ProPAM/Static: A Static View of a Methodology for Process and Project Alignment

Paula Ventura Martins[1], Alberto Rodrigues da Silva[2]

[1] Research Centre of Spatial and Organizational Dynamics,
Universidade do Algarve, Campus de Gambelas,
8005-139 Faro, Portugal
[2] INESC-ID, Instituto Superior Técnico, Universidade de Lisboa, Portugal
{pventura@ualg.pt,alberto.silva@tecnico.ulisboa.pt}

Abstract. Process descriptions represent high-level plans and do not contain information necessary for concrete software projects. Processes that are unrelated to daily practices or hardly mapped to project practices, cause misalignments between processes and projects. We argue that software processes should emerge and evolve collaboratively within an organization. In this article we present a Process and Project Alignment Methodology for agile software process improvement and particularly describe its static view.

Keywords: Software Process Improvement, Process Management, Project Management.

1 Introduction

Software process improvement (SPI) is a challenge to organizations trying to continually improve software quality and productivity and to keep up their competitiveness [1]. Organizations tend to react to: (1) changes in the environment that they operate, (2) changes at a corporate level, (3) unplanned situations not considered in the model, or (4) improve the quality of their final products. Such changes may be caused, for example, by poor performance, by new tools acquired by the company to support its software development teams, changes in the marketing strategy or in clients' expectations and requirements. Thus, an existing process model must be modified or extended to reflect the evolution of the environment and/or internal changes. However, existing process models – that only take into account descriptive aspects, such as work related activities and technical work products – couldn't address such features. Additionally, several surveys and studies [2-4] have emphasized that the majority of small organisations are not adopting standards such as CMMI [5]. Another case is observed in Brazil where software industry and universities are working cooperatively in implementing a successful SPI strategy that take into account software engineering best practices and aligned to Brazilian software organizations context [6].

We argue that the emphasis in SPI should be stressed on communication, coordination, and collaboration within and among project teams in daily project activities, and consequently the effort in process improvement should be minimized

and performed as natural as possible. Little attention had been paid to the effective implementation of SPI models, which has resulted in limited success for many SPI programs. SPI managers want guidance on how to implement SPI activities, rather than what SPI activities do actually implement. Limited research has been carried out on exploring new approaches to effectively implement SPI programs. However, to bridge this gap some initiatives have emerged, namely the MIGME-RRC methodology [7]. On this basis, we propose a new methodology to describe and improve software process based on organizations projects experience.

In this paper we propose a SPI methodology called Process and Project Alignment Methodology (ProPAM). The main goal is to develop a SPI methodology that can evolve with project's knowledge and consequent improvement at software development process level. This methodology takes into consideration project and organization's views and adapts the best practices to define a base software development process. The model is grounded in personal experience and observations from a real case study, related to the extensive literature review, and focused on the three main objectives: (1) further understand how modeling and implementation of software processes can contribute to successful SPI; (2) provide a SPI approach for SPI practitioners to ensure a successful process implementation and (3) contribute to the body of knowledge of SPI with a focus on implementation of software processes based on project experience.

The remainder of this paper is organized as follows: Section 2 presents related work and other initiatives. Section 3 overviews ProPAM and describes the details of its Static View. Finally, Section 4 presents the conclusions and discusses our perception that this proposal has innovative contributions for the community.

2 Related Work

There are several popular SPI models, such as ISO/IEC 15504 [8], CMMI [5] and ISO/IEC 29110 [9]. These models have become well known among practitioners and researchers. However, implementation and adoption of SPI at software development organizations is frequently unsuccessful [10,11]. These SPI models are often prescriptive and attuned to those relative areas for which they are intended and therefore do not take into account other aspects like project and organization specific features. Rather than just repair and adjust the process to specific areas imposed by these SPI models, we should refocus SPI to analyze the current organization practices and introduce practices adapted to the organizations needs. SPI should be based on project's experience and learn with project team member's. Software development is not a rigid or a disciplined manufacture. It has a strong creative and social interaction that can't be totally re-planned in a standardized and detailed process model elaborated by specific groups and without active participation of project teams. These SPI models identify **what** to improve but don't give any information about **how** to do it. Indeed, given the reported problems with existing SPI models, there is a need for a more comprehensive model to SPI.

We argue that SPI requires further researches on SPI models based on project's experience. Following the trend of agile processes [12,13], SPI requires a

commitment to project learning and this organizational knowledge is constructed through collaboration of project team members. Therefore, we need to include guidelines in a SPI model that allow to incorporate project team knowledge in the software process (without constraints imposed by a standard which limits embed tacit knowledge) and that can address features not focused on existing SPI models.

3 Process and Project Alignment Methodology – The Static View

This section presents an overview and details of the ProPAM methodology [15]. ProPAM is about how the process and project are represented and how project teams acquire and use knowledge to improve work. ProPAM is different from existing models in which SPI is seen as starting with the implementation of best practices according to a predetermined scheme, independently of organization's experience of problematic situations. The proposed methodology proposes solving the problems faced in software development projects carried within the organizations. A critical feature in ProPAM is the integration of SPI activities with software development activities. This way, we considered project teams and projects as the baseline for improvement. Project managers and project teams, under the supervision of the process manager, are the foremost responsible for keeping the organization's processes on the leading edge. As Figure 1 illustrates, ProPAM includes SPI activities to monitoring and tracking software projects (project level) besides the SPI activities that intend to develop and implement the software process (process level). The main differences between these two levels are the followings.

At **project level**, ProPAM helps organizations in their efforts to assess and manage problematic situations of specific projects, and to develop and implement solutions to manage these problems. The project level encompasses project(s) information needed to systematically support or reject many of the decisions about the process. At project level**,** team members work together to develop work products. This focus on project team and their collaborative process is important because no one embodies the breadth and depth of knowledge necessary to comprehend large and complex software systems. Project teams are concerned with concrete situations as experienced in all their complexity during software development. Projects context is constantly being created and recreated and it can't be based on a static process model. Participating in a project team is consequently not only a matter of developing software, but also to change organization's knowledge about software development.

At **process level**, project's feedbacks conduct to process reviews and iterative process improvement. The dynamic interplay between these two levels shows the synergy between the activities performed by project roles (project manager and team member) and the activities performed by the process roles (process manager) involved in SPI. At process level**,** actors involved in SPI programs take time to express its shared practices in a form that can meaningfully be understood and exploited by other organizational actors. This includes not only the definition of concepts, models and guidelines, but also the evaluation of success of the improvements.

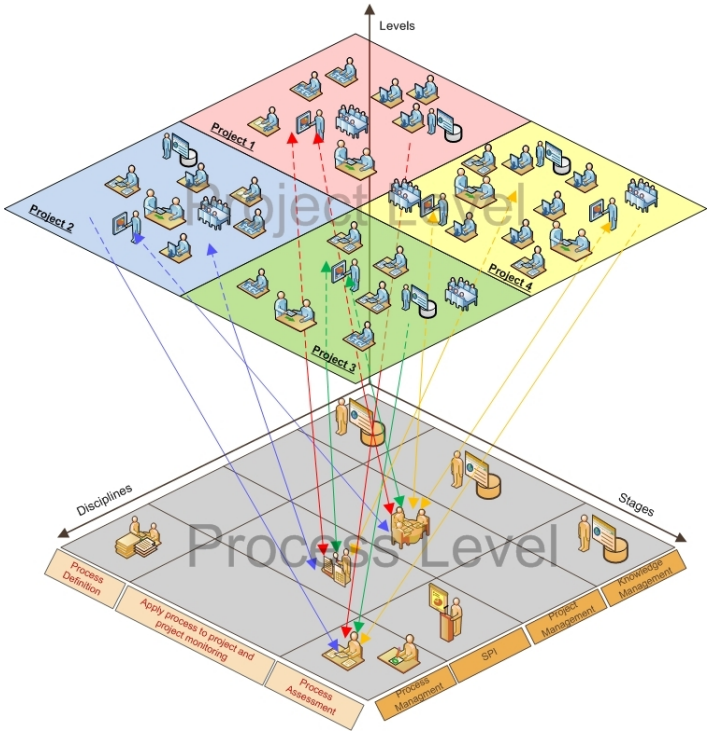

Fig. 1. ProPAM Levels

The scope of the levels is defined considering that process and projects actors collaborate on SPI programs. However, to manage the inherent complexity of these levels, namely ProPAM represented at process level, it is common practice to present views on the models of a level. In general, a view is defined as a projection of a process model that focuses on selected features of the process [14]. ProPAM is organized in two correlated and complementary views, the static view and the dynamic view that represent the behavior at that particular level. Whereas the static view describes aspects of the methodology as core and supporting disciplines in terms of activities, work products and roles, the dynamic view shows the lifecycle aspects of ProPAM expressed in terms of stages and milestones. The remaining of the section is dedicated to details of the static view. Previous work already described the dynamic view [16].

The ProPAM static view describes disciplines involved in SPI and relations between them. Static view is expressed as workflow diagrams, which show structural elements (roles, work products and activities) involved in each ProPAM discipline. Swim lanes in the workflow diagram make obvious the roles responsible to perform specific activities and also identifies involved input and output work products. For each role, control flow transitions between activities are omitted since activities are neither performed in sequence, nor done all at once. Nevertheless, such representation does not describe SPI program changes with time passing. A time-based perspective of the process is left to dynamic view.

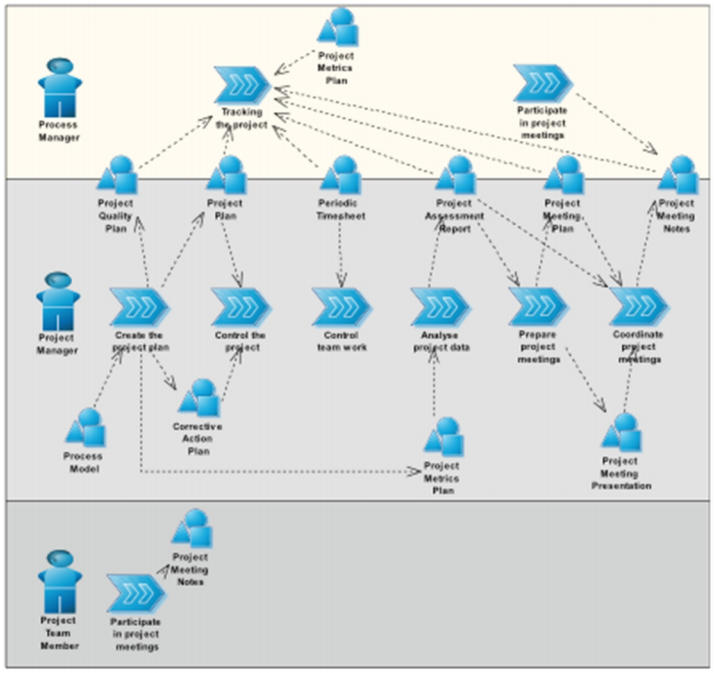

Fig. 2. Project Management View

ProPAM static view integrates project management, process management, SPI and Knowledge Management (KM) disciplines. These disciplines assure alignment of projects with organization vision and goals, and the adopted and improved software process. Other disciplines of concern were omitted, like business modeling, analyze and design, environment, requirements management or configuration management, because those concerns are considered too specific for SPI programs.

Project Management. Project managers are usually interested to be informed about how the project follows its base process and how to handle the changes introduced in the project that are not compliant with the respective process. It is important to detect deviations from schedules (project control and project tracking activities) as soon as possible in order to take corrective actions. Deviations allow identifying elements that do not appear or are incorrectly described in the software process. Therefore, project managers have to be informed about process states in a way that satisfies management needs. This bridges the gap between process management and project management, since project plans should reflect the exact set of activities defined for a given process. To avoid creating detailed plans, project managers may create the plan incrementally, and using only higher-level activities, leveraging lower level tasks only as a guide for how to do the work. The most important goal is to address conflicts and align projects and processes. Figure 2 illustrates the main roles, activities and work products involved in the Project Management discipline.

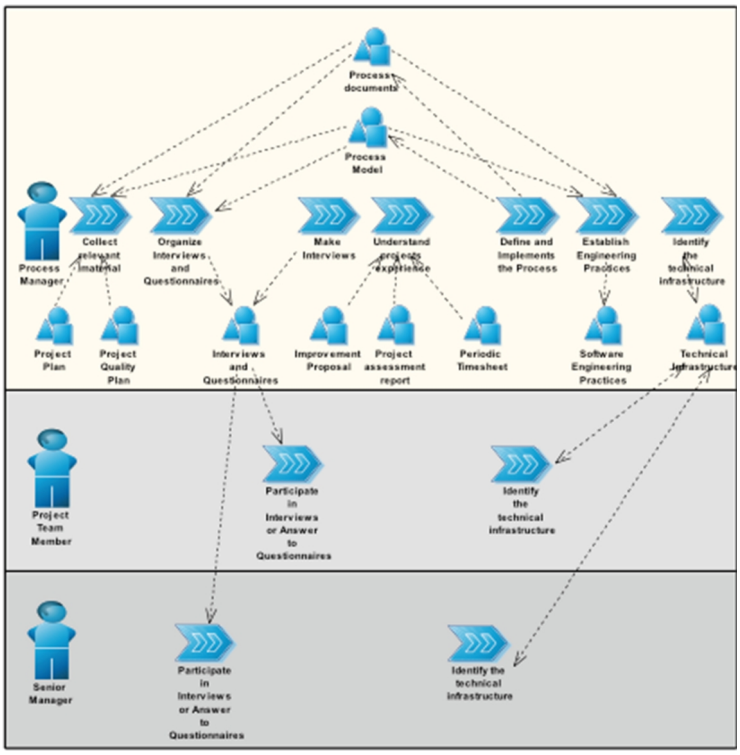

Fig. 3. Software Process Management View

Software Process Management. Software process management discipline involves actions performed to coordinate knowledge acquisition about software processes, to model and to analyze the way teams develop software and, finally, to ensure that future software processes are carried out on the basis of findings obtained in process analysis [17]. Software process management is a collective work involving project managers, senior engineers and the process manager. Nevertheless, at process level the process manager must be concentrated in process definition and implementation. While at project level the process manager coordinate the interaction with projects team members with respect to process assessment. Software process roles should develop the following activities with direct impact on SPI: (1) collect relevant material; (2) organize interviews and questionnaires; (3) make interviews; (4) understand project experiences; (5) define and implements the process model; (6) establish engineering practices; (7) identify the technical infrastructure; and (8) participate in interviews/answer to questionnaires. Figure 3 presents the main roles, activities and work products involved in the Software Process Management discipline. Some details of the main activities allow understanding the importance of this discipline. In this case, the project manager has the same responsibilities of the other team members, so he isn't seen as a specific role. The most important goal is to design a set of solutions for the software process based on performed projects. To help the viewer understanding the diagram in Figure 3, a restriction some flows from

and to work products were omitted, since these work products are inputs or outputs of almost all the activities of the discipline.

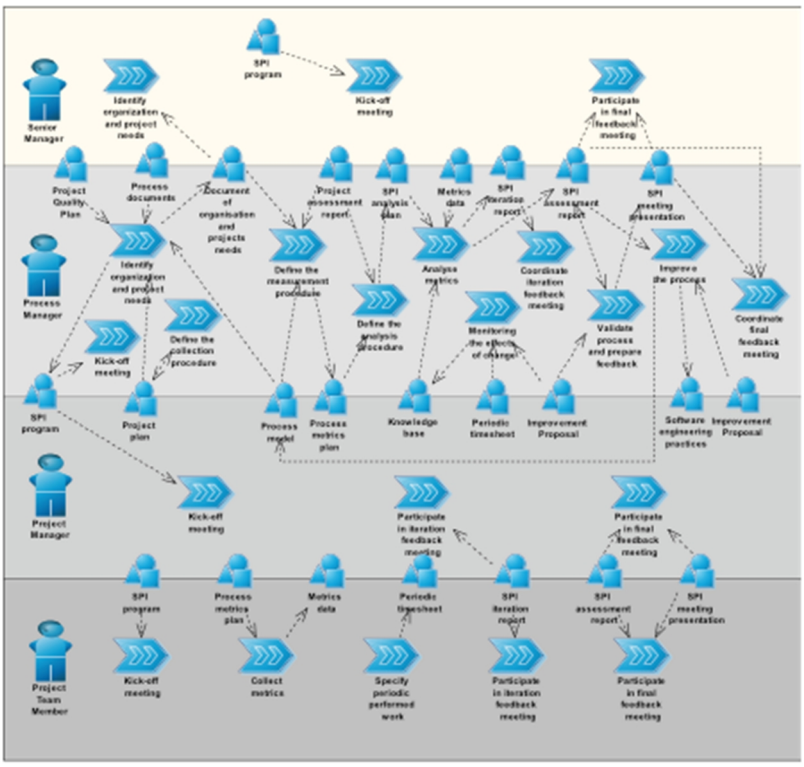

Fig. 4. Software Process Improvement View

Software Process Improvement. The effort of supporting software processes is encompassed by the SPI discipline of the ProPAM methodology. This discipline extends the process management discipline, where the main difference is the scope: the process management discipline is concerned with the process configuration for the organization, while the SPI discipline addresses improvements in the process itself based on assessment results. SPI is the discipline of characterizing, defining, measuring and improving software management and development processes, leading to software business success, and successful software development management. Success is defined in terms of greater design innovation, faster cycle times, lower development costs, and higher product quality, simultaneously [18]. SPI focus is related to the establishment of a set of responsible roles and of the associated competences concerned to the software development process with the aim of improving the organization's software process. The main activity of this discipline is the maintenance of software process knowledge and the improvement of coordination and monitoring activities. The organization must plan to create a stable environment and monitor these activities in order to have clear commitments for current and future projects. The most important goals to be achieved are: (1) software development process and improvement activities are coordinated throughout the organization; (2)

the strengths and weakness of the used software process are identified relative to a base process, if it was previously defined; and (3) improvement activities are always planned. ProPAM also suggests that organizations should identify a group of software managers composed by skilled persons and (internal or external to the organization) advisors, who contribute to identify the process strengths and to improve it when weakness are identified. Figure 4 presents the workflow diagram that illustrates the main roles, activities and work products involved in the SPI discipline.

Knowledge Management (KM). Data is organized into information by combining data with prior knowledge and the person's self-system to create a knowledge representation. This is normally done to solve a problem or make sense of a phenomenon. This knowledge representation is consistently changing as we receive new inputs, such as learning, feelings, and experiences. Knowledge is dynamic, that is, our various knowledge representations change and grow with each new experience and learning. Due to the complexity of knowledge representations, most are not captured by documents; rather they only reside within the creator of the representation. In many cases, the knowledge representation stays within the creator, in which case the "flow of knowledge" stops.

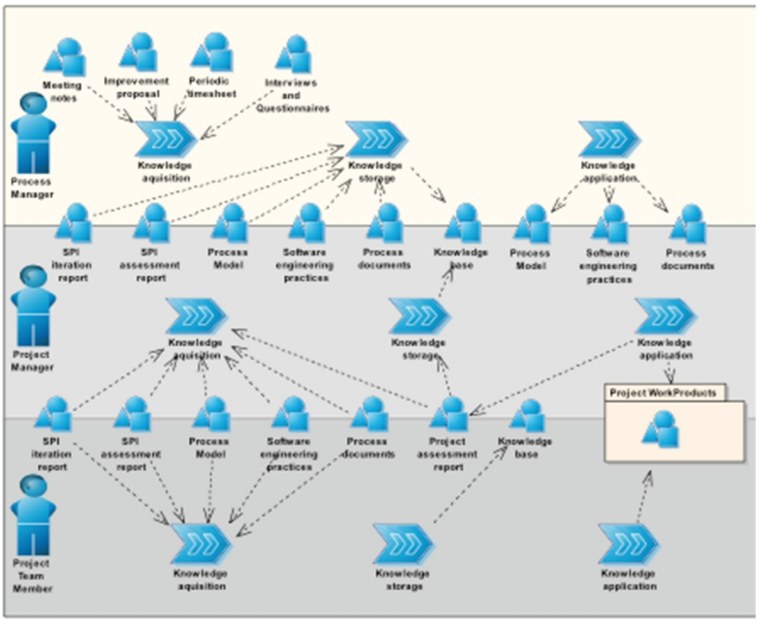

Fig. 5. Knowledge Management View

A KM system, which may be as simple as a story or as complex as an expensive computer program, captures a snapshot of the person's knowledge representation. Others may make use of the knowledge representation "snapshot" by using the story or tapping into the KM system and then combining it with their prior knowledge. This in turn forms a new or modified knowledge representation. This knowledge representation is then applied to solve a personal or business need, or explain a phenomenon. The main goal is to connect knowledge providers with seekers

concerning software processes. Figure 5 presents the main roles, activities and work products involved in the Knowledge Management discipline.

4 Conclusion

In the state of the art, several problems were identified concerning SPI programs based on standard SPI models. ProPAM is presents as an alternative approach to SPI focused on gaps and problems identified on existing SPI standards.

A case study was performed in a small organization without conditions to accomplish a maturity assessment using CMMI. The main goal of the case study was to give us insight into how SPI initiatives can best suit organizational goals and also showed us the impact on the organization and the strengths and weaknesses of ProPAM. Since the focus of the paper is to present ProPAM as an alternative agile SPI approach, the description of the case study is out-of-scope.

As final conclusion, the prescriptive nature of traditional approaches (such as CMMI) and costs necessary to implement SPI programs are the main reasons for further research on SPI based on project's experience. Namely, SPI models must address the importance of using the experience of software teams as an important source to SPI. Another gap observed was the deficient alignment between the process and projects. Nevertheless, the contribution of this work wasn't just an approach to align process and project specifications; we also proposed a mechanism to analyze evolution based on the changing needs of the software development organization.

Acknowledgments. This work was partially supported by national funds through FCT (Foundation for Science and Technology) through projects UID/SOC/04020/2013 and UID/CEC/50021/2013.

References

1. O. Salo, "Improving Software Development Practices in an Agile Fashion," *Book Improving Software Development Practices in an Agile Fashion*, Series Improving Software Development Practices in an Agile Fashion, Agile-ITEA, 2005, pp. 8.
2. M. Staples, et al., "An exploratory study of why organizations do not adopt CMMI," *Journal of Systems and Software*, vol. 80, no. 6, 2007, pp. 883-895.
3. G. Coleman and R.V. O'Connor, "Investigating Software Process in Practice: A Grounded Theory Perspective," *Journal of System and Software* vol. 81, no. 5, 2008, pp. 772–784.
4. S. Basri and R. O'Connor, "Organizational Commitment Towards Software Process Improvement An Irish Software VSEs Case Study," *Proc. 4th International Symposium on Information Technology 2010 (ITSim 2010)*, 2010.
5. SEI, *CMMI for Development, Version 1.3*, S. E. I.-C. M. University, 2010.
6. K. Weber, et al., "MPS Model-Based Software Acquisition Process Improvement in Brazil," *Proc. 6th Quality of Information and Communications Technology (QUATIC 2007)*, IEE Computer Society, 2007, pp. 110-122.

7. M. Mirna, et al., “The results analysis of using MIGME-RRC methodology for software process improvement,” *Proc. 6th Iberian Conference on Information Systems and Technologies (CISTI)*, 2011.
8. ISO/IEC, *15504-2 Information technology - Software process assessment – Part 2: A reference model for processes and process capability*, ISO/IEC TR 15504-2, July, 1998.
9. C.Y. Laporte, et al., “A Software Engineering Lifecycle Standard for Very Small Enterprises,” *Proc. Software Process Improvement, 15th European Conference, EuroSPI 2008*, Springer, 2008, pp. 129-141.
10. D. Goldenson and J.D. Herbsleb, *After the Appraisal: A Systematic Survey of Process Improvement, its Benefits, and Factors that Influence Success*, 1995.
11. N. Baddoo and T. Hall, “De-motivators for software process improvement: an analysis of practitioners’ views,” *Journal of Systems and Software*, vol. 66, no. 1, 2003, pp. 23-33.
12. K. Beck, *Extreme Programming Explained: Embrace Change*, Addison Wesley, 2004, p. 224.
13. L. Rising and N. Janoff, “The Scrum software development process for small teams,” *IEEE Software*, vol. 17, 2000, pp. 26-32.
14. M. Verlage, “Multi–view modeling of software processes,” *Proc. Proceedings of the Third European Workshop on Software Process Technology*, Springer–Verlag, 1994, pp. 123–126.
15. P.V. Martins and A.R. Silva, “ProPAM: discussion for a new SPI approach”, *Software Quality Professional Journal*, 11(2), American Society for Quality, 2009.
16. P.V. Martins and A.R. Silva, “ProjectIT-Enterprise: a Software Process Improvement Framework”, Industrial Proceedings of the 17th EuroSPI Conference, Grenoble, France, 2010, pp. 257-2.66.
17. S. Dissmann, et al., “Integration of Software Process Management and Development History Recording,” *Proc. Second Asia-Pacific Software Engineering Conference (APSEC'95)*, 1995, pp. 468-.
18. D. Rico, *Using Cost Benefit Analyses to Develop Software Process Improvement (SPI) Strategies*, A. E. S. D. ITT Industries, Defense Technical Information Center (DTIC)/ AI, 2000.