LEARNING SOFTWARE ORGANIZATIONS

A Literature Review on Impact of Organizational Learning on Success of Software Process Improvement Activities

Sezen Erdem
ASELSAN INC., P.K. 1, 06172, Yenimahalle / Ankara, Turkey
Senior Expert Software Engineer
E-mail: erdem@aselsan.com.tr

—Abstract —

Software has become an indispensible part of our lives and its importance is continually increasing in each day. Software development organizations are forced to produce software more and more rapidly with an acceptable quality and within an acceptable budget. However, life is not so easy in software development. Software development organizations face with problems in delivery of software products on time within the budget and having required quality and majority of these problems are related with management. Software process improvement (SPI) approaches are introduced to solve the problems in software development. Knowledge is in the core of software development so knowledge management plays an important role in success of SPI activities. Organizations should become learning organizations otherwise continuous improvement in software development cannot be achieved. Software development organizations are knowledge intensive organizations so a special term, Learning Software Organization, has been introduced for explaining the organizational learning in software development organizations. In this paper, the impact of knowledge management and organizational learning on success of SPI approaches will be discussed based on a literature review. The idea behind becoming Learning Software Organization and relationship of it with success of SPI approaches will be investigated.

Key Words: *Software Process Improvement, Knowledge Management, Organizational Learning, Learning Software Organizations*

JEL Classification: L86, Z00.

1. INTRODUCTION

Software is a collection of computer programs and related data that provide the instructions for telling a computer what to do and how to do it. Software has become an indispensible part of our lives and its importance is continually increasing in each day. Sensitive applications ranging from nuclear power plants to telecommunications networks are all based on software. Software is an astonishing invention of human being to make life easier but life is not so easy at software development world.

Reports and surveys about software development projects indicate that there is a problem in delivery of software products on time within the budget and having required quality. (Hayes, 2003; Thibodeau & Rosencrance, 2002; Gibbs, 1994). The findings indicate that something is going wrong in software development and when the vitality of software is considered, something has to be done to solve the problems. Software Engineering term is introduced in 1968 at NATO conference as an answer to the problems in software development. The idea is to make software development based on the principles and practices of engineering so that software development can be treated as a process that can be improved through engineering methods. However the nature of software brings difficulties to application of custom engineering methods. Software development is knowledge oriented area and has a soft nature. Software engineering is different from other engineering disciplines due to its soft nature (Kruchten, 2001). Application of custom engineering methods directly to software engineering is not possible. Software industry has started to search ways of producing good products both faster and cheaper. Many systematic attempts have been applied to produce higher quality software on time and within the budget. Since knowledge is in the center, the approaches include management of knowledge and organizational learning.

The outline of the paper is as follows. The relationship between SPI and Organizational Learning is explained in second part. The term Learning Software Organization and the idea behind the term is explanied in part three. Part four summarizes the idea and the aim of this study.

2. SPI & ORGANIZATIONAL LEARNING

SPI approaches have emerged as the solution to the problems in software development. The fundamental belief of SPI is that good products can only be the result of good processes. The aim of SPI is to make the processes more efficient and, eventually, to raise the product quality (Zahran, 1998).

There are worldwide accepted models and frameworks for SPI. The idea behind the models is to break down processes into sub processes so that processes can be measurable and manageable. Most famous SPI models are IDEAL, CMMI, and ISO 15504. In theory, the models are solution to many of the problems in software development process. But SPI models are not silver bullet to come up all problems. Organizations face with problems in application of SPI models and many of the implementation programs result in failure (SEMA, 2009). There are some critical success factors that should be taken into consideration to achieve success in implementation of SPI programs (Dyba, 2005; Rainer & Hall, 2002; Stelzer & Mellis, 1999; Wiegers, 1996; Goldenson & Herbsleb, 1995). The studies have revealed that business orientation, management support, employee participations, knowledge management and organizational learning and concern for measurement are the most critical factors in success in SPI. It will not be surprising to expect knowledge management and organizational learning concepts to be among the critical success factors of SPI activities.

Zahran (1998) proposed ten critical success factors for successful SPI implementation. One of these success factors is about becoming a learning organization. The critical activities consist of monitoring the results of SPI activities, taking measurements and learning from feedback results. Humphrey (1989) identified basic principles of software process change. One of the principles is continual learning. Basili and Caldiera (1995) stress on reuse of experience and learning by use of quality improvement paradigm for developing core competency, organizational sharing of knowledge and goal oriented measurement for success in SPI implementation. Case studies conducted by Meehan and Richardson (2002) demonstrate that if knowledge management activities can be made explicit, they may help companies improve their software processes. Dyba (2005) performed an empirical investigation of the key factors for success in SPI. The results of the study demonstrated that learning strategies of organizations are one of key factors in SPI. The study of Gasston and Halloran (1999) suggests that in order to achieve optimal benefits from implementing process improvement programs, organizations must move towards becoming learning organizations. Software Experience Center project in DaimlerChrysler company is an example application of organizational learning in SPI activities (Kurt,2002). Results of the project demonstrate that success rate in SPI is higher. Mathiassen et al. (2001) argue that SPI efforts depend on the implicit, individual knowledge of practitioners in an organization. Organizations should improve the existing knowledge of its software practices of software developers. Knowledge about the improved processes should be made available on different

organizational levels. Knowledge management is potentially useful in SPI efforts to facilitate the creation, modification, and sharing of software processes in an organization. The study of Ahlgren (2011) has revealed that systematic organizational learning is a prerequisite for software process improvement and knowledge management provides central improvement targets to increase process efficiency which results in product quality. Organizations require communication, coordination and knowledge management to turn improvement ideas into actual changes (Heikkilä, 2009; Niazi, Wilson & Zowghi, 2006; Dybå, 2005).

As understood from the previous paragraph, knowledge management is important in software development. Software development is a knowledge intensive activity. The main assets for software development organizations are not manufacturing plants, buildings and machines. Knowledge is the crucial source and should be managed carefully. However reality of software organizations brings some barriers to apply systematic knowledge management. Project schedule pressure can limit the time for learning (Nan & Harter, 2009), structures for knowledge sharing may be inadequate (Mathiassen & Pedersen, 2005) and individuals may not be willing to participate in the knowledge sharing (Abrahamsson, Salo, Ronkainen & Warsta, 2002; Nasir, Ahmad & Hassan,2008). There are also other obstacles for learning such as lack of business understanding, inadequate encouragement for learning, unsuitable organizational design and overly high expectations (Mathiassen & Pedersen, 2005). Systematic management of knowledge is especially important to overcome all the barriers in software development.

As mentioned earlier, organizational learning is a prerequisite for success in SPI. Organizational learning concept covers knowledge adoption, knowledge sharing and creation of new knowledge (Argote, 1999; Huber, 1991). Although individual learning is the basis of organizational learning, organizational learning is not equal to the cumulative learning of members of the organization (Huysman, 2000). Learning organization is defined as an organization skilled at creating, acquiring, and transferring knowledge, and at modifying its behavior to reflect new knowledge and insights (Garvin, 1993). Learning organization forms an ideal entity that has the capability to adopt learning processes in practice (Easterby-Smith & Lyles, 2005) and organizational knowledge describes the form and nature of the knowledge that is possessed by the organizations (Easterby-Smith, Crossan & Nicolini, 2000). Knowledge that is collected and exploited for future actions forms organizational memory and in turn building and use of organizational memory is dependent on knowledge management (Ahlgren, 2011). The goal in SPI is to establish organization-wide efficient and useful processes

which are reflection of organizational memory. The critical success factors in SPI programs indicate that shared visions and practices across the organization are required to establish efficient and useful processes. The people should have a mutual understanding and agreement on the process targets (Akgün, Lynn & Reilly, 2002). SPI programs enable sharing knowledge between individuals, teams and departments across organizational boundaries (Mathiassen & Pourkomeylian, 2003; Wenger, 2003). The attributes of learning organization (Garvin,1993), systematic problem solving, experimentation with new approaches, learning from their own experience and past history, learning from the experiences and best practices of others, and transferring knowledge quickly and efficiently throughout the organization, are key elements of the underlying philosophy of SPI models. In the light of the findings, SPI can be seen as a special form of knowledge management (Mathiassen & Pourkomeylian, 2003). SPI models advocate that information about processes should be defined, standardized, and used by the entire organization. To do this, the appropriate process information should be gathered and stored in some central access area, which is available to all employees (Meehan & Richardson, 2002). Mature organizations do not depend on a few employees having all process knowledge. There are mechanisms to support inexperienced employees with explicit process knowledge.

One of the most accepted SPI models, CMMI v.1.3, guides organizations to set up mechanisms to enable knowledge management in organizations and force it to become organizational culture. CMMI Level 3 requires organizations to define their process and generate a set of standards for processes. Basic process management process areas in CMMI provide the organization with a capability to document and share best practices and organizational process assets which leads to learning across the organization (CMMI-DEV, 2010).

3. LEARNING SOFTWARE ORGANIZATIONS

Importance of knowledge in software development organizations introduced a new term for organizational learning; Learning Software Organizations. The term is defined as an organization that learns within the domain of software development, evolution and application (Ruhe, 1999). The idea is based on concept of experience factories (Basili, 1994). In experience factory approach, evaluated experiences are accumulated into a repository of integrated experience models in a form that can be effectively accessed, understood and modified to meet the needs of the current projects. It is a logical or physical organization that supports project developments by analyzing and synthesizing all kinds of experience, acting as a repository for such experience, and supplying that

experience to various projects on demand (Basili, 1994). In learning software organizations, learning objects can include models, tools, techniques and methods applied during different stages of the software development process.

Dyba (2001) identified four foundations for learning software organizations; Social Learning, Sense Making, Knowledge Creation and Purposeful Action. Social Learning focuses on SPI as a socially constructed, collaborative activity. It is important to understand why and how we do what we do in software development to deliberately and repeatedly do in diverse situations. Moreover, understanding of software development process should be shared with others and it should be worked on to improve it. Sense Making aims at developing shared understandings of happenings around. Organization wide constructed shared understanding and vision is regarded as organizational culture. Inside this culture, software developers can continuously make sense external environment and adapt to the environment accordingly. Knowledge Creation aims at generating new knowledge as organization. The steps of knowledge conversion process of Nonaka and Takeuchi (1995) is applied to create organizational knowledge. During socialization, the developers share their experiences with others. Project close-out meetings or lessons-learned sessions after projects are most suitable places to experience learning by sharing. In externalization phase, tacit knowledge of developers is transformed into explicit knowledge by documenting. Different kinds of explicit knowledge both coming from inside or outside of the organization is integrated in combination phase. The purpose is to deliver appropriate knowledge when required. Knowledge bases and experience factories are examples. Finally, in internalization phase, practitioners access the relevant knowledge and adopt it to their context. By this way they learn by experiencing. But there are also ideas arguing that one form of knowledge cannot be converted into the other (Cook & Brown, 1999). In the light of these discussions of knowledge creation, Dyba (2001) concludes that, rather than converting tacit knowledge to external knowledge, software developer uses tacit knowledge to create new explicit knowledge. For learning software organizations, the important issue is to enable conditions for creating knowledge. Purposeful action aims at making use of new interpretations and new knowledge to construct improved courses of action. Purposeful action is at the hearth of developing a learning software organization because without taking any action, no improvement can be gained.

It is non debatable that becoming learning software organization is necessary for achieving continuous improvement in development process. Establishing learning software organizations is not solely a technical issue. It also involves change in

organizational culture (Ruhe, 1999). Organizations are composition of different communities which are interacting with each other and all these communities have different specialized knowledge. Becoming a learning organization requires to set up bridges between the diverse communities and integrate their knowledge to create a shared perspective. In order to understand and improve software development process, appropriate abstractions should be generated resembling the reality of software development. Organizations should find enabling techniques for Learning Software Organizations to help them to improve their processes.

4. CONCLUSION

Software is now a part of life and many critical activities depends on it. So software should be dependable. Software development process is the most important factor for the reliability of software. Reliable software can only be the result of a mature software development process. SPI activities are introduced to fix the problems in software development process and produce better product with low cost and in short time. The importance of knowledge in software development is obvious. Knowledge is the core asset in software development. Systematic management of knowledge is necessary to find solutions to the problems (both technical and managerial) in software development.

This study tries to emphasize the importance of organizational knowledge management in SPI activities. Several publications and case studies are investigated to reveal the importance of knowledge, knowledge management and organizational learning in SPI activities. Continuous improvement in software development process requires learning as organization. A special term for software development organizations, Learning Software Organization, and the idea behind is explained. As conclusion remark, it is obvious that software development organization should become learning software organizations to gain real, continuous and effective success in software development process.

BIBLIOGRAPHY

Abrahamsson, P., Salo, O., Ronkainen, J. and Warsta, J. (2002). Agile software development and methods - review and analysis. VTT publications 478, Espoo.

Ahlgren,R. (2011). Software Patterns, Organizational Learning and Software Process Improvement, ISBN 978-951-39-4173-4. University of Jyvaskyla.

Akgün, A.E., Lynn, G.S., and Reilly, R. (2002). Multi-dimensionality of learning in new product development teams, European Journal of Innovation Management 5(2), 57-72.

Argote, L. (1999). Organizational learning: creating, retaining, and transferring knowledge. 1999. USA: Kluwer Academic Publishers.

Basili, V. R. & Caldiera, G. (1995). Improve Software Quality by Reusing Knowledge and Experience, Sloan Management Review, pp. 55–64.

Basili, V. R., Caldiera, G., and Rombach, H. D. (1994), Experience Factory, In J. J. Marciniak (eds.) in Encyclopedia of Software Engineering, vol. 1, pp. 469-476. John Wiley & Sons, New York

Cook, S.D.N & Brown J.S.(1999). Bridging Epistomologies: The Generative Dance Between Organizational Knowledge and Organizational Knowing, Organization Science, Vol. 10, No. 4, pp. 381-400

Dyba T. (2001). Enabling Software Process Improvement: An Investigation of the Importance of Organizational Issues. Doctoral Dissertation, Norwegian University of Science and Technology

Dyba T. (2005), An Empirical Investigation of the Key Factors for Success in Software Process Improvement, IEEE Transactions on Software Engineering, Vol. 31, No. 5

Easterby-Smith, M., Crossan, M. and Nicolini, D. (2000). Organizational learning: debates past, present and future. Journal of management studies 37 (6), 783–796.

Easterby-Smith, M. and Lyles, M. A. (2005). The Blackwell Handbook of Organizational Learning and Knowledge Management. Blackwell Publishing, USA.

Garvin, D. A. (1993). Building a learning organisation, Harward Bus. Rev., July - August, pp. 78 - 91.

Gasston, J., Halloran, P. (1999). Continuous Software Process Improvement Requires Organisational Learning: An Australian Case Study, Software Quality Journal, 8, 37 - 51

Gibbs,W.W. (1994). Software's Chronic Crisis, Scientific American, Vol. 271, September pp.72-81

Goldenson, D.; Herbsleb, J. D. (1995). After the Appraisal: A Systematic Survey of Process Improvement, its Benefits, and Factors that Influence Success." (Technical report CMU/SEI-95-TR-009). Software Engineering Institute, Pittsburgh, PA.

Hayes M. (2003). Precious Connection, InformationWeek, pp. 34-50,

Heikkilä, M. (2009). Learning and Organizational Change in SPI Initiatives. In F. Bomarius et al. (Eds.): PROFES 2009, LNBIP 32, Berlin Heidelberg,Springer-Verlag, 216–230.

Huber, G.P. (1991). Organizational learning: The contributing processes and the literatures. Organization science 2 (1), 71-87.

Humphrey, W.S. (1989). Managing Software Process. Addison-Wesley Professional

Huysman, M. (2000). Rethinking the organizational learning: analyzing learning processes of information system designers. Accounting, Management and Information Technology 10, 81-99.

Kurt, S., Hunnius, J. P., Basili V. (2002). Experience in Implementing a Learning Software Organization. IEEE Software May/June pp. 46 - 49

Mathiassen, L. and Pedersen, K. (2005). The dynamics of knowledge in systems development practice. In Proceedings of the 38th Annual Hawaii International Conference on System Sciences. IEEE, 233a - 233a.

Mathiassen, L. and Pourkomeylian, P. (2003). Managing knowledge in a software organization. Journal of Knowledge Management 7 (2), 63-80.

Mathiassen, L., Pries-Heje, J. and Ngwenyama, O. (2001). Improving Software Organizations – From principles to Practice. Addison-Wesley.

Meehan B., Richardson I.(2002). Identification of Software Process Knowledge Management, Software Process Improvement and Practice, 7, 47-55 (DOI: 10.1002/spip.154)

Nasir, M.H.N.M., Ahmad, R. and Hassan, N.H. (2008). An empirical study of barriers in the implementation of software process improvement project in Malaysia. Journal of Applied Sciences 8 (23), 4362-4368.

Niazi M., Wilson, D., Zowghi, D. (2006). Critical success factors for software process improvement implementation: an empirical study. Software Process: Improvement and Practice. Special Issue on Free or Open Source Software Development (F/OSSD) Projects 11(2), 193 - 211.

Nonaka I., Takeuchi H. (1995) - The Knowledge-Creating Company: How Japanese Companies Create the Dynamics of Innovation, Oxford University Press, 1995.

Rainer, A., Hall T. (2002), Key Success factors for implementing software process improvement: a maturity based analysis", Journal of Systems and Software, 62 (2): p.71-84

Ruhe G. (1999). 11th International Conference on Software Engineering and Knowledge Engineering, SEKE'99, Kaiserslautern, Germany, June 16-19.

SEI (2010). CMMI® for Development, Version 1.3 Software Engineering Institute, Carniege Mellon Univercity.

SEMA (2009). Process Maturity Profile of the Software Community, Software Engineering Institute, Carnegie-Mellon University.

Stelzer, D., Mellis W. (1999), Success Factors of Organizational Change in Software Process Improvement," Software Process Improvement and Practice, Volume 4, Issue 4, John Wiley & Sons Ltd

The Standish Group, (2009). CHAOS Summary Report, The Standish Group International

Thibodeau, P. & Rosencrance, L. (2002). Users Losing Billions Due to Bugs. Computerworld, vol. 36, no. 27, pp. 1-2

Wiegers, Karl E.(1996). Software Process Improvement: Ten Traps to Avoid." www.processimpact.com

Zahran, S. (1998). Software Process Improvement: Practical Guidelines for Business Success, Harlow, England: Addison-Wesley.