UNIFIED SOFTWARE DEVELOPMENT MODEL FOR FREE/OPEN SOURCE SOFTWARE

Md. Anawarul Kabir[1], Md. Salahuddin Pasha[2] and Mohammad Abdur Razzak[3]

[1]Department of Computer Science, American International University Bangladesh, Bangladesh
kabir@aiub.edu
[2]Aamra group of companies, Banani, Dhaka, Bangladesh
salahuddin.pasha@aamra.com.bd
[3]Master of Science in Software Engineering, Blekinge Institute of Technology, Sweden
razzak@live.se

ABSTRACT

Most of the process models so far have been introduced in the domain of software engineering are meant for proprietary software. Though, in the case of open source software development, the popular Bazaar model can be mentioned, it has some limitations too. The present research has attempted to formulate a process model for developing a software project exclusively for the open source paradigm. We have named it "Unified Software Development Model for Free/Open Source (USDM)".

Not only we have introduced USDM, we have also validated this new model by initiating a project on Bangla pre-processor in the domain of open source development.

KEYWORDS

Free/Open Source, OSS, software development, process model,, software engineering, Bangla OCR & unified software development process model.

1. INTRODUCTION

In the open source paradigm any community led project's success depends on the community interests and size of the community. Most of the open source project developers are working remotely, mostly in their free time and mainly on their personal interest [1] [2]. But in the proprietary software development the project manager sets a target and the developers are bound to work according to the plan [3]. It creates a huge difference between the development of an open source and proprietary software [4]. Bernard Golden [5] also differentiates the software engineering (or software development process model in particular) for open source paradigm from the conventional proprietary software development significantly.

So, the popular process models, so far developed for the proprietary software development, cannot be directly applied to the OSS paradigm, due to its salient features as stated above. The objective of this research is to formulate a new software development model for open source software. To fulfil this objective we have proposed a process model which emphasizes on:

- Development of open source software: Enables the remotely connected volunteers to develop open source software regardless of community size.

- Maximum utilization of developers time: Enable the volunteer developers to contribute to a project by spending little time and efforts.

- Minimizing wastage: Minimizing any kind of wastage.

For this, we have introduced a software development model for open source software to make any project successful with any number of decentralized developers. Contrary to Bazaar[1] model preconditions [6], we have emphasized on developing any open source project regardless of the community size.

Our proposed model will enable the development of open source software with regardless of the community size and ensure maximum utilization of developers' time. We have validated this new methodology by initiating a project on developing a pre-processor for Bengali (Bangla) language OCR.

2. RATIONALE OF THE RESEARCH

Our literature survey has revealed that the existing software development methods emphasize on various metrics of the project, for instance, project type, available resources, timeframe, environment type and some others. But the existing development methods hardly consider any option for development using remotely connected volunteers (developers). However, working with dispersed and remotely connected volunteers (developers) is a common scenario for OSS projects. In a case, arranging a physical meeting among the developers is not possible [7], though which is an important task for most of the software development process models. Moreover, it is hard to develop integrated open source software with a few remotely connected volunteers.

The community developers have to spend a lot of time and efforts to understand any part of the project. Sometimes developers have to go through some irrelevant documentation while working with a small part of the project. These may be termed as wastage. In this context, we have suggested that there is also no mechanism for minimizing wastage in open source[2]. But proprietary software development generally avoids this type of wastage[3]. Duplication is another type of wastage in the OSS paradigm. By duplication we mean, same components of software are developed by different developers due to lack of communication among them.

Developing multiple copies of same component is, no doubt a huge wastage of programmers' efforts[2]. The above stated limitations of OSS paradigm have initiated the present research with a view to finding out a possible solution.

[1] In the Bazaar model the source code is available for public throughout the development process.[6]

[2] Mozilla Add-on developers have to define a minimum and maximum version [9]. Due to the version changes in software, many dependent parts of the software become unusable. As a result, all the modules exceed the defined version have to be updated, though the newer version have no effects on all the modules.

[3] Apple iPad capable to run the software developed for previously released iPhone/iPod (touch) without any modification [10].

3. FREE/OPEN SOURCE AND PROPRIETARY SOFTWARE

The word "Free/Open Source software" describe the software that enable access to the source code of the software and certain[4] freedom to the users [11] [13]. It is like a transparent system.

The Open Source Initiative [12] and Free Software Foundation [11] two separate organizations; who working towards similar goal, but not the same. "Open source is a development methodology; free software is a social movement" [14]. Some people get confused about the word "free". According to the definition, "Free software is a matter of liberty, not price" [14].

On the other hand, the "proprietary software" allows the users to use the program in a restricted[5] way, like a black box system, thus the software freedom is not preserved[15]. The term "proprietary software" often confused with "commercial software". Commercial software can be either open source or proprietary [15].

4. A FOCUS ON SOME OSS PROJECTS

The Linux Kernel project can be cited as a perfect [16] [17] open source software development as it involved a large number of community developers [17]. Initiated by Linus Torvalds in 1991 [18], the success story of this project lies in the involvement of large number of developers and users in the effort.

In the "The Cathedral & The Bazaar" Eric S. Raymond, compared linux kernel development model with the Bazaar model and mentioned, Linus's main success is the design of the development model not the linux kernel [6].

Linux model is an example of high shared conceptualization, high modularity [8]. A stable design usually suits the high-high case. The open source projects lead by any commercial company generally follows the high-low case [8].

The Bazaar model successfully carried out a few large scale software projects with the remotely connected developers communicating via internet mailing lists [6]. One major drawback in the Bazaar model is, in this model remotely connected volunteers work on the project and the success of the project vastly depends on the interest of new volunteers. It has been observed by Karl Fogel, "Most free software projects fail" [19]. Lack of community interest can lead a project to fail at the beginning. Similarly, an active project can become stagnant if the community people lose interest.

Sahana is a Free and Open Source Disaster Management System. The Sahana [20] FOSS project provides an interface using "Sahana Application Framework" that enables modulated development environment for optional and third party modules.

A well designed project can be accomplished within a very short period of time. The Sahana FOSS project was built over a 2-3 week period by a group of volunteers from the Sri Lankan IT industry, spearheaded by the Lanka Software Foundation, a FOSS R&D NPO as the project was well designed [21].

---

[4] "Free software is a matter of the users' freedom to run, copy, distribute, study, change and improve the software" [11]. "Open source doesn't just mean access to the source code". It also ensures other freedom (Free Redistribution, Source Code, Derived Works etc.) [12].

[5] Depends on the copyright owner.

So many open source software are available throughout the Internet. As in June, 2011, more than 296,743 software projects have been registered in SourceForge.net[6] [22]. But a very few open source projects are able to attract the community like the linux kernel.

5. PROCESS MODELS AND OPEN SOURCE SOFTWARE PROJECTS

Process models are different methods of accomplishing a software project in an organized manner. Each process model has some advantages for specific type of project with the available resources and timeline [23].

In most of the open source projects, requirements are not well defined at the beginning [24]. In fact, every release in such a project generates new ideas and specific needs of different groups which, in turn, produce updated requirements. For this reason, the open source software projects generally use the software development model that is designed for rapidly changing environment.

Alfonso Fuggetta mentions [25] that "*rapid prototyping, incremental and evolutionary development, spiral lifecycle, rapid application development, and, recently, extreme programming and the agile software process can be equally applied to proprietary and open source software*".

However, we do not agree with Alfonso's assertion, as the process models he mentioned are exclusively framed for proprietary software and they do not consider the option for remotely connected developers as we have stated earlier.

In the context of project management in OSS paradigm, Gilberto Câamara and Frederico Fonseca [8] describe four types of OSS development model:

- High shared conceptualization, high modularity (the high-high case).
- High shared conceptualization, low modularity (the high-low case).
- Low shared conceptualization, high modularity (the low-high case).
- Low shared conceptualization, low modularity (the low-low case).

High shared conceptualization, high modularity means predefined project plan and large number of developers able to work concurrently. The Linux model is an example of high-high case.

High shared conceptualization, low modularity model generally have predefined project plan and little community efforts. The open source projects initiated by commercial company generally falls under this category.

Low shared conceptualization, high modularity model allows community developers to develop innovative software. There is no stable project plan and allows large number of developers.

Low shared conceptualization, low modularity model used by individual, small group or research projects. There is no predefined project plan and few opportunities for community developers.

[6] "SourceForge.net is a popular open source software development web site" [22].

6. PROPOSED DEVELOPMENT MODEL

In our proposed "Unified Software Development Model for Free/Open Source" (USDM) the project leader [26] initiates the project. His responsibility is to carry out initial requirement specifications and finalize it with the collaborations of the core developers. By core developers we mean a handful of developers who work under the direct supervision of the project leader.

The software process design will be based on the requirement specifications. Designing the core part of the system is the critical one, and it should be carried out by the project leader and the core developers. The core part facilitates the operations of the other modules. However, a common layer is to be designed by the core developers for providing interface between the core part and other modules. This eliminated the possible incompatibility due to version change.

After designing the core part of the system, this model needs to identify the distinct processes, and divide each process into small loosely coupled parts.

As in "The Cathedral & the Bazaar" Eric Steven Raymond suggests, "Good programmers know what to write. Great ones know what to rewrite (and reuse)" [6], we have considered possible reuse of software components in our proposed model. The project leader should at first search for similar open source projects, he/she intends to oat. If similar project is found, the limitations or missing features should be identified by comparing with the initial requirements. It will help to identify reusable components.

Having identification of distinct process along with reusable components, the model is required to create dummy data only for the dependent parts. It will enable the developers to work on any part at any moment even the dependent parts are not yet developed.

Documentation is an important part for any project and this model suggests writing a small documentation on each small parts. It will enable the developers to spend little time and efforts to understand and develop specific part.

In this model, a common information exchange system has been suggested for internal communication. It helps the use of dummy data and allows programmers to use any programming language and library for development.

In framing USDM we have followed the philosophy of UNIX [27] as stated below:

1. "Small is beautiful" - any small part is easy to build and easy to combine with other part and have advantages over large parts.

2. "Make each program do one thing well" - developers able to focus on a single problem. It eliminates extra code required for providing flexibility for multiple jobs.

3. "Choose portability over efficiency" - it will enable the reusability of the code in other parts of the project and even in other projects.

4. "Store numerical data in at ASCII files" - to change data using the common interface/unix pipe between various small parts.

Our proposed model has also similarities with "GNU Hurd Multi-Server Operating System". "The Hurd is a set of objects" [28]. A multi-server model where system services are made available exclusively through objects. It ensures the reusability and modularity by using a client server approach.

In essence, the steps for the USDM are as follows:

1. Requirements Specification
2. Core Design
3. Process Design
4. Dividing of each process into smaller loosely coupled parts
5. Incorporation any reusable components from similar or other Open Source projects
6. Creating dummy data for smellers parts
7. Writing small documentation on each small parts
8. Construction
9. Integration
10. Testing and debugging

7. EMPIRICAL APPLICATION OF THE PROPOSED MODEL

We have applied our propose software development model on two software projects:

7.1. Bengali Encoding Converter

Bengali (Bangla) is the national language of Bangladesh and one of the most popular languages of India [29]. There was no standardization of Bengali language writing system in computer. Several proprietary/non-standard encodings were being used[7] for writing in computer system, before the Unicode[8] standardization of Bengali [31] has become popular.

To validate our proposed model, we have initiated a project for converting documents written in Bengali non-standard encoding to the Unicode standard.

The traditional development model of any non-standard to unicode converter generally use "if else" (figure 1) block structure. As a result, developers have to rewrite the entire "if else" block for developing similar converter for other vendors. In this system, any modification requires altering the source code of the software. This requires knowledge on the specific programming language as well as the design of the software.

[7]Even, today some organizations use non-standard encoding for the legacy software and availability of typist.

[8]"The Unicode Consortium is a non-profit organization founded to develop, extend and promote use of the Unicode Standard, which specifies the representation of text in modern software products and standards." [30]

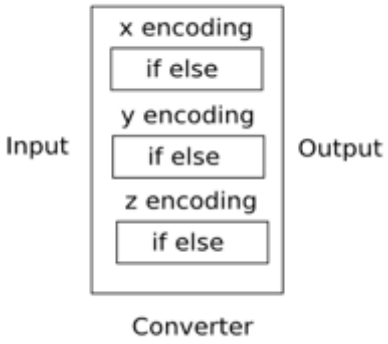

Figure 1: Traditional Method

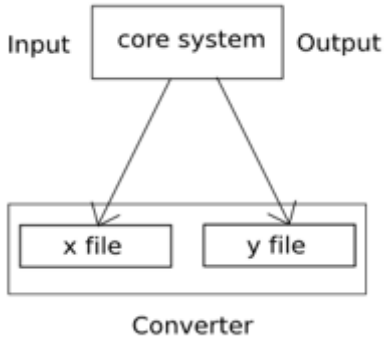

Figure 2: Unified Method

By following our proposed software development model, we have developed a core system (figure 2) and a standard common information exchange file schema. It allows both the developers and users to modify or add any new non-standard encoding by using common information exchange file. The core system read the encoding specific file to understand what it needs to convert.

7.2. Bengali/Bangla OCR Pre-processor

In the Bengali language written and printed system, alphabets are consecutively connected with a line atop called "matra" in most of the word. So, operation of an Optical Character Recognition (OCR) on Bengali language is different from other languages [29].

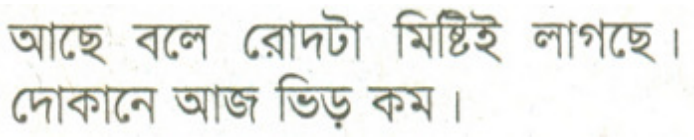

Figure 3: Scan Image

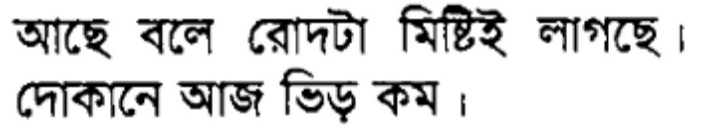

Figure 4: Conversion of scan image to black and white

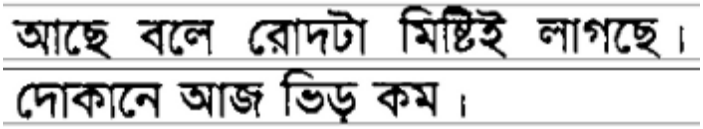

Figure 5: Line detection

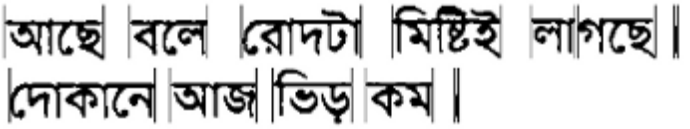

Figure 6: Word detection

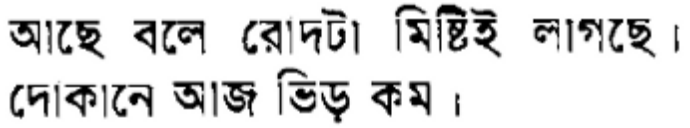

Figure 7: Alphabet detection

We have developed a Bengali/Bangla OCR pre-processor. An image processor for separating connected alphabets and preparing it for OCR system (figure 3-7). Our proposed development model has been applied in the development of Bengali/Bangla OCR pre-processor.

Identified processes for this Bangla pre-processor are:

Task A: Conversion of scan image to black and white.

Task B: Line detection.

Task C: Word detection.

Task D: Alphabet detection.

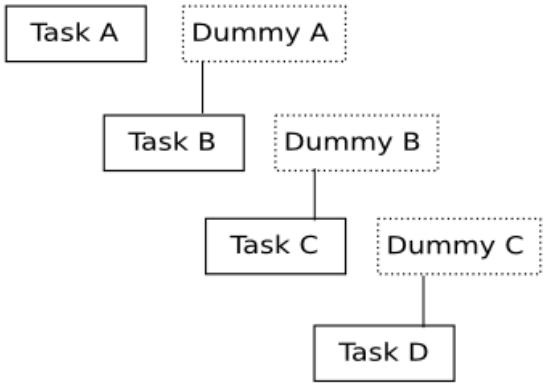

Figure 8: Development Phase

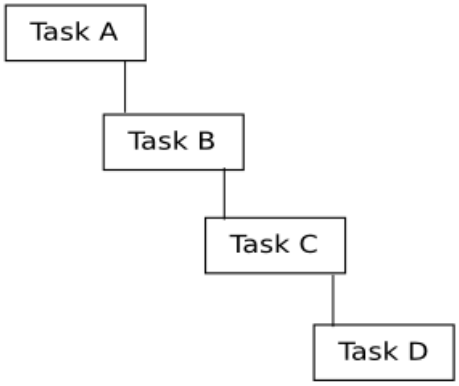

Figure 9: Testing Phase

This model allows developers to develop or improve any part of the system at anytime. Bug tracking is easy in this model. Information passes via a common information exchange file (figure 8-9). It enables the developers to use any preferable programming language or library for development.

8. CONCLUSIONS

Our proposed software development model is exclusively designed for “Open Source” and “Free Software” development with remotely connected volunteers. The project leader's role is very important in this model. The project leader and core developers together manage the project and perform initial tasks. This model suggests low shared conceptual and high modularity (low-high case). It enables the development of any software project regardless of the community size, using less developers time and minimizing wastage. We have validated our project by initiating an empirical project as we mentioned in the paper.

REFERENCES

[1] J. Feller, B. Fitzgerald, S. A. Hissam, and K. R. Lakhani, *Perspectives on Free and Open Source Software*. The MIT Press, 2007.

[2] S. K. Sowe, *Emerging Free and Open Source Software Practices*, 1st ed. IGI Publishing, 2007.

[3] K. Schwalbe, *Information Technology Project Management (with Microsoft Project 2007 CD-ROM)*, 6th ed. Course Technology, 2009.

[4] J. Feller and B. Fitzgerald, *Understanding Open Source Software Development*. Addison-Wesley Professional, 2001.

[5] B. Golden, *Succeeding with open source*. Addison-Wesley Professional, 2005.

[6] E. S. Raymond, "The cathedral and the bazaar [Linux operating system]," *First Monday*, vol. 3, no. 3, Mar. 1998.

[7] F. P. Deek and J. A. M. McHugh, *Open Source: Technology and Policy*. Cambridge University Press, 2007.

[8] Gilberto Caˆmara and Frederico Fonseca, "Information Policies and Open Source Software inDeveloping Countries," *JOURNAL OF THE AMERICAN SOCIETY FOR INFORMATION SCIENCE AND TECHNOLOGY*, vol. 58(1), pp. 121-132, 2007.

[9] "Install Manifests - MDC Docs." [Online]. Available: https://developer.mozilla.org/en/Install_Manifests. [Accessed: 20-Jun-2011].

[10] "Apple Launches iPad." [Online]. Available: http://www.apple.com/pr/library/2010/01/27ipad.html. [Accessed: 20-Jun-2011].

[11] "Free Software Foundation." .

[12] "The Open Source Definition | Open Source Initiative." [Online]. Available: http://opensource.org/docs/osd. [Accessed: 20-Jun-2011].

[13] "GNU Operating System." .

[14] J. Gay and R. M. Stallman, *Free Software, Free Society: Selected Essays of Richard M. Stallman*. CreateSpace, 2009.

[15] L. Rosen, *Open Source Licensing: Software Freedom and Intellectual Property Law*, 1st ed. Prentice Hall, 2004.

[16] P. Aksoy and L. DeNardis, *Information Technology in Theory*, 1st ed. Course Technology, 2007.

[17] Greg Kroah-Hartman, SuSE Labs / Novell Inc., Jonathan Corbet, LWN.net, and Amanda McPherson, The Linux Foundation, "Linux Kernel Development," *The Linux Foundation*.

[18] L. Kaufman and M. Welsh, *Running LINUX*, 2nd ed. O'Reilly & Associates, 1996.

[19] K. Fogel, *Producing Open Source Software: How to Run a Successful Free Software Project*, 1st ed. O'Reilly Media, 2005.

[20] "Sahana Software Foundation." [Online]. Available: http://sahanafoundation.org/. [Accessed: 20-Jun-2011].

[21] M. Careem, C. De Silva, R. De Silva, L. Raschid, and S. Weerawarana, "Sahana: Overview of a disaster management system," in *2nd International Conference on Information and Automation, ICIA 2006, December 14, 2006 - December 17, 2006*, Colombo, Sri lanka, 2006, pp. 361-366.

[22] "About." [Online]. Available: http://sourceforge.net/about. [Accessed: 20-Jun-2011].

[23] R. Pressman, *Software Engineering: A Practitioner's Approach*, 7th ed. McGraw-Hill Science/Engineering/Math, 2009.

[24] S. F. Ochoa and G.-C. Roman, *Advanced Software Engineering: Expanding the Frontiers of Software Technology: IFIP 19th World Computer Congress, First International Workshop on ... in Information and Communication Technology)*, 1st ed. Springer, 2006.

[25] A. Fuggetta, "Open source software-an evaluation," *Journal of Systems and Software*, vol. 66, no. 1, pp. 77-90, Apr. 2003.

[26] S. Koch, *Free/Open Source Software Development*. Idea Group Publishing, 2004.

[27] M. Gancarz, *The UNIX philosophy*. Elsevier, 1995.

[28] N. H. Walfield and M. Brinkmann, "A critique of the GNU hurd multi-server operating system," *SIGOPS Oper. Syst. Rev.*, vol. 41, no. 4, pp. 30–39, Jul. 2007.

[29] S. Sural and P. K. Das, "An MLP using Hough transform based fuzzy feature extraction for Bengali script recognition," *Pattern Recognition Letters*, vol. 20, no. 8, pp. 771-782, Aug. 1999.

[30] "What is Unicode?"

[31] "The Unicode Standard, Version 6.0," in *South Asian Scripts-I*, 1991.

Authors

1. Md. Anawarul Kabir is working as Assistant Professor of Computer Science at American International University- Bangladesh. He has done his undergraduate from the University of Dhaka and completed MS in Computer Science from the University of Wales, Swansea. His research interests are Software Requirement Analysis and Reengineering, Software Process Model, Systems Analysis and Design, Open Source Software, Expert Systems, Data mining. He is also a Member of the British Computer Society. Apart from teaching, he is also a columnist of the Daily Star and the Daily Sun, two leading dailies of Bangladesh.

2. Md. Salahuddin Pasha is working as Software Engg. and Network admin in Aamra Group of Companies in Bangladesh. He is involved in var ious Open Source software development, and core member of local OSS groups BDOSN (Bangladesh Open Source Network), Ankur. He has completed is BSc in Computer Science and Engineering from American International University Bangladesh (AIUB) in 2010. He is also involved with R&D projects for industry, and interested in Open Source related research.

3. Mohammad Abdur Razzak is doing Msc in Software Engineering at Blekinge Institute of Technology, Sweden. He has completed Bsc in Computer Science and Engineering from American International University Bangladesh (AIUB) in 2010. His current research interest is Open Source requirement engineering, Domain Specific Language (DSL) and Wireless Sensor Network (WSN).