A Comprehensive Study of Version Control System in Open Source Software

Nindya Kotwal, Vineeta Bassi

Abstract— The Focus of this paper is to present review about Open Source Software (OSS), Open Source Software Development (OSSD) and how the version control tools are used in OSSD. This paper describes the use of Version Control System (VCS) to coordinate the evolution of open source software projects which mainly includes to manage the communication among various members of a project and to control the hierarchy of the code being developed. The control of the code includes version control and the organization of the team who develops each project.

Keywords— open source software, open source software development process, subversion, version control, version control system.

1 INTRODUCTION

Open Source Software has now a day become the subject of most research and debate; not only for the promise it holds to solve systems development challenges but also due to increase in the OSS development which inturn benefits various organizations in developing nations. Open Source Software development process combines various applicable features found in traditional software processes with some of the other features in such a way that would help in the development and maintenance of high-quality, faster and cheaper software within the rapidly changing Internet environment. Open source development process is not a process without any limitations but it provides numerous applicable advantages and opportunities to the system development process and also to open source society which produces quality softwares very rapidly in open source environment. The one of most important benefit of the open source software is the reduced cost. In addition to this benefit, open source society offers new development models that can be used, particularly within developing nations and economies, and has an extra benefit of skills development. Version control system is an efficient technique that is used in OSS development where work is done in a distributed environment and many version control tools are available.

2 OPEN SOURCE SOFTWARE

2.1 Open Source Software

Open Source Software is the type of software products which are available to the public, with its source code to study, change, and improves its design. However when open source used for commercial purpose, then an open source license is required [1]. Open Source Software is the type of software products which are produced by the distributed communities consisting of huge number of contributors from all over the world who have a sense of commitment towards there work thus in OSS no body is assigning the work and everyone is taking the work willingly which results in skill development.

OSS is software for which the source code is publicly available, though the specific licensing agreements vary as how the code is being used [2]. According to Roets, Minnaar and et al, there is one basic feature of the Open Source Softwares that results to its four fundamental freedoms and that is, users must have access to the source code. OSS is software whose licenses that is access to code which give users these essential 'freedoms':-

1. To run the program for any of the purpose and also study the workings of the program,
2. To modify the program to suit specific needs and requirements.
3. To redistribute copies of the program at no charge or for a fee.
4. To improve the program, and release the modified version of the program.

2.2 Open Source Development Process

Open source development is an alternative approach used in showing how the Internet can change the way software is initiated, constructed, deployed, and evolved. Open source development offers useful information about common problems as well as some possible solutions for globally distributed product development. Process modeling gives a great opportunity to analyze, compare, visualize, and transfer for reuse these possible solutions [3].

The basic principle for the OSS Development Process (OSSDP) [3] is that by sharing source code and ideas, developers cooperate under a model of systematic peer-reviews, and take advantages of parallel debugging that leads to innovation and rapid advancement in the development of the software [3]. The OSSD model has the following features:

1. Collaborative approach to problem solving through feedback and peer review given by the web community.
2. Large pool of globally dispersed, highly talented, motivated professionals from all over the world.
3. Increased user involvement as users are viewed as co-developers.

The Open Source Software Development model (Fig. 1.) incorporates aspects of various previous models used in the development. In addition, the proposed model attempts to encapsulate the phases of the traditional SDLC and previous OSSD models.

The introduction of a generic initiation phase replaces code phase. The initiation phase can be applied at both the micro and macro level of a project development. This phase refers to developed code that is used as a prototype for further development on a particular project, that code may be a small code segment or the initial version of an entire project. The phase is undertaken either by a developer who develops a piece of code for an existing project, or the project founder in a new project.

The initiation phase then moves into a cycle of code reviews and further contribution from the developer community. The possible number of iterations occurring at this phase is dependent on the interest that project or project related components generates within the developer community. Independent peer review and prompt feedback characterize this phase. The cycle of code contribution and review take place within the wider Internet developer community or web developer community, employing tools such as email, bulletin boards and discussion groups.

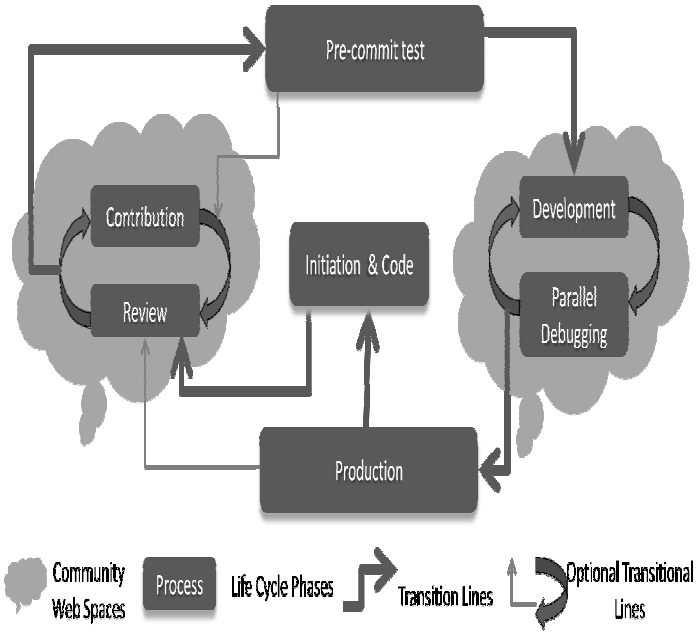

Fig. 1.Open Source Development Model [3]

Once a piece of code is considered adequate for inclusion in a development release, pre-commit testing is performed to ensure that this new piece of code, once added, does not break the existing release. Testing is usually performed by core developers. No rigorous testing schedule exists and, indeed, the process is not even required. However, the consequences of allowing faulty code into a development release can be severe for the programmers involved and in turn the reputation of the project as a whole.

A process of debugging and reincorporation of code into the development release then takes place. This is again an iterative process occurring within the community web space. No formal planned debugging occurs; individuals volunteer. This is another area exemplifying the strength of open source projects. The more people that seek find and remove bugs, the better the quality of the software which is one of the advantages of the OSSD.

Eventually, code forms part of a production release which is generally managed by core developer. Production releases take the form of a prototype that can be used in the initiation phase of the next iteration of that project, component or code segment.

2.3 Some of the Success Stories of OSS

These are some of the most successful projects developed in open source society.

TABLE 1
Some Examples of Open Source Softwares

GNU/LINUX	Operating System
Apache web server	Web Server
MySql	Relational Database Management System
OpenOffice	Office Suite
Mozilla Firefox	Web Browser
Thundermail	Email Client
Tcl,Python	Languages
Emacs,Jedit	Text Editors

2.3 Open Source Software Licenses

An OSS license helps publisher in defining the privileges and restrictions that a user must have to follow if they want to use and modify the software. If any one of the developer wants to publish a program as OSS, the publisher can distribute that program as an un-copyrighted product.

TABLE 2
Comparison among OSS licenses [4]

	Notice of modificationtion	Redistribution of modified work	Original Source code attached to modification	Linking to source code	Liability Notice
BSD	Yes	Yes	No	Yes	Yes
GPL	Yes	Only under GPL	Yes	No	Yes
LGPL	Yes	Only under LGPL	Yes	Yes	Yes
MIT	Yes	Yes	Yes	Yes	Yes
MPL	Yes	Only under MPL	Yes	Yes	Yes

- *Nindya Kotwal is currently pursuing masters degree program in Computer science and applications in Thapar University, India.*
- *Vineeta Bassi is currently working as Assitant Professor in SMCA department in Thapar University, India.*

The users of the OSS program can read, copy, modify, and redistribute the program. However, it is also possible for someone to make the program copyrighted by modifying the original program. Consequently, the modified, copyright-protected program becomes a personal property, and is not OSS any more[4].

To prevent this situation, most of OSS licenses implement "copy left" concept: anyone who redistributes the software, with or without changes, must pass along the freedom to further copy and change it [4].

When an OSS program gets published under an Open source software license, the program gets used and distributed by the users in the way as specified in the license. Developers can make their own licenses as well and the procedure of which is given by open source initiative. The most popular set of open source software licenses are those softwares which was approved by the Open Source Initiative (OSI) based on their Open Source Definition (OSD).

3 VERSION CONTROL SYSTEM

3.1 Version Control

A Version Control System (or revision control system) is a combination of technologies and practices for tracking and controlling changes to a project's files, in particular to source code, documentation, and web pages The reason version control is so universal is that it helps with virtually every aspect of running a project: inter-developer communications, release management, bug management, code stability and experimental development efforts, and attribution and authorization of changes by particular developers [5].

A VCS provides the ability to version resources such as data files. (We use the phrase "a VCS" to mean "an instance of a VCS.") Considerable functionality is associated with modern VCS's; some examples are distributed development, non-linear development and the maintenance of history (versions). A VCS comprises objects such as repositories and branches that help realize such functionality. An authorization scheme for a VCS specifies how these objects are protected, while respecting the semantics of the relationships between the objects.

Version Control System ::={ repository}
repository:: = Branch {Branch}
Branch ::={ Folder} {File} {Tag}
Folder ::={ Folder} {File}

A Version Control System in BNF [6]. The notation::= stands for "comprises," and {} means 0 or more occurrences

3.2 Version Control in OSS

A version control system maintains an organized set of all the versions of files that are made over time. Version control systems allow people to go back to previous revisions of individual files, and to compare any two revisions to view the changes between them.

In this way, version control keeps a historically accurate and retrievable log of a file's revisions. More importantly, version control systems help several people (even in geographically disparate locations) work together on a development project over the Internet or private network by merging their changes into the same source repository which is the base of open source software.

There are various version control tools available such as concurrent version system (cvs), subversion (svn), bazaar, Git [7].

3.3 Version Control Basics

There are some basic terms used in VCS:-

1. commit: To make a change to the project; more formally, to store a change in the version control.
2. log message: A bit of commentary attached to each commit, describing the nature and purpose of the commit.
3. update: To ask that others' changes (commits) be incorporated into your local copy of the project; that is, to bring your copy "up-to-date".
4. repository: A database in which changes are stored.
5. checkout: The process of obtaining a copy of the project from a repository
6. working copy: A developer's private directory tree containing the project's source code files, and possibly its web pages or other documents.
7. revision, change, changeset: A "revision" is usually one specific incarnation of a particular file or directory.
8. diff: A textual representation of a change.
9. tag: A label for a particular collection of files at specified revisions.
10. branch: A copy of the project, under version control but isolated, so that changes made to the branch don't affect the rest of the project, and vice versa, except when changes are deliberately "merged" from one side to the other.
11. merge: To move a change from one branch to another.
12. conflict: All version control systems automatically detect conflicts, and notify at least one of the humans involved that their changes conflict with someone else's.
13. lock: A way to declare an exclusive intent to change a particular file or directory.

Version control system is used to organize software projects. A real life example of managing windows to show the working of the version control system is shown in Fig. 2. [8].

There's a main line with stable builds of Windows. Each group (Networking, User Interface, Media Player, etc.) has its own branch to develop new features. These are

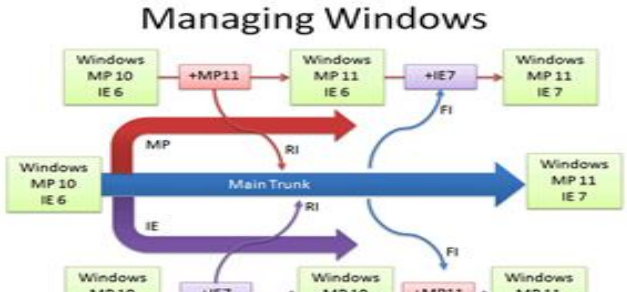

Fig. 2.Managing Windows using VCS

Media Player team makes version 11 in their own branch. When it is ready and tested, there is a patch from 10 – 11 which is applied to Main. This is reverse integration, from the branch to the trunk. The IE team can do the same thing. Later, the Media Player team can pick up the latest code from other teams, like IE. In this case, Media Player forward integrates and gets the latest patches from main into their branch. So it is RI and FI. This arrangement lets changes percolate throughout the branches, while keeping new code out of the main line. In reality, there are many layers of branches and sub-branches, along with quality metrics that determine when you get to RI. But you get the idea: branches help manage complexity. Now we know the basics of how one of the largest software projects is organized using version control system. Subversion is a VCS tool that is used in OSSD.

3.4 Subversion

Subversion is a free/open source version control system (VCS). That is, Subversion manages files and directories, and the changes made to them, over time. This allows you to recover older versions of your data or examine the history of how your data changed. In this regard, many people think of a version control system as a sort of "time machine." [9].

Subversion can operate across networks, which allows it to be used by people on different computers. At some level, the ability for various people to modify and manage the same set of data from their respective locations fosters collaboration. Progress can occur more quickly without a single conduit through which all modifications must occur. And because the work is versioned, you need not fear that quality is the trade-off for losing that conduit—if some incorrect change is made to the data, just undo that change. When working with the data on a daily basis, developers won't be able to copy, move, rename, or delete files the way they usually do. Instead, they will have to do all of those things through Subversion. "Subversion is an open source version control system" that by default uses a command line user interface on the client side.

4 Conclusion and Future Work

Open Source Software is software whose source code may be freely studied, modified and redistributed with few restrictions. Version Control is used in Open Source Software as a tool to manage distributed developers all over the world using the common files.

The data are stored in a Version Control System as configuration items, which are usually files. The most common configuration item is a source code file in Open Source Software Projects. When configuration items are modified, a new revision of the configuration item is created. Configuration items can have multiple modifications between versions, which are published revisions. However, one version of the configuration item can be incompatible with some target systems, so several parallel revisions and versions can be kept in branches. The branches are parallel development lines of the configuration items and software. The source code of the configuration item is duplicated, so different branches can be developed or maintained separately.

Our study provided an evaluation of OSS version control process from the viewpoint of developers distributed all over the world. The research papers and journals were used to produce more understanding about version control process and its basics and how the tools are used to find the evolution of the open source softwares.

The material presented in this paper is only the prelimary review study of the open source software and the importance of version control system in the Open Source Development. In the future the aim is to study the evolution of Open Source Software with large versions, over a period of time using the version control procedures & tool by using a version control tool that is subversion.

References

[1] E.E. Kim, "An Introduction to Open Source Communities, "Blue Oxen Associates Technical report, BOA-00007, 2003.

[2] Margaret Elliott, Mark S. Ackerman, Walt Scacchi, "Knowledge work artifacts: kernel cousins for free/open source software development," Proceedings of the 2007 international ACM conference on Supporting group work, November, 2007.

[3] R. Roets, M. Minnaar, K. Wright, "Towards Successful Systems Development Projects in Developing Countries", Proceedings of the 9th International Conference on Social Implications of Computers in Developing Countries, Sao Paulo, Brazil, May 2007.

[4] A. Beard, H. Kim, "A survey on open source software licenses", Journal of Computing Sciences in Colleges, vol. 22 no. 4, April 2007.

[5] B. C. Sussman, B. W. Fitzpatrick, C. M. Pilato, "Version Control with Subversion For Subversion 1.6", 2010, ISBN 9781440495878.

[6] S. Chamarty, H.D.Patel, M. V. Tripunitara, "An Authorization Scheme for Version Control Systems", SACMAT'11, Innsbruck, Austria, June 15–17, 2011.

[7] P. Weibgerber, M. Pohl, M. Burch, "Visual Data Mining in Software Archives to Detect How Developers Work Together," Computer Science Department, University of Trier, Germany, IEEE Fourth International Workshop on Mining Software Repositories 2007.

[8] Khalid Azad, "A Visual Guide to Version Control," Available: http://betterexplained.com/articles/a-visual-guide-to-version control.

[9] Ben Collins-Sussman, Brian W. Fitzpatrick, C. Michael Pilato "Version Control with Subversion For Subversion 1.5" (Compiled from Revision 2983 Version 1.4.4),2008, ISBN 9780596510336.