Asian Journal of Computer Science and Technology
ISSN: 2249-0701 (P) Vol.7 No.3, 2018, pp.96-105
© The Research Publication, www.trp.org.in
DOI: https://doi.org/10.51983/ajcst-2018.7.3.1888

A Perspective on Open Source Software Development Movement

Amitpal Singh[1], Sunil Kumar Gupta[2] and Hardeep Singh[3]
[1]Research Scholar, IKG Punjab Technical University, Kapurthala, Punjab, India
[2]Associate Professor, Department of Computer Science and Engineering,
Beant College of Engineering and Technology, Gurdaspur, Punjab, India
[3]Professor, Department of Computer Science and Engineering,
Guru Nanak Dev University, Amritsar, Punjab, India
E-Mail: apsohal@yahoo.com
(Received 5 October 2018; Revised 15 October 2018; Accepted 29 October 2018; Available online 6 November 2018)

***Abstract* - The paper presents a thorough analysis of the Open Source Software Development (OSSD) movement. The main focus of this paper is to understand this movement right from its beginning. OSSD models proposed by various researchers are studied to get an insight into the practices and processes followed for OSSD. In order to determine the advantages and disadvantages of open source software (OSS), its strengths and weaknesses are also analysed. Various challenges associated with the development of Open Source Software (OSS) are highlightened. In future, this study will assist in the development of a framework in which OSSD teams can work in coordination for the development of quality software.**
***Keywords*: Open Source Software Development, OSS Strengths, OSS Weaknesses, OSSD Models**

I. INTRODUCTION

A. Brief History

The Open Source Movement first originated at A T & T lab where the first popular operating system known as UNIX was developed. It was the first operating system written in C language which is a high level language and hence machine independent. Programs written at that time were machine-dependent and this gave a new dimension for development of programs which were portable. As people were using different kinds of hardware and this sense of portability attracted people towards UNIX. The research during the 1970s was mostly focused on developing programs for scientific evaluations. The programs had to be written not only for specialists but also for general users interested in computer science and control systems. Thus, the programmers started:

1. Developing program that perform one task but should be done perfectly.
2. Developing programs that can work together by exchange of information.
3. Developing programs to efficiently handle textual data.

Researchers presented papers on UNIX at a conference on operating system principles held in New York, in October 1973. Thereafter, UNIX received even more popularity leading to growth in its user base. AT & T company's mainstream business was telephone and telegraph. But there were certain legal obligations that prevented it from venturing into any business like software development. The company did not want to go against the laws and the team developing UNIX was dissolved. Thereafter, UNIX was sold cheaply without any type of advertisement, support or bug fixes but advance payment was taken. The user base of UNIX was growing and with passage of time bugs were encountered. The company was not providing any kind of support. This leads to users of UNIX to help each other. The interaction of users resulted into formation of a community. The members of the community started working together by sharing ideas for providing solutions regarding bugs and other enhancements in the software [28]. Hence, a new way of problem solving was evolved.

Community members started modifying UNIX as per their own requirements to accommodate new features and to incorporate the latest hardware. Different teams were also working and contributing code into UNIX at Bell Labs. The academic community across various universities also had notable contributions towards UNIX. This unique concept of system designing and development took UNIX to achieve the highest levels of quality and security. This set the basis for designing and development of software from the people and for the people and gave way to concept of open source software development (OSSD) movement [39]. The openly available code was refined by various companies and sold in the market in form of various versions of UNIX operating system. The academic community at University of Berkley started selling its own version of UNIX under BSD (Berkeley Software Distribution) [25]. Free Software Foundation (FSF) was formed by Richard Stallman at MIT in 1984 to develop and distribute free software and to get ahead with GNU project [42]. FSF encouraged volunteer software developers to develop and distribute code under its banner. FSF was trying to launch its own version of Unix operating system but were missing the kernel, a key component for any operating system [22]. This kernel was developed by Linus Torvald, a Finnish programmer, in 1990. He launched his own version of Unix called Linux, uploaded on internet and invited other programmers to use and provide improvements for it [27]. Feedbacks from other programmers were secured as they examined the code of Linux. The different communities modified and enhanced the code which was further released as different versions of Linux. Due to the contributions of community of highly capable programmers and analysts across the globe, Linux achieved highest levels of stability and security. In a short span of time Linux became one of the most popular operating system in the world. In 1998, term 'Open Source Initiative' was coined in Palo Alto, California by software developers which were working on Linux [28].

Another event, in addition to the Linux project, which increased the pace of OSSD was the decision of internet browser company, Netscape, to make its code open for public. They were facing stiff competition from Microsoft's Internet explorer and were losing market share. Netscape wanted to attract the best programmers across the globe to give an edge for its product. They wanted the corporate world to benefit from the contributions of volunteers of open source movement [28]. This development was noticed by Free Software Foundation. They met to plan a strategy for such a new way of software development and also wanted to coin a new term for such an approach. This non-propriety software development approach was thus named 'free software' which after further called 'Open Source' to make it look more business oriented and less ideological [27]. This move was welcome by followers of open source movement. Linux community took no time to advertise the idea of OSSD through its websites, mailing lists and other discussion forums [28]. Another term used is FLOSS and it stands for free, libre and open source software (OSS) [10]. This was the start of a new ideology which would pave the way for development of co-operative environment in which dedicated teams of volunteer software developing communities could participate and contribute in various areas of software engineering. Aim of the communities is always to make high quality and reliable software, no matter how complex may be an application [2]. This study is performed to accomplish following objectives.

1. To get familiar with OSSD movement.
2. To study the evolution of OSS from an idea.
3. To study existing OSS models.
4. To study strengths and weaknesses of OSS.
5. To highlight various challenges associated with OSSD.

B. What is Open Source Software Development?

Free/Open Source Software (F/OSS) is a new way adopted for professional software development. Software is considered to be developed using OSSD strategy if it is distributed under a license, which fulfills "four freedoms" of F/OSS [2]. This is freedom to use the software as per own wish, freedom to copy, freedom to re-distribute and freedom to modify source code with condition that the modifications are also made available for distribution. Two major organizations involved in promotion of free OSS are Free Software Foundation (FSF), which made GNU Project and Open Source Initiative (OSI), which maintains a repository of OSS compliant licenses.

OSSD starts with an idea that originates in the mind of a single person or a group regarding development of software to address a particular need or solve some problem. The team then discuss among themselves or with others about the possible solution for the problem. They also discuss the various alternative solutions available and process of making the code base. After thorough discussions, first release of the software as per their requirements "personal itch" [12] is written. The software is put online and its source code made available to all. All the contributors were inspired to contribute bug fixes and functional improvements into the software. This resulted into formation of a common platform where contributors can communicate and hence a project community [34] is established. Communities use mailing lists, online forums, newsgroups and other online media to advertise the project. The project community allows best programmers from across the globe to share their expertise in various fields. Software code is written and shared in an efficient manner. Software become highly stable as the testing and debugging is performed by contributors across the globe. The cost of software is reduced as freelancers are involved.

Contributions from various contributors are analysed by core team of the project. The core team may have single or more number of coordinators. Coordinators are project creators and responsible for evolution and growth of community. They would take final decision to incorporate the received code into final build and release next test version of the software. After rigorous testing and debugging when required quality of software is achieved, test versions of software are promoted to be the next stable release. Further with passage of time new contributions in form of bug fixes and feature enhancements for the software are received. Same cycle of thorough testing and integration of code into existing software is followed. Every effort is done to attract more and more people towards the project and with passage of time the project grows. Teams members of the community provide feedback, which acts as base for planning of future project enhancing strategies. New features and resources are incorporated during research process. With constant efforts the project attains high quality and upcoming issues are dealt with even better ways.

Based on the above it is worth mentioning that open source is a new software development philosophy and open source licenses apply for distribution of this freely available work. The ways in which developement work is coordinated and communicated among the developers makes it different from existing software development strategy and this is what is unique. In addition to making original code available, OSS foundation set following guidelines to be followed.

1. Software can be freely redistributed and will not have any restrictions for distribution along with or as a part of another software package.

2. Software source code will be available for public to examine and test which in turn improves the quality of software.
3. License of the OSS must allow others to modify and further adapt the code base for some other purpose. Changes to original author's source code by another distributor must be restricted but if required then responsibilities for fixing of errors as a result of changes made by that distributor should also taken.
4. Any person or group should be free to use OSS without any discrimination [42].
5. License should allow the use of OSS without any discrimination against any field.
6. Whenever a software is distributed along with a larger product then license of the larger product should cover all its components. The license must allow for these small components to be distributed alone whenever required [42].

To get an insight into production process of OSS from scratch, about forty five relevant research papers covering various aspects of OSS production have been analysed. Certain other sources like websites, books and magzines covering brief history, nature of OSS, existing models of software production, strengths and weaknesses of OSS, challenges associated with OSSD are also analysed.

This research paper is organized into five sections each covering different aspects of OSSD movement. Section one presents discussion on historical aspects of OSS movement. This section covers the need for development of software using open communities and how open source software development was started and later how it became a full grown movement. Next sub section presents discussion related to what actually is the open source software development. Then various guidelines to be followed by open source software developers is stated. Section 2 and 3 presents strengths and weaknesses of OSS respectively. Section 4 presents review of the existing open source software development models, their evolution and feature addition with time. This section provided important insight into operational aspects of open source software development teams. Section 5 presents various challenges associated with open source software production. These challenges must be overcome for success of OSS. Next section concludes the paper and future work to be done is highlightened.

II. STRENGTHS OF OPEN SOURCE SOFTWARE

OSS are gaining popularity and there market share is increasing with time. They have number of advantages that had resulted into making this movement a success. Some of the strengths of OSS are enlisted below:

A. Less Development Time

Once core team launches the software after planning and design phase, its development kicks in. The product development time is minimised as hundreds of qualified developers across the globe contribute to original product. Thereafter, its functionality is enhanced every now and then [36]. Feedback from initial users is collected and given high importance. Error list is prepared based on this feedback. These bugs are again removed in parallel by hundreds of contributors in a time which is much less than proprietary software development teams. Open source contributors are highly skilled and also the users of their own developed system, therefore they have better insight into software requirements and take less time to finalise the system design [36].

B. Open Source Projectshave Better Quality

OSSD is accomplished in an agile manner [5] using an iterative process of software development, incremental release and process of review followed by further refinement by like minded peers. It ensures high level of quality and system security by employing thorough testing by peers across the globe [37]. Open source communities of peers are decentralized [44][36][20] and share their contributions using web sites repositories.

The latest software repositories are stored online for easy access of the developers. OSS products are distinguished by their quality standards, reliability and efficient working. The process of peer review and involvement of end users in software develoment, testing and debugging leads to high quality of OSS products.

C. Less Software Development Cost

OSSD is accomplished using freely available open source tools [3]. Software development is performed by volunteer software professional. They offers their knowledge, experience and hardware without charging anything. Cost of software development is borne by the contributors themselves. All these factors leads to almost no cost of development. Only cost incurred is web hosting facility cost which would be the responsibility of project core team or community. Proprietary software development companies employ paid staff and have to bear other expenses like hardware, building rents, electricity bills, maintainence charges etc. It is obvious that cost of OSSD is much less compared to proprietory software.

D. OSS Have High Security Levels

No software development company ensures 100% security of its products, but OSS have passed through rigorous testing and debugging of unknown number of reviewers. The software code is open and can be freely inspected for any type of wrong doing. These volunteers are free to publically report any type of backdoor entry or hidden spy code [7] in the OSS. The software is developed for public without any personal gains. It is observed that OSS have employed high levels of security mechanism.

E. Open Source Software are More Stable

Software development envolves number of phases consisting of contributions from hundreds of developers. These contributions have to be clubbed together to achieve the target system. Received code may have some error or deviate from standard coding procedure. Clubbing of code from different sources also results into errors. Bugs are invetiable, be it open source or proprietary software. Solution is to find and fix the bugs. Whoever find and removes bugs quickly is the winner. Proprietary software industry is in great hurry to release next version of the software and may provide solution of old bugs in next release. In case of OSS thousands of contributors with interest in that software are willing to provide solution for any problem. OSS have a bug tracking mechanism which finds and submits encountered errors to the community. There are frequent updates corresponding to OSS used. This results in increasing the stability of OSS [7].

F. Every One is Free to Use OSS

Every one is free to use OSS. Even the corresponding code can be downloaded and modified as per requirements. Freedom to use is there but one is bound by open source licences. Number of licences are certified by Open Source Initiative and used as per the situation. Some of the licences are General Public Licence, BSD Licence, Mozilla Public Licence, Python Licence and there are many more [46].

G. Huge Amount of Manpower Available

Large software projects have large manpower requirements. To develop high quality software products, manpower must be highly skilled. Popular open source projects attract highly skilled people towards it. As popularity of the product increases, more and more people join the community. There is never a shortage of manpower and every software development activity is handled by highly skilled professionals [15].

H. Freely Availabe Resources and Other Infrastructure

Developers involved in OSSD activities contribute various resources required like hardware, office space, furniture, stationary, internet charges, electricity charges etc. One may say that ”OSS is not developed free of cost but it is denoted to public by the community“.

I. Skill Enhancement for Software Professionals

Software professionals join OSSD communties due to certain personal and career specific reasons. Members of these communities devote their time and effort to collaborate, share ideas, acquire knowledge from one another and hence polish their software development skills. Whenever some software related problem arises, huge membership base of the community can cooperate to provide a solution for the same. With time members of community develop self determination and earns recognition among the peers. As number of contributions of a developer increases towards a community, the developer becomes a key member and can be made member of the core group in community. This gives the developer an international recognition and may be followed by job offers from well established software enterprises.

J. OSS Contribute in Price Reduction of Electronic Gadgets

Electronic devices like mobile phones, desktop and laptop computers, palm tops, tablets and many other hardware devices always require software for its smooth operation. Price of these devices is determined by clubbing the software cost with hardware cost. OSS can be employed for almost all electronic devices as a replacement of third party proprietary software used in it. This would help to reduce the cost of large number of electronic devices giving the manufacturers an opportunity to offer more competitive price .

K. No Need for any Government Regulation

Proprietary software development companies are bound to obey rules and regulations of the concerned country. Companies are required to register under the government by paying fees and other taxes. They are under obligation to report annual financial status to the government and other share holders. Failure to do so may be followed by loss of goodwill or penalities. Therefore lots of manpower in terms of accountants, consultants and lawyers are required to complete this task. All of these factors contribute to increase in cost of the software developed by that company. OSSD communities are not under any such government obligation and are only governed by open source licence mechanism. Their only aim is to provide best quality software.

III. WEAKNESSES OF OPEN SOURCE SOFTWARE

OSS is developed by volunteers and one cannot force the things upon others. As one have thorns with flowers and OSS is no exception, OSS has number of disadvantages that hinders its adoption rate. Some weaknesses of OSS are enlisted below:

A. OSS do not have One to One Support Facility

OSS is developed by volunteer from all across the globe and they may be hundreds in number. Developers being volunteers may not turn up when support is needed. OSS is known for their quality but at the same time a strong work force for providing support is required. Some of the well established communities like Linux, Apache, Python and so on, do have a strong support mechanism in terms of mailing lists, FAQs and newsgroups. But what about other communities which are still growing and have not yet matured to that level. Even such communities provide software for critical missions but it is recommended to employ the same carefully.

B. Large Number of Open Source Projects are Abandoned Midway

SourceForge is one of the largest OSS hosting facility with large number of high quality and mature software projects hosted on it. It also have a large number of low quality projects which finds very little amount of downloads. In addition to these huge amount of abandoned projects are also uploaded on such sites [14]. Developers initiate a project but in large number of instances, do not find any takers or contributors as it is not accepted by general public. The original developers themselves leave projects and are attracted towards more successful ones. Poor project handling and mangement skills [39], inability in community formation, member communication, coordination among them, insufficient and uneven distribution of resources are also responsible for failure of the project. Other factor like poor bug handling facility, less feature addition and upgradation reduce quality of software. Some products are unable to compete with an already established product.

C. Source Code is Available to all Hence can be Tampered Leading to Insecurity

An article, published in Washington Post, presents the fact that software gaint, Microsoft, is aggressively advocating the Pentagon to suppress its growing use of freely available OSS and must switch to proprietary systems sold by the software firms [21]. According to Microsoft, open source code is accessible to general public to examine and even to criminals or hackers who may exploit it. Proprietary companies advocate the fact that their software code base which is confined to the company itself, is more secure and ideal for critical systems. UCITA(Uniform Computer Information Transactions Act) [37] has a clause that if users do not adhere to conditions laid in licence agreement then propreitary software development companies can disable their software installed on user’s machines by remotely logging into it. This means that the software manufacturer can put a back door for secretely and remotely accessing their software installed on user machine. The company may turn-off or disable selected features or entire product. This is a major security breach and must be avoided. These back doors could be exploited by hackers to disrupt the functionality of computers. OSS is examined by hundreds of dedicated professionals with an aim to develop high quality public poducts, hence such vulnerabilities are discovered and fixed in short interval of time.

D. OSS are not Compatible with Proprietary Ones

The OSS is developed by community with contributions pouring from across the globe. These software are made as per guidelines issued by the community and are not bound to adopt global standards. These are not made to earn profit and do not get any benefit for sticking to proprietary software production “standards”. In the process of software production, standards are not authorised by any central controlling authority. Each company follow its own standard development mechanism. Proprietary companies design there products so that they are compatible with other similar production houses. They do not care about compatibility issues with other similar OSS. OSSD communities are trying to solve this problem by providing software for various disciplines. Operating systems, database management systems, web servers, office suites, games and software for almost all applications are available as free OSS. Users are recommended to select the software from a wide variety of OSS available and use it. It is assured that these software will not let you down.

E. OSS not Available for all Platform

Communities engaged in the development of OSS want to increase their user base. So they want there software to be used by every individual. But people are using different platform and in order to run OSS on them, some type of intermediate code is required. In order to solve platform issues, OSSD communities have developed many versions compatible with different platforms.

F. OSS development is Community Oriented

OSS production starts with core team giving an initial simple version of the product. Further development is carried on by a dedicated team of developers which with passage of time takes the form of a community. Most important aspect of open production is to advertise the project so that new volunteers are attracted towards it. As the developers would not get any incentive for their contributions, why should they join the community. Therefore, efforts to attract the contributors are required and only then end user participation can be increased. People would use the product only if it is best one with no other alternative. Healthy community consisting of various categories of people to make and analyse design options, write code, performing testing and debugging, provide documentation and support. Communities may be open in nature but joining such communities require one to be well versed with latest technologies.

G. Updated Documentation not Available

OSS is made by combining contributions from hundreds of programmers across the globe. Most of them provide executable code which is embedded into already existing software repository. All the efforts are made to provide quality software whereas very little time is devoted for documentation of system. This unavailability of documentation makes it difficult for old and new developers to have an understanding of the existing system. Lack of documentation makes it impossible for developers to follow similar process for designing and coding of the applications [26].

H. Common Software Development is not Followed

As OSS is developed by independent individuals who hardly communicate with one another, chances of following a common software development model is thus very less. Individual contributors handle and solve the problems in there own way and employ different set of development practices. Most of the efforts are concentrated on development of quality code whereas system design aspects are ignored. This may solve the purpose for small projects but problems arise when size of the software become large. Modern software follow ISO 8402 quality assurance standards and every activity is performed in a systematic and planned manner. The nature of OSSD is unplanned so quality of products developed may not be up to the mark. This is one of the factors which contibute to high failure rate of open source projects. If OSSD have to be applied in critical applications then modern software practices have to be adopted. Some of the established projects like Linux, KDE, Andriod have a dedicated team for planning, defect removal, updation and removal of obselete components. Still good metrics for measuring quality of OSS are not in place.

I. Projects Suffer from Problem of Coordination and Communication

OSSD is carried out by community of unknowns where no one know what others are doing. Whom should a developer consult in case of any development related query? A number of contributors may be working on a particular solution but they do not have any coordination and communication among themselves. OSSD communities must have an established coordination and communication mechanism for attaining higher quality in the developed products. Such mechanism would help to quickly forward bug reports to original developers for fast debugging. Duplication of efforts would also be reduced. Proper coordination would help in finding best solutions and hence highest levels of quality. Communcation channels must be authentic otherwise it ends up with low productivity [1].Professional software development companies have experienced colleagues to guide and support new members of the team [9]. Such support is missing in OSSD teams as contributors do not know each other and may not be interested in guiding others. They may not have time or enough resources to interact with newer members of the community [4][41]. New members are left with options of downloading the code and other documentation to get familiar with the project. This code or documentation may even be outdated. Coordination among members become important with an increase in code base of the project.

J. Interactive User Interface Missing

Many open source users feel that OSSD teams do not concentrate much on designing interactive interface for the products. People are familiar with proprietory software products as they have used them for long period of time. Moreover they have participation of the end user [20] and interface is customised as per desires of the customer. OSS developers have their own choice, preferences and hence design interface as per their requirements. Most of the efforts are focused on developing high quality product than the interface.

K. Irregularity in Removal of Defect

OSS belonging to large size communities have a formal defect removal mechanism which take less time in removal of bugs whereas smaller communites may take much more. At the same time it is seen that solutions are quickly provided for security related bugs but minor bugs may even take months to be fixed. A large number of users with less techincal skills are not able to track the bugs. Therefore, good bug tracking and reporting mechanism is required so that bugs can be timely tracked and removed.

L. Frequent Release of Beta Versions

Contributions to OSS is received frequently and software is updated regularily. Many short time versions which lack high quality standards are released followed by a long time support version. Short time versions are also less stable. Large number of versions and frequent updates may frustrate the end user.

IV. EXISTING OPEN SOURCE SOFTWARE MODELS

Open Source Software development is usually started by individual person or a group of persons. The development is carried forward by formation of a community. Different researchers have explored and explained this mechanism of community formation but still a standard agreed upon model of OSSD do not exist. One factor contributing to the fact of not having a standard development model is that different projects require different development strategies to be followed. Another fact is that different communities have different working and organizational culture, which they adopt for their projects.

Raymond [31] who was a pioneer in the field of OSSD termed it as a Bazaar model and described it as “a great babbling bazaar of differing agendas and approaches”. OSSD is community oriented where development is performed on mega scale in parallel where innovative minds contribute voluntarily [40]. These contributors never have any face to face interactions as they are from different corners of the world. They have different cultures and speak different languages. As soon as problems appear or some innovative idea is to be implemented, some of the best programmers join over the internet to provide solution for software problems without any profit. This model in based on the idea that work is done by collective efforts of people who may be contributing marginally but there joint efforts sum up into huge project. The development activities are publicly visible and open, developed code is available over the web. Projects do not follow any formal project management guidelines, neither there are any budget considerations nor any time schedule [45]. Success of the project depends upon community and joint efforts for community development, skill enhancement of associated developers and maintenance of community should be carried on [35].

The model proposed by Sugumaran *et al.* [42] is based upon the fact that developers of OSS follow certain common practices. They usually perform certain similar action and iterate through them. OSSD process can be divided into seven phases. Initially the process starts with discovery of problem followed by finding out volunteers to handle the project. Next is solution identification phase in which design activity is carried out. After design is validated and verified coding and testing is carried out. Usually number of errors is found during testing which leads to code review and change process to be carried out. When developers are satisfied with the quality of software, code is committed to the build and required documentation is performed. Lastly the software is updated as per latest requirements and future release management is performed.

Another OSSD model was proposed by Wu and Lin [45] and it incorporated licensing for open source distribution and mechanism for version control. According to the author, open source project starts with a personal idea. The developer looks for any existing similar project and if it is found, he joins that project else initiates new project. Developer may have independent hosting facility or may hosts it on some OSSD community website. Bug tracking and other announcements are carried out through mailing lists. One important addition incorporated by the author is use of CVS version control mechanism. Further, documentation is performed and manuals are prepared. Another aspect added by the author into OSS development model is the use of a license model which of course is open in nature. Modifications in form of patches are accepted and incorporated into the system. Official version of the software is released and regular updates are provided.

Model proposed by Schweik and Semenov [38] consisted of three phases for software project life cycle. First phase is termed as project initiation phase, which is performed by the project core team and then they invite other contributors to further develop and enhance the project. In second phase, project is made open by its founders and various open source licenses are followed for its distribution. For success of project, experienced and dedicated developers of core team must have active participation. The core team has to be technically very strong if the developed project is to compete with best in the market and it should be ready for future. Collective efforts of development team are required to incorporate latest innovations into the project. The project development framework should be designed in such a way that future requirements and feature enhancements could take place easily. This phase also deals with selection of relevant technologies in which system is to be implemented. Appropriate web sites for project advertisements, code sharing and recruitment of developers have to be chosen and be at place. Third and final phase deals with various elements of risk encountered during life of the project. Time from project initiation followed by growth, stability and decline is taken into consideration. Other factors like utility of the developed product for global users, interest of users in the developed product and active participation of developers in testing, debugging, documentation and further programming are also taken into consideration.

The model proposed by Fielding *et al.* [13] incorporated a decision making mechanism into life cycle of open source projects. Each phase of the model has task associated with it. This model consists of following six phases. First phase is managed by developers and it involves fixing of roles and responsibilities of various entities involved in the project. In second phase, work to be done is identified. Planning for the work and process of work management is carried out. Third phase performs the job of allocation of work among developers. Various project development activities are carried out. Check is kept that whether the work is done as it was planned earlier. In-house testing and debugging is performed in the next phase and is termed as pre-release testing phase. Fifth phase is inspection phase where final inspection is performed before the software is release. Lastly release management is performed with an aim for up gradation of product with latest features and further innovations.

Jorgensen [17] proposed an OSS which provides information related to processes corresponding to activities performed by specific products. Various phases in this model are as explained ahead. The very first phase consists of code that is contributed by some of the best developers across globe for purpose of reviewing. This code is made available to peers for testing, debugging and further improvements. Further there is a thorough and independent peer review for contributions, which is major strength of this model. Next phase is devoted to pre-commit testing where contributions made by different developers corresponding to some new feature or bug is tested thoroughly. This phase is very important as it is to be ensured here that all contributions are made totally fault free before they are added into the finished product. After newly added code had achieved the desired quality and stability levels, it is released as development release version. This released code would be analysed, reviewed, tested and debugged for flaws by community members in parallel across the globe. Error reports are submitted to the community web sites. Errors are corrected and further improvements are made till required quality is achieved following which this development version is converted into production version and released as a finished product.

Another classification for various stages through which an OSS passes is given by Rothfuss [33] in his work. As usual project starts with planning stage where it is only an idea and scope of the project is still under consideration. Nothing with respect to designing or coding is yet prepared. Whenever any amount of basic source code is ready and project starts to take some shape, the stage is termed as Pre-Alpha. At this stage, the released source code is at very initial level which may not even be expected to compile or run. Contributors other than the core members usually find it difficult to read and understand the intension for which source code is written. When some guidelines for further development are formulated, the project enters next stage termed as Alpha. During Alpha stage, code starts taking shape and the released code starts working at least for small functionality. Documentation pertaining to preliminary development is also released. The team contributes to expand feature set of developed application. When set of intended features are incorporated into the project, it is said to enter into next stage termed as Beta stage. In Beta almost entire codebase is developed but faults are still existing in the system. Faults are found and debugged by the community leading to development of highly reliable software system. When number of faults is below expected quality level then stable version of project is released. This stage is called Stable. The system is now in stable state and reliable enough for use in daily routine. Whenever some changes are required they are applied very carefully and the intension of changes is to increase stability but not to add new functionality. Leaving aside minor issues, if no significant change happens over a long period of time, project will be termed as in mature stage. In this stage developed software performs reliably, its intended purpose is fulfilled and no new development occur. New changes should be applied with extreme care as it may destabilize the already stable system. The project is used as it is over a period of time before it becomes obsolete and its user base decreases to critical low level or it is replaced by better software.

Roets and Wright [32] developed an OSSD model as an extension of traditional SDLC model. Author proposed Initiation phase as the first phase in OSSD life cycle, which takes place of Planning, Analysis, and Design phases of traditional SDLC. Second phase proposed in OSSD consists of Review, Contribution, Pre-Commit Test and Production release as sub phases in place of Implementation phase is SDLC. Third and final proposed phase consists of parallel debugging and development release as replacement of Support phase in traditional software. Taking into consideration the OSSD models proposed by various authors it is evident that OSSD life cycle is altogether different as compared to traditional software development process. Planning of the system, its analysis followed by design is usually carried out by core team and then it is made open for contributions of developers across the globe. OSS projects employs certain own community based processes and practices, but still certain common characteristics are exhibited by almost all of the OSSD projects [11].

The main focus of all OSS projects is to develop software using collaborative development strategy. Community members are globally distributed and they voluntarily participate in software development. There is no central management authority but a core team which had initiated the project. As members are distributed across the globe, they have high diversity of capabilities, qualifications and skills. Most of the members do not have any face to face interaction hence the medium of communication is usually internet or other web-based technologies. Pace of software development is increased as individuals carry various development activities in parallel. Every member develops code independently followed by strict community based peer review. No one will get any type of favour as all are strangers. Core team will decide for release of patches for bug removal and new software releases only after required quality is achieved. Brief features of the above discussed OSS models are tabularised in table I.

TABLE I FEATURES OF OPEN SOURCE SOFTWARE DEVELOPMENT(OSSD) MODELS

Author	**Features of various OSSD Models**
Siau & Tian	Parallel development, contributors across the globe, activities openly visible, formal project development, budgeting and scheduling absent.
Sugumaran *et al.*	Problem discovery, finding volunteers, solution identification, design activity, validation and verification, coding and testing, Software updating and future releases.
Wu & Lin	Open source licensing, CVS mechanism, testing and bug tracking through mailing lists, documentation and manuals, regular updates.
Schweik and Semenov	Project initiation, going open, Project growth, stability or decline taken care off.
Fielding *et al.*	Fixing roles and responsibilities, planning and management, allocation of work and schedule, in-house testing and debugging, final inspection, release management.
Jorgensen	Code submitted for review and improvement, independent peer review, pre-commit testing, development release, rigorous parallel debugging.
Rothfuss	Software Planning, Pre-Alpha, Alpha, Beta, Stable, Mature, Obsolete.
Roets and Wright	Based on the traditional SDLC, review of contributions, Pre-Commit Test and Production, Parallel debugging and future development.

V. CHALLENGES OF OPEN SOURCE SOFTWARE DEVELOPMENT

OSSD is altogether a different approach of software development and hence is associated with number of challenges which must be overcome for its success. Some of the challenges found after surveying literature on OSSD methadology are as listed below.

1. Globally distributed development team
2. No or rare face to face interaction
3. Team of volunteers, can quit any time
4. Healthy environment for community formation [23]
5. Mechanism for mutual support, coordination and feedback [18][24]
6. Maintain the enthusiasm of team members [19]

7. positive leadership [30]
8. Mechanism to deal with frequent changes in team composition.
9. System for detection of betrayal and lying by members
10. System for communication of members who belong to various cultures and speak different languages

The virtual teams responsible for the development of free software must maintain healthy relationship among themselves and the absence of a cordial environment for team members of the community may cause the following effects

1. Decrease the amount of contributions of members [8]
2. Members start concentrating on individual goals [6]
3. Reduces trust among members [43]
4. Insecurity [8]
5. reduced quality and productivity [16].
6. rule-breakers punished by 'flaming', 'kill-filing' or 'shunning' .

VI. CONCLUSION AND FUTURE WORK

This paper concludes the fact that OSSD deploys a new strategy for software development. Volunteers, forming a virtual community, for development of the software are spread across the globe, belong to diverse cultures and have different traditions. They spare their time and resources for the product development and make it available online for free. OSS have grabbed significant market share due to its strengths, which makes it compete with the proprietary software firms. There are certain associated weaknesses which hamper its growth and make people reluctant to adopt them. Further features of various OSSD models are summarized but still certain aspects have to be added into the models to handle highly dynamic nature of open source software development teams. The future work will focus on development of a framework to assist in the coordination and cooperation among the OSSD community members. The community is the backbone of OSSD so there is a strong need to bind its constituent members. Trust will play an important role in the formation and consolidation of a healthy virtual community, therefore relevance of trust with respect to it will be studied. Before development of the intended framework, other OSSD challenges like leadership issues, nature of team composition, diverse nature of members, community policies, support and feedback mechanism will also be taken into consideration.

REFERENCES

[1] P. J. Adams, A. Capiluppi, and Boldyreff, “Coordination and Productivity Issues in Free Software: the Role of Brook's Law”, *IEEE International Conference on Software Maintenance*, September, 2009.

[2] J. Asundi, “Software engineering lessons from open source projects”, *In 1st workshop on Open Source Software*, 2001.

[3] M. Bergquist, and J. Ljungberg, “The power of gifts : organizing social relationships in open source communities” , *Journal of Information Systems*, Vol. 1, No. 1, pp. 305–320, 2001.

[4] L.M. Berlin, “Beyond program understanding: A look at programming expertise in industry”, *In Empirical Studies of Programmers: Fifth Workshop*.

[5] B. Boehm, “Get ready for agile methods, with care”, *Computer*, Vol. 35, No. 1, pp. 64–69, January 2002.

[6] R. J. Bulman, “Shattered Assumptions: Towards a new psychology of trauma”, *Free Press*, New York, 1992.

[7] S. K. Chong, J. Abawajy, M. Ahmad, I. Rahmi, and A. Hamid, “A Multilevel Trust Management Framework for Service Oriented Environment”, *International Conference on Innovation, Management and Technology Research*, Malaysia, September, 2013.

[8] D. E. Chubin, G. S. May, and E. L. Babco, “Diversifying the Engineering Workforce”, *Journal of Communication”*, Vol. 94, No. 1, pp. 73-86, 2005.

[9] C. Daffara, M. Jesús, E. Humenberger, W. Koch, B. Lang and B. Laurie, “Free Software/Open Source: Information Society Opportunities for Europe”, Retrieved from http://www.eu.conecta.it/paper.pdf, 2000.

[10] B. P. DanielBlaney, D. Lenceviciene, and Z. Yang, “Open source software development model”, source internet (scholors.google.com).

[11] S. Dietze, “Agile requirements definition for software improvement and maintenance in open source software development”, *In Proceedings of SREP*, Paris, France, August 2005.

[12] S. R. Eric, “Cathedral the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary”, *O’Reilly Associates, Inc. Sebastopol*, CA, USA, 2001.

[13] R. Fielding, A. Mockus, 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, 2002.

[14] J. Howison, and K. Crowston, “The Perils and Pitfalls of Mining SourceForge”, *Proceedings of the International Workshop on Mining Software Repositories*, Edinburg, UK, pp. 7-11, 2004.

[15] Iqbal, Quadri and Rasool, “ Open Source Systems and Engineering: Strengths, Weaknesses and Prospects”, *TRIM*, Vol 7, No. 2, 2011.

[16] S. L. Jarvenpaa, T. R. Shaw, and D.S. Staples, “Toward contextualized theories of trust: the role of trust in global virtual teams”, *Information Systems Research*, Vol. 15, No. 3, pp. 250-264, 2004.

[17] N. Jorgensen, “Putting it all in the trunk incremental software development in the freebsd open source project”, *Information Systems Journal*, Vol. 11, No. 4, pp. 321–336, 2001.

[18] E. C. Kasper-Fuehrer, and N. M. Ashkanasy, “Communicating trustworthiness and building trust in interorganizational virtual organizations”, *Journal of Management*, Vol. 27, pp. 235–254, 2001.

[19] T. R. Kaywort, and D. E. Leidner, “Leadership effectiveness in global virtual teams”, *Journal of Management Information Systems*, Vol. 18, No. 3, pp. 7-40, 2002.

[20] B. Kogut, and A. Metiu, “Open source software development and distributed innovation”, *Oxford Review of Economic Policy*, Vol. 17, NO. 2,pp. 248–264, 2001.

[21] J. Krim, “Open-Source Fight Flares At Pentagon”, *In Washington Post*, May 2002.

[22] Law and the Internet: Third Edition 3rd ed. Edition by Lilian Edwards, Charlotte Waelde (Editors

[23] R. J. Lewicki, and B. B. Bunker, “Developing and maintaining trust in work relationships”, *In: T. R. Tyler, R. M. Kramer (Eds.), Trust in Organizations: Frontiers of Theory and Research*, Thousand Oaks, CA, Sage Publications, pp. 114-139, 1996.

[24] McAllister, “Affect and Cognition Based Trust as Foundations for D. J. Interpersonal Cooperation in Organizations”, *The Academy of Management Journal*, Vol. 38, No. 1, pp. 24-59, 1995.

[25] M. K. McKusick, “Open Sources: Voices from the Open Source Revolution, *Chapter Twenty Years of Berkley Unix: From AT&T owned to Freely Re-distributable*, Sebastopol, CA, O’Reilly & Associates, pp. 31-46, 1999.

[26] G. Moody, “Rebel Code: Linux and the Open Source Revolution”, *Penguin*, 2002.

[27] L. Mui, M. Mohtashemi, and A. Halberstadt, “A computational model of trust and reputation”, *In Proceedings of the 35th International Conference on System Science*, pp. 280–287, 2002.

[28] History of Open Source Initiative, Retrieved from *http://www.opensource.org/docs/history.html*, 2001.

[29] M. Osterloh, and S. Rota, “Trust and Community in Open Source Software Production”, *Analyse&Kritik, Lucius & Lucius*, Stuttgart, pp. 279-301, 2004.

[30] G. Piccoli, and B. Ives, “Trust and the unintended effects of behavior control in virtual teams”, *MIS Quarterly*, Vol. 27, No. 3, pp. 368-395, 2003.

[31] E. S. Raymond, “The Cathedral and The Bazaar”, Retrieved from *http://www.ojphi.org/ojs /index.php/ fm/article /view /578/499/*, 1998.

[32] M. M. R. Roets and K. Wright, “Open source: Towards successful systems development projects in developing countries”, *In Proceedings of the 9th International Conference on Social implications of computers in developing countries*, Sao Paulo, Brazil, May 2007.

[33] G. J. Rothfuss, “A Framework for Open Source Projects”, *Master Thesis in Computer Science*, rothfuss@abstrakt.ch of Zurich, Switzerl and Register No. 97-711-915, 2001.

[34] W. Scacchi, J. Feller, B. Fitzgerald, A. Hissam, Scott, Lakhani and Karim, “Understanding Free/Open Source Software Development Processes”, *Software Process: Improvement and Practice,* Vol. 11, pp. 95-105, 2006.

[35] W. Scacchi, “Is open source software development faster, better, and cheaper than software engineering”, *In Proceedings of the 2nd ICSE Workshop on Open Source*, 2002.

[36] W. Scacchi, “Understanding the requirements for developing open source software systems”, *IEEE Proceedings on Software to appear*, 2002.

[37] B. Schneier, “UCITA, the Uniform Computer Information Transactions Act, The RISKS Digest Forum on Risks to the Public in Computers and Related Systems”, *ACM Committee on Computers and Public Policy*, Peter G. Neumann, moderator, Vol. 20, No. 87, April 2000.

[38] C. M. Schweik, and A. Semenov, “The institutional design of open source programming: implications for addressing complex public policy and management problems”, 2003.

[39] S. E. Sim, and R. C. Holt, “The ramp-up problem in software projects: A case study of how software immigrants naturalize”, *In proceedings of the 20th International Conference on Software Engineering, IEEE Computer Society Press/ACM Press*, Kyoto, Japan, pp. 361-370, April 1998.

[40] R. Stallman, “The GNU Project”, Retrieved from *http://www.gnu.org/gnu/thegnu project.html*, 1998.

[41] Suchan, and G. Hayzak, “The communication characteristics of J. virtual teams: A Case Study”, *IEEE Transactions on Professional Communication*, Vol. 44, No. 3, pp. 174-186, 2001.

[42] V. Sugumaran, S. Sharma, and B. Rajgopalan, “A framework for creating hybrid-open source software communities”, *Information Systems Journal*, Vol. 12, pp. 7–25, 2002.

[43] M. G. Uddin, and M. Zulekernine, “UML-Trust: Towards Developing Trust Aware Software”, *In Proceedings of the ACM Symposium on Applied Computing*, Brazil, pp. 831-836, 2008.

[44] K. Ullah, and S. A. Khan, “A Review of Issue Analysis in Open Source Software Development”, *Journal of Theoretical and Applied Information Technology*, Vol. 23 No. 2, 2005.

[45] M.W. Wu, and Y. D. Lin, “Open source software development: An overview”, *Computer*, Vol. 34, No. 6, pp. 33–38, June 2001.

[46] Wikipedia, *http://wikipedia.org/licenses*.