Krishnamurthy, A., O'Connor, R. V., An Analysis of the Software Development Processes of Open Source E-Learning Systems, 20th European Conference on Systems, Software and Services Process Improvement (EuroSPI 2013), CCIS Vol. 364, Springer-Verlag, June 2013 pp. 60-71.

An Analysis of the Software Development Processes of Open Source E-Learning Systems

Aarthy Krishnamurthy[1], Rory V. O'Connor[2]

[1]School of Electronic Engineering, Dublin City University (DCU), Dublin Ireland.
aarthy.krishnamurthy2@mail.dcu.ie
[2]School of Computing, Dublin City University, Dublin Ireland
roconnor@computing.dcu.ie

Abstract. In recent years there has been a rapid increase in demand for e-learning systems. The software development process plays a crucial role in the design and development of a high-quality e-learning system. However, to date, there is no comprehensive comparative study of open source software (OSS) development process for different OS e-learning systems. This hinders the development of a generalized OSS development process, a key requisite for rapidly developing high-quality OS e-learning systems. This paper provides a full analysis of different existing and successful OS e-learning software systems and the best practices followed in the e-learning development. In particular, this paper investigates the software development activities of Moodle, Dokeos and ILIAS. An activity flow representation that describes their current development practices is constructed individually for all three OS e-learning systems. Further, a comprehensive comparative analysis is carried out that leads to an explicit identification of various development stages of the three OS e-learning systems.

Keywords**:** Activity flow diagram, e-Learning, open source software, software development processes.

1 Introduction

E-learning can be broadly defined as transfer of knowledge and skills electronically, through different communication medium and devices [1]. Further, in e-learning systems, the learner is not always at a fixed, predetermined location. The learner can take advantage of the opportunities offered by mobile technologies [2]. The principal benefit is the ability to provide users the flexibility of learning and efficiently communicating anytime and from anywhere.

There are many e-learning systems that are developed successfully (Moodle, ILIAS, Blackboard, etc). They are developed either as open source software (OSS) or as closed source software (CSS) systems. Most of the commercial CSS have been developed based on either a traditional software process or some form of tailored traditional process, in order to accommodate local needs. These development processes have associated standards/guidelines that are followed for high quality software development. On the other

hand, OSS systems are developed by a community of like-minded developers, who are geographically distributed, yet work together closely on a specific software product [5].

OSS development (OSSD) has gained significant attention in recent years and is widely accepted as reliable products (e.g. Moodle, Apache, Linux, etc.). In today's times, the educational pattern evolves continuously with time. Hence, in order to keep pace with this evolution, the next generation e-learning systems need to evolve with the educational patterns. Before developing a generalized OSS process for e-learning systems, it is imperative to analyze and understand the existing and successfully running OS e-learning systems. To the best knowledge of authors, there has been no comprehensive study of the development activities of different OS e-learning systems. Hence, this paper carries out this study. In particular, this paper focuses on the development activities/process of three successful and highly popular OS e-learning systems - Moodle, ILIAS and Dokeos.

The development activities of these three OS e-learning systems were identified using two different approaches.

i. The first approach was to collect information from their websites, blogs, wiki pages and/or from any social network/media used by the community to broadcast the information. In addition, information was also collected from bug tracking system (or any other tracking systems), as some of the OSS communities track each of its development activities in such systems.

ii. The second approach was adopted only when the information collected from the first approach was either incomplete and/or ambiguous. It is a direct method wherein, questions were posted in public OSS community forums. The idea was to seek response directly from the community members, either through the same forums or through e-mails/private messages. However, the main disadvantage of this approach was that, there was no clear consensus from the contribution of the community members, on many occasions. In such scenarios, separate e-mails had to be sent to the core members and other experienced developers within the OSS development communities. This helped in identifying many nuances of the current development activities. Importantly, no analysis was done until all the information was gathered. This was strictly followed to avoid any ambiguity due to incorrect assumption of the current development practices.

Once the information on the development practices of each of the three e-learning systems was gathered, an in-depth analysis was carried out. The results of the analysis were then modeled using activity flow diagrams. The activity flow diagram representation was used because it provided a dynamic aspect of an overall flow of the development practices followed by the OS communities. The activity flow diagram was preferred over other approaches like state diagrams and event driven process diagrams as it would indicate an overall flow of activities carried out within the community for its software development.

The paper is organized as follows: Section 2 describes in detail the development activities of the each of the three OS e-learning system. Subsequently, section 3 compares the development practices of the three e-learning systems. Finally, Section 4 provides a brief conclusion along with the next research steps in this direction.

2 DEVELOPMENT ACTIVITIES

Over the years, the development activities of OSS have become openly visible to all and the development artifacts are publicly available over the web. Further, there is little need for formal project management, virtually no budget and often a very flexible schedule. OSSD is oriented towards the joint development by a community of developers [6]. All the three systems considered for analysis in this paper are OSS and follow their own development practices. In this paper, they are investigated through a case-study since there is very little literature available that discusses the OS e-learning systems and their development activities. The case study approach was selected as it will result in a detailed analysis of the developmental activities; thereby leading to comparisons which would help in understanding the similarities and differences on the practices followed by OS e-learning system development community. All three selected OS e-learning systems are quite popular worldwide and importantly, receive significant and frequent contributions from volunteers for performing various development activities regularly. The development activities of Moodle, ILIAS and Dokeos are described as follows.

2.1 Development Activities of Moodle

Moodle stands for 'Modular Object-Oriented Dynamic Learning Environment' and is one of the early and most successful OS e-learning platforms which follow strong pedagogical principles. It has the largest user-base and has the benefit of having the largest market share and highest satisfaction in small companies, educational institutions and government organization's LMS [3]. The different activities of Moodle development are shown in Fig. 1. It should be noted that Moodle does not have a clear bifurcation between various development stages. Contrastingly however, the development always starts with selecting the right candidate feature [7].

The first step of development involves selecting the right candidate feature. For selecting a candidate feature, the community pools the entire feature requests raised in the Moodle moot discussions, user's feature request from forums and feature request from moodle vendors. These candidate features are then voted for entering into the release roadmap list.

Any developer interested in developing the new feature listed in the release road map could initiate a discussion with other fellow community developers, in order to ensure that no one else is working on that requirement/feature. The developer(s) would then discuss their ideas with others, confirm the merits and the need for the particular feature, and importantly, evaluate theirs and other's ideas. Once the feature is selected for development by a Moodle developer, he/she is expected to come with design documents along with other specification documentations. These documents are then posted in the Moodle wiki. In addition, a tracker item is created for the feature and assigned to the developer. Subsequent changes are then carried out, based on the feedback received by the developer in the respective documents which are then updated in the wiki. Once the changes made are agreed by the Moodle community, the developer begins coding. When the development is completed or a major milestone is reached, it is the responsibility of the developer to advertise the feature for testing. Testing could be done by interested candidate(s) within the Moodle community. Subsequently, any bugs found are then reported

and fixed. It is then integrated with the main version of Moodle and then released as a new version, which would be open and freely available.

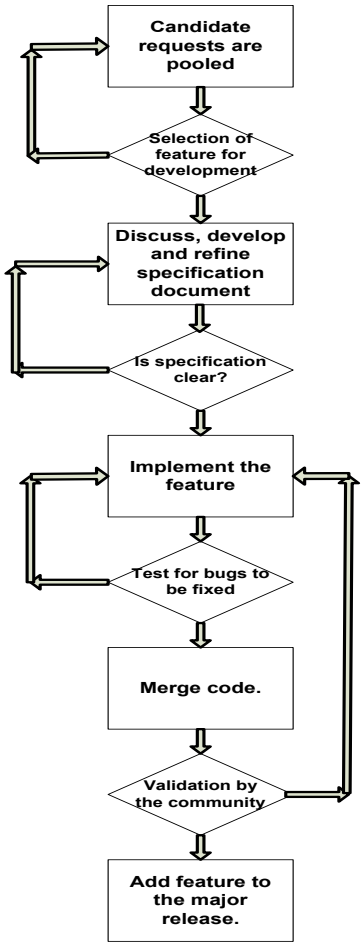

Fig. 1 Activity flow representation of Moodle

Additionally, a tracker item is created for the feature and assigned to the developer. Subsequent changes are made, based on the feedback received by the developer in the respective documents which are then updated.

2.2 Development Activities of ILIAS

ILIAS stands for Integrated Learning Information and co-operAtion System). It is a popular web based learning management system (LMS) / OS e-learning systems and comprises of six stages of development [8]. They are; Vision/Concept, Specification, Implementation, Documentation, Testing and Release & Maintenance. In each of these stages,

the OSS community performs various developmental activities which can be observed clearly in Fig. 2. The details of each stage are described as follows:

- ***Vision/Concept Stage:*** This is the first stage of development wherein, ideas are proposed and published in wiki. The core development team will then decide on how to start the development. If the idea is already been put on to the feature wiki, people with similar interest are requested to work with them and develop the feature collaboratively.
- ***Specification Stage:*** In this stage, all major development is expected to have corresponding use cases or mock up screen-shots. For other minor developments/enhancements, developers would start with the feature wiki where it will describe the feature in detail, the purpose, etc.
- ***Implementation Stage:*** The third stage which is the implementation stage, where the coding/programming is done by the developers. Each module that is developed in this stage is tested by the developer who also fixes the initial bugs that comes across. Further, the developer would either perform a unit-testing using PHP Unit, or get it done by a tester. Subsequently, the code is then merged with CVS.
- ***Documentation Stage:*** There are two types documentation prepared for a feature developed for ILIAS - technical documentation and user documentation. The technical documentation consists of the class and functional documentation generated by PHP Doc. The user documentation will be mainly instructions for the average user on how to use it. The user documentation is only released at the time of release of the product.
- ***Testing Stage:*** The testing stage mainly follows the implementation stage. In this stage, once the unit-testing and code merger is done, an alpha release is carried out for further testing and bug fixing. It is the responsibility of the developer to appoint a tester to test the module developed by him. If the developer is unsuccessful in finding a tester to test his/her module, then the core team would carry out the required testing. However, in any case, the developer himself cannot be a tester for his own developed module
- ***Release Stage:*** The final stage during the development is the release stage wherein, the new modules that have undergone alpha testing are released under the beta version. Errors/bugs encountered after the beta release are then entered into the bug tracker (Mantis bug tracker). These bugs are then fixed and released as the main stable version.

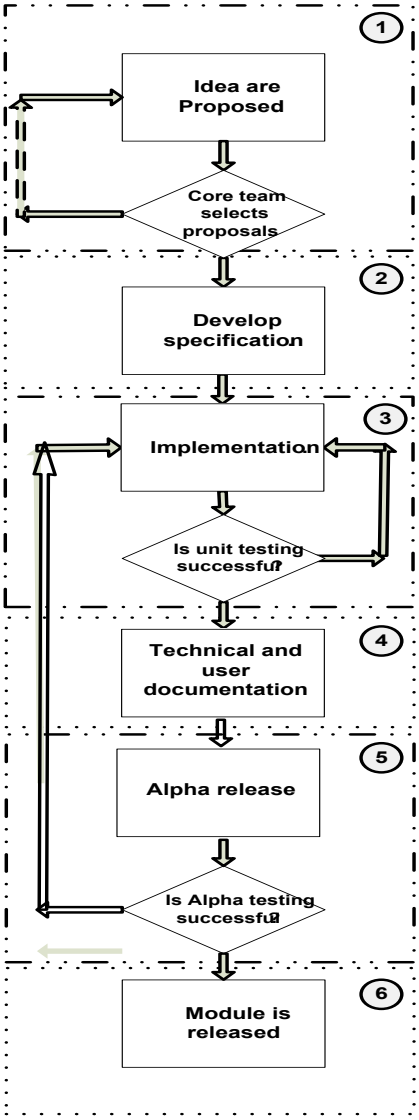

Fig. 2 Activity flow representation of ILIAS

2.3 Development Activities of Dokeos

Dokeos is developed both as commercial and OSS version. The development of OSS version is the responsibility of the Dokeos community – from initiation of idea through release. Although there are two different existing systems, the OS version does provide all the basic features for free without any licensing cost to its users.

Dokeos community does not follow any defined stages as in ILIAS, but often, they do perform some activities in a particular order as shown in Fig. 3. Development of a feature starts with feature selection where the selected feature is added to the roadmap for development. The feature is then developed by the community of developers. The features are first tested before it is given it to the users for further testing. Any anomaly, if and when found are fixed and passed on to the users for user testing.

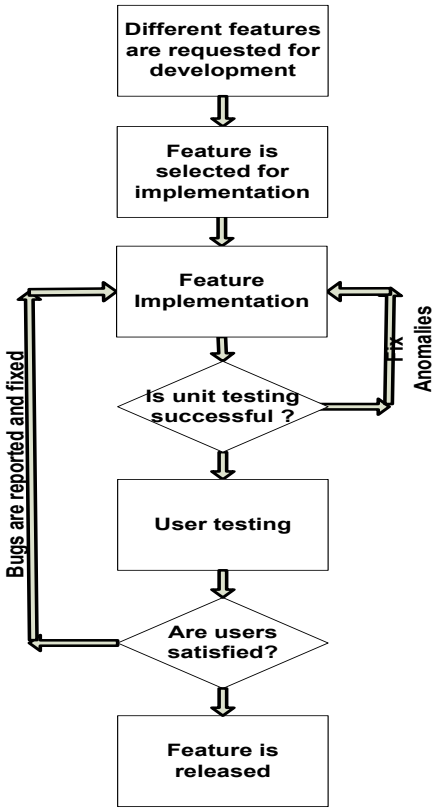

Fig. 3 Activity flow representation of Dokeos

The users would test the developed feature and if there is any bug(s), it would be reported. These bugs are then fixed and sent to the user again for testing. Once the user is satisfied with the features, they are subsequently released to the community as a stable version. All the users could then download it for free and use the same.

3 Comparison of OS E-Learning System Development Practices

The individual analysis of the three OS e-learning systems carried out in section 2 provided details and insight into the different activities carried out at different stages of development. Notably, the manner in which each stage is carried out would depend entirely on the expertise, experience and availability of resources and skills. There are distinct similarities and differences between Moodle, ILIAS and Dokeos on different as-

pects. These are summarized in Table 1. The comparative analysis begins with differences in number of developmental stages. The common developmental activities in each of the stages are then compared, based on different factors like, how it has been performed, who performs it, etc. Each of these differences and similarities are discussed briefly and is described as an observation and critique.

Table 1 Comparison between three OS e-learning system development

LMS / Parameters	Moodle	ILIAS	Dokeos
Number of development stages	Do not categorize development stages	Does categorize six development stages	Does not categorize stages
Who validates the proposed idea	Anyone can validate the idea and comment on it	Only the core team validates the proposed idea	Does not validate the proposed idea at this stage
Detailed development plan	No plan is produced	No plan is produced	No plan is produced
Person(s) responsible for development	A person who volunteered initially & the team that was formed latter on the fly.	A person who volunteered initially & the team that was formed latter on the fly.	Any interested volunteer engages in developing the software.
Testing	Anyone can test at any time.	Anyone can test at anytime.	Anyone can test till the product is released.
Release	Two stage release process is followed.	Two stage release process is followed.	Once the testing is done & bugs are fixed, the product is released. There is no beta release.

- **Number of Software Development Stages**

  Observation: In ILIAS, it is easy to identify different development stages /phases during development. However, Moodle and Dokeos do not categorize different software development stages, even though it has many tasks similar to ILIAS.

  Critique: Having defined stages or phases of development are important as it aids in easy tracking of the development activities as well as assists in planning and testing different phases independently.

- **Scrutiny of the Proposed Idea**

  Observation: New ideas proposed to Moodle and ILIAS is scrutinized immediately after its proposal. At the same time, there is one major difference between Moodle

and ILIAS. In case of Moodle, anyone who is interested in the new idea, including the core team, co-developers, testers, users, etc. can read the proposal document and comment on it. Based on the received feedback, the core team or the core members will signal the development. However, in case of ILIAS, only the core members will review the idea/feature and would decide its future. On the other hand in Dokeos, specifications are not detailed or developed for idea scrutinization.

Critique: Assessing the features credibility and need even before the specifications are developed might lead to inappropriate judgment with regard to the features need and importance.

- **Person(s) Responsible for Specification/Scrutiny**

Observation: In case of Moodle, the entire community could scrutinize the specification by reading the proposal document and commenting on it. Based on the feedback the core team/ members would either agree/ disagree with the idea. On the other hand, as compared to Moodle, ILIAS has a different approach. In case of ILIAS, only the core members would scrutinize the idea/feature decide its future. On the other hand, Dokeos does not have any such activity and therefore the community is not responsible for the same.

Critique: Being open source and built by users for users, the specification validation should be kept open. This will make sure that the specification is acceptable from the OSS user's point of view. This is very important because, in all cases, development happens based on the specification. If the specification happens to be wrong, then the developed feature would go wrong. This is true for all the software products including OS e-learning systems, irrespective of the development method followed.

- **Developmental Plan**

Observation: In all three systems i.e., Moodle, ILIAS and Dokeos, there are no explicit plans portrayed for its development. It is the responsibility of the person incharge to develop the feature as agreed upon. At the same time, it is the individual or team's responsibility to answer all queries regarding the module/feature development.

Critique: Even though having a defined plan is beneficial in tracking the development; it is very complicated to come up with plans and follow it strictly in the OSS environment where the volunteers develop the product during their free time.

- **Person Responsible for Development**

Observation: In Moodle and ILIAS the person who agreed to develop the feature takes responsibility of its implementation. Further, the team formation happen on-the-fly based on the personal interest of the community member(s). If anyone is interested in its implementation, testing, documentation, etc. they volunteer to the working group/person.

Critique: Even though having a defined plan for developing a feature may seem to be a 'failsafe' approach, it is not practical to follow such a structure in an OSS environment. This is especially so, when a feature is developed by geographically distributed community members who volunteer to do the same in their spare time not just for themselves but also for others.

- **Testing**

Observation: In all the three OS e-learning systems, any individual from the community who is interested in a particular feature can test the developed code for any potential bug(s). However, there is one notable difference. In case of Moodle and ILIAS, the common ground testing could be carried out even after new versions are released. On the other hand, in case of Dokeos, this type of common ground testing could be done only till the product is released.

Critique: Testing is one of the important activities in producing a quality software product. OS software products are usually well-tested due to the large number of user-base/testers, who are geographically distributed, have varied skill sets and could test the module/feature independently.

- **Product Release**

Observation: A two-stage testing process is employed in case of Moodle and ILIAS. Once the initial testing is over, both Moodle and ILIAS release their features as a 'beta' version. Subsequently, this is tested again. Once the testing is completed, the features are then finally released along with other items as final version of the major product release. On the other hand, Dokeos does not have any beta release. The feature(s) are tested by users/community once it is developed and the bugs are reported. Once the encountered bugs are fixed, the feature is subsequently released.

Critique: Having a beta test stage will enable identification of problems before the integration to the stable version. This would potentially save any additional costs (in most cases it's the time spent by the OSS community) that might have to be incurred if the stable version is corrupted.

4 CONCLUSION

This research work carried out a detailed individual analysis and a comprehensive comparative analysis of the development activities in the three major open source e-learning systems - Moodle, ILIAS and Dokeos [11]. The results of the analysis were presented using an activity flow diagram representation. The comparison demonstrated the clarity and explicitness of each development stages carried out in the three OS e-learning systems. Further, for any differences identified in the development practices, a corresponding critique has been presented. This resulted in a better understanding of the best practices followed in the three OS e-learning environment.

Significantly, there were two major limitations encountered in the activity flow representation. Firstly, it did not identify the actors involved in carrying out various tasks and secondly, it did not explicitly specify the outcome of a development activity. Hence, having identified the different development stages, the next step in the design of generalized OSSD process for e-learning systems would be to construct a high level abstract model which would focus on the actors performing the activity and their outcome.

Acknowledgment
The author would like to thank Irish Research Council (IRC) - Embark Initiative Program for their support.

References

1. Tavangarian D., Leypold M., Nölting K., Röser M.,(2004). *Is e-learning the Solution for Individual Learning,* Journal of e-learning, 2(2).

2. MOBIlearn, 2003. Guidelines for learning/ teaching/ tutoring in a mobile. Technical report [online]. Available from: http://www.mobilearn.org/download/results/guidelines.pdf..

3. Steve Wexler, Lance Dublin, Nancy Grey, Sheila Jagannathan, Tony Karrer, Margaret Martinez, Bob Mosher, Kevin Oakes, and Angela van Barneveld, 2007. " *LEARNING MANAGEMENT SYSTEMS (LMS)"* in GUILD Research 360 degree report. Available from: http://www.elearningguild.com/research/archives/index.cfm?id=120

4. Clarke, P. and O'Connor, R. 2010. *Towards the identification of the influence of SPI on the successful evolution of software SMEs IN Dawson, et el (eds.).* Proceedings of the 18th Int. Conf. on Software Quality Management.

5. W.Scacchi et. Al., "*Understanding Free/Open Source software Development Process*", Softw. Process Improve. Pract, 11, 2006.

6. W.Scacchi, "Software *Development Practices in Open Source Software Development communities: A Comparative Case Study*", 1st workshop on Open Source Software Engineering, May 2001.

7. Moodle. 2011. *Moodle Development: Overview* [online]. Available from: http://docs.moodle.org/en/Development:Overview.

8. ILLIAS. 2011. *ILIAS Development Guide [online]*. Available from: https://www.ilias.de/docu/goto_docu_lm_42.html.

9. Sakai, 2011. *Sakai development work. [online]*. Available from: https://confluence.sakaiproject.org/display/MGT/How+Sakai+Development+Works.

10. Nichols, D.M. and Twidale, M.B. 2003. The usability of open source software. *First Monday*, 8(1).

11. Krishnamurthy, A. and O'Connor, R. V. Using ISO/IEC 12207 to Analyze Open Source Software Development Processes: An E-Learning Case Study, In: Woronowicz, T., Rout, T., O'Connor, R.V., and Dorling A.. (eds.) SPICE 2013. CCIS, vol. 349, pp. 107–119. Springer, Heidelberg (2013)