SOFTWARE PROCESS ASSESSMENT AND IMPROVEMENT MODEL IN SOFTWARE DEVELOPMENT LIFE CYCLE

Manoj kumar *

Rajeev kumar Sharma**

Rupak Sharma***

ABSTRACT

Software development is a challenging undertaking that is often critical to the safety of humans and the welfare of businesses. Problems caused by low quality, unreliable software applications include widespread inconvenience and loss of life. Software process assessment and improvement is an integral part of the software development life cycle. The improved software quality that results from these efforts increases product reliability and enhances customer satisfaction. The goal of this paper is to provide an overview of contemporary software process assessment and improvement models, to discuss the economics of these models, and to investigate techniques to increase the success rates of assessment and improvement projects.

***Keywords**: SDLC, SPI, Software quality, CMM.*

*[A.P], Organization: MCA –Dept, SRM-UNIVERSITY,NCR campus Modinagar-201204[U.P]

** [A.P], Organization: MCA –Dept, SRM-UNIVERSITY,NCR campus Modinagar-201204[U.P]

*** [A.P], Organization: MCA –Dept, SRM-UNIVERSITY,NCR campus Modinagar-201204[U.P]

INTRODUCTION

Review of Literature

The literature review that follows is organized by subject heading. Those subjects include the major software assessment and improvement models in use today. Included are BOOTSTRAP, Capability Maturity Model (CMM), ISO 9001, Personal Software Process (PSP), SPICE, Team Software Process (TSP), and Trillium.

BOOTSTRAP

BOOTSTRAP is a European method for software process assessment and improvement that was developed to speed up the application of software engineering technology in the European software industry [13]. The BOOTSTRAP methodology is based on the CMM . However, it has been extended and adapted to include ISO 9000 guidelines and the European Space Agency software engineering standard (ESA-PSS-05). In addition, BOOTSTRAP has proven suitable for use by all kinds and sizes of software development organizations. The main features of BOOTSTRAP are:

• Questionnaires for both site and project evaluation
• Uniform procedure and mandatory assessor qualification/training
• Constructive instead of a normative approach
• Open questions
• Immediate feedback and action planning

The BOOTSTRAP method adopted a process model which addresses processes and practices for both the software producing unit and the project. Process areas were divided into organization, methodology, and technology.

CMM

The Capability Maturity Model (CMM) plays an important role in the software improvement efforts (SPI) of organizations worldwide [13]. The process was developed by the Software Engineering Institute at Carnegie Mellon University in 1986. Its goal is to improve, over time, the application of an organization's software technologies. The model provides a guide for organizations to select software process improvement strategies by facilitating the determination of current capabilities and the identification of critical issues.

The CMM process is made up of five well-defined levels of sequential development: initial, repeatable, defined, managed, and optimizing [4]. These maturity levels provide a progressive scale for measuring the maturity of an organization and its ability to use software technologies. Organizations that depend on formal rules, instead of individual performers, to manage software projects are considered to be more "mature."

ISO 9001

ISO 9001 is an international standard for quality assurance in design, development, production, installation, and service [11]. It is broken down into twenty elements. ISO 9001-3 relates to the development, supply, and maintenance of software. Almost 90 percent of the companies that completed ISO 9001 implementation reported improved internal documentation as one of the most important benefits of registration.

ISO 9001 is similar to the CMM in the following areas: emphasis on process, documented processes, practiced processes, address the "what" and not the "how" [13]. Differences between the two approaches occur in the areas of focus, dimensions, assessment and certification, coverage, supplier's role, and level of detail. For example, an ISO 9001-compliant software organization would not necessarily satisfy the requirements for a level 2 in the CMM. However, it would satisfy most of the level 2 and some of the level 3 goals.

Personal Software Process (PSP)

The Personal Software Process is a process-based method developed by the SEI for software engineers to use to apply process definition and measurement to their personal tasks [6]. Most important, the PSP shows developers how to manage product quality, meet commitments, and justify their plans with data. In addition, the PSP follows the concepts of the CMM. The key message of the PSP is that developers should use process management concepts to identify the methods most effective for them.

SPICE

ISO/IEC 15504 was the result of the SPICE (Software Process Improvement and Capability dEtermination) project [13]. It provides a reference model for focused self assessments and includes a capability scale that is simple to understand. SPICE defines a two-dimensional model used in a process assessment to describe processes and process capability [3]. The first dimension details the processes an organization should use to supply, develop, operate,

evolve, and support software. The second dimension is made up of nine generic attributes used to characterize the capability of a process. These attributes are grouped into six capability levels (0-5).

Trillium

The Trillium model was initially designed for use with embedded software systems (e.g. telecommunications) and is based on the CMM [1]. Its architecture differs from the CMM in the following ways:

• Architecture is based on roadmaps instead of key process areas

• A product rather than a software perspective

• Wider coverage of capability impacting issues

• Customer focus and a telecommunications orientation

Trillium is comprised of five levels (1-5). These are unstructured, repeatable and project oriented, defined and process oriented, managed and integrated, and fully integrated [13].

METHODOLOGY

Research Type

This paper was a research-based descriptive study. The key outcome of the investigation was the exploration contemporary software process assessment and improvement models along with a look into the economics and success rates of assessment and improvement projects.

Research Methods Employed

The primary research method employed throughout the course of writing this paper was browser-based Internet searches. The literature reviewed included textbooks, journal articles, and magazine articles referenced by a select set of online resources. Relevant texts were located, ordered, and delivered using the Fatbrain.com Internet site. The full text articles from journals and magazines were located and subsequently downloaded.

Online Tools and Resources

A variety of online resources were used to locate and download literature relevant to the goal of the paper. These resources included ACM Search (www.acm.org/dl/search.html), IEEE Digital Library (http://computer.org/search.htm), and ProQuest Direct (http://proquest.umi.com/). Perhaps the most powerful search tools to be employed were the intelligent search agents [2] and [9].

Copernic [2] is a well-documented freeware search agent[2]. It uses predefined channel sets, which allows researchers to target inquiries to all major Web search engines and also search for relevant text in newsgroups. Copernic conducts fast, multithreaded, full Boolean searches with progress displays and customizable search depth. Once results are compiled, Copernic displays returns (including name, location, and introductory text) in a right-click-enhanced list box sorted by relevance.

Results

Competition among software developers continues to intensify as customers demand shorter cycle times, added functionality, and improved quality [5]. As they strive to deliver improved software at an ever increasing rate, software companies are often reluctant to sacrifice quality and incur increased development costs in order to decrease the time to market. Many are investing in software process assessment and improvement projects as one way to reduce costs, shorten cycle time, and increase quality.

Economics

The cost per capita of major software process improvements can exceed $20,000 and the implementation time last as long as five years in large software development organizations [7]. These per employee costs include training, consulting fees, software licenses, capital equipment, and office improvements. A recent study found that the “smallest” amount that can be spent and still achieve tangible benefits is around $700.

These are incurred through the use of methodological improvements that do not require capital investments: Joint Application Design (JAD), usage of function point metrics, and the usage of inspections. However, improvements in schedule, cost, and quality do not exceed 10 percent for this low an investment.

At the other extreme, several companies and government agencies have reported spending in excess of $10,000 per employee with little or no benefit [7]. Studies have also shown that smaller software companies are able to implement software process improvements methodologies at a lower cost and more rapidly than large corporations and government agencies.

Benefits

The benefits of software process assessment and improvement include lowered costs, increased quality, and reduced time to market [8]. For example, the SEI reported the following:

• Annual productivity gain of 37 percent.
• Annual reduction in time to market of 19 percent.
• Annual reduction in post-delivery defects of 45 percent.
• Average ROI for SPI of 5.7:1

Often the biggest payoffs of software process assessment and improvement projects are in human terms and not in dollars [8]. These include factors such as pride in work, increased job satisfaction, and improved ability to attract and retain software experts.

IMPROVEMENT TECHNIQUES

Implementing an SPI program is not as easy as teaching a new design method or purchasing a new software system [10]. Software organizations must experience a "paradigm shift" to be successful. This includes effectively dealing with an organization's resistance to change. One effective way of dealing with this resistance is to convince stakeholders with concrete and tangible examples of both problems and improvements. The successful implementation of SPI programs requires focusing on many people, organizational, process, quality, and methodological issues. Examples include [12]:

• Consistent management leadership with a focus on quality.
• Time to identify and implement improved processes.
• Realization that SPI is based on common sense and commitment.

Another technique to increase the chances of success is to watch for and eliminate the following factors that undermine SPI deployment [12]:

• Lack of management commitment.
• Unrealistic management expectations.
• Time-stingy project leaders.
• Stalling on action plan implementation.
• Achieving a CMM level becomes the primary goal.
• Inadequate training.
• Expecting defined procedures to allow staff inter changeability.

• Ineffective process assessments.

CONCLUSION

Software process assessment and improvement initiatives by the software industry are in response to low quality, high cost software applications. Developers have committed large sums to these programs. However, SPI programs must be embraced at all levels of an organization in order to be successful. Business managers should understand their business benefits. Project managers must welcome the increased visibility into the software process that they provide, and end users should be interested because of the higher probability of meeting cost, quality, and timing goals provided.

REFERENCES

[1] Coallier, F., Mayrand, J., & Lague, B. (1999). Risk Management in Software Product Procurement. In K. Emam & N. Madhavji (Eds.), *Elements of Software Process Assessment and Improvement* (pp. 23-44). Washington, DC: IEEE Computer Society Press.

[2] Copernic (2001). Copernic 2001. Retrieved June 6, 2001, from the World Wide Web: http://www.copernic.com.

[3] Drouin, J. (1999). The SPICE Project. In K. Emam & N. Madhavji (Eds.), *Elements of Software Process Assessment and Improvement* (pp. 45-55). Washington, DC: IEEE Computer Society Press.

[4] Freedman, A. (2000). *Computer Desktop Encyclopedia* (Vol. 13.4). Point Pleasant, PA:The Computer Language Company.

[5] Harter, D., Krishnan, M., & Slaughter, S. (1998). *The life cycle effects of software process improvement.* Paper presented at the International Conference on Information Systems.

[6] Humphrey, W., & Over, J. (1997). *The Personal Software Process (PSP).* Paper presented at the International Conference on Software Engineering, Boston, MA.

[7] Jones, C. (1999). The Economics of Software Process Improvements. In K. Emam & N.Madhavji (Eds.), *Elements of Software Process Assessment and Improvement* (pp. 133-149). Washington, DC: IEEE Computer Society Press.

[8] Krasner, H. (1999). The Payoff for Software Process Improvement: What it is and Howto Get it. In K. Emam & N. Madhavji (Eds.), *Elements of Software Process Assessment and Improvement* (pp. 151-176). Washington, DC: IEEE Computer Society Press.

[9] LexiBot (2001). Our Technology - Results: The LexiBot Expression. *BrightPlanet*. Retrieved June 6, 2001, from the World Wide Web: http://www.brightplanet.com/technology/results2.asp.

[10] Sakamoto, K., Nakakoji, K., Takagi, Y., & Niihara, N. (1998). *Toward computational support for software process improvement activities.* Paper presented at the International Conference on Software Engineering.

[11] Weissfelner, S. (1999). ISO 9001 for Software Organizations. In K. Emam & N. Madhavji (Eds.), *Elements of Software Process Assessment and Improvement* (pp. 77-100). Washington, DC: IEEE Computer Society Press.

[12] Wiegers, K. (1999). Software process improvement in Web time. *IEEE Software, 16*(4),78-76.

[13] Zahran. (1998). *Software Process Improvement: Practical Guidelines for Business Success*. Reading, MA: Addison-Wesley.