economy by incentivizing developers and organizations to develop an open-source software ecosystem. The incentivization to the developers will also aid the government's flagship Digital India program and help India become a major tech-oriented economy in the world[97]."

Open-source solutions are starting to be used by numerous more Indian firms. Indian banks have utilized Open Source Software (OSS) for their core banking systems for several years. This has allowed the banks to lower their IT expenses while enhancing security and dependability. Kerala Police, the state's law enforcement organization, is another example of a

[95] Supra note 92 page at 15
[96] Supra note 84
[97] Supra note 84

government body utilizing OSS. SUSE's solutions laid the foundation for Kerala Police's software needs[98].

OOS can be an additional choice, or as many would say, a lifesaver, even though many Indians in the IT field have been laid off by multinational behemoths like Amazon, Meta, and Accenture due to apparent over-hiring during the pandemic. Global IT behemoths reportedly released 38,000 workers in March 2023, many of whom were Indian. To support India's growth and innovation as a global leader in open-source innovation, businesses such as SUSE plan to keep employing it.

"We intend to continue hiring in India to fuel our growth and innovation, i.e., we did not over-hire during the pandemic," stated Manu Dhir[99].

To take advantage of Indian talent, the corporation established a "Centre of Excellence" in Bengaluru, India, last year. "All SUSE departments are included in our CoE, particularly emphasizing engineering and technical support. To support our expansion and creativity, we plan to keep hiring people in India," Manu Dhir said. In the face of global corporate layoffs, India's bet on open-source software may hold the key to growing its tech sector[100].

Over the years, open source software (OSS) has grown significantly in India thanks to corporate involvement, government laws, community involvement, and educational activities. Government initiatives include the National Policy on Open Standards for e-Government (2010) and the Policy on Adoption of Open Source Software for the Government of India (2015). The Digital India Campaign uses open-source platforms like MyGov, DigiLocker, and UMANG to deliver government services electronically and strongly emphasizes digital infrastructure and literacy. State-level initiatives such as Kerala and Tamil Nadu have implemented policies encouraging Open and Shared Knowledge (OSS) in education and governance.

Corporate adopters include start-ups and SMEs, multinational IT behemoths like Google, Microsoft, and IBM, and Indian IT behemoths like Infosys, Wipro, and Tata Consultancy Services (TCS). Participation in academia and the community involves membership in open-source groups like ILUG. FSFI and academic institutions incorporating OSS into their course offerings. Open-source events offer forums for industry executives, enthusiasts, and developers to interact and present open-source initiatives.

---

[98] Kd, R. (2007). Is the Future of Software Development in Open Source? Proprietary vs. Open Source Software: A Cross Country Analysis. *Journal of Intellectual Property Rights*, *12*(2). http://nopr.niscair.res.in/bitstream/123456789/267/1/JIPR%2012%282%29%20%282007%29%20199-211.pdf

[99] India embraces open-source software to boost tech industry amid layoffs - India News News (wionews.com)

[100] Ibid

Aadhaar, Bharat Operating System Solutions (BOSS), and the National Repository of Open Educational Resources (NROER) are essential initiatives and platforms. In the future, OSS will likely be used more by the government, integrated into new technologies like block chain, IoT, and artificial intelligence, and given more attention to the educational system. It will also likely be used more responsibly and ethically.

On a comparison of the factors mentioned above and the scenario, we can see that even though there are similarities all over the globe concerning the usage and licensing of open-source software, there are specific differences concerning the licensing and legal protection of the same in the country of India which are to be discussed briefly in the following chapter.

CHAPTER 4

LEGAL ISSUES AND CHALLENGES FACED BY OPEN SOURCE LICENSING

4 INTRODUCTION

Chapter 3 discussed the different status of Open Source Licensing in the international and national spheres. This being the case, there are considerable challenges to protecting OSS in a developing country like India. This chapter will briefly discuss the legal issues and challenges faced by OSS and the licensing of the same in India.

The Licensor may have needed to learn who the licensees are. To differing degrees, each of these licenses presents the licensed code and invites adoption and use, subject to the conditions of each license[101]. With these open-source and free software licenses, licensees are not required to notify the Licensor of their agreement or take any other deliberate step to alert them that they have done so[102].

Open source software (OSS) licenses allow free sublicensing of the licensor's work to other licensees, but may strain relationships between the original licensee and subsequent licensees. Many licenses do not require affirmative action before granting access to the licensed work, but some do. The use of the licensed work is subject to accepting the license's conditions, which restricts rights and minimizes affirmative responsibilities. OSS also has liabilities, such as copyright notices and the availability of specific codes. Legal counsel may be needed to verify license conditions and ensure all required notices are included. FSOS communities have many licenses with inconsistent conditions, with the most popular being the GPL. FSOS licenses can be diverse and have different problems depending on the developer's or company's point of contact. There has been a surge in interest in non-proprietary software licenses (FOSS), which include both Free Software and Open Source Software. An increasing number of software projects use FOSS licenses, and legal research has examined the enforceability and validity of current licenses to protect the emerging open-source software market.

---

[101]. *Andrew M. St. Laurent*., chapter 6- legal impacts of Open source and free software licensing, Understanding Open Source and Free Software Licensing, page 147

[102] Ibid

4.1 ENFORCEABILITY OF OSS IN CONTRACT

It is a controversial question to determine whether an FSOS license constitutes an enforceable contract. The enforcement of FSOS licenses largely depends on the general contract law, which is the legal framework under which courts consistently maintain the enforceability of shrink-wrap, online, and ordinary standard form agreements[103]. While some FSOS licenses need to clarify whether a contract is a transactional goal, others are not built to establish contractual acquiescence[104]. Divergent views exist in the community; some regard FSOS licenses as limiting copyright notices that are not contractual, while others view them as any other type of licensing agreement.

The assertion that a partial release of copyright (or other intellectual property rights) can be enforced by both the grantor (the licensor who wishes to impose restrictive terms) and the release-recipient (the licensee who uses the program) determines whether an FSOS license is enforceable[105]. The distinction between classifying the licenses as contractual or non-contractual affects the remedies issue.

FSOS licenses can be found in two distinct contexts: in actual transactions or software releases and as simple standard forms that others can use[106]. One can only assess whether or not the license became a contract component in this scenario. Because the licensor retains property rights in the software and conditionally releases or authorizes the licensee to utilize part of those rights, the idea that FSOS licenses can be regarded as enforceable, non-contractual restrictive notices or conditional releases is predicated on this[107].

This is a mere promise not to sue if the licensee uses the property in a particular way. Egan Moglen, General Counsel for FSF, is a leading proponent of this view, stating that a license is a unilateral permission to use someone else's property. At the same time, a contract is an exchange of obligations[108].

The claim that an FSOS license is not a contract is supported by FSOS proponents for two pragmatic reasons: it allows for the variety of informal ways FSOS software is distributed and

---

[103] Raymond T Nimmer, chapter 11, legal issues in open source and free software distribution, The Law of Computer Technology (1997, 2005 Supp.) page 16

[104] Ibid

[105] Ibid

[106] Ibid page 18

[107] Ibid

[108] Ibid

removes the possibility that the licensee may be entitled to contractual remedies. An infringement lawsuit is the sole way to pursue a breach of a "non-contractual" license[109].

The most contentious FSOS licenses, the GPL (and LGPL), are the subject of Moglen's remarks because their copyleft clauses jeopardize businesses' trade secrets that may utilize GPL software in their goods. Some FSOS standard forms, like the Open Software License, expressly anticipate a contractual relationship and stipulate that by using any of the rights provided to you in Section 1, you agree to this license and all of its terms and conditions[110].

Specific FSOS licenses, like the BSD license, permit the distribution and use of the program in source and binary formats, with or without modification, and concentrate on non-contractual ideas[111]. The permit requires redistribution of the source code to preserve the copyright notice, conditions, and disclaimer. It also prohibits the notice from being reproduced in documentation and promoting or endorsing software-derived goods without express written consent[112]. The program is given "as is," and any direct, indirect, incidental, special, exemplary, or consequential damages resulting from using the program are not covered by the copyright owners or contributors[113].

Standard form language is not promissory because the contractual nature of the relationship is contingent upon the parties' circumstances and actions. The licensing terms are unmistakably a part of the parties' agreement, and the stated conditions exclude any guarantees and remedies while requiring the user to behave affirmatively[114].

Property owners may conditionally waive the enforcement of their rights without needing to get a contractual agreement to the limiting terms, according to the principle that a waiver or notice is enforceable without a contract. In intellectual property law, however, more case law needs to be used to support this idea[115]. In certain situations, a substantial body of case law refuses to uphold such releases' restrictive provisions[116].

Suppose the recipient properly relies on the non-contractual grant until the grant is revoked and the effects of harmful reliance are mitigated[117]. In that case, copyright notices extending the recipient's rights beyond what would otherwise exist will likely be enforced. In this context,

---

[109] Supra note 103 page 20
[110] Ibid
[111] Supra note 103 page 22
[112] Ibid
[113] Ibid
[114] Ibid
[115] Ibid
[116] See Bobbs-Merrill Co. v. Strauss et al., 210 U.S. 339 (1908) (restrictive resale notice not enforceable when not contractual);
[117] Supra note 103 page 23

enforceability refers to the licensee's ability to do so. Although the enforceability of restricted waivers has been tested in most cases, the general premise still holds.

A non-contractual notice or waiver may not impose restrictive guidelines not established by a contractual transaction between the parties[118]. The past unwillingness of courts to allow the anti-competitive use of a patent or copyright to encompass things or conduct outside the scope of the right lends credence to this view.

To fully comprehend the issue, consider whether a non-FSOS ("proprietary") publisher could distribute software under a non-contractual license that restricts use, eliminates warranties, only permits redistribution under specified circumstances, and applies those conditions to all transferees—initial and subsequent. Other licensors can do the same if FSOS licenses are allowed without a contract[119].

Numerous FSOS standard forms have language that anticipates contractual arrangements, including what constitutes acceptance of the license and the effect of the "agreement." This is a standard contract writing technique[120]. However, it is frequently combined with additional phrases that do not specifically mention promissory duties. Suppose someone inquires as to whether an FSOS license is a component of a contract. In that case, the guidelines for constructing an FSOS contract are the same as those that apply to any other standard form or contractual license utilized in a commercial connection. Similar requirements are outlined in both UCITA and the Restatement (Second) of Contracts: the licensee must indicate consent to the FSOS license after having a chance to evaluate its terms for the form to describe the conditions of the contract and become a part of it[121].

Shrink-wrap and online licenses that are appropriately provided to gain consent are regularly enforced by case law[122]. The same standards govern FSOS standard forms and share all the features of such permits, depending on their use. The FSOS license must be shown and accepted for it to be enforceable. Generally speaking, this means that the licensee must have reasonable knowledge that terms are being or will be presented, have the option to object after reviewing the terms, and still take actions that it reasonably believes will indicate its consent to the other party.

---

[118] Ibid

[119] Supranote103 page 25

[120] Ibid

[121] Supranote103page 27

[122] Supra note103 page 27

Although Specht claimed that giving clear notice of particular actions amounts to consent, the rules are more lenient than that. In *Register.com v. Verio, Inc*[123]., the Second Circuit emphasized the flexibility of contract doctrine in its approach. Even though the licensee was never prompted to click and express their explicit consent to the terms, the court decided that the licensing terms were nonetheless enforceable. Verio, however, is unable to use Specht's logic because the case concerned Verio regularly contacting Register's computers to view the conditions of the offer and obtain WHOIS data[124].

The idea of freely consenting to license terms changes depending on the situation and the environment. The GPL and LGPL, among other FSOS licenses, are designed to be used by several parties in a license chain, which makes references to "you," "the licensor," and "the licensee" frequently unclear[125]. Specific licenses mandate that the licensee either automatically distributes the software under the license terms or passes them through. This means that when a transferee does not deal directly with the business or individual who created the software, there are three ways in which the licensing conditions may directly affect the licensee[126].

The GPL (and LGPL), according to Eben Moglen, General Counsel of the Free Software Foundation (FSOS), is a non-contractual release or license instead of a contract. Many within the FSOS communities, however, see it as a contract. It is reasonable to assume from reading the GPL that it contains language consistent with both a non-contractual release (restrictive notice) and an attempted contract. How the standard form is employed in a transaction determines the relationship that is formed, not the terms of the form. The test of whether a relationship is formed, if any, looks at the actual transactional usage of the standard form.

The GPL and LGPL's respective terminology results in incredibly conflicting text. The GPL includes wording consistent to create a contract and a document independent of the contract's terms. Overall, the majority of the terminology is contractual. Furthermore, it is argued that more than a non-contractual waiver is intended due to the length and complexity of the GPL and LGPL and the creation of affirmative responsibilities for licensees. Indeed, irrevocability for the GPL can only be achieved if the form as employed becomes a contractual duty if "free software" demands an irreversible license.

---

[123] Register.com v. Verio, Inc., 356 F.3d 393 (2nd Cir. 2004).

[124] Raymond T Nimmer, chapter 11, legal issues in open source and free software distribution, The Law of Computer Technology (1997, 2005 Supp.) page 14

[125] Supranote103 page30

[126] Ibid

The language in Section 6 of the GPL states that "each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute, or modify the Program subject to these terms and conditions," is the most explicit indication of a potential non-contractual basis[127]. It is not your responsibility to enforce compliance with this license on behalf of other parties. The license cannot be based on a contractual relationship unless the remote transferee consents to the terms of the license concerning the relationship with the copyright owner. This language contemplates an attempted pass-through license from the copyright owner to a remote transferee. This GPL pass-through is not contractual if the wording accurately describes the procedure in a specific transaction.

GPL makes use of more than just this pass-through idea. The licensee is required to, for instance, "cause any work that [it] distributes or publishes [containing the Program] to be licensed... under the terms of this License[128]," according to Section 1(b)[129]. This assumes that the license terms will be applied to the following transfer due to some action taken by the parties. Comparably, Section 4 goes on to say that "parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance," in addition to stating that "rights under the License shall terminate if the Licensee copies, modifies, or distributes other than in compliance with the License[130]." This implies that the latter transferees have rights from the licensee, presumably granted by contract, and cannot be revoked.

Although this is not always the case, the General Public License (GPL) may assume the formation of a contractual relationship[131]. The licensee must agree to the conditions after being given a chance to examine them for the GPL (or LGP) terms to become an enforceable contract. Assent by conduct is permitted under modern contract law, and terms of shrink-wrap and online licenses are consistently upheld by case law where two conditions are met: 1) the terms of the license were made available in a way that gave the licensee a chance to study them, and 2) conduct following that opportunity implies assent[132].

---

[127] Supranote124 page 33

[128] GPL u/s 6

[129] The reference to causing the “terms of this License” to apply could be referring to the GPL as a written template (e.g., B licenses this under the GPL template) or to the actual agreement between the transferor and the copyright owner by which the first transferee obtained rights in the software. If the latter case governs, GPL is contemplating a partial assignment of the license to subsequent parties. As with any other contract, the effectiveness of an assignment agreement would be judged under contract law.

[130] Ibid

[131] Ibid

[132] Supraanote124 page 36

To the extent that GPL terms apply to a contract, it is pointless to inquire whether the software has been "GPL." That expression indicates an attempt to license one's software under the GPL. Such a ruling must be more conclusive regarding whether a contract was made and consent received. The legal question is whether the license is given in a way that forms a contract and whether the transferee's conduct signifies consent to that contract[133].

Regarding this matter, three broad observations can be made: First off, there is a lot of variation in how open source and free software are distributed, and in certain situations, at least, it's possible that the conditions don't turn into legally binding agreements. Second, as happened in Register.com, the licensee had reason to know the impact of its conduct in this case, as part of the usage of trade or previous interactions with the licensor, so when assessing the enforceability of GPL in any context, it is essential to consider the extent to which the terms of GPL might become part of the agreement. A legally binding agreement is an agreement that is based on something like the usage of trade, course of dealing, or something similar.

Other requirements of contract law must be met by the parties' connection for a contract to be enforceable, including consideration, mutuality, and the requirement that any consent to terms come from a person with the power to bind (or estop) the organization against which contract terms are enforced. The terms of GPL may still be relevant under non-contractual analyses even if it fails to create an enforceable contract.

Subject to the conditions of each license, licensees may adopt and utilize licensed code through open-source and free software licenses. The licensor's work may be freely sublicensed to other licensees under these licenses, which do not demand any active action from licensees. Affirmative action is unnecessary to access licensed material under specific permits, such as the BSD, MIT, and Apache Licenses. Others, such as the GPL and LGPL, don't need this consent.

Acceptance of the license's terms is required to utilize the licensed material. In contrast to traditional forms of agreements, open source, and free software contracts primarily restrict the rights granted under the license, with relatively few positive responsibilities placed on licensees. Due to this feature, the enforceability of these licenses is shielded from potential challenges based on the need for thought or consent between parties.

Even the most permissive open-source licenses entail a minimal requirement to guarantee that legal consideration is given and that the permit creates an enforceable contract[134]. For instance,

---

[133] Ibid

[134] Supranote102 page 148-149

licensees under the MIT License are subject to this actual and non-onerous requirement, and breaching it would be considered a breach of the license agreement.

Arguments against mutual consent are rare and only occur in specific situations. According to the traditional contract interpretation, two parties have conferred and worked out a final agreement, which is formalized in a signed document. In such cases, any party's permission may be contested in one of two ways: first, by claiming that the other party misled them about a material fact related to the contract, the consenting party may claim that their consent was obtained through fraud[135].

Oral contracts are more prevalent but can be created without a signed document[136]. Software contracts are more appropriately governed by the "shrink-wrap" license, which regulates the use of commercial software[137]. The buyer is advised that opening the package and removing the shrink-wrap signifies acceptance of the license agreement. While some courts have supported establishing these terms in a contract, others have not. The degree to which the buyer knew or might have learned that the software was being delivered subject to a license and that the license's conditions would restrict the software's usage is a crucial distinction[138].

Offer and acceptance occur in a virtual environment where the license and the product are located. Seemingly insignificant details can determine the existence of a contract[139]. The contract, for instance, may be located on a website, but software stored on a physical medium may also give rise to similar problems. By clicking on a hyperlink, a user can get the license terms by downloading the program from the website or by seeing the terms on a page that displays the license terms[140]. Because the user is not compelled to agree to the conditions before accessing the licensed work but is at least made aware that the software is produced subject to a license, this "browse wrap" license may result in an enforceable contract.

An enforceable contract may or may not be created via a variation of the "clickwrap" and "browse wrap" licenses, in which consent is inferred from another action (often downloading the licensed program). Courts find it unsettling when affirmative consent is lacking because it appears unjust to enforce the terms of a contract when one side still needs to take concrete steps to support them[141].

---

[135] Ibid
[136] Ibid
[137] Ibid
[138] Ibid
[139] Ibid
[140] Supranote102 page 150
[141] Supranote101 page 150-151

Contracts recorded in electronic media have legal force thanks to model legislation like the Uniform Electronic Transactions Act (UETA) and E-Sign[142]. UETA and E-Sign do not change the standard state law governing contract interpretation. The Uniform Computer Information Transaction Act (UCITA) also mods ordinary state contract rules about software transactions. The goals of UCITA are to standardize the interpretation of contracts governing information transactions and to facilitate such transactions. It has only been embraced by two states, Maryland and Virginia; hence, it has yet to be generally accepted. Several states have passed anti-UCITA laws. UCITA must be further addressed because its influence is limited and unlikely to spread soon. The goals of UCITA and UETA are to offer consistent contract interpretation and to streamline information exchanges[143]. In proprietary software, implied and express guarantees are typically disclaimed and replaced with a restricted express warranty. However, these disclaimers are only helpful in a few states, causing issues for end users and developers using FSOS code. Some argue that licensees can fix qualitative issues without warranties, similar to automakers exempting transmissions from warranties. To address this, it's important to consider applicable laws, such as UCC Article 2, Article 2A, or UCITA, which acknowledge that "express warranties" can be produced by actions or words that become part of the parties' contract.[144]

A “savings” rationale underpins the MIT License and other open-source and free software licenses, maintaining their validity without express authorization[145]. These licenses limit the rights under the license rather than imposing positive obligations on licensees. These limitations, which can include the need to reprint the copyright and permission notice, can be simple or intricate.

One example is the GPL License, which does not impose the usual restrictions on most applications that use software licensed under the GPL. Users can install, use, and modify software licensed under the GPL without restriction. Unlike proprietary agreements, there are no restrictions on the quantity of installations or royalties paid in exchange for use. The GPL only applies if the user plans to distribute modified or original code versions[146].

Open source software (FOSS) is a valuable business strategy, but its legitimacy is often challenged by detractors who use legal means to contest its legitimacy. The disclaimer of

[142] Ibid
[143] Ibid
[144] Ibid
[145] Ibid
[146] Supranote101 page 150-151

warranties is a crucial component of open-source licensing, as developers cannot contribute without being held personally liable or forced to make a warranty. However, lawsuits have been filed against software companies, such as MySQL and NuSphere, for copyleft license noncompliance. Despite these challenges, FOSS remains a valuable tool for collaboration and productivity.

Versata Software Inc. V. Ameriprise Financial Inc & Ors[147]**.**

In 2013, Versata sued Ameriprise in Texas state court, alleging that Ameriprise breached a software license for Versata's Distribution Channel Management (DCM) software. The litigation involved two contracts, a Master License Agreement (MLA) and a patent infringement lawsuit. Versata argued that copyright law pre-empted Ameriprise's complaint for breach of contract. Ameriprise submitted the case to a federal court, where XimpleWare filed a federal lawsuit against Versata and Ameriprise for copyright infringement and breach of contract. In February 2015, Versata and XimpleWare reached an out-of-court settlement, but the federal court determined that Ameriprise's counterclaim was not pre-empted due to additional obligations outside the Copyright Act. The case was remanded to the state court.

In the Patent case, the court stated: "*Even if the original licensee—[here, Versata]—breaches its license for whatever reason, third-party customers of that original license retain the right to use XimpleWare's software so long as the customer does not itself breach the license by 'distributing' XimpleWare's software without satisfying [any] attendant conditions*[148]*."* In other words, if one party violates the license, it would not automatically terminate other licensees' rights who have complied with the terms of the license[149].

Mark Radcliffe, a licensing expert and partner at law firm DLA Piper, exclaims that *"The days of open source software free lunches are rapidly coming to an end, and that means enterprises that fail to stick to the terms of open source licenses can expect to be sued*[150]*."*

4.2 LEGAL ISSUES IN RELATION WITH IPR

Publishers of open-source software depend on intellectual property regulations to uphold their preferred development approach. Software developers have exclusive interests in their works thanks to various rights and protections collectively called "intellectual property," such as trade

[147] Versata Software Inc. V. Ameriprise Financial Inc & Ors**.** May 3, 2013 (the "Texas case"),
[148] Aadhisree Jadhav, Legal issues and compliance pertaining to open source software, dec 29, 2018, Open Source Software Legal Issues And Compliance (globalpatentfiling.com)
[149] Ibid
[150] Supranote147

secrets, copyrights, and patents. Worries have been raised about the potential violation of intellectual property rights if open-source software is utilized.

Third-party allegations of intellectual property infringement[151]: Individuals are prohibited from claiming intellectual property rights in open-source software under licenses like the GPL[152]. Software may have been provided as open-source by one person even though it may already be protected by intellectual property rights held by another business or person. Intellectual property rights holders and others who think the program is public domain may disagree. These people are known as third parties. For instance, A develops a novel technique for data sorting that is helpful for computer programs. After that, A submits a patent application, stating that the method is new. Subsequently, B, an independent programmer, creates a software application using the same data that A had noted in his application, which was still waiting[153]. Without knowing about A's patent application, B releases his software under the GPL for public use. A may file a patent infringement lawsuit against anyone using B's software if A's application is granted. Even though no one knew of the patent violation and that these users thought the software was open-source, they would still be held legally liable. The most famous case involving third-party infringement allegations is the SCO Group and IBM legal dispute.

Utilizing market competition to maintain low prices for the benefit of customers is one of the objectives of antitrust legislation. According to antitrust law, it is illegal to artificially lower a commodity's price in certain situations where doing so is likely to hurt consumers and competition in the long run. However, it has yet to be determined that an OSS platform's agreement to offer its license for free violates antitrust laws or is illegal.

How to control the use of OSS in your company[154]:

Adopt an OSS Policy[155]: If an organization decides to employ open-source software, it should specify how it will be used. This prevents future problems, which may be expensive and time-consuming.

Update as soon as possible[156]: If a defect is discovered in open-source software, it should be patched immediately. Additionally, it's critical to ensure that any apps that use the frameworks and the primary source code for these projects receive the necessary changes.

---

[151] The legal issues to think about when using open- source software, http://www.qlegal.qmul.ac.uk
[152] Ibid
[153] Ibid
[154] Ibid
[155] Ibid
[156] Ibid

Ensuring Quality: Selecting the appropriate program for the job when using open-source software rather than relying solely on familiarity is critical. Utilizing well-known software does not ensure excellence. The software should be selected with the assurance that all of the needs of the business may be satisfied in mind.

Forking[157] - The most remarkable thing about open-source software is that its source code may be altered and customized to suit the user's requirements. Because it will always have a link to the original software, forking allows the organization to keep track of any changes made to the open-source software.

Using Tools[158]: Automated tools like Jenkins or SOAR should be continuously integrated to monitor and manage security vulnerabilities in the OSS. With the increasing concern over cyber-security, these tools aid in identifying and resolving possible security vulnerabilities. The process is quicker and more secure with these automated technologies.

Your Company's Effect on the OSS Community

When developers use open-source software, they feel they contribute to the community. Ignoring this idea is a narrow-minded strategy that could harm the enterprise. For example, suppose someone on your team discovers a flaw in an open-source program, and they promptly address the issue in just your local copy without engaging with the community. The best action would be to use open-source bug-tracking software or bug reports and publish your solution to the larger community.

Control over copyrights and related patents raises ownership concerns in free and open-source software[159]. Control is granted by copyright law to the copyright owner, whether they are the original owner or the beneficiary of a transfer of ownership. Because the authorship-based system outlined in the Copyright Act and the horizontal image of software development promoted by open source are in direct opposition, copyright law is likely to create split or joint ownership in the open-ended world of free and open source software, with potentially disastrous results. Because contributions to FSOS software typically involve changes to already-existing code rather than distinct, expressive code introduced to a program, ownership of FSOS software is complicated. The scenario can get more complex due to copyright principles related to multi-party ownership, as there are three possible formats: one of three types of works: (1) a collaborative effort, (2) a derivative work, or (3) a joint work[160].

---

[157] ibid
[158] Ibid
[159] Supranote124 page 38
[160] Ibid

Co-writers of collaborative Work are joint tenants and have the equal right to grant a nonexclusive license for the job, subject to reporting earnings to the other authors. Essentially, neither co-author has complete and sole rights to the work[161]. If a downstream joint owner is present, they have the independent authority to grant licenses for the copyrighted work. That joint author may, however, assert the copyright even about expressions authored by the other co-author since they are co-owners of the complete work[162].

On the other hand, a derivative work expands upon an earlier work (which might belong to a different author) and produces a new, independent work that the new author owns. Still, it contains components from the earlier work with the first author's consent[163]. The party carrying out the new work is the copyright owner of the new elements in and of itself and has the right to manage the entire cohesive, copyrightable work. According to Section 103 of the Copyright Act, the creator of a derivative work is the copyright owner, even if the copyright owner is not required to grant permission to prepare derivative works. This copyright in the derivative work only applies to the parts of the work that the author of the derivative work developed, and it is independent of the underlying copyright[164].

The result is similar to a collaborative work or compilation. In all cases, authorship is selecting and arranging content into a new, cohesive work[165]. The author of the collective work is the one who chose and arranged the pieces that make up the expressive composite. However, if the work gathers works by many authors with their permission, ownership, and copyright of the individual authors' works will be retained, excluding any authorized compilations or edits[166].

The parties' intentions, the nature of the contributions, and the ways they are combined determine the differences between these three categories of works and the various forms of ownership they generate. Unless the parties asserting ownership contributed expression to the final work product, no co-ownership develops by operation of law. The potential complexity and level of fact-intensive investigation needed to determine ownership where community-wide development occurs is evident compared to the development paradigm supported by free and open-source software.[167]

---

[161] Supranote124 page 39
[162] Ibid
[163] Weismann v. Freeman, 868 F2d 1313 (2d Cir. 1989).
[164] Ibid
[165] Supranote124 page 42
[166] Ibid
[167] Ibid

Instead of collaborative authorship, fresh contributions are expected to produce derivative or collective works in many Free Software Operating Systems (FSOS) environments[168]. The form of some licenses, like the GPL and LGPL, suggests this: they do not declare any purpose to co-author with subsequent contributors, but they do account for program changes and community evolution. The writers' intentions at the time of the contributions determine whether or not a joint work exists. Later, alterations are more likely to be separately authored derivatives of the original work if the primary or initial author does not indicate they intend to become co-authors. Joint authorship does not necessitate co-authors working on the work simultaneously under copyright laws. It suffices that when they both create their expression, they wish for their contributions to be combined into integral or interdependent pieces of a single, cohesive whole. The idea of collaborative authorship differs from situations in which one person dominates the creative process and from those in which distinct works or forms of expression are created with no intention of combining them into a "collective work[169]."

The difference between joint and communal works depends on illusive intent problems when there isn't a clear contract. Two possible justifications for joint ownership seem to exist: first, in cases where the contributions of each author are inseparable, and second, in cases where the contributions are interdependent and each was written with the implied understanding that the result of the many contributions will be recognized as a single, indivisible work. [170]

The free software and open source software (FSOS) community, the idea of "copyleft" license provisions is widely accepted[171]. This is because it safeguards the user's ownership rights and, when combined with FSOS software, may affect the user's control over the software written entirely by it. The purpose of these restrictions is to protect the licensee's rights, as stated in the Preamble to the GPL. To that end, the licensee's right to transfer in any way they choose is curtailed by these provisions, which also mandate that any distribution be subject to the rights and restrictions outlined in the GPL alone[172].

People impacted or potentially impacted by the terms typically refer to the risk of "viral" license terms that reach out to infect their own separately developed software, as well as improper market leverage and misuse of copyright to control other people's works, even though proponents refer to such restrictions as creating "free" software and protecting rights[173]. The

[168] Supranote124 page 44
[169] Ibid
[170] Supranote124 page 45
[171] Ibid
[172] Preamble GPL Version2.0
[173] Supranote124 page 46

question should be framed in terms of how much FSOS software licenses affect transactions outside of the particular agreement or transaction with the initial licensee or aim to affect such transactions. What impact do these clauses have on licenses obtained after the initial FSOS licensor? Pure unrestricted licenses, pass-through licenses, and expanded licenses are the three categories that FSOS licenses fall into.

Pure unrestricted licenses are the least demanding among the FSOS licenses regarding licensee restrictions. This framework permits the transferee to copy, alter, and distribute the software provided that the licensor makes it available (via contract or otherwise)[174]. The license does not require significant terms for any program transfer to a third party. However, it may include minor requirements for the right to transfer the software (such as keeping any copyright notice). The transaction agreement between the transferor and transferee determines those terms.

The "pure unrestricted" category includes many licenses approved by the Open Source Initiative. Most licenses are built on the so-called BSD license model, which does not reference required substantive terms of any redistribution but instead depends on a copyright notice to grant transferees the right to copy, modify, and distribute. This is regarded by many as the most "open" and "free" FSOS variety. The licensee retains the autonomy to determine the format of ensuing transactions about the original or altered code[175].

Copyright (c), all rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted, provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions, and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions, and the following disclaimer in the documentation or other materials provided with the distribution.
- Neither the name of the licensee nor its contributors may be used to endorse or promote products derived from this software without specific prior written permission[176].

Notably, this license's language does not apply to control terms in any later software distribution by the licensee, whether or not it has been modified, except the disclaimer clause. By enclosing their copies of the software in transactions with third parties with restrictive terms, licensees can use the framework to protest new terms and revert to the source. This poses no issue for verbatim copies of the original software because licensees have unrestricted access

[174] Ibid
[175] Supra note124 page 48
[176] Bsd License, available at www/opensource.org.

to the original code. More risk arises, though, if the first licensee significantly improves the original program, possibly turning it into a non-FSOS framework. Free software proponents contend that strong viral or pass-through terms justify using separate licenses to govern subsequent terms[177].

With pass-through licensing terms, the licensor can manage the conditions of software redistribution to third parties, but only as long as they are relevant to the program that was supplied to the licensee, who then redistributed the software[178]. These conditions guarantee that third parties who redistribute the program will receive the same rights. The most prevalent use of pass-through provisions is for distributing verbatim copies of the original program.

Expanding licenses apply the Free Software Foundation's (FSF) copyleft principles and impose limitations on altered software versions and newly developed content produced by the licensee[179]. The word "viral" license originates from this possible influence on freshly developed software code by the original licensee. These phrases are intended to stop modification processes from effectively removing formerly "free" software packages from the free ecosystem by caulking them in with proprietary changes[180].

Viral provisions reduce the licensee-developer's authority over their work and, in cases where source code disclosure is mandated, risk destroying trade secrets that could be worth a lot of money. The magnitude of the viral impact under the FSOS license and the licensee's intended program usage determine how risky the situation is.

Viral or expansive effects are usually reserved for situations where the licensee distributes altered code; they do not apply to modifications developed and kept for individual use. In FSOS licensing, it is customary to demand an extension of the license's terms to encompass the dissemination of "derivative works." A work based on one or more pre-existing works is referred to as a "derivative work" in the Copyright Act[181]. Examples of such works include translations, musical arrangements, dramatizations, fictionalizations, motion picture versions, sound recordings, art reproductions, condensations, abridgments, and any other form in which a work may be recast, transformed, or adapted.

Because of its restrictive conditions that limit licensees' options for redistribution of the work, the GNU General Public License (GPL) is the most popular free software license. Although it is the least well-written of the central Free Software Operating System (FSOS) licenses and

---

[177] GPL, version2.0 preamble
[178] Supranote124page 49
[179] Ibid
[180] Ibid
[181] Ibid page 50

presents difficulties for licensees, it is also related to Linux. Instead of taking definite stands on the permit and related legislation, the Free Software Foundation's public interpretations frequently demonstrate its dedication to free software. Because of the GPL's inconsistent interpretation of contractual and non-contractual words, its requirements are unclear, making it difficult to use software covered by the license near other significant software[182].

Copyright rights were the primary concern of open source and free software (FSOS) licenses until the mid-1990s when patent rights gained significance[183]. Nevertheless, how patent rights are handled in most FSOS licenses could be more varied and lacking. While some agreements try to address issues linked to patents, many ignore the question and concentrate on copyright. A consensus that would shield licensees from the risk of patent infringement without imposing an obligation that extends too far into a participant's patent portfolio to be economically feasible has yet to be reached by the FSOS community. [184]

Although patent concerns are covered in the GPL, no explicit patent rights or exclusive rights under patent law are granted, including the right to manufacture, use, sell, or offer to sell the patented invention[185]. Instead, it says that licensees are still required to abide by the GPL's terms even if they are subject to restrictions that conflict with it. In the event of a settlement or other attempt to avoid liability for patent infringement, the licensee's continued use of the software may be blocked by the settlement unless the settlement conditions align with the GPL. This may be a partial consequence of licensing regimes. Still, in contrast to most commercial licenses, the licensee in this situation is unlikely to be able to negotiate a GPL terms exception to allow for a patent license or settlement; in fact, the FSOS philosophy would make such an exception challenging. The rigidity often impedes such specific discussion in terms created by viral FSOS licensing.

Three approaches to addressing patent infringement issues have emerged in the context of FSOS licensing: the first addresses patent grants that are covered by the license; the second addresses infringement warranties or patent indemnity duties; and the third addresses "patent retaliation," which entails adverse outcomes if a party in an FSOS chain files a patent infringement claim. While many licenses recognized by the Open Source Initiative contain both a grant of copyright and a license to patent rights, some FSOS licenses, such as the GPL, do not create a patent license.[186]

---

[182] Supranote124 page 52
[183] Supra note 124 page 70
[184] Ibid
[185] U. S Copyright Act u/s 271(a)
[186] Supra note 124page 71

The patent license has a relatively limited scope; it only covers the patent rights of a specific licensor and the possibility that those claims may be included in the software that licensors have provided to licensees. The attempt to limit the patent license's scope to that which is necessary to carry out the FSOS idea is evident in this statement. A potential risk associated with establishing a patent license chain is that the patent holder may be bound by an overly expansive award, making FSOS unaffordable for them[187].

A patent cross-license in which licensees issued to the licensor and third parties was produced by the Real Networksneee Public Source License Version 1.0. The patent grant covers after-acquired patent rights for both the licensee and the licensor. However, the award only applies to claims that are inevitably violated when the Original Code is used or created, not when combined with other hardware or software[188].

As part of the FSOS license paradigm, there has generally been a reaching out in FSOWS licenses into the field of patent law. Crucially, the licensee's liability risk for infringement of third party rights is not covered by these licenses. FSOS licenses disclaim any promise of non-infringement, regardless of the extent of their separate patent and copyright awards, to the extent that they deal with such claims expressly.

One typical solution to patent difficulties in open source or free software (FSOS) initiatives is the employment of patent retaliation clauses. These restrictions are intended to prevent patent litigation against any license chain participant by leveraging the leverage and sunk costs that may result from an individual using the FSOS software as a licensee. This clause, for instance, is used in the Apache license to terminate the licensee's patent rights if they file a patent lawsuit alleging that the Work or a Contribution incorporated into the work constitutes a direct or contributory patent infringement against any entity (including a cross-claim or counterclaim in a lawsuit).

The termination rule covers any patent claim against the software, whether in its original or modified form and is not limited to the code written by the contributor. This would give the FSOS project more protection against patent claims than it currently has against copyright claims. The statement above does not exclude a third party from suing for copyright infringement if the third party "contributes" copyrighted code to the software. Still, another contributor owns the copyright and chose not to contribute that code.

[187] Supra note 124 page 72
[188] Supra note 124 page 73

However, it needs to be clarified if a court upholds this clause regarding automatic termination. In this country, courts frequently refuse to uphold provisions that would result in a forfeiture of rights and inflict a loss on the party in violation, much beyond relatively minor offenses. Furthermore, the clause controls rights not covered by the FSOS property by using the power of the FSOS software.

In 2004, Larry Rosen, the Open Source Initiative's general counsel, suggested changing the patent retaliation clause in his organization's two licenses. Version 2.1 contains new language stating that as of the date you file an action, including a cross-claim or counterclaim, against the licensor or any licensee, alleging that the Original Work infringes a patent, you may no longer exercise any of the rights granted to you by this license and this license will automatically terminate[189].

Software licenses frequently come with quality and non-infringement risk assurances. Nonetheless, the standard license style often disclaims any warranties in the context of free software and open source, enabling licensors to provide licensees with explicit warranties. The absence of warranty protection distinguishes this licensing model[190]. According to guidelines released by the FDIC on FSOS software and warranty matters, proprietary software licenses typically come with a guarantee and indemnity for user defense in infringement cases[191].

On the other hand, FOSS is now being marketed by VARs with dual licenses, which encompass the software's obligations and rights. Institutions should carefully consider the conditions of any indemnity that a VAR offers and its ability to pay for a strong defense. If available, institutions might also take into account third-party insurance. Concerns around indemnity and warranty raise issues for businesses regarding the relative worth of various software packages. The existence of dual licensing schemes suggests that selecting between proprietary software and FSOS is a challenging decision for businesses. Conflicts increase as "proprietary" providers provide more excellent protection through warranties and indemnities. Microsoft started a behavior that does not exist in FSOS: requiring volume licensees to repay them for patent infringement. [192]

A disclaimer must be included with software redistribution under certain Free Software and Open Source (FSOS) licenses, such as the GPL. This is only sometimes the case under regular FSOS licenses. On the other hand, specific licenses let holders make and provide warranties on

[189] Supranote124 page 77
[190] Supranote124 page 78
[191] FDIC, Risk Management of Free and Open Source Software, p5 (Oct. 21, 2004).
[192] Supranote124 page 78-79

their liability if they so desire. Specific licenses establish a presumption of protection for those engaged in a production or distribution chain. The justification for not offering warranties is to safeguard the purportedly voluntary and accessible nature of contributions made to the FSOS software.

A user considering contesting the license's applicability needs help. One way they can argue that there is no legally binding agreement between them and the licensor is to reject the license's obligations. However, this leaves them in the unfortunate situation of needing access to all the fundamental rights of the free or open-source software license.

Because software is freely transferred across national borders and business is conducted globally, open-source and free software licenses are difficult to enforce in other jurisdictions. International copyright laws are frequently not enforced strictly, and agreements like the Berne Convention are commonly broken. The spread of "pirated" CDs and DVDs, as well as file-sharing software, makes copyright enforcement more difficult, especially when the extraordinary power of these licenses shows itself when enforcement issues arise. Users would not be permitted to retain, utilize, or alter the copyrighted code without a license[193]. The laws of the United States and other countries protect works that aren't expressly marked as copyright[194].

Most software is created in developed nations, where copyright enforcement is commonplace. Reliability and support are more important to users than software's additional cost. It takes a robust, aboveground organization to deliver these services and dependability. This probably applies to work licensed as free or open-source software since "pirating" usually refers to distributing a job without paying royalties.[195]

Even in the absence of solid legal enforcement by the licensee, market factors and social dynamics likely limit the level of infringement, which accounts for the comparatively modest amount of litigation that open source and free software licenses give rise to. Even without enforceable safeguards, market forces and societal dynamics tend to restrict the amount to which these licenses can be violated[196].

The challenges of breaking open source or free software licenses, particularly in terms of copyright enforcement. For example, XYZ Corporation used a significant portion of the Duchess code in their program, Vulcan, without giving credit to the creators. This led to Vulcan

---

[193] Supranote101, the global scope of Open Source Software Licensing, page 153- 154
[194] Ibid
[195] Ibid
[196] ibid

being released under a proprietary license without the necessary copyright and permission notice. Years later, ABC Corp. released Virgo, a similar application, but based on the Duchess source. XYZ's legal counsel suggests filing a lawsuit for copyright infringement. However, XYZ's attorneys learn that they need to follow the Duchess license and advise against proceeding with the lawsuit. If an infringement claim is issued, XYZ could lose its exclusive right to distribute Vulcan and have its copyright declared invalid. The creators may file a lawsuit against XYZ for copyright infringement, potentially awarding damages for XYZ's infringement.[197].

The effective enforcement of open-source and free software licensing also depends on the open-source and free software communities[198]. It is just wrong to violate the conditions of these licenses or to know to violate them. There still needs to be more in the way of open-source and free software licenses. Most of the code published under these licenses results from volunteer labor, with many dedicating substantial portions of their lives and a great deal of time to creating and disseminating high-quality code for the benefit of the most significant number of people.

This community feels strongly about this principle. It is unimaginable to take someone else's work and distribute it as one's own without violating it egregiously. It is not the language in the licenses themselves nor the courts that uphold them. However, this moral precept enforces free and open-source software licenses[199].

Although the concepts of open source and free software licenses are widely acknowledged, the two groups have different ideologies. One rule is that it is improper to distribute or alter someone else's work without the author's consent. The "cross-over" of programmers from open source to proprietary projects should not be confused with this. Notable open-source programmers such as Eric Allman and Bill Joy have transitioned from open-source to proprietary projects while adhering to the original license's rules and philosophy in both distributions.[200]

Programmers and users are defenders of these ideals, even though contracts and the legal system are essential to their protection. When writing code, a programmer could wish to combine features from one or more applications under various licenses to create a new program. Programmers must verify their compatibility since seemingly benign clauses in one

---

[197] Ibid

[198] Supranote101 community enforcement of open source and free software licenses, page 156-157

[199] Ibid

[200] Ibid

or both licenses could render them incompatible. Copyright infringement occurs when programs are distributed or modified, including code incompatible with a permit[201].

A user contemplating the combination of works licensed under various licenses should always carefully review the licenses in question. It is more straightforward to list the incompatible licenses than to state that two licenses are compatible confidently.

For instance, the code cannot be mixed with work covered by another license (except by cross-licensing) if either of the works is licensed under a proprietary license. The user's only right concerning the work under a traditional proprietary license is to utilize one copy. It is not permitted to alter or distribute two works covered by proprietary licenses together, even if the licenses are the same, without breaking the terms of the license(s).

One license that is incompatible with code released under most licenses is the GPL License. The author of the purported "new program" is imposing a constraint (compliance with the conditions of that license) that is not included in the GPL License and is thus in violation of the GPL by combining GPL-licensed code with code under any but the most permissive licenses.

The protection of these ideals depends on open-source and free software licensing policies.

Specific works are not covered by any license, either because their creators have released them into the public domain or because time has worn out the copyright protection on the work. Therefore, as long as the conditions of each license are met, code in the public domain may be integrated with code licensed under any other license. Programmers, for instance, can combine GPL-licensed code with public domain code, alter the resultant work, and distribute it as long as they abide by the GPL.

The following circumstances make the issue of licensing compatibility more challenging to understand. Generally speaking, "research style" licenses work well together. For example, if all license requirements are followed, two programs licensed under different licenses—one under the MIT License and the other under the BSD License—can be merged and delivered under the licensee's proprietary license[202]. In general, licenses that permit the addition of new constraints to the code to be licensed are also compatible with licenses that follow the research style. For instance, the Q Public License allows for the release of altered versions of the licensed work in the form of patches added to the original work. Users are free to draw the code for a patch from an MIT-licensed program and distribute it by both licenses, provided they also

[201] Ibid

[202] Supranote101 page 161

abide by the other terms of the Q Public License and include the MIT License's required permission and copyright notices.

Through a technique called cross-licensing, authors of a work can grant licenses under terms different from those stated in the original license. Barring the sale or licensing of those rights to third parties, this is so that the original licensor will always maintain all copyright-related rights. Free and open-source software licenses confer certain rights on the licensee in exchange for licensee compliance with predetermined requirements rather than requiring the creator to cede full rights to a third party[203]. The licensor does not consent to provide the license to those licensees or under those conditions alone.

ABC Corp. uses the pre-1999 BSD License to license its Mudd program in the example provided. John Smith wants to use Mudd code in his open-source project, Pond. Pond is built using code from Audrey Strauss's previous GPL-licensed game, River. He knows that the pre-1999 BSD Licenses and the GPL are incompatible, but he can find a solution if ABC or Strauss agree to cross-license their software[204].

Due to the culture of free and open software, which promotes free dissemination of work and discourages duplication of effort, most open-source software developers are amenable to cross-licensing. Specific licenses, such as the GPL, expressly outline this possibility. Section 10 of the GPL grants rights to incorporate program components into other free programs with differing distribution circumstances[205]. If it succeeds, all challenges and possible ambiguity around various licenses vanish, and the coding process will be controlled by whichever license all parties can agree upon without worrying about future legal issues.

Open-source organizations, like the Apache Software Foundation, are not permitted to cross-license their copyrighted works. When hundreds or thousands of programmers contribute to a project, as is the case with "bazaar" project structures, cross-licensing may not be a feasible option. For instance, numerous authors must relicense their work under a different license for the Linux operating system, which is licensed under the GNU/Linux (GPL) license. Due to the strict copyright restrictions, it is impractical for each contributor to claim that their work can only be used in ways compatible with the terms they agreed to join the project. Most of the time, cross-licensing such projects is not feasible. Nonetheless, most accessible and open-source software projects don't pose any logistical challenges because they are typically

---

[203] Ibid
[204] Supranote101 page 162
[205] GPL Version 2.0

maintained by one person or a small group of individuals, who can usually be reached only by asking permission to share the work under a different license.

The SCO Group sued IBM in March 2003, claiming that the business violated its intellectual property rights about the UNIX kernel[206]. Because SCO keeps some of the most specific elements of the code they claim to have been protecting under wraps, the suits exact facts still need to be discovered. Meanwhile, SCO asserts that IBM and AT&T agreed to a contract in 1985 to develop a UNIX variant known as AIX. SCO filed a claim against IBM since they acquired all of the UNIX-related intellectual property from AT&T in 1995[207].

Additionally, SCO asserted that any Linux user needed to get a license from them and threatened to prosecute any business user for copyright infringement[208]. When they sued DaimlerChrysler and auto parts retailer AutoZone, two corporate Linux users, in March 2004, this threat ultimately materialized[209]. Following this, IBM filed a suit counter against SCO, saying the latter violated its patents and copyrights. Additionally, IBM claimed that SCO breached the GPL due to its use and modification of the Linux kernel, which is licensed under the GPL[210].

Although it is too soon to judge how strong SCO's claims are, this case has undoubtedly raised the stakes for copyleft licenses' financial significance and, consequently, the significance of ensuring that the license conditions are legitimate. If the goal of SCO's assertions is to create doubt and uncertainty in the minds of potential FOSS users, then they may succeed. Businesses are trying to reduce their possible liabilities in an already hazardous and challenging business environment by ensuring they will be safe and have enough insurance if this happens.[211]

Due to their heavy reliance on copyright protection, open source software (FOSS) licenses have made the patentability of software a contentious topic. The U.S. Patent Office has been awarding more and more software patents, which has worsened the U.S[212]. The rationale behind this shift has been linked to concerns with software protection and the relevance of the Abstraction-Filtration-Comparison theory in U.S. copyright law.

---

[206] Andrés Guadamuz González, Legal Challenges to Open Source Licences, Volume 2, Issue 2, June 2005, page 257

[207] Caldera Sys. Inc. v. Int'l Bus. Machs. Corp. (D. Utah 2003) (No. 03-CV-0294).

[208] Supranote206, page 259

[209] Ibid

[210] Ibid

[211] Ibid

[212] Supranote206 page 260; K Nichols, *Inventing software: the rise of "computer-related" patents*, (1998)

Because software is listed as a non-patentable substance in the European Patent Convention, Europe has historically been seen as having a separate patent environment[213]. Practice and case law, however, have permitted "computer-implemented inventions" with a technical effect or contribution to be eligible for a lower patentability requirement[214]. To improve the precision of the phrasing used to describe technological impact, the European Union has proposed a directive to harmonize various areas of software patentability[215].

Software developers, especially Free and Open Source (FOSS) developers, have opposed the plan because they believe the American system is too broad and permits the patentability of any kind of software, including those that have existed for a long time[216]. FOSS developers see this as a challenge to the fundamental tenet of FOSS licenses: copyright. Additionally, despite the new language and current limits, broad patents in the American model will be granted in Europe.

U.S. Patent 6,330,551, granted to safeguard automated online dispute resolution systems, is one instance of this[217]. The patent's primary function is to protect a blind-bidding-like system in which disputing parties submit their financial claims to a computer system that computes them and, in the end, automatically determines a settlement that satisfies each party's stated expectations. Although this patent has also been granted in the U.K. (GB2345997), it should not be subject to the purportedly strict European standards on the patentability of software ideas[218].

Many situations are similar to this one, rendering the directive's suggested wording useless. Due to disagreements between the European Parliament and the European Commission on the precise language of the definition of technical contribution, the draft directive has needed help getting approved and implemented[219]. Although the directive's future is yet unknown, it is desired that if it is accepted, the language will be as clear-cut and comprehensive as possible to reduce the likelihood of issues with how the concepts are applied.

---

[213] Article 52(2) European Patent Convention

[214] Cases such as VICOM [1987] 2 EPOR 74; Merrill Lynch [1989] RPC 561; Gale [1991] RPC 305; Fujitsu [1997] RPC 608

[215] Proposal for a Directive on the Patentability of Computer-implemented Inventions COM(2002) 92

[216] Supranote206 page 261; for example, US Patent No. 6865546, which protects a system that determines a buyer's age according to previous buying records.

[217] Supranote206 page 261

[218] Ibid

[219] Ibid

In conclusion, as licenses mostly rely on copyright and may result in unenforceable licenses and lawsuits, expanding the patenting of software concepts would be detrimental to the growth of free and open-source software.

The Free Software Foundation (FOSS) development community faces formidable legal obstacles; an adverse decision in the SCO case might have disastrous consequences for the movement as a whole[220]. The open-source community can withstand these challenges by continuing to be a robust force that disseminates information online and refutes arguments against it. Well-executed instances of coordinated online activism include issue-focused blogs and websites such as Groklaw and the Foundation for a Free Information Infrastructure (FFII)[221]. To refute the numerous errors presented in attacks, academics should get more active and produce more scholarly works.

Ensuring these issues are addressed by strong, legitimate licensing is the most robust protection against FUD. Redrafting FOSS licenses to allow for patents is necessary since the current reliance on copyright in the patent space needs to be revised. "Patent clauses" in specific licenses, like the GPL, permit users to assign patent claims derived from the protected software and use the copyrighted portion of the program.[222] The GPL clause states, "We have made it clear that any patent must be licensed for everyone's free use or not licensed at all." Other recent licenses, such as the Apache License (version 2.0) contains a patent assignment clause that allows users to use the copyright part of the software and assigns patent claims that arise from the protected software.[223]

Recent findings imply that there might be alternative approaches to safeguarding free and open-source software from software and that stringent licenses are not required to create a shared pool of patents for open-source uses. The announcement by IBM that it will not enforce its 500 software patents if they are used in open-source software projects has generated news[224]. This unprecedented step, which provides a precise definition of an open-source project, has been made possible by deft contract law. However, this revelation should be taken with a grain of salt, given that IBM was granted over 3,000 patents in 2004 and possesses a sizable portfolio of software patents[225].

---

[220] Supranote206 page 262
[221] Supranote206
[222] Ibid
[223] Guadamuz, A. (2005). Legal Challenges to Open Source Licenses. https://core.ac.uk/download/429718168.pdf
[224] Ibid; “IBM frees 500 software patents”, BBC News (11 January 2005), @: <http://news.bbc.co.uk/1/hi/technology/4163975.stm>.
[225] Supranote206 page 262

Understanding that the copyright holder is not given an exclusive license when software is released under an Open Source license is crucial. To make money, a copyright owner is more than welcome to offer the software's proprietary version under standard commercial licenses and the source code under an Open Source license. Multiple Open Source licenses may also be used to release software. It is unusual for software to be released under the GPL and the Artistic license in certain instances involving the Artistic license[226].

The main challenge with this strategy will be utilizing source code included in the program's proprietary version created by the Open Source community[227]. Under the GPL, this is a complicated (and presumably unachievable) problem. The Open Sources licensing brings up numerous significant legal concerns. There are disagreements concerning copyright regulations and fair use principles concerning how copyleft is understood.

Significant legal questions are brought up by the ownership of source code in community-based development projects. These include the terms, interpretation, and enforceability of Open Source licenses and their susceptibility to software patents. There are also concerns regarding who oversees an Open Source initiative and what occurs when influential individuals depart from the movement. The GPL has not been the subject of any litigation filed by the Free Software Foundation, and no decisions that specifically interpret Open Source licenses or license-related concerns have been adjudicated.

After the first BSD Unix licensing disputes were resolved out of court, there were only unanswered legal questions about Open Source licenses[228]. The community-based methodology known as "open source development" maintains the copyright ownership of the code creator while allowing them to use the Open Source license on their creations[229]. Some authors have centralized ownership under one organization by transferring their copyright rights to the Free Software Foundation. All Open Source projects, however, eventually contain code that many copyright holders own.

The community model used to build open-source software presents concerns over who owns what specific parts of the source code. Conflicts may arise around decisions made regarding software development, version control, release schedules, programming standards, and other aspects of managing the development of a particular product. Programs have occasionally "forked," indicating disagreements over which version is the official, leading to alternate

---

[226] Kennedy, Dennis M. (2001) "A Primer on Open Source Licensing Legal Issues: Copyright, Copyleft and Copyfuture," Saint Louis University Public Law Review: Vol. 20: No. 2, Article 7.
[227] Ibid
[228] Supranote227 page 345
[229] Ibid

programs' development. In these situations, issues about project ownership and who is entitled to pursue copyright enforcement in an infringement lawsuit will arise[230].

It is unclear how copyrights and permits will be enforced when a license is violated or an infringement. The strategy used by the Free Software Foundation is significant because it allows the Foundation to act independently to enforce permits and take legal action in the case of infringement without having to locate all copyright owners by persuading authors to assign copyright ownership to the Foundation[231]. However, this strategy is unique to the Open Source movement.

Open Source licenses are distinct in that the community has consistently interpreted them over time, frequently thanks to the loud efforts of influential members like Richard Stallman[232]. The community's intense discourse and idea-sharing have given rise to interpretation disputes and license changes that aim to "improve" or amend the provisions of standard Open Source licenses. Since he founded the Open Source movement, Richard Stallman's interpretations have been crucial in interpreting the GPL[233].

The informal, community-based, simple language approach may need to be revised as Open Source licensing gains traction and courts and attorneys scrutinize the licenses more closely due to specific legal issues. It will be interesting to observe how much weight courts assign to existing community interpretations.

The restriction on the number of Open Source licenses has been beneficial. Still, as more commercial software products are converted to the Open Source model, demand will continue to mount to expand the number of licenses and their variety. Interpretation issues and creating programs with a combination of possibly incompatible licenses will persist with any growth in permits or allowable changes.

Open Source licenses' streamlined structure presents interpretive challenges; for example, UCITA licenses are not indefinite. When a court decides that an Open Source license has expired because the "reasonable" length has passed, this could lead to issues. There may also be other "gap filler" or default rules that raise similar issues[234].

One argument for why the BSD Licenses are essential for contract formation is that they limit responsibility for all losses, including direct damages[235]. A severability clause in many

---

[230] Ibid page 371
[231] Ibid
[232] Supranote227 page 372
[233] Ibid
[234] Supranote227 page 373
[235] The Open Source Initiative, *Licenses*, *at* http://www.opensource.org/licenses (accessed on June 2, 2024)

commercial software licenses enables judges to ignore or alter just the objectionable portion of the permit while upholding the remaining terms. A software licensing attorney may be tempted to suggest changes to Open Source licenses that the community would not approve of, but this clause is particularly crucial under UCITA[236].

The enforceability of shrink-wrap categories of agreements may be called into doubt given that Open Source licenses need to be negotiated or signed. State law may vary on these matters. By confirming shrink-wrap agreements in general, UCITA would benefit the Open Source movement, but it also presents other challenging problems for Open Source licensing.

Since developers won't contribute to projects if they could be held personally liable or forced to make a warranty regarding their source code, the disclaimer of warranties is a crucial component of open-source licensing. According to open-source license agreements, software source code is provided "as is" without warranties[237]. These disclaimers make sense because the licensee/user has access to the program's source code and can alter it to correct any errors in the code, even though its primary purpose is to shield it from personal liability[238].

Certain conditions or restrictions on warranties may be stipulated by state law, particularly rules about consumer protection. Regarding computer information, UCITA has introduced new implicit guarantees and established precise guidelines for warranty waivers. A general disclaimer of warranties and a statement that the software is being given "as is" will raise questions about whether or not UCITA includes a license.

The Open Source license presents two critical questions when licensing source code used in software. Since no licensee is signing any Open Source license agreements, the first question is whether the applicable Open Source license will obligate downstream licensees of programs. Since downstream licensees are intended to be bound by the Open Source license, this question is crucial in the event of a dispute within the Open Source movement.

The second concern is that should the number of open source licenses be permitted to increase, maintaining track of licenses may become crucial in open source systems. Anybody creating an Open Source program needs to be highly aware of which licenses apply and how those licenses affect the program. This is especially important regarding Section 2(b) of the GPL and whether or not a software program's inclusion of GPL code entails that the program as a whole is subject to all of the GPL's terms[239].

---

[236] Supranote227 page 374
[237] Supranote227 page 374
[238] Ibid
[239] Supranote227 page 375

Since the early days of developing open source licenses, the Open Source movement has been concerned about software patents' possible potentially harmful impact. To address the patent issue, some Open Source licenses, like the GPL, require the licensee to consent to any patent or decline to apply for any software patent that could impact the license.

The significance of Open Source software in the servers and infrastructure of the Internet has increased dramatically between 1998 and 2001. The Open Source approach to software development has garnered recognition due to its dependability, affordability, and collaborative methodology. On the other hand, as Open Source is utilized more frequently in business settings, doubts regarding the legality of licenses issued during an informal period are beginning to surface.

Due to potential compatibility with communitarian development methodologies and legal concerns, a thorough evaluation of the Open Source licenses is necessary. Changes in copyright legislation, especially UCITA, may unexpectedly affect licensing. Future alterations to fundamental Open Source licenses and reconsidering licensing tenets are anticipated. The majority of licenses will be reviewed and revised to reflect any legal concerns brought up by UCITA or any new legislation. Commercial developers will probably keep lobbying for their licenses, prolonging the fight to restrict and narrow the number of Open Source licenses. While the community will always be essential to this process, judges will probably have a more significant say in how Open Source develops.

One of the most important ideas to come out of open source is copyleft, which protects freedom and takes an approach to intellectual property that serves the public interest by using licenses. This idea will remain critical to intellectual property law, safeguarding not only individual rights but also the community's ability to use intellectual property created for the benefit of the community.

To sum up, the evolution of the Open Source movement, licensing, and concepts like copyleft and software freedom will significantly influence intellectual property law development.

4.3 ISSUES AND CHALLENGES OF OSS IN INDIA

India nurtures benefits from adopting and using open-source software (OSS) with substantial cost savings, agility, and encouragement towards innovation. However, these advantages come with several legal hurdles and issues, including many regulations, mainly concerning intellectual property rights (IPR). Developers, organizations, and governments are constituents who need to attend to such challenges to make contributions and learn sustainable things in India through OSS.

Business and development perspectives can be used to understand the utilization of open-source software. From a business standpoint, open source software (OSS) gives clients specialized software solutions because its source code may be changed and shared. OSS vendors make money by letting users download their apps for free, turning them into paying customers later, or selling OSS-compatible hardware. It encourages a collaborative work approach from a development standpoint because several teams from different firms/entities actively create the OSS. OSS is typically less expensive than proprietary software and is provided for free. Because proprietary software is more complex to obtain, it is more common. It is fiction that open source software (OSS) is in the public domain. In order to use OSS(s), a license of some kind is needed. The traditional copyright system does not protect this license; instead, the open source model has developed a legal innovation of the traditional IP rights system known as "copyleft[240]." The process involves copyrighting the software and granting mass licenses for use, improvement, or modification. In this case, the changes are made subject to reciprocal terms, meaning that the changes are made in exchange for the software development community. Notable licenses include the GPL (General et al.) and the Mozilla Public License. Permissive open-source licenses are another type of OSS[241].

Computer programs are protected by copyright in majority of nations. Copyright protection guarantees that inventors have the legal right to their inventions and are paid for the goods that result from their labor, which promotes, supports, and rewards creative endeavours. In an industry where innovation and the growth of ideas have led to revolutions in technology and society, copyright protection is therefore critical to producing new and original work. In India, copyright protection is granted to proprietary software.

The Indian software industry is another example. According to the revised NASSCOM-MCKINSEY Study in 2002, by 2008, the industry would employ 4 million people, account for 7% of India's GDP, and contribute 30% of the country's foreign exchange inflows[242]. An industry rapidly expanding its market size would invest more and more in R&D to continuously innovate and provide better services to its expanding customers. Thus, the need for protection increases. This is evident in the Indian Film Industry, where, over the last ten years, the country

---

[240] *OPEN-SOURCE SOFTWARE – USAGE & ITS LEGAL IMPLICATIONS – Legal Developments*. (n.d.). The Legal 500. https://www.legal500.com/developments/thought-leadership/open-source-software-usage-its-legal-implications/#_ftn19

[241] Ibid

[242] Indian Software Industry Development: International and National Perspective, January 2001, Economic and Political Weekly 36(45):4278-4290; DOI:10.2307/4411353

has seen a massive increase in its market, ranging from the UAE to the US and Europe. The film industry has now actively started campaigning against piracy and for copyright protection. It would be pertinent here to look into the characteristics of the software Industry in general and the Indian software Industry in particular. The software industry is prone to high research and development costs. Adding to its woes is the meagre cost of reproduction. While developing a typical program might take years and run into millions of dollars in terms of costs, the copy takes possibly a floppy disk or two and two minutes of inexpensive computer time.

The copyright law in India provides IPR protection for computer software. Consequently, the provisions of the Indian Copyright Act of 1957 safeguard computer software's copyright. Significant amendments to Indian law were introduced in 1994 and took effect from 10 May 1995. Due to these modifications, Indian copyright laws are among the strictest worldwide.

Free usage does not necessarily mean that the software is in the Public Domain**,** *i.e.*, they are not covered under copyright laws. A copyright comes into existence as soon as the software is created. In *Tata Consultancy Services v. State of Andhra Pradesh*[243], the Supreme Court of India held software as Intellectual Property. It is covered in section 29(o) of the Indian Copyright Act, 1957 ("Copyright Act") under "literary works" which includes computer programs. Section 14(b) (ii) of the Act gives the right to sell or to give the software on commercial rent. [244]

The catch is that the person who created the Open Source Software is not eligible for a royalty from its sales. In the Indian setting, how can one freely distribute Open Source Software (OSS) while still maintaining its copyright? The OSS is not expressly recognized in India by the Indian Patents Act of 1970, the Copyright Act, or the Information Technology Act of 2002.

Nevertheless, if we examine Copyright Act Section 14, a computer program's copyright owner is permitted "to issue copies of the work to the public not being copies already in circulation"[245] under Sub clauses (b)(i) and 14(a)(ii). Since section 14, cell a (ii), does not mention whether copies being given to the public should be free, the OSS developers can license and re-distribute their original work without charge. The copyright owner is granted permission to license "any

---

[243] Tata Consultancy Services v. State of Andhra Pradesh 271 ITR 401

[244] Singh, A. (2017, October 6). *Are open source licenses recognized under Indian Copyright Law? Explore the best open source license for your business. - iPleaders*. iPleaders. https://blog.ipleaders.in/open-source-licenses-recognized-indian-copyright-law-explore-best-open-source-license-business/

[245] *Intellectual property of copyright in case of derivative work prepared from sources in the public domain*. (2018, October 15). Advocatetanmoy Law Library. https://advocatetanmoy.com/intellectual property/intellectual-property-of-copyright-in-case-of-derivative-work-prepared-from-sources-in-the-public-domain/

interest" in his work under Section 30 of the Copyright Act[246]. The rights the licensee has obtained are identical to those of the copyright assignee. Section 19 clarifies the assignment form, and clause 3 of that section gives the licensor the express choice to provide a free license. Currently, the licensor must indicate the rights licensed, their tenure, and their extent by clause 2 of section 19[247]. As a result, the open-source licenses covered above perfectly comply with Indian legal regulations.

When developing computer software, the creator usually wishes to ensure that the time and effort expended is somehow protected against misappropriation by, for example, a competitor. The software developer wants to prevent others from making verbatim copies of the software and copying as much of the innovation that went into the software as possible. When a computer runs software, it performs a process. Typically, this process can be represented by various distinct (even if functionally equal) software code sequences. Because a rival may watch the software's functions and then build comparable code without knowing the specifics of the software code underlying the functions, a software developer does not only want to rely on the protection of verbatim copying of the program. Because of this, most software developers want to keep control over their work to avoid having the functionality stolen.

Software copyright is violated, just like any other literary work, when a copy or a significant portion is made without the owner's consent.

Making or distributing copies of copyrighted software without the required authorization is prohibited by Section 14 of the Copyright Act. The lone exemption is given by Section 52 of the Act, which permits the use of a backup copy only as a short-term safeguard against the original copy being misplaced, distributed, or damaged. The Copyright Act was amended in 1994, and it is now illegal to sell, provide on hire, or offer to sell or hire any copy of a computer program without the copyright holder's express consent.

The software creates unique problems because it is so easy to duplicate, and the copy is usually as good as the original (although many times plagued with computer viruses). However, the fact that the copy is as good as the original does not legitimate piracy. Copyright law makes no distinction between duplicating software for sale and free distribution. A civil and criminal action may be instituted for an injunction, actual damages (including infringer's profits), or statutory damages per infringement. Moreover, with the amendments to the Indian Copyright Act in 1994, criminal penalties have substantially increased. According to Section 63, now

[246] Supranote244
[247] Ibid

there is a minimum jail term of not less than seven days, which may extend to 3 years, and a fine, which shall not be less than Rs. 50,000, which shall be extended up to Rs.2 00,000.

However, the Copyright (Amendment) Act 1994 has enlarged the scope of protection of computer programs. The Amendment Act confers the copyright holder with an additional exclusive right to sell, give on hire, or offer for sale or hire any copy of the computer programs regardless of whether such a copy has been sold or given on will on earlier occasions. In other words, even the legitimate owner (e.g., purchaser) of a copyrighted work cannot sell or rent his copy.

An apparent conflict has emerged concerning copyright law between the proponents of the OSS and traditional copyright models. The reason is that both models are antithetical to each other. This is because the latter protects the proprietary rights model and has the effect of 'restricting access to the general public.

Regarding patents, open-source software components (OSS) in creating proprietary software frequently carry the risk of infringement (much like it does under copyright law). As a result, even minor infringement can stop the development process completely. Including terms about patents in Open Source licenses is one-way open source addresses patent-related concerns.

The lawsuit between Google and Oracle attracted the attention of the open-source community because of its possible effects on API usage. A ten-year legal struggle between the tech giants ended when the US Supreme Court ruled in *Google v. Oracle*[248] that Google's copying of more than 11,000 lines of Oracle's Sun Java application programming interface (Java SE) code was allowed under copyright law. Google was charged with stealing the names and syntax of 37 API (s) to create Android, its mobile operating system. In defense of itself, however, it claimed that it had simply changed the implementing code—which carried the Java methods' functionality—and kept the declaring code necessary for method declaration. The Supreme Court determined that Google did not violate fair use provisions because Android did not threaten Java SE's market share, the use was transformative, the amount and substantiality of the copying was minimal, and the nature of the use did not separate it from the overall organization of the API.

In this case, the computer sector, and the OSS community in particular, is given a break by the SC ruling. By employing the same declaring code for every enhanced method, an inventive cross-compatible software ecosystem based on universal and common standards can be created thanks to equitable access to the APIs. This boosts the open-source development ecosystem

[248] Google LLC v. Oracle America, Inc., 141 S. Ct. 1183 (2021)

and is a significant benefit. Start-ups will not have to create their APIs or pay for licensing, for instance (s). Gatekeeping of APIs would be avoided, creating a level playing field for start-ups and tech enterprises of all sizes. Developers would also benefit from it because they would not have to learn any new APIs, and their abilities would be transferable. Most significantly, "interoperability" is promoted by the freedom to reimplement API(s) and by having open API(s). The Internet's interoperability is its "backbone." Computers and the internet work because they have "open, documented sets of instructions that third parties can use to ensure that information can successfully pass back and forth between programs." On the other hand, since the reimplementation of the API does not result in the creation of any derivative work, the scope of copyleft provisions in such circumstances may be limited. Additionally, avoiding the critical issue of whether or not APIs are copyrighted will increase ambiguity in the tech sector and jeopardize established API-related procedures.

All modules should be web-based and network-centric, with the library database being released online. From their homes or other faraway locations, members can view the items. Employees at libraries in India can enter data remotely and from various locations. Libraries need to catch up if they use a LAN or a single machine to operate their dynamic website and database unless it is required. Regarding open source, the library will need to put in much effort to get it published via its ISP. Hosting a dynamic website on an ISP differs from merely publishing and hosting a static website. The reason is that it is challenging to decipher other people's code. Authentication and permission should undergo security audits when data is posted online.

Precautions should be taken extra when it comes to open-source software. Most Indian libraries use a single PC to keep their databases on local networks or LANs. Upgrade to the newest technologies; this automated system is not suitable.

It is also apparent here that India's widely used software is web-based and not just open-source. Before choosing any program, it is essential to consider the computing infrastructure of our company and the Internet service provider for online publishing. Based on this information, one should choose either commercial or open-source software.

The platform could be Windows or Linux. An LMS might be proprietary or open source. However, using the OSS or any other software in an isolated environment, such as a LAN or inside a library, is not recommended. Any solution—whether open, accessible, or closed—will need servers, network infrastructure, labor to manage the setup, modification, and alignment of system processes (in this case, the library process), and staff and user training. Since libraries are designed to last for centuries, the best software for libraries would be one whose creator has also survived for many generations. Not only should software be able to survive, but it

should also continue to be developed with new needs and in line with the finest technology available at the time.

Open-source solutions have a high total cost of ownership (TCO). The cost will be significant when considering software installation, server setup, training, AMC, hosting, security, follow-up, and customization. The cost of the program is minimal when considering the entire implementation. The goal should be the Proper Information Systems Solution; open or closed software should not be a concern.

The Indian Copyright Act, among other things, protects literary works that include computer programs, tables, compilations, and databases. The computer Program, for copyright protection, means a set of instructions expressed in words, codes, schemes, or in any other form, including a machine-readable medium, capable of causing a computer to perform a particular task or achieve a particular result. Copyrights are limited exclusive rights owned by an author of original expressions fixed in a tangible medium of expression. The copyright owners of the computer Program have the following exclusive rights:[249] a) To reproduce the work in any material form, including the storing of it in any medium by electronic means; b) To issue copies of the work to the public not being copies already in circulation; c) To perform the work in public or communicate it to the public; to make any cinematograph film or sound recording in respect of the work; d) To make any translation of the work; e) To make any adaptation of the work f) To sell or give on commercial rental or offer for sale or commercial rental any copy of the computer program. Such commercial rental does not apply to computer programs where the program itself is not the essential object of the rental. g) The open source falls in the category of derivate works, but unlike the American copyright Law, the Indian Copyright Act does not expressly enlist the derivative works in the category of copyrighted works. However, the copyright owner has the right to adapt his work. Adaptation, among other things, refers to any work, any use of such work, and any use involving its rearrangement or alteration. The open-source code would fall under the adaptation category in the Indian context. The software copyright owner has the exclusive right to modify or improve his software. He can license this right.

Copyrights cover software since it involves expression. Owners of open-source projects benefit from copyright rules by asserting that copies protect their software. After that, they are the only ones with the authority to duplicate, distribute, and produce derivative works. If they publish

[249] M. Tariq Banday, Indian legal Approach to Licensing of Open Source Software Distribution: an explanatory study- conference paper- June 2011

the work and allow others to edit, duplicate, and change it, producing a derivative work, contributors risk facing copyright violations. Owners grant licenses with conditions permitting such behavior to comply with copyright laws and guarantee that no contributor to the project will be held accountable for copyright infringement[250].

The following parties may bring a lawsuit for copyright infringement: a) the copyright owner; b) the copyright owner's assignee, if the owner has assigned the copyright by this Act's provisions; c) an exclusive licensee, if the copyright owner has granted an exclusive license[251]; d) Legatee, if the copyright is transmitted by testamentary disposition[252]; e) the publisher of the work, if it is anonymous or pseudonymous unless the author's identity is revealed.

Any of the criteria above do not apply to open-source code in and of itself. Since open-source software, by definition, includes multiple contributors rather than a single licensee, there may be numerous litigants. The plaintiff must prove that the copyright violated in an open source is his. Since there will be various owners of multiple copyrights existing in an open-source project, it is necessary first to determine which copyright has been violated and who the owner of that property is.

There would be two classes of literary works: (a) primary or prior works, which are the literary works not based on existing subject- matter and therefore would be called primary or prior works, and (b) secondary or derivative works. These are literary works based on the existing subject matter- the Supreme Court in *Eastern Book Co.v. D.B. Modak*[253] outlined the originality requirement for derivative works. The apex court opined: The originality requirement in derivative work should originate from the author by applying a substantial degree of skill, industry, or experience. A precondition to copyright is that work must be produced independently and not copied from another person. Where a compilation is produced from the original work, the compilation is more than simply a re-arranged copyright of the original, often referred to as skill, judgment, labor, or capital. The copyright has nothing to do with originality or literary merit. The author creates copyrighted material through his skill, labor, and capital investment; maybe it is a derivative work. The courts only have to evaluate whether derivative work is not the end-product of skill, labor and capital, which is trivial or negligible but substantial[254].

---

[250] Ibid
[251] Section 54(a) of the Act
[252] Section 20 of the Indian Copy Right Act
[253] (2008) 1SCC 1
[254] Supranote249 page 4

The apex court did not follow the traditional standard of originality in the above case and relied on the US, Canadian, and English decisions[255]. It was observed that the sweat-of-the-brow approach to originality is too low a standard, which shifts the balance of copyright protection too far in favor of the owner's right and fails to allow copyright to protect the public's interest in maximizing the production and dissemination of intellectual works. On the other hand, the standard of creativity and originality needs to be lowered[256]. A creative standard implies that something must be novel or nonobvious concepts more appropriately associated with patent law than copyright law.

The author of a compilation must exercise his skill and judgment to produce the material in question, which may not be creative in the sense of being novel or nonobvious, but it also cannot be the result of labor and capital alone in order to be entitled to copyright. The author's derivative work must bear distinctive characteristics and the original text's flavour[257]. The trivial variation or inputs in the judgment would not satisfy an author's copyright test. Since Open Source Code is a derivative work, it must adhere to the higher standard of originality set forth by the highest court. A claimant for copyright protection over an improvement must meet the conditions outlined in the ruling.

When someone does something for which the owner of the copyright has the exclusive right granted by the copyright Act, without permission from the copyright owner or the Registrar of the copyrights under the copyright, or in violation of the terms of a license so granted, or of any condition imposed by the competent authority under the Copyright Act, it is considered to be an infringement of copyright in work. Therefore, distributing, copying, or creating a derivative work without the owner's consent will be an act of infringement.

Indian courts have not firmly established the standards for identifying software infringement. The queries that need to be addressed are: How does one determine whether a source code violation exists? To what extent must a code be identical before assuming it was copied rather than written from scratch?

American courts have determined infringement using side-by-side comparisons of source codes. The court determined that there was a chance of success for an infringement claim in

---

[255] Supranote249-The Supreme Court in Feist Publications Inc. v. Rural Telephone Service Co. Inc., 499US 40(1991) observed thus: The sine quo non of copyright is originality; CCH Canadian Ltd. v. Law Society of Upper Canada, 2004(1) SCR339 observed thus: "I conclude that the correct position falls between these extremes. For a work to be "original" within the meaning of the Copyright Act, it must be more than a mere copy of another work

[256] Supranote249

[257] Supranote249

Cadence Design System Inc. v. Avant! Corp[258]., even though there were only fifty-six lines of code that match between the two programs. According to the argument, "quantitatively, what has been duplicated is tiny, but it can still be qualitatively substantial[259]." These guidelines may help identify instances of open-source code infringement in India.

A major case involving open source software (OSS) and intellectual property rights (IPR) is *Campus Eai India Pvt. Ltd. V. Neeraj Tiwari & Ors*[260]. Neeraj Tiwari & Ors And Campus EAI India Pvt. Ltd. are at odds in this litigation for purported software plagiarism. Campus EAI India Pvt. Ltd., the plaintiff, asserted that the defendants were utilizing their software without the required license and had duplicated it without authorization.

The Delhi High Court found in the defendant's favor that the program was not copyright protected and that there was no plagiarism proof. In addition, the court granted the defendant's request for summary judgment, resolving the matter without requiring a full trial. The difficulties in defending intellectual property rights when dealing with open-source software are brought to light by this case. The court's decision highlights the significance of having the appropriate license and the necessity of substantial proof of infringement to prove plagiarism.

The court's decision highlights the significance of having the appropriate license and the necessity of substantial proof of infringement to prove plagiarism. This case has significant ramifications for the open source community since it emphasizes how crucial it is to carefully abide by licensing agreements and how crucial it is to develop robust community plans in order to guarantee the success of OSS projects.

A notable court case highlighting the difficulties and ramifications of software piracy and the enforcement of intellectual property rights in India is *Microsoft Corporation v. Vishal Mehta*[261]. Microsoft Corporation, a significant player in the worldwide technology industry, and Vishal Mehta, a person charged with violating Microsoft's software copyrights, are parties to the lawsuit. According to Microsoft, Vishal Mehta was involved in illegally using and distributing its proprietary software, namely Windows and Microsoft Office. According to Indian law, Vishal Mehta violated the company's intellectual property rights when he sold pirated versions of Microsoft software.

The following legal issues are related to copyright infringement in this case:

---

[258] Cadence Design System Inc. v. Avant! Corp., 1999 U.S. App. LEXIS 18302

[259] Lord Denning in Thornton v. Shoe Lane Parking Ltd., (1971) 1 All ER 686CA said: No customer in a thousand ever read the conditions. If he had stopped to do so , he would have missed the train or the boat

[260] Campus Eai India Pvt. Ltd. vs. Neeraj Tiwari & Ors on 29 March, 2019

[261] Microsoft Corporation v. Vishal Mehta on 7 November, 2013

Vishal Mehta violated copyright when he distributed and reproduced Microsoft software without authorization. Software that has been pirated is distributed illegally, undermining the market for legitimate software and resulting in significant losses for the copyright owners. The case highlights how important it is to enforce intellectual property rights to safeguard software creators' rights and promote innovation.

The case highlights multinational companies' difficulties when attempting to safeguard their intellectual property in foreign markets, especially in areas where piracy is prevalent. The Delhi High Court found in Mehta's favor, concluding that Mehta had obtained the software license legally and that there was no proof of violation. The court granted Mehta's request for summary judgment, so the matter was resolved without a full trial. The ruling highlights how crucial it is for India to have strong IPR enforcement laws.

It promotes lawful software use by incentivizing software businesses to take aggressive legal action against software piracy. The case draws attention to the persistent problem of software piracy in India and its effects on the software sector. The lawsuit fosters a just and competitive marketplace by defending software developers' rights. It highlights how important it is for businesses and consumers to be better informed about intellectual property rights.

By educating stakeholders on the legal and financial ramifications of intellectual property rights infringement, legal education programs can aid in decreasing instances of piracy. The historic case of *Microsoft Corporation v. Vishal Mehta* deals with the critical problems of software piracy and protecting intellectual property rights in India. The case highlights the court's role in preserving intellectual property rights and the legal options for software companies to safeguard their intellectual property. By establishing a precedent, the case supports the continuing initiatives in India to counter software piracy and advance a lawful and moral software industry.

Thus as cited in the beginning, an open-source license must be examined from the perspective of contract law since it is fundamentally a contract. This license is comparable to software shrink-wrap/click wrap licensing. While these permits have withstood legal examination in other countries, Indian courts have not discovered such change. Other jurisdictions' courts have established guidelines that may help settle disputes about open-source licensing. The terms visible through the shrink-wrap boxes containing the software application are contained in shrink-wrap licenses. A contract is created, and the buyer is presumed to have accepted the terms for using the program. [262] In contrast, unless the user hits the "I accept button" on the

[262] Supranote249 page 5

screen after downloading software from the Internet, he cannot access the program. A contract between the parties is ripened when he hits this button. This is known as a "clickwrap contract" in the industry. The legality of these contracts has not yet been discussed in Indian courts. Nonetheless, American case law has evaluated the enforceability of clickwrap licensing based on several circumstances[263]. The questions include "Did the consumer receive enough notice of the license? Was there enough competition in the market to equalize the parties' respective bargaining power, and did the customer have enough time to evaluate the license and return the software? Subject to specific procedural constraints, the enforceability of the shrink-wrap/clickwrap licenses has been upheld overall. The future of open-source licenses is still in the air, but case law regarding shrink-warp and click-warp licensing could provide a helpful basis for further research.

To exemplify the multifaceted nature of OSS we can look into the various questions that has popped up in this regard and which are duly answered by the honourable courts such as the follows:

Is the licensee of open-source software considered a consumer of the Consumer Protection Act,

Is the licensee of open-source software considered a consumer of the Consumer Protection Act, 1986 (CP Act)? This intriguing subject is likely to be discussed in India before the consumer courts established by this Act. This Act protects those who buy items to resell or use for commercial purposes but does not protect those who employ services or buy goods for consideration[264]. A customer can make a compensation claim if he discovers that the items he bought are defective or the services he hired are subpar[265]. Can this compensation claim be brought under the CP Act against the individual who licensed a flawed open-source program? Alternatively, can he profit from the licensee's further licensing of this open-source program after producing enhancements or changes to the program that would come under the purview of commercial or resell?

Tort law may be used to open source software licenses to establish liability for product liability, carelessness, property damage, and financial loss to companies using the software. Although there are well-established guidelines for judging negligence, how these guidelines should be applied to open-source software is still being determined[266]. The courts have determined the following elements of negligence: (a) duty of care to the plaintiff[267], (b) violation of such

[263] Id
[264] Section 2(d) of Consumer Protection Act
[265] Section14 of the CP Act
[266] Supranote249
[267] Donoghue v. Stevenson(1932)A.C 562

obligation[268], and (c) resulting damage[269]. How courts apply these ideas is yet to be determined. Sometimes, it will take much work to verify (a) who was responsible for the duty. (a) By whom was the duty breached? (c) To whom did the obligation belong?

The open-source application has brought up several previously unheard-of legal issues that may be discussed in court in the coming days. Open-source programs contain the source code incorporated within them, allowing others to make the required improvements and modifications. This would effectively raise questions about copyright violations. Like other software, the open-source program is licensed under a standard contract. It has been noted that severe or irrational terms are included in these conventional contracts. In order to lessen the strictness of these exemption clauses, the courts have established some guidelines. Will the standard terms found in the open-source distribution also be subject to these guidelines?

Will the idea of privity contrary to the fundamental tenets of open source be abandoned in the event of open-source content distribution? Whether the open-source community can continue to have the same degree of influence over open-source software is yet to be seen. Is the owner of the open source license deemed a consumer subject to the protections granted by the CP Act? Due to the lack of a statutory definition of merchantability, will the principles of the sale of goods be extended to open-source projects, and how would courts define the doctrine's application in these situations? Will open-source distribution be subject to the same negligence rules? If so, who is the rightful owner of the duty, and to whom does it belong?

Other challenges include:-

Vulnerability**:** Since many people have access to the source code, only some who work with the code do so with excellent intentions, which might lead to vulnerabilities. While most open-source contributors use their access to identify and address flaws, others can use it to introduce bugs and develop vulnerabilities that contaminate hardware and occasionally steal identities. This problem is avoided with proprietary software because the licensing business has stringent quality control procedures that guarantee security limits are not crossed[270].

Steep Learning Curve[271]**:** Using open-source software may be more complex than it once seemed. It is said that operating systems like Linux have a much higher learning curve and are impossible to become proficient with quickly. Many people find Linux challenging to work

---

[268] Nirmala v. Tamil Nadu Electricity Board, AIR 1984 Mad.201

[269] State of MP v. Asha Devi, AIR1989MP.93

[270] Thinksys. (2023, November 20). Benefits and challenges of going open source. *ThinkSys Inc.* https://thinksys.com/development/benefits-and-challenges-open-source-software/

[271] Ibid

with despite its technically superior to competing proprietary software. Finding the proper candidates to close the skills gap might be time-consuming.

Insufficient Assistance[272]**:** Despite the vast size of the open-source community, obtaining assistance to resolve issues may occasionally require additional time. Open source relies on the community to identify and repair problems. Thus, the issue is resolved as soon as the community has had a chance to consider it. Furthermore, the identity of the person who conceptualized, planned, and produced open-source software is often unknown. Determining who is responsible for such incidents is challenging when the program is not working. Organizations may also have unintended expenses related to paying for outside support services.

At times, the risk of abandonment exists for open-source and private technologies. The primary programmers involved can give up on the project and go on to the next great thing if they get disinterested in it. Another thing to remember is that before using any open-source software, a compatibility analysis must be done to determine if the hardware platform is compatible with the open-source platform. Notwithstanding these obstacles, open-source emphasizes teamwork, community involvement, and volunteerism all of which contribute to creating superior, highly personalized products utilizing cutting-edge technology.

Open-source communities are familiar with the obstacles to open-source software that have been discussed, and a wide range of subject matter experts have offered some potential remedies for what may prove to be a problematic scenario shortly. It is essential to carefully consider the Open Source licenses in light of the legal questions they raise and how they might work in tandem with the communitarian model of development—changes in copyright law. The fight to maintain a cap on the number of Open Source licenses will continue, and the pressure on commercial developers that release software into Open Source to issue their licenses will only increase. The community will continue to play a significant role in this process of evolution, although courts will probably play a more significant part in Open Source. Intellectual property rights will be protected, and the community's right to exploit intellectual property created for the benefit of the community will be represented at the table. Future developments in the Internet and our understanding of intellectual property law will be significantly influenced by the growth of the Open Source movement, Open Source licenses, and the concepts of copyleft and software freedom.

---

[272] Ibid

As discussed in this chapter, the legal issues and challenges regarding the validity of OSS concerning IPR are still in question as there still exist a gap and a potential insufficiency regarding the same concerning the aspect of protection rendered under traditional IPR laws to the OSS

CHAPTER 5

CONCLUSION AND SUGGESTION

5.1 CONCLUSION

Open Source software has gained a lot of attention, especially during the years 1998–2001. Open-source software is essential to the Internet's infrastructure and is increasingly prevalent in servers and operating systems. The contribution of Open Source software to the growth of the Internet cannot be overstated.

The debate over whether intellectual property rights (IPR) protection and open source software (OSS) or proprietary software (PSs) will advance the economic growth of developing nations is rife in both the business community and at the political level. IPR protection is more critical and contentious now than ever regarding politics and the economy. It is also essential for developing countries' policy-making in any area related to human development. India needs a comprehensive federal policy regarding using either OSS or PSs. Certain governments, notably West Bengal and Kerala, have adopted an aggressive strategy against using and disseminating PSs as part of their political goal. These states used to be vocally opposed to computerization and believed that computers would cause unemployment. As a result, these governments' governance and ability to produce skilled labor for the global workforce declined. The central government should immediately adopt a comprehensive policy, and the States should follow suit to their local needs and development, given the subject's significance and the debate's timeliness. It is possible to concentrate on providing OSS services for software development, advancements in related technologies, support, and maintenance through easy access to software products, and facilitating the transfer of software technologies at a reasonable cost by looking at OSS development at an organizational level. High levels of IT infrastructure and internal skill development are necessary for the organizational success of OSS.

Open source software has several noteworthy drawbacks in addition to its many notable benefits, which include higher lifecycle costs, compatibility problems, a lack of specialized support, and the perception of feature and user-friendliness limitations. These drawbacks include security, customizability, cost-effectiveness, and community collaboration. The argument over open source vs. proprietary software is complex; each model has advantages and disadvantages that companies should carefully consider in light of their particular needs and objectives.

For employees to successfully assume these new responsibilities, organizations implementing OSS must give them more training and development opportunities and a sustained commitment to the projects.

Open source projects that are now accessible span a variety of application domains, from conventional library management systems to cutting-edge solutions like Ganesha, DSpace, and Greenstone, which enhance them.

Regarding stand-alone programs that enhance conventional commercial library management systems, open-source software (OSS) is something to consider. Managers of libraries and systems librarians should keep an eye on this trend to see if there are any new advances.

Staff time and knowledge are the most crucial resources for the entire project. Most of the work involved in setting up and maintaining a digital library is labor-intensive, even if there is a lot of technology and computer usage. Emergency resources must be considered, and backup plans (such as standby equipment, temporary workers, etc.) must be created. Open Source Software (OSS) allows users to view and alter its source code without restriction.

Every open-source project has a proprietor. Every open-source project is subject to a license agreement: General public license, Berkeley Software Distribution Mozilla Public License, etc. An interface built into OSS makes it simple for users to establish their library collections. Collections can be created and provided remotely on a shared digital library host or locally from the user's web server.

All evidence points to a significant increase in the production and use of open source.

This will probably lead to more judges looking at the wording and legislation around open source, which should help clear up some ambiguities. More needs to be determined on how lawmakers and governments will respond to the popularity of open source legislation, which aims to rewrite certain sections of the law to consider software's unique qualities and essential role in the economy and society.

Governments can adopt free software for several reasons: financial savings, open standards, more robust, more adaptable, secure software, and social advantages. They must understand, however, the restrictions in indemnity provisions, the implications of implicit guarantees, and the responsibilities associated with using and redistributing Free Software (FOSS). Licensing options must be carefully considered when using public monies for software development. Large commercial markets can profit from traditional closed-source licensing, while developers who significantly alter or integrate the software can profit from dual licensing. More and more solutions becoming available should lead to a more significant and widespread use of FOSS.

The Open Source approach is receiving more attention and interest as a means of creating software that is more affordable, stable, and less prone to bugs. Concerns regarding the community-style approach to software development are beginning to fade.

Interest in open-source initiatives has increased due to increased financing and awareness of open-source work. As a result, the Open Source licenses—which serve as the movement's cornerstone—have also received more attention. However, there is a worry that as Open Source is used more and more in commercial settings and the boundaries of the licenses are pushed, the licenses, which were written in a possibly more informal era when things could be done without the involvement of lawyers, will no longer hold up. It is improbable that the Open Source licenses will be able to withstand legal challenges for an additional 15 to 20 years, as they have done thus far.

Consider the Open Source licenses carefully in light of the legal questions they raise and how they might work with the communitarian development model. Changes in copyright legislation, particularly UCITA, may have significant unforeseen effects on open-source licensing.

Additionally, be made to likely the fundamental Open Source licenses and a deeper consideration of the underlying principles. Furthermore, it is reasonable to anticipate that the majority of the licenses will be the subject of some debate and amendment to address some of the legal concerns that have come up and may later be brought up by UCITA or other new laws. The fight to maintain a cap on the number of Open Source licenses will continue, and the pressure on commercial developers that release software into Open Source to issue their licenses will only increase. The community will continue to play a significant part in this process of evolution, while the courts will probably play a more significant role in developing Open Source. Using licenses to safeguard freedom and a public interest approach to intellectual property, copyleft—possibly the most essential idea to come out of open source—will continue to be crucial to developing intellectual property law. Not only will intellectual property rights be protected, but there will also be a seat at the table to protect the community's rights to use intellectual property created for the benefit of the community. Growth of the Open Source Community: In the future, as the Internet and our understanding of intellectual property law expand, source licenses, copyleft, and software freedom will all be crucial concepts.

Through an in-depth analysis, the study reveals that the scope of open source is drastically increasing and that there is a need for a proper legal framework. Copyright law largely governs open-source licensing restrictions; norms are established by international treaties such as the Berne Convention and agreements under the World Trade Organization (WTO). The copyright

and contract laws of each nation govern open-source licensing. Nonetheless, there are loopholes in the law, including the absence of particular clauses addressing difficulties with enforcement, ambiguity in the application of the law, and problems with cross-border enforcement and compliance. To tackle these obstacles, the legislation may include specific clauses or modifications in copyright statutes, set up dispute resolution procedures, encourage awareness and education among interested parties, and unify global agreements and norms.

The study concluded that there needs to be an appropriate legal framework for open-source software and its licensing at the international and national levels. At the global level, countries like the U.S. and E.U. were developing legal frameworks regarding the conflict arising from the OSS about IPR, such as UCITA. As per India, the Copyright Act of 1957 was used for the judicial interpretation of the OSS and is insufficient and ineffective in some cases.

Generally speaking, copyright law gives the author of a work the right to use it without registering it. With proprietary software licensing, the buyer only gets a license to use the program; the inventor retains all copyright and related intellectual property. On the other hand, when the software is shared or redistributed, Open Source Software (OSS) licenses from organizations like the Free Software Foundation (FSF), GNU, and Apache Software Foundation (ASF) guarantee that the author's copyright is respected.

Copyright infringement may arise from violating the terms and conditions of OSS. In legal instances such as Cisco v. Free Software Foundation, Cisco was required to supervise adherence to the free software licenses and put a portion of the proprietary code into the public domain. Suppose OSS components are added to proprietary software. In that case, the complexity of compliance rises, increasing the risk of third-party infringement lawsuits and further disputes between the software's distributors and the I.P. owner. The original licensors may take legal action if OSS is broken, which could harm a company's reputation. As a result, handling the complexity of using an OSS requires extreme caution.

5.2 SUGGESTIONS: -

It is advised that users or entities intending to integrate open source software (OSS) into their systems thoroughly examine the license terms for these items before creating custom software for industry usage or internal use. This will provide a better understanding of how intellectual property rights affect copyright enforcement and protection for the software. Additionally, they should consider concerns regarding intellectual property law, highlighting how it may hinder innovation and restrict the work of other innovators while promoting the original authors'

commercial exploitation. Entities must ensure that the required notices, source code availability, and other obligations are being upheld, as the last thing they would want is to face legal action for copyright infringement or, in a worst-case scenario, have to pay damages and have their software license revoked.

In light of the above analysis, the following suggestions are put forward as part of the study based on the inferences arrived at by conducting the same.

1. Ensure adherence to international principles and accords and adopt international models of protecting OSS.
2. Ensure timely participation and cooperation of the different wings of Government for bringing about timely changes in the existing IPR laws
3. Enact a new law protecting open-source software and licensing therein.
4. Existing IPR laws are to be amended to incorporate the protection of open-source software and allied aspects.
5. Promoting FOSS in government is the first step. All I.T. providers are required by the government's open-source software adoption strategy to submit proposals utilizing open-source solutions. A policy framework will go one step further by publicly awarding departments that implement FOSS efforts with recognition, like a particular category under the Digital India Awards, and by formally giving FOSS-specific indicators more weightage in the assessment criteria in RFPs (request for proposals).
6. India needs to make the most of its technological might. Open-source software is, in fact, beneficial to India's national interest, considering the evolving technology industry's politics and economy. It is significantly more fruitful to concentrate on open-source initiatives rather than trying to assert technological sovereignty by demanding localization and completely rewriting the rules. Reducing reliance on multinational technology firms and their supporting governments might be achieved by this dependable method.
7. Encouraging Open-Source Economy: India needs to push several governmental levers to encourage developers and businesses to invest more in creating open-source software. It should produce globally competitive developers and businesses that establish themselves as vital components of the tech industry. In the post-pandemic era, the gig economy will expand. Thus, you are encouraged to contribute to this area.

8. The role of technology institutions is to encourage students to work on open-source projects. This is especially true for engineering colleges. Maintaining a robust open-source ecosystem is, in fact, a social responsibility issue for a nation with a significant I.T. sector. If acknowledged as fulfilling Corporate Social Responsibility (CSR) obligations, more developers will gravitate toward open-source projects. It will lessen the likelihood that a vital component of the global information infrastructure will depend on a few people.
9. A Center of Excellence for FOSS: To serve as a home for FOSS-led innovation in India, a reputable institutional anchor that can unite disparate FOSS groups and champions is required. One organization that helped Kerala become a FOSS pioneer was the International Centre for Free & Open Source Software (ICFOSS). A national "FOSS Centre of Excellence" can bring together funding, materials, and assistance with capacity-building, providing the much-needed impetus to develop top-notch "made in India" FOSS goods.
10. Start educational programs to help the OSS community understand the value of intellectual property rights protection. Offer training and materials on best practices for handling and safeguarding intellectual property in open-source software projects. Educate companies, developers, and the public about the legal ramifications of utilizing and contributing to open-source software through public awareness campaigns.
11. Establish alternative dispute resolution (ADR) procedures to address IPR conflicts about OSS. All parties concerned should be able to access, utilize, and economically benefit from these systems. Encourage arbitration and mediation as a less combative and more cooperative form of conflict resolution for OSS-related problems.

Overall, OSS is an inadequate paradigm since it must provide a sufficient product development incentive. Adoption of OSS is accelerating across all domains and businesses, from business to government. Thus, gaining a thorough grasp of the best way to design an open-source software project to increase its likelihood of success is a crucial academic and practical undertaking. By putting these recommendations into practice, the legal problems and difficulties OSS presents in the context of IPR can be handled more skillfully, creating an atmosphere more conducive to the expansion and advancement of open-source software.

BIBLIOGRAPHY

BOOKS/REPORTS/ARTICLES

- David McGowan, Legal Implications of Open-Source Software, 2001 U. ILL. L. REV. 241 (2001).
- Dennis M Kennedy, A primer on Open Source Licensing Legal Issues: Copyright, Copyleft and Copyfuture, 2001, Intellectual Property: Policy Considerations From a Practitioner's Perspective (Volume XX, No. 2)
- Di Penta, M., German, D. M., Guéhéneuc, Y., & Antoniol, G. (2010). *An exploratory study of the evolution of software licensing*. https://doi.org/10.1145/1806799.1806824
- Ballhausen, M. (2019). Free and open source software licenses explained. *Computer*, *52*(6), 82–86. https://doi.org/10.1109/mc.2019.2907766
- Steve H. Lee, Open Source Software Licensing (Pre-Publication Version as of Apr. 28, 1999), https://cyber.harvard.edu/openlaw/gpl.pdf (last visited June 22, 2024).
- Public licenses: open source, Creative Commons and IP pledges. (2022). In *Cambridge University Press eBooks* (pp. 592–636). https://doi.org/10.1017/9781009049436.020
- Nabi Hasan, Issues and Challenges in Open Source Software Environment with Special Reference to India, ICAL 2009 – Technology, Policy and Innovation (2009).
- Kemp, R. (2009). Current developments in Open Source Software. *Computer Law and Security Report/Computer Law & Security Report*, *25*(6), 569–582. https://doi.org/10.1016/j.clsr.2009.09.009
- Raymond T. Nimmer, Legal Issues in Open Source and Free Software Distribution, in the Law of Computer Technology ch. 11 (1997, 2005 Supp.).
- Josh Lerner & Jean Tirole, the Scope of Open Source Licensing, 21 J.L. Econ. & Org. 20 (2005)
- Mikko Valimaki, The Rise of Open Source Licensing- A challenge to the use of Intellectual Property in the Software Industry, 2005, Turre Publishing, A Division Of Turre Legal Ltd.
- Halina Kaminski & Mark Perry, Open Source Software Licensing Patterns, University of Western Ontario (2007).

- Hendrik Schöttle, Open Source License Compliance—Why and How?, IEEE Computer Soc'y (Aug. 2019)
- Knut Blind, Mirko Böhm & Nikolaus Thumm, Open Source Software in Standard Setting: The Role of Intellectual Property Right Regimes, in Open Source Law, Policy and Practice (Amanda Brock ed., Oxford University Press 2022), DOI: 10.1093/oso/9780198862345.003.0011.
- Pearl: J. Libr. & Info. Sci. (Oct. 2012), Recent Growth and Developments in Open-Source Software on Digital Libraries in India, 6 DOI: 10.5958/j.0973-7081.6.4.019
- Meera Sarma, Development and Use of Open Source Software in India, in ICTs in Developing Countries (B. Dey et al. eds., 2016)
- St Laurent, A. M. (2004, August 16). *Understanding Open Source and Free Software Licensing* (1st ed.) "O'Reilly Media,
- Brett Smith, A Quick Guide to GPLv3, Free Software Foundation, Inc. 2007
- D. M. Germ´an and J. M. Gonz´alez-Barahona. An empirical study of the reuse of software licensed under the GNU General Public License. In *Proceedings of the International Open Source Systems Conference (OSS'09)*, pages 185–198. Springer, 2009
- Bibek Debroy and Julian Morris, *Open to Development: Open Source Software and Economic Development*, Rajiv Gandhi Foundation, http://www.rgfindia.com/rgf2/text/oss.pdf., 2.06.24
- Kd, R. (2007). Is the Future of Software Development in Open Source? Proprietary vs. Open Source Software: A Cross Country Analysis. *Journal of Intellectual Property Rights*, *12*(2). http://nopr.niscair.res.in/bitstream/123456789/267/1/JIPR%2012%282%29%20%282007%29%20199-211.pdf
- De, R. (2009). Economic impact of free and open source software: A study in India. *Interop, Mumbai, October*, 7–9.
- Suzor, N., Fitzgerald, B and Perry, M.,(2007), legal issues for free and open source software in government, IFIP International Federation for Information Processing, Volume 234, Open Source Development, Adoption and Innovation, pp. 353-354
- Brock, A. (2022) (n.d.). *Open Source Law, Policy and Practice*. Oxford University Press, USA, DOI: 10.1093/oso/9780198862345.001.0001

- Punita Bhatt a. (n.d.). *AliJ.Ahmad b*.(2016)Social innovation with open source software: User engagement and development challenges in India, Elsevier Ltd.
- Jorge Colazo, Yulin Fang(2009) Impact of License Choice on Open Source Software Development Activity, Journal of the American Society for Information Science and Technology, 60(5):997–1011
- Framework for Adoption of Open Source Software Ine-Governance Systems, *April 2015 Version 1.0*, Government of India, Ministry of Communications and Information Technology, Department of Electronics and Information Technology, New Delhi –110 003
- Ranjan Kumar. (n.d.). In *Subhash Kumar, Sanjay K. Tiwari*. (2018) Adoption of Free and Open Source Software in India, International Journal of Applied Engineering Research ISSN 0973-4562 Volume 13, Number 16 pp. 12725-12731
- Vikrant Narayan Vasudeva (May 2012) Open Source Paradigm and Intellectual Property Rights, journal of Intellectual Property Rights, Vol 17, November 2012, pp 511-520
- Eve, M. P., & Suber, P. (2014). *Open access and the humanities: Contexts, controversies and the future*. Cambridge University Press.
- Open Source Software Country Intelligence Report, 2021

INTERNATIONAL INSTRUMENTS

- The Agreement on Trade- Related Aspects of Intellectual Property Rights(1994)
- The World Trade Organization
- Berne Convention for the Protection of Literary and Artistic Works(1886)
- WIPO Copyright Treaty(1996)
- European Union Copyright Directive(2001)

STATUTES

- Indian Contract Act, 1872
- Indian Copyright Act, 1957
- The Uniform Electronic Transactions Act (UETA), 1999
- The Uniform Computer Information Transaction Act (UCITA), 1999
- The U.S Copyright Act, 1976

- The Information Technology Act, 2000
- The Indian Patent Act, 1970
- The Digital Personal Data Protection Act, 2023

NATIONAL POLICIES AND GUIDELINES

- National Policy on Open Standards for E-Governance (2008)
- Policy on Adoption of Open Source Software for Government of India (2015)
- Framework for Adoption of Open Source Software in E-Governance Systems (2016)

WEB RESOURCES

- Pal, S. (2021, August 16). History of Open Source Software (with an interactive timeline). Btw. https://www.btw.so/blog/history-of-open-source-software
- Opensource.com. What Is Open Source? | Opensource.Com. https://opensource.com/resources/what-open-source. Accessed 21 Dec 2023
- Opensource.com. What Is Linux? | Opensource.Com. https://opensource.com/resources/linux. Accessed 21 Dec 2023
- What Is Linux?" Linux.Com, https://www.linux.com/what-is-linux/ Accessed 21 Dec 2023.
- What Is Apache? | Definition from TechTarget." WhatIs, https://www.techtarget.com/whatis/definition/Apache. Accessed 29 Dec 2023.
- Welcome! - The Apache HTTP Server Project. https://httpd.apache.org/. Accessed 29 Dec 2023.
- "What Is Apache? | Definition from TechTarget." WhatIs, https://www.techtarget.com/whatis/definition/Apache. Accessed 29 Dec 2023.
- 20 Years Ago, Mozilla's Move to Open-Source Its Browser Was Radical. Now Even Microsoft's a Convert." CNET, https://www.cnet.com/culture/mozilla-open-source-firefox-move-helped-rewrite-tech-rules-anniversary/. Accessed 29 Dec 2023.
- The Future of the Open Web — White Paper." Mozilla, https://www.mozilla.org/en-US/foundation/reimagine-open/. Accessed 29 Dec 2023.
- What Is Firefox? | Definition from TechTarget." WhatIs, https://www.techtarget.com/whatis/definition/Firefox. Accessed 29 Dec 2023.

- Chromium. https://www.chromium.org/Home/. Accessed 30 Dec 2023
- [1] "What Is Chromium Browser Used For?" Lifewire, https://www.lifewire.com/chromium-web-browser-4171288. Accessed 30 Dec 2023.
- What Is LibreOffice? | LibreOffice - Free and Private Office Suite - Based on OpenOffice - Compatible with Microsoft. https://www.libreoffice.org/discover/libreoffice/ Accessed 30 Dec 2023.
- Who Are We? | LibreOffice - Free and Private Office Suite - Based on OpenOffice - Compatible with Microsoft. https://www.libreoffice.org/about-us/who-are-we/. Accessed 30 Dec 2023.
- AlternativeTo, OnlyOffice, https://alternativeto.net/software/onlyoffice/about/ (accessed on Jan 3, 2024).
- Online Office Applications for Business. https://www.onlyoffice.com/. Accessed 3 Jan 2024.
- What Is MySQL? https://www.oracle.com/in/mysql/what-is-mysql/. Accessed 3 Jan 2024
- "What Is MySQL?" GeeksforGeeks, 11 Oct. 2022, https://www.geeksforgeeks.org/what-is-mysql/
- Welcome to Python.Org." Python.Org, , https://www.python.org/about accessed on 4 Jan 2024
- "Introduction to Python." GeeksforGeeks, 6 Nov. 2015, https://www.geeksforgeeks.org/introduction-to-python/. Accessed on 4 Jan 2024
- Java, what is Java? https://www.java.com/en/download/help/whatis_java.html (accessed on Jan 22, 2024).
- Guru99, 23 Best CMS Platforms (2024 Update), Guru99, https://www.guru99.com/best-cms.html (accessed on Jan 22, 2024).
- Newcomer, Colin. "21 Best CMS Software to Build a Website (And Manage Content Effectively)." Kinsta®, 20 July 2020, https://kinsta.com/blog/cms-software/. Accessed on 22 Jan 2024
- GIMP." GIMP, https://www.gimp.org/. Accessed 22 Jan 2024
- admin. "What Is Inkscape? All You Need to Know." OnWorks, 16 Dec. 2022, https://www.onworks.net/blog/what-is-inkscape/. Accessed on 22 Jan 2024
- Developers, Inkscape Website. About | Inkscape. https://inkscape.org/about/. Accessed 22 Jan 2024.

- mijacobs. What Is Git? - Azure DevOps. 28 Nov. 2022, https://learn.microsoft.com/en-us/devops/develop/git/what-is-git. Accessed on 24 Jan 2024
- Singh, R. (2023, December 13). What is Eclipse and use cases of Eclipse? DevOpsSchool.Com. https://www.devopsschool.com/blog/what-is-eclipse-and-use-cases-of-eclipse/ accessed on 26 Jan 2024
- Open source licenses: Types and comparison. (n.d.). Snyk., from https://snyk.io/learn/open-source-licenses/ Accessed on Jan 23, 2024
- The gnu general public license v3. 0—Gnu project—Free software foundation. (n.d.)., from https://www.gnu.org/licenses/gpl-3.0.en.html accessed on June 22, 2024
- Open Source Software Licenses 101: The LGPL License - FOSSA. (2022, October 26). Dependency Heaven. https://fossa.com/blog/open-source-software-licenses-101-lgpl-license/ or (Open Source Software Licenses 101: The LGPL License - FOSSA, 2022)
- Open Source Software Licenses 101: The LGPL License, FOSSA, 2022, https://fossa.com/blog/open-source-software-licenses-101-lgpl/ (accessed on Jan 25, 2024
- Horcasitas, J. (2021, November 2). Understanding Open-Source Software Licenses. DigitalOcean. https://www.digitalocean.com/community/tutorials/understanding-open-source-software-licenses. Accessed on April 2024
- GNU Affero General Public License - GNU Project - Free Software Foundation. (n.d.). https://www.gnu.org/licenses/agpl-3.0.en.html accessed on 3 Feb 2024
- What is mit license? (n.d.)., from https://memgraph.com/blog/what-is-mit-license accessed on 5 Feb, 2024
- Librarian, U. L. S. (n.d.). Guides: Open licenses: creative commons and other options for sharing your work: bsd licenses. Retrieved Feb 6, 2024, from https://pitt.libguides.com/openlicensing/BSD accessed on 5 Feb, 2024
- What is BSD licenses? | Definition from TechTarget. (n.d.). WhatIs., from https://www.techtarget.com/whatis/definition/BSD-licenses Accessed on Feb 22, 2024
- The Apache License (v2) - An Overview. (2012, May 14). http://oss-watch.ac.uk/resources/apache2 accessed on 22 Feb 2024
- Academic Free License v. 3.0. Open Source Initiative. https://opensource.org/license/afl-3-0-php (accessed on 2023, June 27).

APPENDIX

grammarly Report: Chapter 1,2

Chapter 1,2

by VYDEHI P

General metrics

79,164	11,944	675	47 min 46 sec	1 hr 31 min
characters	words	sentences	reading time	speaking time

Score

97

This text scores better than 97% of all texts checked by Grammarly

Writing Issues

155	63	92
Issues left	Critical	Advanced

Plagiarism

8 %

44 sources

8% of your text matches 44 sources on the web or in archives of academic publications

Chapter 3

by VYDEHI P

General metrics

37,713	5,484	331	21 min 56 sec	42 min 11 sec
characters	words	sentences	reading time	speaking time

Score

This text scores better than 98% of all texts checked by Grammarly

Writing Issues

46	18	28
Issues left	Critical	Advanced

Plagiarism

13 sources

4% of your text matches 13 sources on the web or in archives of academic publications

Chapter 4

by VYDEHI P

General metrics

98,054	14,916	792	59 min 39 sec	1 hr 54 min
characters	words	sentences	reading time	speaking time

Score

This text scores better than 99% of all texts checked by Grammarly

Writing Issues

121	32	89
Issues left	Critical	Advanced

Plagiarism

41 sources

5% of your text matches 41 sources on the web or in archives of academic publications

Chapter 5

by VYDEHI P

General metrics

15,878	2,328	109	9 min 18 sec	17 min 54 sec
characters	words	sentences	reading time	speaking time

Score

99

This text scores better than 99% of all texts checked by Grammarly

Writing Issues

11 Issues left

Critical

11 Advanced

Plagiarism

2 %

3 sources

2% of your text matches 3 sources on the web or in archives of academic publications