Empirical Software Engineering, 7, 387–390, 2002.

Enabling Software Process Improvement: An Investigation of the Importance of Organizational Issues[☆]

TORE DYBÅ tore.dyba@sintef.no
SINTEF Telecom and Informatics, S.P. Andersensv. 15, NO-7465 Trondheim, Norway

Abstract. This doctoral study was initiated to explore the relative importance of organizational issues in software process improvement. It is based on a pilot case study, a multiple case study of twelve organizations, and a quantitative survey among 120 software organizations. The findings from the investigations suggest that the key to successful learning is a continuous and simultaneous dialectic interplay between the knowledge that the organization has established over time, and the knowing of the organization's members in their respective contexts. Also, the findings indicate that success depends critically on six organizational factors. Finally, the findings show that there are important differences between small and large software organizations, specifically in the ways in which they react to unstable and changing stimulus situations.

Keywords: Software process improvement, organizational knowledge creation, learning software organization, critical success factors, survey research, grounded theory.

The term "software engineering" was coined at the historic NATO conference in Garmisch-Partenkirchen in 1968 (Naur and Randell, 1969) as an answer to the problems in software development. It was deliberately provocative, implying the need for software development to be based on the principles and practices seen in engineering. Thus, the point of departure for most of the subsequent efforts in addressing the problems in software development has been to treat the entire task of software development as a process that can be improved through engineering methods.

However, software development and software process improvement (SPI) involves organizational as well as technical issues. While software engineering has been offered as a way of resolving the intrinsic technical problems, the organizational problems need another approach. Nevertheless, both classes of problems need to be jointly resolved to improve the process of software development—they are co-producers of the outcome.

Concern with the relationship between the problems in software development and organizational issues is not a new phenomenon. More than 25 years ago, Lucas

[☆] Doctoral dissertation defended at the Norwegian University of Science and Technology, December, 2001, ISBN 82-471-5371-8, 350 pages. Advisors: Prof Dr Reidar Conradi, Prof Dr Tor Stålhane.

(1975) wrote a classic book on why information systems fail, suggesting that "the primary cause for system failure has been organizational behavior problems." There is, however, an increasing amount of evidence to suggest that organizational issues are now more important, and will become even more critical, to the successful improvement of software development than they were when Lucas wrote his book (see e.g. the October 1993 issue of *Communications of the ACM* and the May/June 2001 issue of *IEEE Software*).

Still, the typical response to the persistent problems in software development has been to apply even more rigorously the principles of engineering. Organizational issues are often ignored or not properly addressed in much of the recent literature, with only a handful of studies being reported. Furthermore, organizational issues are only treated implicitly, or in many cases not at all, even by managers who perceive organizational issues to be of more importance than technical issues in determining the successful outcome of systems development projects (Doherty and King, 1998). To help fill this gap, this doctoral study was initiated to explore the relative importance of organizational issues in enabling software process improvement.

To narrow the focus of the investigation, the research problem was addressed by studying three research questions. The first question focused on identifying the key processes that are part of a successful software organization's learning cycle. This part of the investigation was grounded in prior consulting experience in software development and observations done in a pilot case study as well as in a multiple case study of 12 software organizations. In addition to identifying SPI as a social, collaborative activity, these investigations also led to the identification of sense making, knowledge creation, and purposeful actions as the foundations of the learning software organization. Therefore, to explain why some SPI initiatives are more successful than others, a dynamic model of SPI was constructed that incorporated these foundations. The main conclusion on the first research question was that the key to successful learning is a continuous and simultaneous dialectic interplay between the knowledge that the organization has established over time, and the knowing of the organization's members in their respective contexts.

The second question focused on identifying the key factors for success in SPI. In addressing this question, a literature review of more than 600 references and a quantitative survey of 120 software organizations were performed. Based on the results of bivariate correlational and multiple regression analyses, the main conclusion on the second research question was that SPI success depends critically on six organizational factors: business orientation, involved leadership, employee participation, concern for measurement, exploitation of existing knowledge, and exploration of new knowledge. This suggests that, rather than trying to imitate technical procedures, software organizations should focus their SPI efforts on creating an organizational culture within which these procedures can thrive.

The third question focused on finding the relationships between organizational context and modes of learning in software organizations. Two important results emerged from the investigation of this research question. First, the results showed that environmental turbulence had no influence on the level of SPI success. What's more important, however, was that large successful and small successful organizations differed fundamentally in their respective approach to SPI. The most important conclusion on the third research question was that small software organizations, in turbulent environments, require learning strategies that are closely aligned with explorative behavior, while at the same time promoting the exploitation of past experience.

The conclusion on the research problem was that SPI cannot be managed, but only enabled through the space in which the software organization creates the possibilities for sense making, knowledge creation, and purposeful action. Taken together, the answers to the three research questions constitute the main contributions of the dissertation. So, returning to the original question on how software organizations can enable SPI, the study found that a different context than that proclaimed by current "best practice" SPI approaches must be created.

The main contribution of the dissertation is to increase the understanding of the influence of organizational issues by empirically showing that they are as important in SPI as technology, if not more so. This suggests that the large literature on organizational theory, developed in non-software settings, has more relevance to software development and SPI than has previously been recognized in software engineering. Also, it suggests that software organizations that want to prosper in the 21st century should synergistically combine technology with social collaboration to become learning software organizations.

Acknowledgement

This research was supported by the Research Council of Norway through grant no. 118206/221.

References

Doherty, N. F., and King, M. 1998. The Importance of Organisational Issues in Systems Development, *Information Technology and People* 11(2): 104–123.

Lucas, H. C. 1975. *Why Information Systems Fail*. New York: Columbia University Press.

Naur, P., and Randell, B. (eds.) 1969. Software Engineering, *Proceedings of the NATO Conference in Garmisch-Partenkirchen*, NATO Science Committee, Scientific Affairs Division, NATO, Brussels, January.

Selected publications from the dissertation*

Conradi, R., and Dybå, T. 2001. An Empirical Study on the Utility of Formal Routines to Transfer Knowledge and Experience, *Proceedings of the 8th European Software Engineering Conference* (*ESEC'2001*), Vienna, Austria, 10–14 September.

Dybå, T. 2000a. An Instrument for Measuring the Key Factors of Success in Software Process Improvement, *Empirical Software Engineering* 5(4): 357–390.

Dybå, T. 2000b. Improvisation in Small Software Organizations, *IEEE Software* 17(5): 82–87.

*In total, 14 papers and articles were published during the study period.

Dybå, T. (ed.): 2000c. *SPIQ—Software Process Improvement for Better Quality*: *Methodology Handbook* (in Norwegian), IDI Report 2/2000, Trondheim, Norway: Norwegian University of Science and Technology (NTNU), Department of Computer and Information Science.

Dybå, T., and Moe, N. B. 1999. Rethinking the Concept of Software Process Assessment, *Proceedings of the European Software Process Improvement Conference* (*EuroSPI'99*), Pori, Finland, 25–27 October.

Dybå, T., Stålhane, T., and Palmstrøm R. 1997. Experience of Goal-Oriented Measurement using ami and GQM, *Proceedings of the 8th European Software Control and Metrics Conference* (*ESCOM'97*), Berlin, Germany, 26–28 May.