Approaches for Measuring Software Project Management Effectiveness

Kadir Alpaslan Demir, James Bret Michael, John S. Osmundson
Department of Computer Science
Department of Information Sciences
Naval Postgraduate School
Monterey, CA, 93943, USA
{kdemir,bmichael,josmundson}@nps.edu

Abstract

Evaluating, monitoring, and improving the effectiveness of project management can contribute to successful acquisition of software systems. According to our empirical studies on a range of software development projects, half of the variation in software project success ratings may be explained by project management effectiveness measurements. To improve the quality of the management, to focus our improvement efforts on the right issues, and to increase the odds of success in software projects, it is essential to measure the effectiveness of software project management. In this paper, we introduce four different approaches for measuring software project management effectiveness. Two of these approaches discussed in this paper were successfully used in the development of a metric for software project management effectiveness. The contribution of this paper is the introduction of these approaches for guiding the researchers for the development of other project management metrics.

1. Introduction

There are various studies reporting the success and failure rates of software projects [2,3,4]. Even with the lowest failure rates reported, the software projects are significantly failing when compared to projects in other fields. In [5], current project management issues in leading project-based industries are listed. Among nine industries, in only software industry column, overruns and poor performance is explicitly listed as an issue among others. The average software project is likely to be six to 12 months behind schedule and 50 to 100 percent over budget [6]. One would expect that our record in software projects should have been much better with all the latest technical advancements in the software engineering body of knowledge. In order to achieve better software project outcomes, we also need significant advances in the software project management field. Therefore, proposals and discussions for applicable and viable theories, models, tools and practices in software project management are important steps in achieving better project outcomes.

Ineffective software project management is among the main reasons for the failures in software projects [7]. In addition, effective project management is a determinant in the success of software projects [7]. DeMarco and Lister state that “For overwhelming majority of the bankrupt projects we studied, there was not a single technological issue to explain the failure.” [8]. Robertson et. al. emphasize that “In several decades of project experience, we have never seen a project fail for technical reasons. It has always been human failures that have caused otherwise good projects grind to a halt.” [9]. Various other studies, researchers and practitioners report similar issues regarding the importance of software project management in the success and failure of software projects [10, 11].

According to Boehm, poor management can increase software costs more rapidly than any other factor. COCOMO [12], a method for software project cost and effort estimation developed by Barry Boehm and his colleagues, does not include project management related factors. Therefore, in COCOMO II [13], the estimation model incorporates some project management related factors such as PCON (personnel continuity) and PMAT (process maturity). We believe, in order to keep the rate of the software cost overruns and schedule slippages down, measuring and therefore improving the quality of project management is an enabler. In addition, such project management metrics can be incorporated to cost estimation techniques yielding better estimates.

According to Morris, “One of the major areas of project management development over the next years, I believe, will be establishing and refining inter-industry metrics for quantifying performance improvements. Much of this work will be IT-related.” [14]. Hyvari investigates the effectiveness of project management based on four different factors [15]. These factors are organizational structures, technical competency, leadership ability, and the characteristics of an effective project manager. He does not state the reasoning for selecting these factors and whether this is a complete list or not.

According to our empirical studies on a range of software development projects, half of the variation in software project success ratings may be explained by project management effectiveness measurements [1]. To improve the quality of project management, to focus our improvement efforts on the right issues, and to increase the odds of success on software projects, it is essential to measure the effectiveness of software project management. Project management is a complex endeavor and development of a metric for project management effectiveness is clearly not an easy task [1]. However, measurement and evaluation of management effectiveness in software projects opens up a lot of opportunities for improvement.

The motivation for this study was the lack of literature on approaches for measuring the effectiveness of software project management. During the development of a metric for software project management effectiveness [1], it became clear that the first task should be identifying possible solution alternatives. This study is the result of such endeavor.

In this paper, we introduce four approaches for measuring the effectiveness of software project management. We further discuss each approach and present examples of the existing implementations. Two of the approaches discussed in this paper were successfully used in developing a software project management effectiveness metric [1]. The contribution of the paper is the guidance for the development of project management metrics.

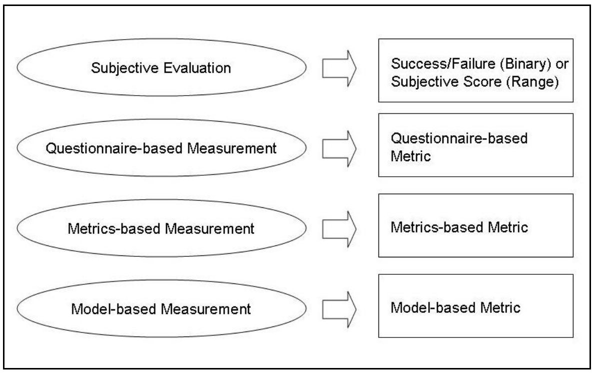

Figure 1. Approaches for Software Project Management Effectiveness Measurement

2. Discussion of Approaches

We have identified four different approaches that can be used in the development of methods to measure the effectiveness of software project management. Figure 1 shows these four approaches and corresponding metric types. Each of these approaches is discussed in detail.

2.1. Subjective Evaluation

In this approach, the project participant's perception is used in the evaluation of the project management. This participant may be the project manager, the technical manager, or the developers. Since it is based on the perception of the participant, this is a subjective evaluation. In this approach, the project participant is simply asked to categorize the project management as a success/failure or to rate the project management effectiveness based on a scale. This approach is the simplest one and used in some studies. For example, Osmundson et. al. [16] requested the project managers and project developers rate the project's success based on a scale from 0 to 10 in their study. In another study, Verner and Evanco investigated the project management practices leading to success in in-house software development projects [17]. They analyzed 42 successful and unsuccessful projects based on the senior software practitioners' categorization of their projects. In his doctoral dissertation, Procaccino used the same approach and his study is based on the view of software practitioners [18].

It is important to point out that even though such an approach is subjective; it is hard to disregard the validity (to some extent) of the project participant's perception. The practitioners have a sense of what the best practices are and if those are followed or not. However, as Pinto and Slevin [19] pointed out there is a significant risk of mislabeling a project as a success or failure without a well-established set of success criteria. This risk is more significant when the study compares the successful and failed projects based on the subjective evaluation approach. Because when the project is in fact a failure and the participant mislabels it as a success, then this evaluation skews both results such as boosting the success rate and decreasing the failure rate.

Another important consideration is that the measures resulting from this approach do not provide any insight on how to improve the management of the project. Just labeling a software project as a success or a failure without understanding the causes of it, has limited use for practitioners and researchers.

2.2. Questionnaire-based Measurement

In this approach, the measurement of project management effectiveness is based on the evaluation of responses to a questionnaire. Questionnaire-based evaluations are common in management and organizational sociology study areas (for example [20, 21, 22, 23, 24]) because abstract concepts such as teamwork, organizational commitment, communication, leadership etc. are hard to quantitatively analyze.

This approach has been used in the development of a quality management metric for software development [16]. In the study by Osmundson et. al., a questionnaire was developed to investigate which best project management practices are followed to what extend in a software project. Then, based on the responses to the questions, the quality of the project management is measured. They also compared the resulting metric (QMM) with a metric gathered via subjective evaluation discussed in the previous section. The questionnaire investigates four important areas of software project management. They are requirements management, project planning and estimation, risk management and people management [25]. People management is further divided into four areas: Human resources, leadership, communication, technical competency of the program manager. The complete questionnaire instrument included 457 questions. The QMM metric is based on a scale from 0 to 10, 0 being the lowest quality score, and 10 being the highest quality score. The importance of the QMM study is the focus on the development of a metric for the quality or effectiveness of project management in software projects.

Demir has successfully used this approach in developing a metric for software project management effectiveness [1]. While, QMM study is similar with this research, Demir's research provides a theoretical framework while incorporating a more comprehensive questionnaire with investigating 15 project management areas. These areas include communication, leadership, teamwork, quality engineering, configuration management etc.

COCOMO II incorporates a process maturity factor (PMAT) as a scale factor to the effort estimate [13]. It is important to note that scale factors affect the effort estimate exponentially. In COCOMO II, this PMAT factor is determined using one of two methods [26]. The first method is based on the SW-CMM rating of the organization when there is one. The second method is used when the organization does not have a SW-CMM rating. The second method uses another rating (Equivalent Process Maturity Level – EPML) which is based on the percentage of compliance for each key process area goal in SW-CMM model. This compliance is (EPML rating) evaluated via the responses to a questionnaire derived from 18 key process areas.

2.3. Metrics-based Measurement

Another approach for measuring the effectiveness of software project management is via the use of other software metrics. For example, metrics such as the number of defects over time, software complexity, requirements stability, staff turnover rate etc. can be used as inputs for a metric model for software project management effectiveness metric. This type of measurement is in fact an indirect measurement. When complex attributes are measured in terms of simpler sub-attributes, this measurement is indirect [27]. Many effort predictions use several levels of indirect measurement [27]. Erdogmus presents a cost-effectiveness indicator for software development. He uses base measures such as nominal output, production effort, rework effort, issue count, staffing profile to derive a breakeven multiple as an indicator aggregating productivity, quality, and staffing needs [28]. This is a good example for this approach in a different context. Wohlin and Maryhauser provide a detailed method for assessing software project success using subjective evaluation factors [29].

In 2007, Systems Engineering Leading Indicators Guide version 1.0 [51] resulted from a project by the Lean Aerospace Initiative, International Council on Systems Engineering (INCOSE), Practical Software and Systems Measurement (PSM), and Systems Engineering Advancement Research Initiative (SEARI). This guide mainly focuses on presenting leading indicators for evaluating the goodness of systems engineering on a program. The guide explains a leading indicator as "a measure for evaluating the effectiveness of how a specific activity is applied on a program in a manner that provides information about impacts that are likely to affect the system performance objectives." Some of the leading indicators are requirements trends, system definition change backlog trend, risk handling trends, and work product approval trends. While the guide lists and explains a set of leading indicators, it does not provide a guide, method or framework to combine these measures to form an effectiveness measure of systems engineering.

We provide a metric model for such measurement to guide the future researches. The model is presented below:

$$SPMEM = Measurement\_function(\sum_{i=1}^{n} w_i m_i)$$

In the model above, m is a metric that is hypothesized to relate to the metric for management quality, which is denoted by $SPMEM$. There can be n number of metrics. There may also be only one metric and in that case n equals to 1. Examples of such metrics may include programmer productivity, defect reduction rate, certain earned value metrics (EVM) etc. w_i is the weight associated with a certain metric, m_i. Such weights may be required since different metrics may relate to the resulting management quality metric differently. Then these metrics are combined via a measurement function based on the hypothesized metric model.

Above we presented a generalized metric model. Development of a management effectiveness or quality metric for software projects using this approach requires significant research ideally supported with extensive empirical studies.

2.4. Model-based Measurement

In this approach, the metrics for effectiveness or quality of the management are derived from models of management of software projects. Currently, this approach is also conceptual and there are no examples implemented. There has not been any attempt to measure the management effectiveness of software projects based on a model of project management.

For quite some time, researchers are focused on developing software development life-cycle methodologies. There are many examples of methodologies such as waterfall, spiral, win-win, rapid prototyping, agile development, SCRUM etc. There is also a field called software process research within the software engineering discipline. Software process research started back in 1980’s through a series of workshops and events. Due to many software application failures, researchers are focused on improving the software process. The assumption is that there is a direct correlation between the quality of the software process and the quality of the software application developed. A good example in the software process research is the development of the CMM series models. An area of software process research is software process modeling. There are a number of Process Modeling Languages (PMLs) developed [30]. Some examples are Process Interchange Format (PIF) [31,32], Process Specification Language (PSL) [33], Unified Process Model (UPM) [34], Core Plan Representation (CPR) [35], Workflow Management Coalition Process Definition (WfMC) [36], Architecture of Integrated Information Systems (ARIS) [37]. A review of these PMLs can be found in [38]. In June 2005, Business Process Management Initiative (BPMI) and Object Management Group (OMG) merged their activities and formed the Business Modeling & Integration (BMI) Domain Task Force (DTF). They have developed various standard proposals for different views of process management such as Business Motivation Model (BMM) specification [39], and Business Process Definition Metamodel (BPDM) [40]. Fuggetta [30] pointed out that few (if any) of the proposed PMLs and related Process-centered Software Engineering Environments (PSEE) have been transferred into industrial practice. Fuggetta states that the goal should be to ease the adoption of PMLs. Most of the PMLs are heavily technical and formal. Since these PMLs were not widely adopted, we do not have actual project data based on models developed with these languages. As a result, Pinto stresses the importance of modeling the business, technical, financial, environmental, and other dimensions of the project before committing any significant sources or even before the go-ahead [41]. Jaafari provides a simplified highest-level representation of a project model and lists the ideal requirements for a project model [42]. He stresses that we still have a long way to go in realizing such sophisticated modeling systems. We have developed a simple, visual and formal modeling language called PROMOL for modeling project management [43]. This modeling tool achieves most of the ideal requirements listed by Jaafari. According to Demir, as presented with a theory of project management [1], there are two main concepts in the core of project management. This theory and the project management modeling language, PROMOL, is explained in [42]. Furthermore, the applicability and the scalability of the theory and PROMOL are shown with examples in [49,50]. These two main concepts in project management are activities and entities. An activity is a named process, function or task that occurs over limited time. An entity is something that has a distinct, a separate existence, though it does not need to be a material existence. These two concepts can be used in modeling project managements. Then, the quality or effectiveness of these activities and entities in a project management model can be used as inputs for a metric model for effectiveness of project management. A high-level metric model may be formulated as follows:

$$SPMEM = Measurement\_function(\sum_{i=1}^{m} qa_i + \sum_{j=1}^{n} qe_j)$$

In the metric model above, qa_i is the quality of an activity and qe_i is the quality of an entity. These activities and entities are the components of a project management model. There can be m number of activities and n number of entities in the model that is of interest as inputs for the $SPMEM$ metric model. The measurement function is a function that combines the quality measures of activities and entities. This function is specific to the metric model and it is defined in the metric model. Different metric models may require quite different measurement functions. It is important to emphasize that there can be a number of variations of this high-level model. Examples of these variations may be including only activities, or including only entities or basing the metric model to a specific life-cycle development model and deriving the activities and entities from this life-cycle development model.

The success of the model-based measurement will be highly dependent on the representation capability of the project management model. When these project management models are far from satisfactory, then the resulting metric will likely be unsatisfactory.

3. Conclusions and Future Work

According to Evans, Abela and Beltz, the first characteristics of dysfunctional software projects is failure to apply essential project management practices [44]. This is derived from 841 risk events in 280 software projects. 480 out of 841 risk events (57%) in software projects are due to not applying essential project management practices. Jones reports that an analysis of 250 software projects between 1995 and 2004 reveals six major areas effective in successful projects and inadequate in failing projects [7]. They are project planning, project cost estimating, project measurements, project milestone tracking, project change management, project quality control. All of these areas are related to software project management. These studies clearly show the importance of project management in achieving software project success. Therefore, project management metrics are the keys to rationally focus and substantiate the management improvement efforts.

It is important to note the recognized work by Basili and Rombaugh on the Goal/Question/Metric (mostly known as GQM approach) approach for development of software metrics [45]. They provide an overall approach on how to develop metrics. First, it is very important to define the goal of the measurement activity. This sets up the context for the measurement. Second, we have to find the right questions for identifying the metrics that are going to be used in the measurement effort. Third, we have to choose or develop the right metrics for achieving the goal. The GQM approach is completely applicable to all of the approaches presented here. The goal referred in GQM is already defined via the context and it is measuring the quality or effectiveness of management of a software project. These approaches help us to refine and ask the right questions. The examples and high-level models presented in the previous section guide us in identifying and combining the necessary metrics.

In [1], we successfully used the first two approaches in developing a metric for gauging the effectiveness of software project management. Again, in another research, the same first two approaches successfully guided the development of another similar metric called Quality Management Metric (QMM) [46] [47] [48].

In this paper, we aimed for:

- A good review of the literature related to the effectiveness measurement of project management
- The introduction of four different approaches for effectiveness measurement for software project management
- The guidance for the development of project management metrics via high-level metric models

Future work may include the development of other project management metrics with the other two approaches and comparing these metrics with the metrics developed in [1] and [46]. Comparison of metrics developed with different approaches may provide direction in improving these metrics for achieving better accuracy and precision in the measurements.

It should be noted that there may be other approaches in addition to the ones discussed here. Identification of other approaches is another line of future work.

4. Disclaimer and Acknowledgements

The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of any affiliated organization or government.

5. References

[1] K. A. Demir, *Measurement of Software Project Management Effectiveness,* PhD Dissertation in Software Engineering, Naval Postgraduate School, Monterey, CA, December, 2008.

[2] Contracting for computer software development, FGMSD-80.4, US General Accounting Office, Washington, DC, 1979.

[3] CHAOS, in: The Standish Group Report, Standish Group, West Yarmouth, MA, 1995

[4] K. E. Emam, and G. Koru, "A Replicated Survey of IT Software Project Failure Rates", *IEEE Software*, Vol. 25, No. 5, pp. 84-90, 2008.

[5] Slevin, D. P., Cleland, D. I., & Pinto, J. K. *The Frontiers of Project Management Research*, Project Management Institute, Pennsylvania, 2002.

[6] E. Yourdon, *Death March,* 2nd Edition, Yourdon Press, Pearson Education Inc., Publishing as Prentice Hall Professional Technical Reference, Upper Saddle River, NJ, 2004.

[7] C. Jones, "Software Project Management Practices: Failure versus Success", *Crosstalk,* Vol. 17, No. 10, October 2004.
[8] T. DeMarco, and T. Lister, *Peopleware: Productive Projects and Teams,* 2nd Edition, Dorset House Publishing Company, New York, NY, 1999.
[9] S. Robertson, and J. Robertson, *Requirements-Led Project Management,* Pearson Education Inc., Boston, MA, 2005.
[10] G. Weinberg, *Quality Software Management: Volume 3 Congruent Action*, Dorset House, New York, 1994.
[11] Defense Science Board, "Report of the Defense Science Board Task Force on Defense Software", Office of the Under Secretary of Defense for Acquisition and Technology, Washington, D.C. 20301-3140, http://www.acq.osd.mil/dsb/reports/defensesoftware.pdf, accessed in March 2008.
[12] B. W. Boehm, *Software Engineering Economics,* Prentice-Hall, Upper Saddle River, NJ, 1981.
[13] B. W. Boehm, C. Abts, A. W. Brown, S. Chulani, B. K. Clark, E. Horowitz, R. Madachy, D. Reifer, and B. Steece, *Software Cost Estimation with COCOMO II,* Prentice-Hall PTR, July 2000.
[14] P.W. G. Morris, *Chapter 1 in Project Management Handbook*, Editor J. K. Pinto, Project Management Institute, Jossey-Bass Inc., San Francisco, California, 1998, p. 23.
[15] I. Hyvari, "Project management effectiveness in project-oriented business organizations", *Int. Journal of Project Management,* Vol. 24, 2006, pp. 216-225.
[16] Osmundson, J. S., Michael, J.B., Machniak, M.J., Grossman, M.A., "Quality Management Metrics for Software Development", *Information and Management*, Vol. 40, 2003, pp. 799-812.
[17] Verner, J.M.; Evanco, W.M., "In-house software development: What project management practices lead to success", *IEEE Software*, Vol. 22, Issue 1, Jan.-Feb. 2005, pp. 86–93.
[18] J. D. Procaccino, "Quantitative Models for Early Prediction of Software Development Success: A Practitioner's Perspective", Doctoral Dissertation, Drexel University, October 2002.
[19] J.K. Pinto, Project Management Handbook, Project Management Institute, Jossey-Bass Inc., San Francisco, California, 1998, p. 379.
[20] B. B. Brown, *Employees' Organizational Commitment and Their Perception of Supervisors' Relations-Oriented and Task-Oriented Leadership Behaviors*, Dissertation for Doctor of Philosophy in Human Development, Virginia Polytechnic Institute and State University, Falls Church, Virginia, March 25, 2003.
[21] S. G. Baugh, and R. M. Roberts, "Professional and Organizational Commitment among Engineers: Conflicting or Complementing", *IEEE Transactions on Engineering Management,* Vol. 41, No. 2, May 1994.
[22] A. K. Paul, and R. N. Anantharaman, "Impact of people management practices on organizational performance: analysis of a casual model", *Int. Journal of HRM,* Vol. 14, No. 7, November 2003, pp. 1246-1266.
[23] D. C. Kinlaw, *Superior Teams: What they are and how to develop them,* Gower Publishing Ltd., Hampshire, England, 1998.
[24] R. Muller, "Determinants for external communications of IT project managers", *Int. Journal of Project Management*, Vol. 21, 2003, pp. 345-354.
[25] Machniak, M.J., *Development of a quality management metric (QMM) measuring software program management quality*, Master's thesis, Naval Postgraduate School, Monterey, CA, December 1999.
[26] B. K. Clark, "Quantifying the effects of process improvement on effort", *IEEE Software,* November/December 2000, pp. 65-70.
[27] Norman Fenton, and Shari Lawrence Pfleeger, Software Metrics: A Rigorous and Practical Approach, Revised Printing, PWS Publishing Company, MA, 1997.
[28] H. Erdogmus, "Cost-Effectiveness indicator for software development", *Proceedings of the International Symposium on Empirical Software Engineering and Measurement (ESEM 2007)*, Madrid, Spain. September 20-21, 2007.
[29] C. Wohlin, and A. von Mayrhauser, "Assessing Project Success using Subjective Evaluation Factors", *Software Quality Control*, Vol. 9, Issue 1, January 2001, pp 43-70.
[30] A. Fuggetta, "Software Process: A Roadmap", Proceeding of the Conference on the Future of Software Engineering, Limerick, Ireland, 2000, pp. 25-34
[31] J. Lee, M. Grunninger, Y. Jin, T. Malone, A. Tate, G. Gost and other members of the PIF Working Group, *The PIF Process Interchange Format and Framework Version 1.1,* May 1996.
[32] J. Lee, M. Grunninger, Y. Jin, T. Malone, A. Tate, G. Gost and other members of the PIF Working Group, "The PIF Process Interchange Format and Framework Version 1.2"*, The Knowledge Engineering Review*, Vol. 13, No. 1, pp. 91-120, Cambridge University Press, March 1998.
[33] C. Schlenoff, A. Knutilla, S. Ray, "A Robust Process Ontology for Manufacturing Systems Integration", *Proc. of the 2nd Int. Conference on Engineering Design and Automation,* Maui, Hawai, August 7-14, 1998.
[34] P. Kruchten, "Unified Process Model (UMP)- A Model of the Rational Unified Process", *Proc. of the International Process Technology Workshop,* Villard de Lans, France, September 1999.
[35] A. Pease, Core Plan Representation, version 4, November 1998.
[36] Workflow Management Coalition, Interface 1: Process Definition Interchange Process Model, WfMC TC-1016-P, November 1998.
[37] A.-W. Scheer, ARIS-Business Process Frameworks, 3rd Edition, Springer-Verlag, Berlin, 1999.
[38] E. Breton, and J. Bezivin, "An Overview of Industrial Process Meta-Models", *13th International Conference on Software & Systems Engineering and their Applications,* December 5-8, Paris, France, 2000.
[39] Object Management Group, Business Motivation Model Specification, version 1.0, September 2007.
[40] Object Management Group, Business Process Definition Metamodel, Beta 1, July 2007.
[41] J.K. Pinto, Project Management Handbook, Chapter 1, Project Management Institute, Jossey-Bass Inc., San Francisco, California, 1998, pp. 11.
[42] A. Jaafari, "Modeling of Large Projects", Chapter 13 in *The Wiley Guide to Managing Projects,* Edited by P.

W.G. Morris, and J. K. Pinto, John Wiley and Sons, Inc., Hoboken, New Jersey, 2004, pp. 301-302.
[43] K. A. Demir, and John S. Osmundson, “A Theory of Software Project Management and PROMOL: A Project Management Modeling Language", Technical Report, NPS-IS-08-006, Naval Postgraduate School, Monterey, CA, USA, March 2008.
[44] M. E. Evans, A. M. Abela, and T. Beltz, “Seven Characteristics of Dysfunctional Software Projects”, Crosstalk, Vol. 15, No. 4, April 2002, pp. 16-20.
[45] V.R. Basili, and H. D. Rombach, “The TAME Project: Towards Improvement-Oriented Software Environments”, *IEEE Trans. On Software Engineering*, Vol. SE-14, No. 6, June 1988, pp. 758-773.
[46] Osmundson, J. S., Michael, J. B., Machniak, M. J., & Grossman, M. A., “Quality Management Metrics for Software Development.” *Information and Management, 40*, pp. 799-812, 2003.
[47] M. J. Machniak, *Development of a quality management metric (QMM) measuring software program management quality*, Master’s thesis in Software Engineering, Naval Postgraduate School, Monterey, CA, 1999.
[48] M. A. Grossman, *Validation of a Quality Management Metric*, Master’s thesis in Software Engineering, Naval Postgraduate School, Monterey, CA, 2000.
[49] K. A. Demir, and A. Erguner, “Software Project Management Modeling with PROMOL”, Software Quality and Software Development Tools Symposium, Istanbul, Turkey, 9-10 October 2008
[50] A. Erguner, *A Novel Project Management Theory and Its Applicability,* M.S. Thesis in Information Technology Management, Naval Postgraduate School, Monterey, CA, March, 2008.
[51] *Systems Engineering Leading Indicators Guide Version 1.0*, Lean Aerospace Initiative (LAI), International Council on Systems Engineering (INCOSE), Practical Software and Systems Measurement (PSM), and Systems Engineering Advancement Research Initiative (SEARI), editors: G. Roedler, and D. H. Rhodes, 2007.