An Integrated Software Development Method with built-in Software Process Improvement Framework

Partha Protim Roy, Hasan Sarwar, Thoukir Hasan Ha-Mim

Abstract*: The concept of a startup is growing around the world where innovation is the major driving force. Managing such a small startup company from both technical and administrative perspective is a challenging issue. Though several approaches for software development, like the waterfall model, agile model, SCRUM model are available, it is difficult to implement such a model with full features due to limited number of resources, lack of experienced resources and lack of enough financial capability. The scopes of software process improvement are even narrower in a situation where they hardly manage or adapt a methodology. As a result, small companies suffer and work without a good quality framework model. Companies struggle to produce quality software. In this work, we have suggested an integrated software engineering methodology, in which, the process improvement framework is a built-in phenomenon. Our system is dubbed as 3D model, where the Ds stand for Design, Develop and Deploy. Design indicates the design of the product making as well as the process design. Develop and Deploy is also hinting to the product and process. The process improvement is managed through observation and management, which are fed back into the system through proper documentation. We did some client surveys, organization surveys, and observed work patterns of a group of human resources for a period. We measured some of the parameters before and after the implementation of 3D model. We found that implementation of 3D model gives an improved process and product development environment with no extra manpower or extra costing.

***Keywords*: Software Engineering Methodology, Software Process Improvement, 3D Model.**

I. INTRODUCTION

The concept of a startup is growing around the world. Innovation is the major driving force for emerging economies and more startups are coming with new innovations. Innovative ideas are generating new companies. As a result, many new small software companies around the globe are emerging. Recent statistics suggest small companies account for 93% in Europe, 56% in the US, 66% globally [1]. These companies employ around 15-30 resources. In spite of the fact of being small, many of the companies are developing large ERPs for their clients. They are also providing support services for those clients. Clients always expect the best service while the companies struggle to provide the best service as the system is sold at a cheaper price and not possible to maintain different specialized departments by the software companies. Many of the times, a single resource acts as a developer or requirement analyst or even a tester or trainer. In such an overlapping job responsibility, maintaining a strict workflow like Analysis, Design, Development, Testing, and deployment for a large project is complicated. This situation gets worsened while a lot of change requirements are generated even before the real use of the software.

There are certain methodologies that can be used for software development purposes, known as the waterfall model, agile model, and SCRUM model. Small companies try to follow such models or a combination of these models. Every model has its own specifications. Waterfall demands that the client requirement should be collected fully before the design process begins. Small companies are not always capable to extract real requirements from clients due to their inexperience in system building or inexperienced resource persons. Especially in a culture where clients are not aware of the importance of the requirement collection process and always explain their requirement after the product is deployed. Agile models demand serious domain knowledge from the team leader. Since in agile models, developers collect requirements in iterative style and develop an iterative partial version of the product, lack of domain experience from analysts' part may guide the project in a wrong direction. Moreover, such situation arises at such a stage of the project, where starting from scratch is not at all possible.

There are also some frameworks known as the Software Process Improvement framework. These frameworks are specially meant to initiate a quality practice in any kind of software development methodology. Of them, CMMI, ISO, and 6 sigma are commonly known and widely practiced industry standards. Implementation of these quality frameworks requires certain specifications of the company format and development process. Small companies cannot afford such process standards as these require the implementation of certain engineering practices. As a result, small companies suffer without a good quality framework model. In such a scenario, companies are struggling to produce quality software with minimal cost and effort.

In most of the works of SPI for large or small industries, SPI is an added process on top of the regular engineering methodology.

Revised Manuscript Received on January 2, 2020.

***** Correspondence Author

Partha Protim Roy, Department of Computer Science and Engineering, United International University, Dhaka, Bangladesh. Email: pproy.cse@gmail.com

Hasan Sarwar*, Department of Computer Science and Engineering, United International University, Dhaka, Bangladesh. Email: hsarwar@cse.uiu.ac.bd

Thoukir Hasan Ha-Mim, Edusoft Consultants Ltd, Dhaka, Bangladesh. Email: thoukir@gmail.com

One approach suggests improving the global level of software practices by introducing some high-level goals and operational goals [2-3]. In another work, SPI has been tried by aligning the software processes with reference models like CMMI-DEV and ISO/IEC 15504 [4]. The software process matrix has been developed along with some practices [5]. MESOPYME is another model that tries to create a method to assess and improve software processes [6]. PRISM is another model for implementing process improvement. Here, a separate team is employed that finds plans to improve the current process and comes up with a revised process model. All these efforts indicate a common pattern of process improvement approach. The pattern is that the process itself is in its place. Process improvement is a separate approach where surveys are conducted, and teams come up with an improvement plan in the process.

II. RESEARCH METHODOLOGY

As an exploratory survey, we conducted some surveys on a few micro-companies. We carried the survey in three main parts. In the first part, we took a detailed interview of software companies and about their processes. In the second part, we collected some clients' feedback about the software company and its services. Finally, in the third part, we analyzed and planned a work process at a micro software company. Survey briefs are given below.

A. Company Survey

There were questions in 9 different categories. Survey questions were related to employee information, requirement analysis, project design, development process, deployment, testing, service, support, and client.

B. Client Survey

There were 21 questions in 6 categories. The categories were requirements, product assessment, communication, support, software training, and general assessment. Questions were more targeted to get client feedback on the above issues.

C. Developer Inspection

A questionnaire was created that was circulated among the developers. The questionnaire was given each day on a regular basis for a certain time period. Developers used to fill up the form at the end of each day. The filling pattern, the data entered into the form suggests the work pattern of the employees.

III. AN INTEGRATED PROCESS MODEL

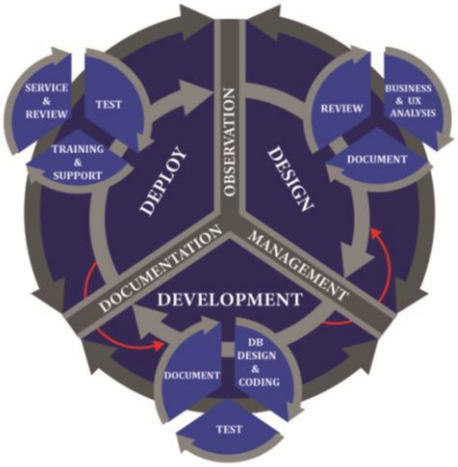

Fig. 1. A Software Life Cycle with Integrated SPI

We have come up with an integrated approach. The software engineering methodology has been re-engineered in a way that encompasses the process improvement framework and product development framework in one structure as shown in Fig.1. Against the usual practice of finding the scopes where the process can be improved by a separate set of activities, the following diagram shows ways where the process itself is insisted to get improved by following a secondary circle. Basically, the main cycle of software development is presented as DESIGN, DEVELOP, DEPLOY in these 3 stages.

Secondary circles for Design are Business & UX Analysis, Document, and Review. Now while Design is concerned, we will focus on the Design of the process as well. And the same secondary cycle will work here for process improvement. Especially the part of Review and

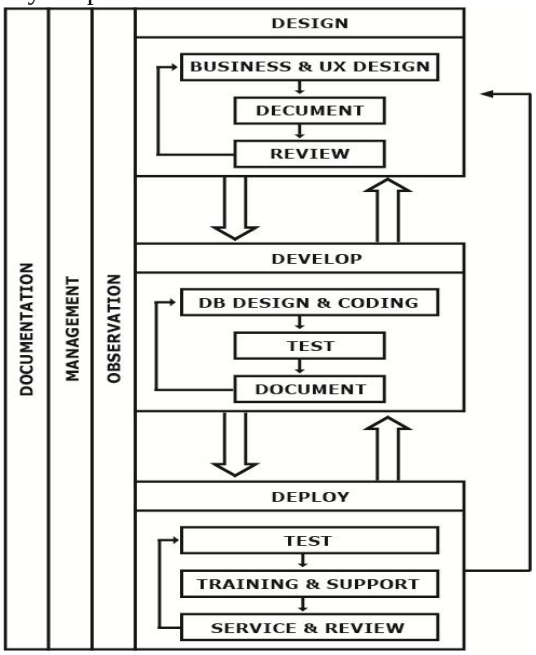

Fig. 2. Block Diagram of the Integrated Software Model

document corresponds to the process and the process improvement together.

The next stage is Develop, which has the corresponding secondary cycle of DB Design & Coding, Test, and Document. Here especially Testing and Document checking can be used as part of process improvement as well. The third D is Deploy, As the cycle is concerned, deploy stage is used to transfer the program to the user. The secondary circle may be used for performance improvement. It is not essential that all the secondary client-side is to be followed with equal importance in the 3 stages. The level of emphasis will be determined by another circular (outer) top-view operation consists of 3 activities, which are Observation, Management, and Documentation. Apart from the developers and support personnel of them, a team of 1 or more members is appointed as a monitor of a project. Monitor's job responsibility is to observe the work progress, find out the bottlenecks, discuss with the management of both the developer organization and the client organization and improve or emphasize the processes mentioned at the secondary circles.

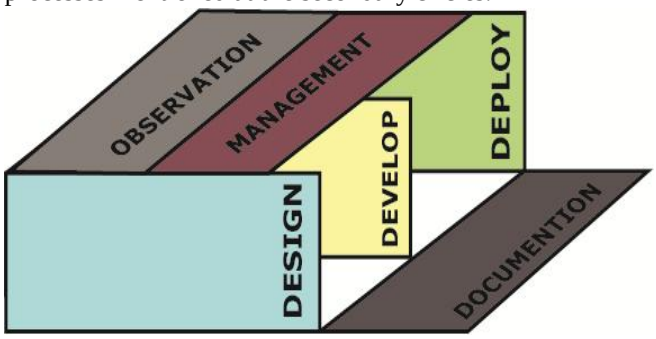

Fig. 3. A block diagram depicting the relations of Development and Process Improvement Life Cycle

Figure 2 shows the block diagram of the Software life cycle phases along with the SPI operations. Observation, management, and documentation are the 3 operations that are viewing the complete development cycle. This phenomenon is better understood in Figure 3. Observation is done as a top view for DESIGN, DEVELOP, and DEPLOY stages. The parameters that are observed are eventually the level of engagement of the developers, quality of the produced software, or client's reaction to using the product. Based on the observed value, the Management changes the work pattern of the 3 phases, do the required documentation and provide intermittent training or counseling to the respective teams for better outputs.

IV. CASE STUDY

We have used this approach in a project. Before the application of this approach, based on the survey mentioned above and the collection of some more data we found out the following values that were most important in describing the process.

Table- I: Improvement in some parameter after the use of the 3D Model

Parameter	The value found before the application of the approach	The value found after the application of the approach
Developer Satisfaction	0.46	0.63
Rework Level	20 (approx.)	5 (approx.)
Client Satisfaction	0.60	0.80
Company's process maturity	0.20	0.40

Table 1. shows improvement of some parameters after the use of the 3D Model

As an exploratory survey, we went through some surveys on a few micro-companies. We decided to conduct a survey in three main parts. In the first part, we managed a detailed interview of software companies and their processes. In the second part, we collected some clients' feedback about a software company and its services. Finally, in the third part, we analyzed and planned a work process at a micro software company. Survey briefs are given below.

A. Developer Satisfaction

The survey consisted of a few questions that indicate the comfort zone of an engineer involved in the development process. For example, if the requirement specification was not documented, the developer took more time to fix his work resulting wastage of time that generated a lower satisfaction level. Lack of cooperation and understanding between the design and development team and the process definition for the same was improved. The developer is now firmer about the requirement and less time is wasted due to confusion or indecision. In the table I, we can see that the implementation of the 3D approach improved the developer satisfaction level from 0.46 to 0.63.

B. Rework Level

This parameter defines the number of reworks. Here, Rework is defined as a number which indicates how many times a single task is sent back to the developers for further fixing of bugs or change requirement. As a part of the implementation of the above approach, the observation was made to find the root cause and subsequently, the design of the solution was made again and then implemented again. In order to improve the process, the management decision was to include the consumer team as a direct stakeholder of the development team. Finally, it reduced the rework level as shown in the table. There were 21 questions in 6 categories. The categories were requirements, product assessment, communication, support, software training, and general assessment. Questions were more targeted to get client feedback on the above issues.

C. Client Satisfaction

Client satisfaction is also measured by asking a few questionnaires. Clients were initially disturbed for repetitive requirement specification due to poor data collection approach. Besides, the lack of proper review and coordination generated a certain level of mistrust among the customers. Unorganized support maintenance service created irritation among the clients resulting in dissatisfaction. Implementation of the 3D approach made life easier.

D. Process Maturity

Here process maturity refers to the knowledge state among the team members.

Knowledge is categorically shared among the developers, designers and deploy representatives in a collaborative and coordinated way. Earlier, before the implementation of the 3D model, collaboration and cooperation among the team members of both the developer side and client-side was a nightmare. However, proper knowledge sharing and a respectable atmosphere reduced the gap of collaboration and cooperation in terms of knowledge transfer. Having developed proper knowledge before doing any sorts of activities ensures appropriate output from that part.

E. Observation, Management, and Documentation

Initially, the project was built in a short time constraint and every module and component was developed on an ad hoc basis. The project was almost on the verge of failure. The top management of the client side sat several times with the software service provider and the project was almost on the verge of collapse. The project at that moment was on LIVE and there were lots of inconsistencies in data and operation. The developers were asked to fix all the errors that were popping up and the level of untrust on service providers, especially to handle and level the situation, was growing. In such a scenario, we opted to apply the 3D Model to come out of this situation. We took refuge using the observation phase; a specific role was created as monitor cum manager. The monitor's job was to review the Design, Develop, Deploy process. There was another role created as business monitor (works at the ground level). Monitors found that that issues raised by the clients are repeating again and again even after fixing anomalies. The monitor verified the code, the business flow and found that some serious issues remain with the implementation of business flow. The 2 types of monitors were as below:

Business Monitor: This monitor observed all the bugs in the business process, analyzed the issues and designed perfect solution for implementation. The solution was either to change the business flow or to change the code according to the business flow.

Development Monitor: Inspects the code and logic and modified the code by the developer to inhibit future errors.

V. DISCUSSION AND CONCLUSION

We implemented some of the features of the 3D model into the case study. Still, we experienced some improvement in the output. It is understood that a team working in an unstructured way will find it hard to adopt a new approach to a systematic process. So, our goal was not to put a rigid order in the development process. The team we worked with was not used to follow a very formal process of development. We also did not put them in pressure to follow the 3D model. Rather the implementation of this model was very flexible.

In comparison to the different approaches of the software engineering life cycle, our 3D model reduces the number of steps of the software development life cycle. Compared to 5 steps of Waterfall model – Requirement, Design, Construction, Testing, Release; 4 phases of RUP approach – Inception, Elaboration, Construction, Transition; or 4 segments of Spiral model – Objectives, Risk Resolution, Develop, Plan; our model has only 3 steps, namely DESIGN, DEVELOP, and DEPLOY.

As a result, many time-consuming actions have been reduced and from specification to user acceptance, time spent was also reduced by a significant amount.

A comparison with Waterfall, Spiral, and Incremental Model is shown in table II. The initial comparison is shown in [7].

Another comparison with some SPI models is shown in table III below.

Table- III: A Comparative Study with CMMI and other SPI approaches.

Model	Characteristics
CMMI	A complex model works in parallel with the process model to improve Process
Other SPI approaches	Assessment setup is done, then SPI planning is done, afterward SPI implementation is done [8].
3D approach	SPI is not a separate activity. It is integrated with the Process Model and continuously adjusting the development process.

The 3D methodology is really effective, while we used the 3 functions of the outer circle to be simultaneously implemented, namely, Observation, Management, and Documentation shown in Fig.4. The observation function was focused to observe the overall system with a critical analysis and continuous suggestions to the middle management or monitor to introduce changes in various activities of the 3D process. It is not mandatory that every use case of the development will follow strict cycle times of all the primary and secondary circles. It is the management decision whether for each scenario we should go through the steps of all primary and secondary cycles. The management, by observation, can dynamically decide to skip some secondary cycles while puts

emphasis on other secondary cycles. In this way, the total cost of implementation of the project is reduced. The practice here was to introduce change management based on the observation. Finally, a documentation of the whole process is

emphasis on other secondary cycles. In this way, the total cost of implementation of the project is reduced. The practice here was to introduce change management based on the observation. Finally, a documentation of the whole process is maintained for future reference.

Table- II: A comparative study of the 3D model with Waterfall, Spiral, Incremental Approach

Model/ Feature	Waterfall	Spiral	Incremental/ Iterative	3D
Specification of All the Requirements in the beginning	Yes	Not all and Frequently Changed	Not all and Frequently Changed	Not all and Frequently Changed
Long term project	Inappropriate	Appropriate	Appropriate	Appropriate
Complex Project	Inappropriate	Appropriate	Appropriate	Appropriate
Frequently Changed Requirements	Inappropriate	Appropriate	Appropriate	Appropriate
Cost	Not costly	Costly	Costly	Not costly
Cost estimation	Easy to estimate	Difficult	Difficult	Medium
Flexibility	Not	Less flexible	Flexible	Flexible
Simplicity	Simple	Intermediate	Intermediate	Simple
Supporting high risk projects	Inappropriate	Appropriate	Appropriate	Appropriate
Guarantee of Success	Less	High	High	High
Customer Involvement	Low	Low, After Each Iteration	High, After Each Iteration	High, After Each Iteration
Testing	Late	At the end of each phase	After every Iteration	After every Iteration
Maintenance	Least maintainable	Yes	Maintainable	Maintainable
Ease of Implementation	Easy	Complex	Easy	Easy

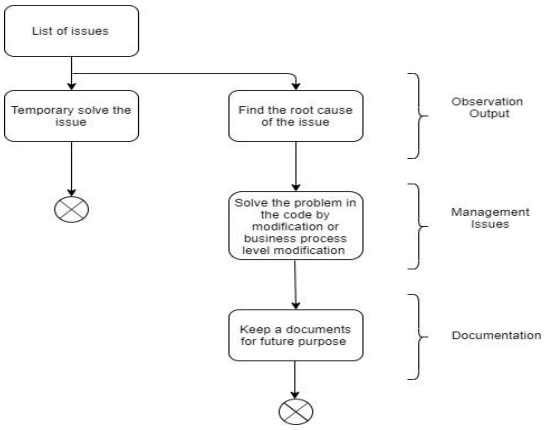

Fig 4 Use of Observation, Management, and Documentation to improve the product

VI. FUTURE WORK

A more detailed analysis of this model will be done considering more observed parameters from the developer side, client-side and management side.

REFERENCES

1. Laporte, C.Y., O'Connor, R.V., Fanmuy, G., "International Systems and Software Engineering Standards for Very Small Entities," CrossTalk - The Journal of Defense Software Engineering Vol. 26(3), 2013, pp. 28–33.
2. N. Habra, A. Renault, S. Alexandre, and M. Lopez, "OWPL Micro Assessment," in Software Quality Workshop, 24rd International conference on Software Engineering ICSE, Orlando, Florida USA, 2002.
3. Alexandre, S., Renault, A., and Habra, N., "OWPL: A Gradual Approach for Software Process Improvement in SMEs," Proceedings of the 32nd EUROMICRO Conference on Software Engineering and Advanced Applications (EUROMICRO- SEAA 2006), 2006, pp. 328–335.
4. Jean Carlo R. Hauck, Christiane Gresse von Wangenheim, Richard H. de Souza1, and Marcello Thiry, "Process Reference Guides – Support for Improving Software Processes in Alignment with Reference Models and Standards," Springer. EuroSPI, CCIS, vol. 16, Heidelberg, 2008, pp. 70–81.
5. Ita Richardson, "Software Process Matrix: A Small Company SPI Model," Software Process: Improvement and Practice, Vol. 6, September 2001, pp. 157–165.
6. Calvo-Manzano, J.A., Cervera, J., and San Feliu, T., "Software process improvement: MESOPYME model," Journal of Computing and Information Technology (CIT), vol. 43: 159-165, 1997.
7. Alshamrani A and Bahattab A, "A comparison between three SDLC models waterfall model, spiral model, and Incremental/Iterative model" International Journal of Computer Science Issues (IJCSI). 12, 106, 2015
8. Deepti Mishra and Alok Mishra, "Software process improvement in SMEs: A comparative view" Computer Science and Information Systems · June (2009), 658.5:004.4, 10.2298/CSIS0901111M, 2009

AUTHORS PROFILE

Partha Protim Roy is a Master's student in the Department of Computer Science and Engineering at United International University (UIU). He graduated in Computer Science and Engineering from Ahsanullah University of Science & Technology (AUST). In addition, Mr. Roy is a passionate entrepreneur and the Managing Director of Advance Computer Expertise IT (A.C.E.), a consulting and web development company in Dhaka. He has a strong background in researching Software engineering and Rabindranath Tagore song, Classical Music and Music Direction.

Hasan Sarwar is a Professor and former Head of the Department of Computer Science and Engineering of United International University (UIU). Mr. Sarwar holds a Ph.D. in Applied Physics, Electronic and Communication Engineering from the University of Dhaka. He graduated from the Computer Science and Engineering Department of Bangladesh University of Engineering and Technology (BUET). In addition, Mr. Sarwar is an entrepreneur and Director of Edusoft Consultants Ltd., a consulting and software development company in Dhaka. He has a strong background in researching the Bangla OCR, Software engineering, Telemedicine, Education Management, Distance Learning, and Plasmonics.

Md. Thoukir Hasan Ha-Mim is a Solution Architect of Edusoft Consultants Ltd. He completed the Master of Science in Computer Science and Engineering (MSCSE) from United International University (UIU). He graduated from the Department of Business Administration of Northern University Bangladesh (NUB). He has a strong background in system analysis and web-based software design & development of Business Administration of Northern University Bangladesh (NUB). He has a strong background in system analysis and web-based software design & development.