Square Model- A Software Process Model for IVR software System

Devesh Kumar Srivastava
Department of CSE
Uttrakhand Technical University
Dehradun India

Durg Singh Chauhan
Vice Chancellor
Uttrakhand Technical University
Dehradun India

Raghuraj Singh
Professor, Department of CSE
Hercourt Bulter Technological Institute
Kanpur India

ABSTRACT

A process is not a static entity. Improving the quality and reducing the cost of products are fundamental goals of software engineering discipline. As the qualities are determined by the process to satisfy the objectives of quality improvement and cost reduction, the software process must be improved. Nowadays it has been widely accepted that the quality of software highly depends on the process that is carried out in an organization. As part of the effort to support software process engineering activities, the research on software process modeling is to provide an effective means to represent and analyze a process and, by doing so, to enhance the understanding of the modeled process. Thus the enforcement of the process model can directly contribute to the improvement of the software quality. The process must be improved on the basis of past experience on software projects. The activity of analyzing and improving the process is largely done in process development component of the software process. From time to time so many software process models has been developed in order to maintain reliability and quality of software .Our research work focus on a new proposed software process model for BPO software application which is based on BPO software industries. Such software are based on Gensys technology which interact with Interactive Voice Response (IVR), Computer Telephony Integration (CTI), Call Recording, Call Monitoring, Call Center Analytics / Performance Management, Workforce/Agent Management and others.

Keywords

Business process outsourcing (BPO), Software Process Improvement (SPI), Verification & Validation (V & V).

1. INTRODUCTION

The software process is a major factor for delivering quality software systems. It aims to manage and transform the user need into a software product that meets these needs. Software process means the set of activities required to produce a software system, executed by a team of people organized according to a given organizational structure. The ultimate goal of research into the software process modeling is to improve software development process through a better ways of organizational design at the level of individual processes and the organization as a whole accompanying innovations in technological support. Software process modeling describes the creation of software development process models. A software process model is an abstract representation of the architecture, design or definition of the software process [14]. Each representation describes, at different detail levels, an organization of the elements of a finished, ongoing or proposed process and it provides a definition of the process to be used for evaluation and improvement. A process model can be analyzed, validated and simulated, if executable. The process models are used mainly for software process control (evaluation and improvement) in an organization, but they can be also used for experimenting with software process theory and to ease process automation. Further there is a question how to evaluate the software process. So software processes evaluations are improved by means of which to judge and decide on the quality of the object under analysis that is, the software process of a given organization propose a process improvement strategy. The efforts of the scientific community in this field have led to quite a number of maturity models and standards, such as ISO 9000 [8], CMM (Capability Maturity Model). All these models have two goals .First to determine the aspects for improvement in a software development organization and second to reach an agreement of what a good process is. As use of software is increasing in business, industry and services due to its ability to improve the operational and managerial efficiency of conducting various activities. Good software is known to be one which satisfies the customer requirement .A software is declared to be good by the customers if it possesses certain attributes. It should be bug free, reusable, easy to maintain and user friendly. To define the process model we need to define process definition language. Because of the complexity of the software process, various process definition languages (PDL), process modeling tools (PML), and workflow representation language have been created to represent and formalize software processes and their related activities, roles, artifacts, and tools. [4]. Main purposes of these languages are process understanding, process design, training and education, process simulation and optimization, and process support [4]. According to [11], a software process model is a framework that enables specific software processes to be defined. The structure, standards and relationships of the process elements are established within the framework. Therefore a software process defines the approach that is taken as software is engineered .Another reason why it is important is that a process provides organizational stability and more control to its activity. However, there are some common misconceptions in the industry about process. It is assumed that a process interferes with creativity, creates bureaucracy, useful only for large projects and implementing a process costs high. Although everyone in the software industry understands the importance of having motivated, quality workforce and the latest technology. Few software companies follow capability maturity model and have the maturity level of CMM level 5. Our research work shows that effort put into these models and standards can assist to produce high quality software, reduce cost and time and increase productivity. However, little attention has been paid to the effective implementation of these models and standards [6] which has resulted in limited success for many Software Process Improvement (SPI) efforts. Studies show that 67% of SPI managers want guidance on how to implement SPI activities, rather than what SPI activities to actually implement [7].In this paper we describe a new process model as collaborative framework that supports software process improvement based on BPO software application. The remainder of this paper is structured as follows: Section 2 explains the related work, Section 3 explains the collaboration framework proposed for process assets reuse in the project under development and finally in section 4 represent the conclusions and future works.

2. Related Work

The purpose of this paper is to propose the most accurate architecture for the collaborative framework. Therefore the authors reviewed some tools and prototypes to find out the critical gaps in the software tools that software engineers use to manage software process improvement programs and projects execution. Today there are three main actions that can be taken with respect to the software process: define, evaluate and improve the process model. The definition of the software process refers to the definition of the processes as models, plus any optional automated support available for modeling and for executing the models during the software process. In the software development process the author focuses on the activities directly related to production of the software, for example analysis and planning, designing, coding, testing and deployment. Due to the importance of the development process, various models have been proposed .These models are waterfall model, prototyping, Iterative development, spiral model, Time boxing model and few more.

Software Life Cycle*:* A software life cycle model is either a descriptive or prescriptive characterization of how software is or should be developed. A descriptive model describes the history of how a particular software system was developed. Descriptive models may be used as the basis for understanding and improving software development processes or for building empirically grounded prescriptive models (Curtis, Krasner, Iscoe, 1988). A prescriptive model prescribes how a new software system should be developed. Prescriptive models are used as guidelines or frameworks to organize and structure how software development activities should be performed, and in what order. The classic software life cycle is often represented as simple prescriptive waterfall model, where software evolution proceeds through an orderly sequence of transitions from one phase to the next in order. See figure -1

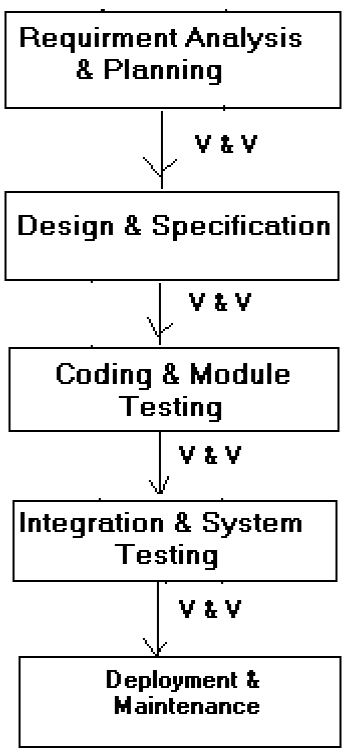

Figure-1

2.1 The Stepwise Refinement

In this approach, software systems are developed through the progressive refinement and enhancement of high-level system specifications into source code components (Wirth 1971, Mili 1986). This model has been most effective and widely applied in helping to teach individual programmers how to organize their software development work.

2.2 Incremental Development

Developing systems through incremental release requires first providing essential operating functions, then providing system users with improved and more capable versions of a system at regular intervals (Basili 1975). This model combines the classic software life cycle with iterative enhancement at the level of system development organization.

2.3 Joint Application Development (JAD)

It is a technique for engaging a group or team of software developers, testers, customers, and prospective end-users in a collaborative requirements and prototyping effort (Wood and Silver 1995). JAD is quintessentially a technique for facilitating group interaction and collaboration.

2.4 Rapid Application Development Model (RAD)

This model is proposed when requirement and solutions can be modularized as independent system or software components, each of which can be developed by different team. Then after it smaller component are developed, they are integrated to produce the larger software system.

3. Methodology

The author has been reviewed several software process model which are currently running in software industry, its advantage and disadvantage. On the basis of BPO software application, he has proposed a new software process model which will be useful for developing the software application based on BPO software industry. The call is made either inbound or outbound through software application by either agent or customer through lucent / noble dialer or gensys technology. This model is named Square model.

3.1 Proposed Model: The square model proposed by author “Devesh” is a software process model combining elements of both designing and prototyping and iteration in-stages. This model is a combination of the features of the waterfall model, prototyping model, iterative model and spiral model. This proposed model covers risk factors, which seeks to identify situations that might cause a development effort to fail or go over budget/schedule, occurs during each cycle. The angular dimension of this model denotes progress made in accomplishing each development. During development the model has to be simulated and benchmarked to get the progress. See Figure-2

This model is very specific for BPO / calls center software application and should be developed in dedicated environment. The steps in this model can be generalized as follows:

1. The requirements are defined in document as much detail as possible.
2. A preliminary design is created for the new system. In this phase all possible and available alternatives can help in developing a call center effective software project. This phase has been added specially in order to identify and resolve all the possible risks in the project development. If risks indicate any kind of uncertainty in requirements, prototyping may be used to proceed with the available data and find out possible solution in order to deal with the potential changes in the requirements.
3. A first protype of the new system is constructed from the preliminary design. This is usually a scaled-down system, and represents an approximation of the final product.
4. A second prototype is evolved by a fourfold procedure:
    1. evaluating the first prototype in terms of its strengths, weaknesses, and risks.
    2. defining the requirements of the second prototype.
    3. planning and designing the second prototype.
    4. constructing and testing the second prototype and so on.
5. After passing rigourously the various testing, the software has to be implemented and deployed.
6. The final software is used for calling through lucent / noble dialer or through gensys tools to test, verification the inbound call and outbound call in call monitoring phase.See the figure of proposed model.

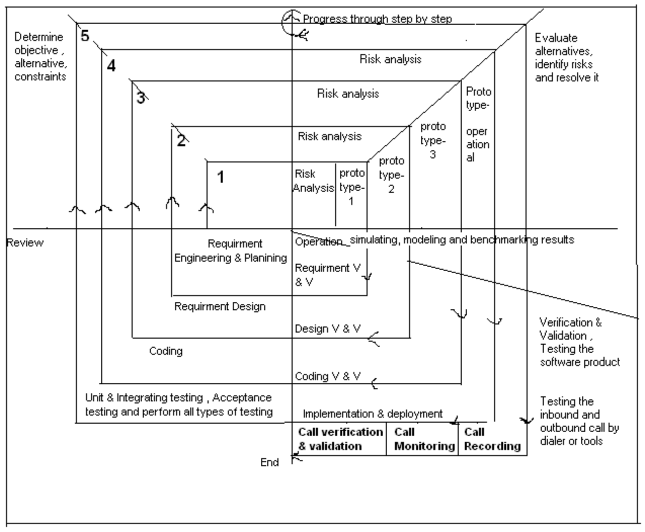
Progress through step by step
Determine objective , alternative, constraints
Evaluate alternatives, identify risks and resolve it
5
4
3
2
1
Risk analysis
Risk analysis
Risk analysis
Risk Analysis
proto type-1
proto type-2
proto type-3
Proto type-operational
Review
Requirment Engineering & Planining
Operation simulating, modeling and benchmarking results
Requirment V & V
Requirment Design
Design V & V
Coding
Coding V & V
Verification & Validation , Testing the software product
Unit & Integrating testing , Acceptance testing and perform all types of testing
Implementation & deployment
Testing the inbound and outbound call by dialer or tools
Call verification & validation
Call Monitoring
Call Recording
End

Figure-2 – Square Model (A proposed model)

3.2 Comparisons of Software Process models

Process model	Strengths	Weakness	Types of projects
Waterfall	Simple , easy to execute , logical	Requirements are frozen early, Disallow changes , user feedback not allowed, can not updated the hardware technology during development	For well understood problems, short duration’s projects.
Prototyping	Help in requirements elicitation;, reduce risks, leads to better systems.	Heavy process, disallow later changes	When uncertainties in requirements
Iterative	Quick delivery , reduce risk, allows user feedback, avoids requirement bloating	Each iteration have planning overhead, cost increase during iteration, software architecture may suffer as frequent changes.	For business where time is important, when risk of a long project can not be taken, when requirement are not known earlier.
Spiral	Planning and negotiations easier	Requirement not clear , needs confirmations and high risks	For large projects

Square	Planning and negotiations easier, requirement clear , no complexity of software applications, reduce risks	Cost increased due to hardware , software execute only after make a call	For small and medium type projects.

4. Conclusions and Future Work

Models of software development must account for software the interrelationships between software products and production processes, as well as for the roles played by tools, people and their workplaces. Modeling these patterns can utilize features of traditional software life cycle models, as well as those of automat able software process models. In this paper a new model is presented that has the potential to help companies to improve their Software Process Improvement implementation processes. This model provides a very practical structure with which to implement SPI programs for call center software application. However, this model is in initial stage and need further improvement and evaluation. In order to progress on this model and to have a statistically representative sample, we are planning many more interviews with SPI practitioners to reinforce the findings and to enrich our model with more fine grained activities within each phase of our SPI implementation process improvement.

5. REFERENCES

[1] Boehm, B., Anchoring the Software Process. Software. (July 1996) 73-82.

[2] Baumert, John; Mc Whinney &Mark.(1992).Software Measures and the Capability Maturity Model., Software Engineering Institute,CMU/SEI-92-TR-25, Pittsburgh, PA USA,September. 1462-77 379-389,

[3] Baddoo, N. and Hall, T. 2002. Motivators of software process improvement: An analysis of practitioner's views, *Journal of Systems and Software* (62). 85-96.

[4] Fuggetta, A., Software Process Roadmap. Future of Software Engineering Session. Proc. of 22nd Int. Conf on Software Engineering (2000) 27-34.

[5] Grundy, J.C.; Apperley, M.D.; Hosking, J.G.; Mugridge, W.B. A decentralized architecture for software process modeling and enactment, *IEEE Internet Computing,* Volume: 2 Issue: 5, Sept.- Oct. 1998, 53 -62.

[6] Goldenson, D. R. and Herbsleb, J. D. 1995. *After the appraisal: A systematic survey of Process Improvement, Its benefits, And Factors That Influence success*. SEI, CMU/SEI-95- TR-009 59.

[7] Herbsleb, J. D. and Goldenson, D. R. 1996. A systematic survey of CMM experience and results. *18th international conference on software engineering (ICSE-18).* Germany. critical success factors approach, *Journal of Information Science* (19). 425-437.

[8] ISO, ISO 9000-3:1997. Quality management and quality assurance standards. Part 3: Guidelines for the application of ISO 9001: 1994 to the development, supply, installation and maintenance of computer software. (International Organisation for Standardization, ISO, 1997).

[9] Jianguo Li, Jinghui Li, and Hongbo Li Research on Software Process Improvement Model Based on CMM World Academy of Science, Engineering and Technology 39 2008, pp 368-371.

[10] K. Benali, J. C. Derniame, Software processes modeling: what, who, and when, *Proceedings of the Second European Workshop on Software Process Technology* (September 1992).

[11] Kellner, M.I., and Hanser, G.A., Software Process Modeling. Technical Report, Software Engineering Institute. Carnegie Mellon University (May 1988).

[12] M.P.Thapliyal, Pratibha Dwivedi, Software Process Improvement in Small and Medium Software Organisations of India *International Journal of Computer Applications (0975 – 8887) Volume 7– No.12, October 2010*.

[13] Osterweil, L., et.al., Engineering Medical Processes to Improve Their Safety: An Experience Report. Proc of Method Engineering, (2007)

[14] P. H. Feiler, W. S. Humphrey, Software process development and enactment: Concepts and definitions *Proceedings of the Second International Conference on Software Process* (February 1993) 28-40.

[15] Pankaj Jolte , An Integrated Approach to Software Engineering Third edition Narosa Publishing house New DelhiPressman, R.S. and D. Ince, Software Engineering: A Practitioner's Approach. 5 ed. 2000, Maidenhead: McGraw-Hill

[16] Raffo, D., W. Harrison, M.I. Kellner, R. Madachy, R. Martin, W. Scacchi, and P. Wernick, Special Issue on Software Process Simulation Modeling, *Journal of Systems and Software*, 46(2-3), 89-211, 1999.

[17] Scacchi, W., Understanding Software Process Redesign using Modeling, Analysis and Simulation. *Software Process --Improvement and Practice* 5(2/3):183-195, 2000.

[18] Silvia T. Acuña, The Software Process: Modelling, Evaluation And Improvement, Handbook of Software Engineering and Knowledge Engineering Vol. 0, No. 0 (2000)

[19] Yogesh Singh, K.K.Aggarwal . *Software Engineering Third edition, New Age International* Publisher Limited New Delhi.

[20] Kathleen Coleman Dangle, Patricia Larsen, Michele Shaw, and Marvin V. Zelkowitz, “Software Process Improvement in Small Organizations: A Case Study”, IEEE Software, November / December 2005.