ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 11, November 2016

Comparative Study: Proprietary Software vs. Open Source Software

Gauri Sood, Shipra, Dr. Rachna Soni

Asst. Professor, Dept. of Computer Science, DAV College for Girls, Yamunanagar, Haryana, India

Asst. Professor, Dept. of Computer Science, DAV College for Girls, Yamunanagar, Haryana, India

HOD, Dept. of Computer Science, DAV College for Girls, Yamunanagar, Haryana, India

ABSTRACT: The Life cycle for the expansion of traditional commercial software is well established and discussed in various research papers in detail. On the contrary, in case of Open Source Software (OSS),life cycle for the development is not being discussed in much detail since there exist no standardized life cycle approach for Open Source Software (OSS) development. Different researchers and developers have projected various life cycles for the development of OSS with respect to their own development experience, requirement or application. The popularity of the open source software development in the most recent decade, has brought about an increased interest from the industry on how to use open source components, participate in the open source community, develop business models around this type of software development, and learn more about open source development methodologies. The main focus of this paper is on reviewing the Open Source Software (OSS).

KEYWORDS: Open Source Software, Proprietary Software.

I. INTRODUCTION

A. *OPEN SOURCE SOFTWARE*

OSS[1,2] is defined as the software whose source code is available along with the software and user has the liberty to run, copy, distribute, study, alter and perk up the software under the licensing policies of OSS. The development of OSS gains esteem due to the wide accessibility of the internet facility to each and every region of the world, parallel development, peer review, parallel debugging, specialist developers, and feedback. Many different developers, user, or co-developers can contribute in the development of the OSS. The development of OSS is always initiated by single developer or a solo group, who initiates the development of software for its own “personal itch”. The development of OSS is dissimilar from the traditional industrial software also called as “closed software”. It is found by various researchers that the traditional Software Development Life Cycle (SDLC) [3,8] and development processes models cannot be used for the progress of OSS. Various researchers and practitioners are working on developing the standard development life cycle of OSS.

Open source software is by description software for which users have access to the source code [8]. This distinguishes it from the current common practice by commercial software publishers of only releasing the binary executable versions of the software. Largely open source software is also distributed at no cost with limited restrictions on how it can be used; therefore the term “free” when used,carries two meanings: 1) free of cost and 2) free to do with the software as you desire (i.e., most importantly. free to read the code). It is claimed that open source development produces more bug-free code, quicker than closed proprietary developed code, although this has yet to be conclusively demonstrated. Open source software development teams, are usually comprised of volunteers working not for monetary return, but for the pleasure and pride of being part of a successful virtual software development project. Team members often come from around the world and hardly ever meet one another face-to-face. The open source projects are self-organized, employ enormously rapid code evolution, massive peer code review, and rapid releases of prototype code. Several of these practices are counter intuitive and the reverse of what conventional software engineering holds as the accurate processes for the production of high quality code. The Open Source Software movement is a prototypical illustration of a decentralized self-organizing process. There is no inner control or central planning and supporting their software with no monetary compensation for their efforts.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 11, November 2016

A.1 FEATURES OF OSS

A Source software has several features of their own. Some of the features of Open Source Software has been discussed below [8].

1. **Free Redistribution:** The license does not limit any party from selling or giving away the software as a component of an cumulative software distribution containing programs from numerous different sources. The license shall not require a royalty or additional fee for such sale.
2. **Source Code:** The OSS includes source code and allows distribution in source code as well as compiled form.
3. **Derived Works:** Nearly all license allow modifications and derived works to be disseminated under the same terms as the license of the original software.
4. **No Discrimination against Fields of Endeavour:** The license does not restrict anybody from making use of the program in a specific field of endeavour. For example, it will not restrict the program from being used in a business or from being used for genetic research.
5. **Is not specific to a Product :** The rights attached to the program does not rely on the program's being part of a particular software distribution. If the program is used or distributed within the terms of the program's license, all parties to whom the program is redistributed have the similar rights as those that are granted in conjunction with the original software distribution.
6. **Not Restrict Other Software :** The license does not place limitations on other software that is distributed along with the licensed software. For example, the license does not insist that all other programs distributed on the same medium must be open-source software.
7. **Technology-Neutral :** Most of OSS is technology neutral.

B. *PROPRIETARY SOFTWARE*

The word proprietary is derived from the Latin word “proprietas” meaning property. Proprietary software is a software that is owned by an individual or a company (generally the one that developed it). There are almost always major limitations on its use, and its source code is almost always kept secret. Source code is the form in which a program is formerly written by a human using a programming language and prior to being converted to machine code which is straightforwardly readable by a computer's cpu (central processing unit). It is essential to have the source code in order to be able to modify or improve a program. Virtually all Microsoft software is proprietary, counting the Windows family of operating systems and Microsoft Office. This includes software that is given away at no charge, such as Internet Explorer. Other main producers of proprietary software include Adobe, Borland, IBM, Macromedia, Sun Microsystems and Oracle. The restrictions on the use of proprietary software are generally enumerated in the end user license agreements (EULAs) that users must consent to. For software provided by large companies, EULAs are generally long and complicated contracts. Among the most familiar of the prohibitions for such programs are making unauthorized copies, using it on more than a certain number of computers and reverse engineering it.

II. RELATED WORK

Open source software has been around since the very commencement of electronic computing. In the early days of information technology it was pretty natural and financially sound for developers to share source code among very few and very costly computing machines. As the machines became smaller, diversified and cheaper, the number of developers grew, and the source code, in general, became more multifaceted. Development of free software was especially flourishing in the educational environments. Barkley Software Distribution (BSD) is license developed for distribution of BSD version of Unix operating system urbanized by University of California Berkeley from 1977 to 1995 in collaboration with AT&T labs, as described by Raymond (2001) [3]. At the initiation of the development, code was shared between AT&T and Berkeley. Due to a divestment in 1984 it became a proprietary AT&T product. Since the commencement of 1980s, the thought of closesourced/proprietary software became mainstream, captivating the place that free software sharing has held for a long time. The open source supporters went to create their own organizations such as free software foundation (FSF) founded by Richard Stallman, as described by Weber (2004). The FSF did not have a desired influence on bringing back open source software development to the mainstream. Though this situation was about to change with the successful release of Linux kernel. The system primarily developed by the Linus Torvalds as a part of academic project, with the support of the developers community came to produce very complicated, sophisticated software that was free for everyone to use. Eric Raymond was very much enthused by this set of events, that in his now famous book ”The Cathedral and the Bazaar” he talks about the significance of Linux, as

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 11, November 2016

it was the very first time the open source developer community showed that not only complicated and sophisticated software can be built in such manner but also that business models can be built around such fashion of software development and distribution. In 1998, Raymond was one of the main contributors to the Open Source Initiative (OSI) [3,9], an organization that is envisioned as open source educational and advocacy organization. Several companies have followed the suit, and determined to open source a part of their proprietary software as a part of business strategy to deal with the competition. Thus, amongst the preliminary suitors we can find Netscape corporation, who by means of open sourcing Netscape internet browser tried to battle against closed source and free of charge distribution of Microsoft's Internet Explorer (Raymond (2001))[20.23]. In 2006 Africa have anticipated as they have changed their laws to protect IPRs; using non-proprietary software will enable them to deploy extensive computerization without making large payments to suppliers from the developed countries[3,8]. In 2007 McGhee discusses the rise of free and open source software (FOSS) as a substitute to traditional commercial software products and how this new software brings a host of unique legal issues that must be evaluated before any gains may be realized[4,7]. Mr. McGhee reviews the case law and explains when companies should be aware of the particular situations that emphasize FOSS licensing compliance. Stol and Ali Babar (2009) have made a review of the broad area of "open source" from the conference on Open Source Systems, OSS[7,9]. They manually selected empirical papers from the conference and investigated them. In the past 10 years, a lot of companies have entered the open source business arena, using some of the business models proposed by Raymond (2001). Unfamiliar with the surroundings ,companies had very quickly to readjust their way of doing business in order to ripe several perceived benefits of open source trends. Besides open sourcing software, companies tend to take part and contribute to open source projects, as well as adopt some of software development methodologies such distributed and voluntary based development society as open source utilizes.

III. PROPRIETARY SOFTWARE VS OPEN SOURCE SOFTWARE

Details	**Proprietary Software**	**Open Source Software**
Cost	Varies from only some thousand to a few hundred thousand dollars, depending on the complexity of the system needed. This fee is made up of a base fee for software, integration and services and annual licensing/support fees. This cost may be prohibitive for some; on the other hand what the user is paying for is a more customized product from a trusted brand that includes advanced levels of security and functionality, incessant innovation, a greater ability to scale, on-going training and support and a lesser requirement for technical skills.	OSS comes at a little cost because of Open source software. We don't need an expensive software or hardware to run the system. Organizations can employ this system as long as they like, without thinking of paying any set up, activation and monthly subscription charges.
Service and Support	Service is perhaps the supreme advantage of using proprietary software. Proprietary software providers offer ongoing support to users, a crucial selling point for users without technical expertise. If the user manual or guide is not enough, or if a user experiences a difficulty with the software, there is an immediate point of call to turn to for assistance. There is a certain fall in the risk undertaken with proprietary software because users are working with companies that are viable, and individuals with intimate knowledge of the products and services being	Although Service is one of the key issues regarding open source software. Open source software relies on its online community network to deliver learning support by means of forums and blogs. While there are loyal, massive and engaged online communities that users are turning to. This needs a little basic knowledge and skill set from the user to be aware of feedback from online community and resolve them. Occasionally the trouble shooting is faster than those of proprietary software. But users who question should know what their real problem is, else, they

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 11, November 2016

	used should any questions arise. Because service is one of the key reason users choose proprietary over open source software, many proprietary software providers compete on service, mounting the bargaining power of buyers and thereby increasing customer service levels among providers.	will not receive the right feedback.
Innovation	Proprietary software providers do not permit users to view or alter the source code. While this may be viewed as a disadvantage to a few ,it ensures the security and reliability of the software. Furthermore, several proprietary software providers customize software for specific users to offer more flexibility while investing in R&D in order to regularly propose new products and upgrades. Proprietary software providers have online user communities that create value by sharing new ideas, strategies and best practices through feedback mechanisms such as forums and surveys, which also foster innovation and let the product to adapt with changing needs. This innovation comes fully tested, and is available to all users of the software. It does not require venture in R&D or the technical understanding of source code, and assistance with implementation in general part of the package. Because vendors must ensure their software does not become redundant, users also benefit from the kind of targeted innovation undertaken- continuous investment in R&D rather than "innovation for innovations sake", business focused rather than technology focused.	OSS enables innovation by providing users the freedom and flexibility to adapt the software to suit, without restriction. However, innovation may or may not be passed on to every user of the software. It is a users prerogative whether they desire to share their innovation with any online communities, and users must be actively participating in these communities to become conscious of such innovations. It has been debated whether customized changes to the original source code limit the future support and development of the software, as these can potentially result in a limited ability to apply future updates, fixes or modules intended to improve the software, leaving the user with a version that may have irresolvable issues. It is pertinent to note that open source software providers in the initial stages generally struggle to attract large scale R&D funding.
Security	There is always a debate on security. A lot of proprietary software based developed from proprietary operating systems are perceived comparatively less secure to those from OSS. But the total solution from proprietary software is viewed as secure as it is developed in a controlled environment by a concentrated team with a common direction. Moreover, the source code may be viewed and edited by the team only, and is heavily audited, eliminating the risk of back door Trojans and reducing the risk of several bugs or issues with the software.	Open source operating software is apparently the most secure OS. Linux being a classic example. However total solutions developed from OSS are perceived less secure as the source code is already available freely to all, hence there may be chances for sabotage. Great players using OSS have robust security policies, hence security in big organizations using OSS is not a concern. Earlier, open source software was not always peer reviewed or validated for use. But currently, with the rising importance of OSS, they are reviewed and corrected. Examples of Drupal, wordpress, Joomla etc are there to see.
Usability	Proprietary software normally employs expert	Earlier OSS had been greatly criticized for its

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 11, November 2016

	usability testing, and as the software is usually aimed at a more targeted audience, and therefore further tailored, usability is generally ranked quite high. In addition, detailed user manuals and guides are provided. This enables quicker training and provides an immediate reference, allowing users to move all along the learning curve more quickly. Supporting services include seminars, targeted training courses and extensive support to aid maximise use of the software. It is also important to note that while many people see proprietary software as "closed", todays proprietary software offers a vast array of mechanisms for enhancement by third party systems and developers	lack of usability, as generally, the technology is not reviewed by usability experts and does not cater to the immense majority of computer users. Nowadays, OSS is tested by the community for usability and superior solutions are being developed globally. Though, open source software does not legally need documentation such as user manuals or guides, hindering the construction of such tools, they are now well documented.
Standards	Describes the software interfaces, protocols and electronic formats that are developed by and controlled by a given company and have not been made freely accessible for adoption by the industry. A number of proprietary software uses proprietary standards, i.e. non-public interfaces or electronic formats. When an interface, a protocol or an electronic format is non-public, the proprietor of the proprietary interface controls it, including when and how the interface changes, who can adopt it, and how it is to be adopted (consequential in user lock-in).	"Open standards" software interfaces, protocols, and electronic formats that are openly documented and have been accepted in the industry either through formal or de facto processes, and which are freely accessible for adoption by the industry. The open source community has been a head in promoting and adopting open standards. Some of the success of open source software is due to the accessibility of worldwide standards for exchanging information, standards that have been implemented in browsers, file sharing applications, email systems and many other tools. Without open standards it would be impossible to interrelate and exchange information on the Internet.
Availability	These are available from their respected companies that own the rights to the packages. Sometimes, trial versions are available for free download and testing	These are freely available over the net. Several OSS are also developed into a limited proprietary software with 24X7 support from online community and the developer as well
Transparency	PS does not present an open look to the internal structure. Only user interfaces are provided to work with it. User cannot know the internal processing and other details.	The source code of OSS is freely accessible along with the product. Any person can read, modify, build and distribute a tailored version of original product. Thus, it gives a transparent look at the core structure of the product.
Reliability	PS is developed by specialized teams at vendors end only. Only finished products are provided at outlets. Since there is no un-authenticated modification, the result is always reliable.	Since OSS are available on a number of unverified websites and even most of these distributions may be modified by any technologically sound user, every distribution is reliable in terms of security, robustness,

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 11, November 2016

		performance. The reason is that if a user adds/modifies some component; it may works good individually, but, may clash with other components and ultimately degrade the product.

IV. REASONS BEHIND OSS NOT FOLLOWING TRADITIONAL DEVELOPMENT MODEL

The traditional commercial software can be developed by using the various classical conventional development models like waterfall model, iterative model, prototype model, spiral model etc. The development of OSS cannot be done by using these traditional development life cycles models because of these following reasons [3,8].

1) Requirements may not be properly planned at the start of the development process.
2) Number of users, co-developer, and developer participating in the development procedure may be large and moreover different users may geographically locate at different location.
3) No accurate specification and documentations are available for OSS development.
4) OSS development process may or may not consider all the phases of traditional development process of closed software.
5) Difference in motives of development of OSS and closed software.
6) OSS need collaborative team structure for its development.
7) The way of OSS evolution is different, as OSS allows changes to be included.
8) OSS Development is in open environment.
9) In OSS development the key focus is on coding and implementation.
10) No apparent design process.
11) No explicit list of deliverables, schedule and project plan.
12) User can select work of their own preference in OSS. .

V. CONCLUSION & FUTURE SCOPE

OSS development has gained so much popularity due to a variety of features it provides. Many successful OSS developments have attracted a large fraction of the researchers for their own personal interest and to earn geek fame. The development of OSS is at peak at present. But the standardized life cycle for the development is not published yet as the SDLC for traditional software exists. In this paper an attempt is made to review the OSS proposed by various researchers in various research articles. This paper gives a compact study of OSS. The comparison between Proprietary and open source software provides the better understanding about OSS. When contrasting previous literature on older "platforms-wars", such as the ones from the PC and game-console industries, with the current and under-studied mobile platforms-war, we empirically notice that many of the market players remain the same (Microsoft and Apple). There is a scenario of convergence: same firms push for similar technological standards across different platforms, i.e. Microsoft Windows within X-box, Surface Tablets, PC, Netbooks and Mobile phones. This convergence between industries remains unexplored by academia. Interesting research questions dealing with the implications of such convergence remain unexplored, i.e "should firms concentrate on one platform-war or run several platform-wars in parallel?

REFERENCES

1. M. Waaijer, "Handbook of Research on Open Source Software B. Parens, "The Open Source Definition", Available at: http://peresens.com/Articles OSD. html,2002.
2. A. Zoitl, T. Strasser and A. Valentini, "Open Source Initiatives asbasis for the Establishment of new Technologies in Industrial Automation: 4DIAC a Case Study", published in IEEE International Symposium of Industrial Electronics
3. Munish Saini,"A review of open source software development lifecycle models, available at http://dx.doi.org/10.14257/ijseia.2014.8.3.38.
4. M. Ueda, "Licenses of Open Source Software and their Economic Values", Published in Applications and the Internet Workshops, pp. 381-383, 2005.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 11, November 2016

5. M. Waaijer, "Handbook of Research on Open Source Software: Technological, Economic and Social Perspectives", IEEE Transactions on Professional Communication, vol. 52, no. 1,pp. 109-112, 2009.
6. E. S. Raymond and B. Young, "The Cathedral and the Bazaar: Musingson Linux and Open Source by an Accidental Revolutionary", O'Reilly, Sebastopol, CA,2001.
7. R. S. Pressman, "Software Engineering A Practitioner's Approach", McGraw Hill, 4th Edition, 1997.
8. S. Mandal, S. Kandar and P. Ray, "Open Incremental Model- A Open Source Software Development Life Cycle Model (OSDLC)", International Journal of Computer Applications (0975 – 8887), vol. 21, no. 1, May, pp. 33-39, 2011.
9. V. Potdar and E. Chang, "Open Source and Closed Source Software Development Methodologies", International Conference of Software Engineering (ICSE), pp. 105-109,2004
10. J. Feller and B. Fitzgerald, "A framework analysis of the open source software development paradigm", Proceedings of the twenty first international conference on Information systems', International Conference on Information Systems, Brisbane, Queens-land, Australia, pp. 58-69, 2002
11. Jawadekar, "Software Engineering- Principles and Practices", TMH, pp 18-27.
12. K. Beck, "Manifesto for Agile Software Development", http://agilemanifesto.org/. Retrieved, (2010) June 14.
13. K. Schwabe, "Agile Process and Self-Organization", http://aanpo.org/article/index, Agile Alliance, 2002.
14. A. Mockus, R. Fielding and J. D. Herbsleb, "Two case studies of open source software development: Apache and Mozilla", ACM Transactions on Software Engineering and Methodology, vol. 11, no. 3, pp. 309-346,2000.
15. P. Vixie, "Software Engineering", M. Stone, S. Ockman & C. Dibona (Eds.), Open sources: Voices from the open source revolution", Sebastopol, California: O' Reilly &Associates.
16. E. S. Raymond, "The Cathedral and the Bazaar", O'Reilly & Associates, Sebastopol, CA, USA, 1999.
17. A. Capiluppi and M. Michlmayr, "From the Cathedral to the Bazaar: An Empirical Study of the Lifecycle of Volunteer Community Projects", International; Federation for Information Processing (IFIP), open Source Development, Adoption and Invotion, eds.J.Feller, B.Scacchi, W.Sillitti, A..,(Boston Springer), vol. 234, pp. 31-44.
18. C. M. Schweik and A. Semenov, "The institutional design of open source programming: Implications for addressing complex public policy and management problems" ,from http://firstmonday.org/issues/issue8_1/schweik/index, Retrieved 11th May 2013.
19. L. Torvalds, "The Linux edge", M. Stone, S. Ockman & C. Dibona (Eds.), Open sources: Voices from the open source revolution, Sebastopol, California: O' Reilly & Associates.
20. D. E. Wynn, "Organizational structure of open source projects: A life cycle approach", Paper presented at the Proceedings of the 7th Annual Conference of the Southern Association for Information Systems, Savannah, 2004.
21. Open source Licence Proposal" http://cio-nii.defense.gov/sites/oss/open_source_software _(OSS)_FAQ.htm".
22. M.-W. Wu and Y.-D. Lin, "Open source software development: an overview. Computer", vol. 34, no. 6, pp. 33-38, 2001.
23. "Open source software development method", http://en.wikipedia.org/wiki/Open_ source_ software _development_ method", 2007.
24. Narender pal singh and Rachna Soni,"Agile Software:Ensuring Quality Assurance and Processes", http://link.springer.com/chapter/10.1007/978-3-642-22577-2_86,pp 640-648,2011
25. Arti Rana, Dr. S.P Singh, Dr. Rachna Soni and Dr. Ashish Jolly," Incremental Software Version Release Planning For Adaptive Maintenance",International Journal of Software Engineering and Its Applications,Vol. 9, No. 8 ,pp. 217-228, 2015.