Wave Process Model for Open Source Software Development

Ermias Kebede
Software Engineer, Switzerland
ermi.kbd@gmail.com

Oliver Petzoldt
iceaddis Innovation Hub, Addis Ababa, Ethiopia
oliver.petzoldt@cimonline.de

Abstract

Wave Process Model (WPM) is a new form of modeling approach to represent the relationships between processes and activities in the form of a wave graph. In this explanatory research, WPM is applied to describe the open source software further development process. As being a lightweight model, on the y-axis we placed processes of the selection, the reuse, the modification, the extension and the contribution. On the x-axis there are activities which can be realization of internal requirements or design patterns. The intersection of processes and activities are tasks to be accomplished by the software enhancement.

In the research result, we find an open-end description of the open source software further development and its process model having parallel and concurrent WPM parts. With a prototype development, we could see that the WPM concept can be applied for project management and controlling tool. These facts are more elaborated with case studies of deploying, internal requirement, redesign and source code further implementation parts to enhance open source software for additional use.

Keywords: Process Model; Software Development; Open Source Software.

1. Introduction

Process models have a view of interest, which are activity, product, resource or role based models [4]. Software development life-cycle models are divided into descriptive or prescriptive [12]. The methodology or models which are used to develop computer systems are said to be related to the activity chain of the development process namely from planning till release [2, 3]. We preferred to describe the WPM as a chain of activity in the process where activities and processes are related with tasks or assignments.

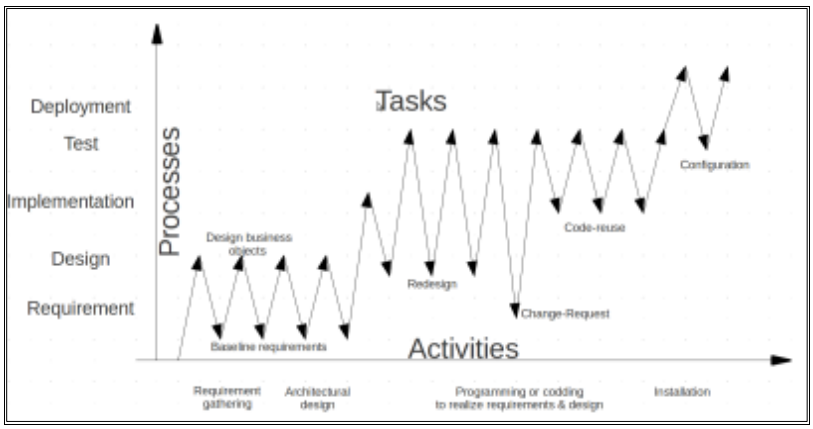

Figure 1: WPM-Sample

Software development process models like the waterfall model, spiral model and agile method seem to lack describing some process and activities involved in the open source software further development OSSFD [8, 9]. Activities by OSSFD were not best compatible to be managed with existing process models [5].

As the primary purpose of the life cycle model is to communicate the work to be done among human beings, work unit leveling is achieved by decomposing [7]. We took the decomposition of work units into waves, waves into activities and activities into tasks. This means a wave describes a kind of phase. What we add in the WPM is the tasks

to demonstrate the relationship between processes and activities as shown in Figure 1.

Requirement engineering is essential when an institute wants to adopt an OSS, because when we customize or add features, we are dealing with requirements which may not exist in the OSS. The other use of it is also if we should select an OSS we may evaluate it with our internal requirement. In this case we added internal requirement in case of OSS further development process. The important part of our research is the concept of the WPM itself which was enforced by the prototype development and the case studies.

2. Background

From the classifications of OSS development framework [11], deploying OSS product, using the OSS CASE Tools and integrating OSS Components talks in general about usage. The other three parts participating in the OSS products are related to development. OSS is there in the Internet cloud; we can use it or develop it further. In OSS, we do not only have software, but also communities [1]. Additionally, How-To, FAQ, and Blogs are where we find Information [8]. Information are like bug request (as requirement), definitions and even design documents.

To select an OSS, we may evaluate it to choose the appropriate one [11]. After selection we can deploy an OSS to use it or to develop it further. Basically there are contributors which are users and developers. Users are those who use OSS and developers are those who provide or develop OSS further. There are individuals, companies, or organizations who participate in OSS development [10].

Those who provide OSS may start from scratch at the beginning but when they released it, OSS can be selected and further developed from interested participants. Further development activities are done with reusing, modification, and extension. Reusing can mean to integrate code-example by copy and paste or calling a predefined method or class. Usage means using OSS applications, tools, concepts, and communication media.

Communications are in chat-rooms, e-mail or community websites. Contribution can be anything like feedback, bug request or financial matter [8]. There is no as such a defined life cycle, rather an open-end further development process as shown in Figure 2.

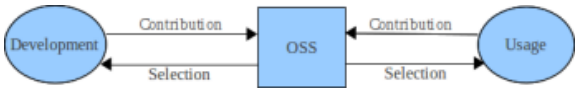

Figure 2: Open-Source Software Life Cycle

If we are going to further develop an OSS, it means we are going to reuse what is there and add more modules or features by modifying and extending some parts [6]. If we are going to have our own individual requirements, we can do it by ourselves. If we are going to have our own development process model, we should not forget to include selection and contribution. Selection is because of using OSS tools and other products [11]. Contribution is not only because of the philosophy but also for information storage.

3. The Proposed Solution

The life cycle for OSSFD can be described in the open-end interaction as shown in Figure 1. Like production and consumption of a product, we have OSS development and usage.

The process modeling approach for further development of OSS is a group of processes which have their own WPM. They can communicate with selection and contribution at any time.

The selection process is there to get information from OSS communities, FAQ, How-To, Repositories, issue lists, code-examples, etc. It suggests contributions of comments, blogs, multimedia materials, etc. The WPM features for the OSS further development are selection, reuse, modification, extension, test and contribution as shown in Figure 4 in each WPM graph of deployment, requirement, redesign and implementation. The activities can be gathering requirements, database design, customization, realization of specific class, and others where we get the input for specific assignments or tasks.

Table 1: Process Names

Process	*Reusing*	*Modification*	*Extension*	*Test*
Deployment	Installation	Configuration	Plug-In, Add-on	Executing,
Requirement	Use-Baseline	Change-request	Feature-request	Validation
Redesign	Use-pattern	Modify	Extend	Validation
Further Implementation	Copy-Paste, Class-Reuse	Customization, Re-factoring	Add-New class, method or else	Run, review, Experiment

The WPM graphs for the parallel processes in Figure 3 can have appropriate names.

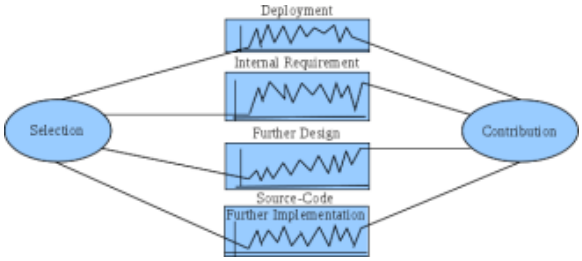

Figure 3: WPM approach for OSSFD

Let us clarify the process for Figure 3 and Table 1. In case of deployment why we choose installation as reusing is because it is the first thing what we do after selecting an OSS. By the configuration tasks, we may modify configuration files. It is also true that some OSS may need other plug-in installation if we need to have additional features. A test can be simply executing the installed application; if it fails we may need to troubleshoot by selecting again a solution in the community website or by applying our previous experience.

Figure 4: WPM for OSSFD Processes

Internal requirements are there if we would like to customize or add new features in the original OSS. In this case we may gather additional requirements. Baseline requirements could be existing artefacts, known business processes like tax calculations given from the government, etc. We can reuse a document from previous work process, reuse universal formulas, facts and domain specific requirements. But when we would like to modify some behaviors we are talking about a change-request. The other part what we mean with the extension is a kind of new feature request. Since we have a test part, the appropriate word for that is validation. Contributions can be then communicating or writing the requests and issue lists in the form of a notice, a table or other form of a document.

Design patterns are there to be reused. Normally in the OSS itself we have already the design in the source code. But in case of forking we may even need to redesign. Modification may take place when we have to change the user interface. Extension examples are there when we have to add additional tables and relationships in the database design.

By source code enhancement or further implementation, we can select an internal requirement, a bug-request, a source code, a design concept, an interface, etc; by exploring, inspecting, analyzing etc; as an input for translating a concept to the source code. Reusing can be performed with copying and pasting a code example, see how class can be implemented in the programming language, how to call a specific method; etc. Modification can be applied by changing configuration values or renaming some attributes to use a specific variable for other purposes. Moreover, we can add new classes, new methods, new algorithms, new scripts, new concepts or else, to add new features, to realize a plug-in or to append additional modules. The test part can be just running the application to find out if the reuse, the modification or the extension we have implemented has worked properly. Contributions can be done with documenting in the source code, commenting while committing and more.

4. Prototype and Case-Studies

4.1 Prototype

The prototyping let us demonstrate how we can show the application of the WPM. Moreover, it supported us to refine the WPM. For example, associating the tasks with activities let us notice that tasks are repeated, feeding the case study data in to the database to display them and practicing to select and reuse OSS products and tools.

The requirement were to realize the WPM graph for recording and displaying the processes, activities and their relationship with the tasks or self assignments. For that matter, we had three tables respectively and their relationships with the corresponding work units on the database and in the source code we had also the mapping classes. How the prototype is realized was also by applying the reuse, modify and extension methodology.

In the development of the prototype, we used the three tier architecture of the application as an input for the activities. Activity-1 = user-interface realization, the second is the application logic for the WPM graph, the third is the database design with source-code mapping, and the forth is their integration as further development activity.

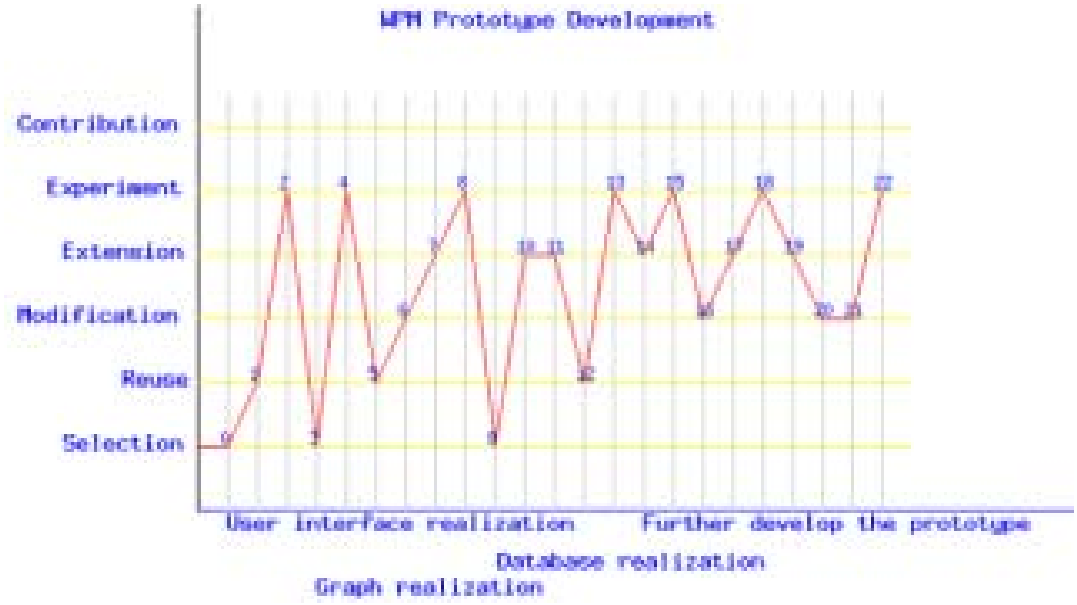

Figure 5: WPM for Prototype Realization

We can see the WPM graph instance in the prototype development in Figure 5. The numbers are those assignments done in the development activities. Those activities were recorded in the database and displayed in a list-box. For example, the following notices are for the tasks numbered here from 3 to 8.

3. Take code example for drawing lines.
4. Test the selected code-example to display a simple graph.
5. Reuse a simple code-example of PHP class concept for creating a WaveGraph.
6. Modify the created class to have the graph attributes (String, int, Array, image) and group the structural code to different functions (Coloring, axis, lines).
7. Extend the WaveGraph class by adding functions for writing strings on the x-axis (Process), on the y-axis (activity), intersection lines from-to (for tasks), and for having extra dimensions
8. Experiment with the new graph image output.

4.2 Case Studies

The case-studies, which are listed in Table 2, were conducted as an action research to practice the methodologies of selection, reuse, modify, extend, test, and contribution processes. In that way we could also record those tasks in the prototype database to display them as a WPM graph. The case-studies produce plenty of information and WPM graph for

deploying different OSS products, gathering internal-requirement, customization and plug-in development over existing OrangeHRM OSS for human resource management. The WPM graph had different forms like a sound wave or sometimes like a heart bit.

Table 2: Activity Samples

Case Study	*Activity Example*
Deployment	Installation of Ubuntu, Dspace, CVS and OrangeHRM
Requirement	Customization request of different user-interface and realization of payroll module from baseline requirement.
Design	Extending the payroll module for each Model, View and Controller by following the design pattern in the OrangeHRM OSS.
Implementation	Modifying the source-code for each customization request. Extending the source-code by reusing existing code-examples in the OrangeHRM itself.

5. Related Work

For WPM, it was not possible to find a related work, since this is a new concept. We have presented somehow related works for OSSD.

In our literature review "Do Best Practice Frameworks fit Open Source Software Customization" [5], we have seen the efforts made to use existing frameworks for the open-source customization in case of ITIL, SPICE and V-Model.

Another thing we discussed is the part which tried to use agile method for their open source further development activities in the thesis research paper of "Using Open-Source Solutions in Agile Software Development" [9].

What we find is also a generalized process modeling in the form of UML for describing the OSS development activities [10].

The above works are nearly related to our specific objective of our research, which we have already included them in the literature review part.

6. Conclusion

We have validated that WPM can be used to detect activities and to control what we were doing. More precisely, we described the further development process using WPM. In finding out the types of activities, in case of internal requirement gathering, which is inspecting and then feedback, lets us to see a methodology of implementing the requirement as they came. The other methodology we identified to have the activities per the design pattern is also another methodology how the design concept can be an input for the activities. The data selected by the case studies are not for evaluating purpose but for describing the WPM concept as a whole and that the WPM is a valued concept. As a future work, the WPM concept can be researched to apply it for other types of software development processes or for any kind of work process management and controlling.

References

[1] Gautam Guliani and Dan Wood, "Open Source for the Enterprise", O'Reilly, July 2005.

[2] Jeremy Lewis, "Software Development Life Cycle (SDLC) 100 Most Asked Questions, SDLC Methodologies, Tools, Process and Business Models", 2008.

[3] Sommer Vill, "Software Engineering", 8th Edition, Addison-Wesley, 2007.

[4] Jean Claude Derniame, Badara Ali Kaba, and David Wastell, "Software Process Model Principles, Methodology and Technology", Lecture Notes in Computer Science, Springer Verlag, 1999.

[5] Keßler Steffen and Alpar, Pau, "Do Best Practice Frameworks Fit Open Source Software Customization?", Institute of Information Systems, Philipps-Universität Marburg, Germany, 2009.

[6] Markus Pizka, "Adaptation of Large-Scale Open Source Software - An Experience Report", Technische Universitaet Muenchen, Institut Fuer Informatik, Munich, 2004.

[7] Life cycle Model, The concept behind the design of Software Life Cycle Model http://www.chambers.com.au/sample_p/c_pmodel.htm, Last accessed on April 05 2012.

[8] Free/Open Source Software Development: Results and Research Methods Master's Thesis, Otso Kivekaes, University of Helsinki, Department of Computer Science, June 1, 2008.

[9] Tuukka Haapasalo, "Using Open-Source Solutions in Agile Software Development", Helsinki University of Technology, Department of Computer Science and Engineering Laboratory of Software Technology, January 17, 2007.

[10] Stefan Dietze, "Modell und Optimierungsansatz für Open Source-Softwareentwicklungsprozesse", Potsdam University, 2004.

[11] Adoption of Open Source Software in Software-Intensive Industry, Department of Computer and Information Science, Norwegian University, Øyvind Hauge, NTNU-trykk, Trondheim, June 17th, 2010.

[12] Walt Scacchi, "Process Models in Software Engineering", Institute for Software Research, University of California, Irvine, J.J. Marciniak (ed.), Encyclopedia of Software Engineering, 2nd Edition, John Wiley and Sons, December 2001.