Chapter 10
Applicability of Lehman Laws on Open Source Evolution

Nisha Ratti
Rayat Institute of Engg. and Information Technology, India

Parminder Kaur
Guru Nanak Dev University, India

ABSTRACT

Software evolution is the essential characteristic of the real world software as the user requirements changes software needs to change otherwise it becomes less useful. In order to be used for longer time period, software needs to evolve. The software evolution can be a result of software maintenance. In this chapter, a study has been conducted on 10 versions of GLE (Graphics Layout Engine) and FGS (Flight Gear Simulator) evolved over the period of eight years. An effort is made to find the applicability of Lehman Laws on different releases of two softwares developed in C++ using Object Oriented metrics. The laws of continuous change, growth and complexity are found applicable according to data collected.

BACKGROUND

Open Source Software

The fundamental idea behind open source is making the source code available to public, so that any user can use it or modify it and redistribute it in the improved form. Open source help the users to interact with and learn from other users. There is no particular way to run an open source project. Some are democratic in nature and they welcome volunteers to contribute in all activities. In some projects, all the users work for one company, do all the development work and share the bugs and thereby think of the solution together. In some other projects, developers do not make any community at all, just share a web page. On that particular web page, they just share the development, let the other people download. And many a times, they send the response by emails. There are some wrong conceptions regarding Open Source Software Development (OSSD). Some people say that open source software development is a new software development paradigm. In fact, this paradigm is working since the age of ARPAnet and

DOI: 10.4018/978-1-4666-8510-9.ch010

UNIX. Another misconception regarding OSSD is that it is not done by professionals. But truth is that professional programmers are hired and paid huge remuneration to do the software development using this paradigm. Some people still believe that OSSD produce the low quality software. In fact, developers do the quality assurance during the development itself.

Open Source Software Development Process

In open source software development process, a number of groups of developers are formed. Every group has their own leaders. OSSD follow very less formalized method for development. Only a little number of projects has any explicit process for development. Overhead related to the introduction and enforcement of the formal method are the main reasons for not having any particular process model. Email is the prime means of communication among the developers. So the log of messages is maintained which in turn help in design decisions (Boldyreff, 2003).

W. Scaachi (Scaacchi, 2001) has given five type of development processes for open source software development:

1. Requirements analysis and specification
2. Coordinated version control, system build, and staged incremental release
3. Maintenance as evolutionary redevelopment, refinement, and redistribution
4. Project management
5. Software technology transfer.

Open Source Software Development Models

There are several basic differences between Open Source Software Development (OSSD) and traditional methods. The system development life cycle (SDLC) of traditional methods have generic phases into which all project activities can be organized such as planning, analysis, design, implementation and support (Satzinger et al, 2004). Also, open source life cycle for OSSD paradigm demonstrates several common attributes like parallel development and peer review, prompt feedback to user and developer contributions, highly talented developers, parallel debugging, user involvement, and rapid release times.

Vixie (1999) holds that an open source project can include all the elements of a traditional SDLC. Classic OSS projects such as BSD, BIND and SendMail are evidence that open source projects utilize standard software engineering processes of analysis, design, implementation and support. Mockus et al (2000) describe a life cycle that combines a decision-making framework with task-related project phases. The model comprises six phases like roles and responsibilities, identifying work to be done, assigning and performing development work, pre-release testing, inspections, and managing releases. Jorgensen (2001) provides a more detailed description of specific product related activities that support the OSSD process. The model (figure. 1) explains the life cycle for changes that occurred within the FreeBSD project (Kaur, 2011)

Jorgensen's model is widely accepted (Feller et al, 2001; FLOSS Project Report, 2002) as a framework for the OSSD process, on both macro (project) and micro (component or code segment) levels. However, flaws remain. When applied to an OSS project, the model does not adequately explain where or how the processes of planning, analysis and design take place (Kaur, 2011).