Available online at www.sciencedirect.com

SCIENCE DIRECT®

Information & Management 41 (2004) 279–288

INFORMATION & MANAGEMENT

www.elsevier.com/locate/dsw

An exploration of the relationship between software development process maturity and project performance☆

James J. Jiang[a,*], Gary Klein[b], Hsin-Ginn Hwang[c], Jack Huang[c], Shin-Yuan Hung[c]

[a]Department of Management Information Systems, University of Central Florida, Orlando, FL 32816-1400, USA
[b]University of Colorado, Colorado Springs, USA
[c]National Chung Cheng University, Jai-Yi Taiwan

Accepted 30 June 2003

Abstract

Software projects have a high rate of failure. Organizations have tried to reduce the rate through methodological approaches but with little perceived success. A model of software development maturity (the capability maturity model (CMM)) describes managerial processes that can be used to attack software development difficulties from the managerial control perspective at five maturity levels. Our study examined performance of projects in relation to the activities at these various levels of maturity. A survey of software engineers indicated that the activities associated with the managerial control of development related positively to project performance measures. However, not each level of maturity demonstrated observable benefits, indicating that greater caution is needed in the planning and implementation of the activities.

Keywords: Software development; Capability maturity model; Project performance

Software projects continue to grow more critical to the organizations that employ them. Still, software failures are frequent. According to Standish Group International [38], about 15% of all software development projects never deliver a final product. Software project problems cost US companies and government agencies an estimated US$ 145 billion annually. In large software development projects, more than 80% are excessively late and over budget [13,26]. One approach to combat the failure rate has been technical, with organizations introducing new design methods [4,25]. However, after significant resources are poured into software development methods, such as CASE and Rapid Application Development, projects are often still considered ‘‘run-aways’’ [2,5,28,39,40].

The managerial side of the software development project, meanwhile, is often conducted without adequate planning, with poor understanding of the overall development process, and a lack of a well-established management framework [36,37]. A formal addition to management practice was developed in the 1990s to help organizations along an evolutionary path to a more disciplined development process. The Software Engineering Institute (SEI), for the US Department of Defense (DoD), recommended a number of key software process improvement (SPI) areas. Later, these activities were formed into an evaluative framework called the capability maturity model (CMM) [33,34].

☆ For the Research Section of Information & Management.
* Corresponding author. Tel.: +1-407-823-4864;
fax: +1-407-823-2389.
E-mail address: jjiang@bus.ucf.edu (J.J. Jiang).

0378-7206/$ – see front matter © 2003 Elsevier B.V. All rights reserved.
doi:10.1016/S0378-7206(03)00052-1

This model requires a considerable amount of time and effort to implement and often needs a major shift in culture and attitude [7,19]. One study found that the median time for an organization to move up one level of the five-level CMM is between 21 and 37 months [16]. Over three-quarters of the organizations reported that implementing any key SPI activity took longer than expected. In fact, resources expended amount to billions of dollars every year in the US alone. In addition to these, an organization's culture can be adversely impacted by adding to its rigid bureaucracy and reducing creativity and freedom on the part of the developers [21]. Thus, given the costs and potential disadvantages of implementing the CMM, benefits must be evident to justify its continued use.

The CMM has already achieved wide interest and acceptance in the software industry. It has spread far beyond its origins and is now used by thousands of major organizations worldwide [12]. But is the CMM an appropriate model for guiding improvements in the software development process? Do the suggested SPI activities really apply to a variety of organizations? To our surprise, no empirical evidence examining the effects of the various suggested SPI activities on project performance can be found in the information system (IS) literature. Therefore, the focus of this study is on the relationship between SPI activities and software project performance. Specifically:

1. Is there a relationship between the implementation of the CMM's SPI activities and an organization's software project performance?
2. Are certain SPI activities more likely to influence the final project outcomes than other suggested SPI activities?
3. Which dimensions of project outcomes are more likely to be influenced by which SPI activities?

1. The software capability maturity model and software project performance

The CMM was originally developed by the Software Engineering Institute and it has been enhanced since then. It was primarily based on the experiences and extensive feedback of software practitioners and designed to assist the US Department of Defense in software acquisition. In September 1987, the SEI released a brief description of the process-maturity framework. It described an evolutionary software development process improvement path from an ad hoc, immature process to a mature, disciplined process. The CMM maps organization's software project

Table 1
Key process activities for CMM

Maturity levels	Characteristics	Key process activities
Level V: Optimizing	Process optimization	Process change management Technology change management Defect prevention
Level IV: Managed	Measuring quality of development process and its product	Software quality management Quantitative process management
Level III: Defined	Processes engineering	Peer reviews Intergroup coordination Software product engineering Integrated software management Training program Organization process definition Organization process focus
Level II: Repeatable	Basic project management	Software configuration management Software quality assurance Software subcontract management Software project tracking and oversight Software project planning Requirements management
Level I: Initial	Chaotic, few if any process	None

process on a five-level system. The five-levels are defined according to a set of activities described by the SPI areas. When combined into the five-level model, each level represents one of five stages of maturity. The key process areas are summarized in Table 1.

Project performance is viewed differently by each of the stakeholders in the system development effort. It is desirable to incorporate a breadth of success aspects when considering project performance. As such, project performance includes software engineering issues of efficiency and effectiveness, as well as organizational issues of control, communication, and organizational knowledge [20,22,31]. *Efficiency* is often considered to be measured by the quality of the software product, adherence to budgeted time and money, and cost of the software operation. *Effectiveness* is considered to be the applicability and adaptability of the software. The *organizational issues* involve the knowledge gained by the organization during development, the interpersonal relations maintained, and the ability to control the resources used by the project.

The CCM model has become influential as a means for software process improvement. The fundamental belief is that project performance can be improved by implementing the recommended SPI activities. The suggested software improvement areas include having competent people, establishing basic project management processes, documenting and standardizing engineering processes and organizational support, measuring and controlling product and process quality, and facilitating continuous process improvement. Thus, as an organization matures in the CMM model, the software process becomes better defined and more consistently implemented throughout the organization. As a result, many researchers and practitioners have argued that, through the CMM, managers can better monitor the quality of products and the process that produced them.

Limited evidence on the success of the CMM consists of case studies including Hughes Aircraft [17], Raytheon [10], and Tinker AFB [8]. These show that organizations adopting the CMM model tend to have substantially higher quality software, a faster cycle time, and higher development productivity. Herbsleb and Goldenson [15] found evidence, in a sample of 61 organizations, that software development process maturity is associated with better organizational performance. As these studies indicate, evidence is accumulating that CMM-based software process improvement is paying off.

A positive relationship between project success and software process management can be expected [23]. Managerial control refers to the manager's attempt to influence employees to behave in accordance with organizational procedures and goals. In the context of a software project, control-based activities by an organization could include activities such as defining and documenting how the work is to be done; assigning tasks to team members; establishing performance guidelines and standards through task feedbacks; and initiating corrective actions as necessary [14]—areas which CMM has identified.

Beath [6] emphasized the importance of management control on software project implementation. She argued that, for fairly simple projects, project managers need to establish an outcome-oriented governance structure between the project team and users. In a more complex project, project managers need to master a variety of governance techniques that can be matched to a variety of project characteristics. Along the same line, Keider [24], surveying 100 software professionals, found that the most critical project failure reason was a lack of management control of project development efforts.

Some management control researchers argued that managerial control can be established through process control: attempting to guide employees' behaviors [11,35]. According to this, the proposed CMM would establish a set of specific activities for project team members to conduct and allow software management to monitor and evaluate behavior. Furthermore, this view is consistent with the literature that finds that monitoring project progress against standards is positively related to project performance. Based upon managerial control theory and the limited empirical findings, we expect:

H1: There is a positive relationship between software development process management maturity activities and project performance.

2. Research methodology

2.1. Sample

Questionnaires were mailed to 1000 randomly selected IEEE Computer Society members.

These members are likely those familiar with the activities of the CMM and work in organizations that adopt some CMM activities for managing their software development. Postage-paid envelopes were enclosed with each questionnaire. All respondents were assured that their responses would be kept confidential. Of the 1000 initial surveys mailed in the Spring of 2001, a total of 103 responses were received. In order to increase the sample size, a second mailing to the same sample was conducted in the Summer of 2001. The response from both mailings totaled 160, for an overall response rate of 16%. Six questionnaires were eliminated due to missing data, leaving a final sample of 154.

Non-response bias occurs when the opinions and perceptions of the survey respondents do not accurately represent the overall sample to which the survey was sent. One test for non-response bias is to compare the demographics of early versus late respondents [1].

Table 2
Demographic information

Gender	
Male	137
Female	17
Position	
IS manager	41
Project leader	52
IS professional	51
Others	8
No response	2
The industry type of your company	
Service	76
Manufacturing	61
Education	8
Others	6
No response	3
Average IS project duration in your organization	
1 years and under	65
1–2 years	54
2–3 years	12
3–5 years	7
6 or more years	12
No response	4
Average size of IS project teams in your organization	
7 and under	86
8–15	47
16–25	8
26 and over	9
No response	4

The *t*-tests were computed on the means of key demographics (work experience, gender, recent project duration, and team sizes) to examine whether significant differences existed between early and late respondents. No significant difference was found; therefore, the two rounds of respondents were combined for further analysis.

The respondents consisted of software managers (27%), project leaders (34%), and software professionals (33%), About 42% worked in companies that had average size of 8–15 person or more in project teams. The sample included employees in firms from manufacturing (40%), service (49%), education (5%), and other industries. Demographic features of the sample population are shown in Table 2.

2.2. Constructs

2.2.1. Software process management maturity activity

The 38 items used to measure software process management maturity was adopted from Delkleva and Drehmer [9]. Items of this instrument describe key processes representing the CMM repeatable (level II: items 1–12), defined (level III: items 13–26) and managed (level IV: items 27 and up) maturity thresholds. A factor analysis indicated a close structure to the one proposed. Since few organizations have achieved level V, no items were employed at this level. Key phrases representing these items are in Table 3, full descriptions are available in the original source. The respondents were asked to evaluate the overall extent of each structure and procedure implemented in their organization's software projects. Each item was scored using a five-point Likert scale ranging from ''not at all'' (1) to ''extremely'' (5) and these were averaged across all relevant items to arrive at the scale measures.

2.2.2. Project performance

Project performance has often been defined as the extent to which the software development process has been undertaken as well as performance of the delivered system from the viewpoint of the users. It is important to study both the process performance and the product performance aspects, because even though the software delivered by the project may be of high quality, the project itself may have significantly

Table 3
Factor analysis of software process-maturity activities

Items	Project management process	Process engineering and organizational support	Product and process quality	Key word
C1				Software quality assurance
C2	0.51			Configuration control
C3	0.66			Formal management review
C4	0.56			Size estimated
C5	0.72			Software development scheduled
C6	0.65			Software cost estimated
C7	0.54			Profiles of software size
C8				Software design errors
C9				Software code and test errors
C10	0.63			Managers sign off
C11		0.84		Requirements change control
C12		0.74		Code changes controlled
C13				Software engineering process
C14			0.52	Developers training required
C15			0.57	Training review leaders
C16		0.66		Development standardized
C17		0.68		Standards documented, used
C18				Senior managers review
C19		0.66		Design review items tracked
C20		0.67		Code review items tracked
C21		0.55		Compliance with standards
C22		0.71		Design reviews conducted
C23		0.59		Design changes controlled
C24		0.65		Code reviews conducted
C25		0.50		SQA sample verification
C26				Adequacy of regression test
C27			0.60	Process metrics database
C28			0.50	New technology intro; managed
C29			0.45	Test coverage measured
C30			0.75	Review efficiency analyzed
C31			0.73	Design review data analyzed
C32			0.91	Code, test errors projected
C33			0.70	Error cause analysis
C34			0.48	Code review standards
C35			0.47	Software process assessed
C36			0.60	Design and code coverage
C37			0.93	Forecast remaining errors
C38			0.95	Design errors projected
Cronbach α	0.90	0.94	0.96	

Note: Only loadings >0.45 are shown.

exceeded time and cost projections. On the other hand, well-managed projects which adhere to cost and schedule may deliver poor systems. The project performance construct used in this study was adopted from Nidumolu. The specific items are identified in Table 4, the full descriptions may be found in the original source. The respondents were asked to evaluate each item based upon a five-point Likert-type scale ranging from "never" (1) (goal is never achieved) to "always" (5) (goal is always achieved). Five categories are represented by the items, including organizational learning, process controls, interpersonal communication quality, operational efficiency, and software flexibility.

Although the project process management maturity activity construct has been examined in prior studies, a

Table 4
Factor analysis project performance

Item	Learning	Control	Interaction quality	Operational efficiency	Flexibility
(1) Knowledge is acquired by your organization about use of key techniques	0.92				
(2) Knowledge is acquired by your organization about use of development techniques	0.84				
(3) Knowledge is acquired by your organization about supporting users' business	0.62				
(4) Effective control over project costs		0.88			
(5) Effective control over project schedules		0.79			
(6) Adherence to audit and control standards		0.71			
(7) Complete training provided to users			0.86		
(8) Quality communication between IS unit and users			0.82		
(9) Users' feeling of participation			0.79		
(10) Reliable software				0.83	
(11) Efficient cost of software operations				0.54	
(12) Wide range of outputs that can be generated and queries that can be answered				0.71	
(13) Efficient cost of adapting software to changes in business					0.90
(14) Rapid adapting of software to changes in business					0.96
(15) Efficient cost of maintaining software over lifetime					0.66
Cronbach α	0.80	0.81	0.84	0.80	0.88

Note: Only loadings >0.45 are shown.

principal components analysis (PCA) followed by varimax rotation was conducted. The PCA found three eigenvalues greater than one. Table 3 includes the results of the varimax rotation on the original 38 items constrained to three factors. The criteria used to identify, distinguish, and interpret factors were that a given item should load 0.45 or higher on a specific factor and have a loading no higher than 0.35 on other factors. Thirty-two items remained. The first factor is defined as basic project management activities and corresponds to CMM Level II. The second factor contains process engineering and organizational support activities, including items for code review, change controls, and development of standards. These items are specified in CMM Level III. The third factor is defined as product and process quality and equates to CMM Level IV. These three factors explained 64% of the total variance. The Cronbach α-test [32] suggested reasonable reliability for the scales of basic project management process ($\alpha = 0.90$), process engineering and organizational support ($\alpha = 0.94$), and product and process quality ($\alpha = 0.96$). The descriptive statistics of each factor of this construct are given in Table 5.

Table 5
Descriptive statistics of the constructs

	Learning	Control	Interaction quality	Operation efficiency	Software flexibility	Overall project performance	Project management activities	Process engineering activities	Product and process quality	Overall software maturity level
Mean	3.66	2.76	3.01	3.50	3.27	2.82	2.89	3.08	2.29	3.02
S.D.	0.86	0.92	0.91	0.79	0.89	0.93	0.99	1.01	0.99	0.95
Median	3.67	2.73	3.00	3.67	3.33	2.78	2.86	3.18	2.08	3.08
Skewness	−0.41	0.20	−0.04	−0.27	−0.14	−0.01	0.12	−0.17	0.70	−0.14
Kurtosis	−0.20	−0.76	−0.69	−0.31	−0.64	−0.51	−0.71	−0.94	−0.28	−0.67

Table 6
Regression results

Dependent variable	Independent variable	Coefficient	P-value
(1) Project performance	Software development maturity	0.39	0.0001
(2) Project performance	Project management process	−0.03	0.7500
	Process engineering and organizational support	0.26	0.0036
	Product and process quality	0.16	0.0711

Similarly, a factor analysis with varimax rotation was conducted for the project performance items. Adopting the eigenvalue greater than one rule, five factors were extracted. A subsequent varimax rotation constrained to five factors yielded the results. Again, the criteria used to identify, distinguish, and interpret factors were that a given item should load 0.45 or higher on a specific factor and have a loading no higher than 0.35 on other factors. The five factors matched exactly with the a priori structure. The Cronbach α-values for basic learning, control, interaction quality, operational efficiency, and software flexibility are 0.80, 0.81, 0.84, 0.80, and 0.88, respectively.

3. Data analysis and results

To test the hypothesis, a regression analyses was conducted by taking project performance as the dependent variable and activity level as the independent variable. The results are shown in Table 6. The *P*-value (0.001) indicated that there was a significant relationship between project performance and software development maturity levels. This indicates that the extent of the CMM specified SPI activities implemented by organizations has a positive relationship (0.39) with project performance, as measured by both the process performance and the product performance.

To further examine the relationships between the specific SPI activities and project performance, we conduct another regression by breaking the overall software maturity construct into the three levels of activities—project management process, process engineering and organizational support, and product and process quality. Process engineering and organizational support activities (CMM-based Level III recommended activities) are significantly related to project performance (0.26) in terms of predictive ability. Product and process quality related suggested activities are marginally significant. On the other hand, the basic project management process activities (CMM-based Level II activities) were not significantly related to project performance in the regression analysis. These results suggest that organizations may not experience much benefit until they reach Level III maturity. The basic project management process activities may be a necessary foundation for project success, but not in delivering observable benefits, certainly not as measured in this study.

Lastly, we examine the relationships between the different project performance dimensions and the SPI levels of activities. A total of five independent regressions were conducted, one for each category of performance as the dependent variable. Each regression uses the three activity categories as the independent variables. The results are shown in Table 7. The performance dimensions of learning, control, operations efficiency, and flexibility were all positively related to process engineering and organizational support either significantly ($P < 0.05$) or marginally ($P < 0.10$). Likewise, control, interaction quality, and flexibility are positively related to product and process quality. Only flexibility is related to project

Table 7
Regression coefficients of project performance and SPI activities

	Learning	Control	Interaction quality	Operation efficiency	Flexibility
Project management process	0.05 (0.69)	0.06 (0.58)	0.03 (0.82)	−0.05 (0.64)	−0.22 (0.08)
Process engineering and organizational support	0.22 (0.07)	0.34 (0.00)	0.10 (0.42)	0.30 (0.00)	0.32 (0.01)
Product and process quality	0.07 (0.56)	0.20 (0.07)	0.23 (0.07)	0.07 (0.54)	0.22 (0.08)

Note: *P*-values in parentheses.

management processes, and this in a marginal negative direction. This latter result should not be surprising, as the rigid activities that provide a strong management process add to the bureaucracy that stifles flexibility.

4. Conclusions

CMM was originally designed to help characterize the maturity of software practices, guide a program of continuous process and workforce development, set priorities for immediate actions, and establish a culture of software development process quality excellence. CMM assumes that the quality of a software project outcome depends on the extent of suggested activities actually implemented by an organization [18]. Today, the CMM has spread far beyond its original application area and is widely used by software organizations in the US and around the globe.

Our study examined the relationship between CMM software process development activities and project performance. From the responses of 154 experienced software project developers, the analysis confirmed that software process management maturity is positively associated with project performance. This result is largely consistent with the many individual cases reported in [27]. To software managers, this result suggests that CMM, in general, could be a useful guide to improving their current state of software processes in order to improve project performance.

Results indicate that project performance is most related to the process engineering and organizational support activities of the CMM (Level III) but that product and process quality activities (Level IV) also have a positive relationship with project performance. On the other hand, basic project management process activities (Level II) were not significant at all. Organizations therefore need to realize that benefits may not be reached until they achieve Level III. This requires a great amount of time and money before benefits can be realized. Also, strong relations to benefits seem to tail off after Level III. Not all organizations may wish to pursue the CMM under these conditions.

Further relationships between the various specific project performance criteria and the SPI activities were explored. Process engineering and organizational support activities are positively associated with learning, control, interaction quality, and software flexibility. Interestingly, adherence to basic project management activities was negatively related to software flexibility. However, software flexibility can be improved with Levels III and IV activities. Furthermore, the quality of the interaction between users and IS staff was positively related to product and process quality related activities (Level IV) but not activities in other levels.

These insights may prove crucial to organizations that seek to follow the CMM model for their software development process improvements. Identifying strengths and weakness in an organization's current software development processes against a community standard is a necessary first step to build consensus around the fundamental software process development problems of their organization. Organizations then set priorities for their improvements so the resources can be effectively allocated to a few vital areas and activities. Planning sufficient attainment of the levels in CMM is important in realizing any benefits. An expectation that certain fundamentals must be in place to implement the higher level activities is important in planning for realization of the benefits. Furthermore, achieving higher levels of software development process maturity requires a long-term commitment to continuous process improvement. It may take organizations years to achieve the next level of maturity and to realize the benefits.

The CMM focus is on "what" organizations should do and not "how" they should do it. Sometimes the practices follow strict organizational requirements, at other times it is left to individuals who exercise autonomy to determine what actions are required and how to execute these activities [29,30]. Some studies support the concept that technical project performance can be improved if team members engage in higher levels of self-control as opposed to rigid organizational control [3,41]. This is one example of how the CMM provides "what to do" but allows flexibility on the part of the team members in "how to" accomplish their tasks.

References

[1] J.S. Armstrong, T.S. Overton, Estimating non-response bias in mail survey, Journal of Marketing Research 15, 1977, pp. 396–402.

[2] U. Apte, S.C. Sankar, M. Takur, E.J. Turner, Reusability-based strategy for development of information systems, MIS Quarterly 14 (4), 1990, pp. 421–433.
[3] L. Bailyn, Autonomy in the Industrial R&D Lab, MIT Press, Cambridge, MA, 1984.
[4] S. Bandinelli, A. Fuggetta, Modeling and improving an industrial software process, IEEE Transaction on Software Engineering 21 (5), 1995, pp. 440–454.
[5] H. Barki, S. Rivard, J. Talbot, Toward an assessment of software development risk, Journal of Management Information Systems 10 (2), 1993, pp. 203–223.
[6] C.M. Beath, Managing the User Relationship in Management Information Systems Projects: A Transaction Governance Approach, Unpublished Ph.D. dissertation, Graduate School of Management, UCLA, Los Angeles, CA, Summer 1986.
[7] E.P. Brooks, No silver bullet: essence and accidents of software engineering, IEEE Computing 20 (4), 1987, pp. 10–19.
[8] K.L. Butler, The economic benefits of software process improvement, Cross Talk, July 1995, pp. 14–17.
[9] S. Delkleva, D. Drehmer, Measuring software engineering evolution: a rash calibration, Information Systems Research 8 (1), 1997, pp. 95–104.
[10] R. Dion, Process improvement and the corporate balance sheet, IEEE Software 10, 1993, pp. 28–35.
[11] K.M. Eisenhardt, Control: organizational economic approaches, Management Science 31 (2), 1985, pp. 134–149.
[12] B. Fltzgerald, T. O'Kane, A longitudinal study of software process improvement, IEEE Software 16 (3), 1999, pp. 37–45.
[13] R.L. Glass, Software Runaways, Prentice-Hall, New York, NY, 1998.
[14] J.C. Henderson, S. Lee, Managing I/S design teams: a control theories perspective, Management Science 38 (6), 1992, pp. 757–777.
[15] J.D. Herbsleb, D.R. Goldenson, A system survey of CMM experience and results, Proceedings of ICSE 18, 1996, pp. 323–330.
[16] J.D. Herbsleb, D. Zubrow, D.R. Goldenson, W. Hayes, M. Paulk, Software quality and the capability maturity model, Communication of the ACM 40 (6), 1997, pp. 30–40.
[17] W.S. Humphrey, T.R. Snyder, R.R. Willis, Software process improvement at Hughes Aircraft, IEEE Software 8 (4), 1991, pp. 11–23.
[18] W.S. Humphrey, Managing the Software Process, Software Engineering Institute, The SEI Series in Software Engineering, Addison-Wesley, New York, NY, 1989.
[19] C.W. Ibbs, Y.H. Kwak, Assessing project management maturity, Project management Journal 31 (1), 2000, pp. 32–43.
[20] J.J. Jiang, W. Muhanna, G. Klein, User resistance and strategies for promoting acceptance across system types, Information & Management 37, 2000, pp. 25–36.
[21] T. Jones, Managing the behavior of people working in teams—applying the project management method, International Journal of Project Management 13 (1), 1995, pp. 47–53.
[22] M.C. Jones, A.W. Harrison, IS project team performance: an empirical assessment, Information & Management 31, 1996, pp. 57–65.
[23] B. Katz, N. Lerman, Building a three-dimensional work breakdown structure, Data Base, 1985, pp. 14–26.
[24] S.P. Keider, Why systems development projects fail? Journal of Information Systems Management 1 (3), 1984, pp. 33–38.
[25] J.P. Kuilboer, N. Ashrafi, Software process and product improvement: an empirical assessment, Information and Software Technology 42, 2000, pp. 27–34.
[26] K.R. Linberg, Software development perceptions about software project failure: a case, The Journal of Systems and Software 49 (12), 1999, pp. 177–192.
[27] W.H. Lipke, S. Rosenbaum, Software improvements in an international company, in: Proceedings of the 15th International Conference on Software Engineering, 1993, pp. 212–220.
[28] F.W. McFarlan, Portfolio approach to information systems, Harvard Business Review 59 (5), 1981, pp. 142–150.
[29] C.C. Manz, H.P. Sims, Organizations, Wiley, New York, NY, 1980.
[30] P.K. Mills, Self-management: its control and relationship to other organizational properties, Academic Management Review 8 (3), 1983, pp. 445–453.
[31] S.R. Nidumolu, The effect of coordination and uncertainty on software project performance: residual performance risk as an intervening variable, Information Systems Research 6 (3), 1995, pp. 191–219.
[32] J.C. Nunnally, Psychometric Theory, McGraw-Hill, New York, NY, 1978.
[33] M. Paulk, B. Curtis, M.B. Chrissis, C.V. Weber, Capability maturity model for software: version 1.1, IEEE Software 10 (6), 1993, pp. 18–27.
[34] M. Paulk, C.V. Weber, S.M., Garcia, M.B. Chrissis, M. Bush, Key Practices of the Capability Maturity Model, Version 1.1, CMU/SEI-93-TR-25, Software Engineering Institute, Pittsburgh, PA, February 1993.
[35] K.D. Peterson, Mechanisms of administrative control over managers in educational organizations, Administrative Sciences Quarterly 29 (4), 1984, pp. 3–14.
[36] A. Rai, H. Al-Hindi, The effects of development process modeling and task uncertainty on development quality performance, Information & Management 37, 2000, pp. 335–346.
[37] K. Schwalbe, Information Technology Project Management, Course Technology, New York, NY, 2000.
[38] The Standish Group, Chaos, Standish Group Report, 1995.
[39] H.E. Thomson, P. Mayhew, The software process: a perspective on improvement, The Computer Journal 37 (8), 1994, pp. 683–690.
[40] P. Weill, M. Broadbent, Leveraging the New Infrastructure—How Market Leaders Capitalize on Information Technology, Harvard Business School Press, Boston, MA, 1998.
[41] G. Weinberg, The Psychology of Computer Programming, Van Nostrand Reinhold, New York, NY, 1971.

James J. Jiang is a professor of management information systems at the University of Central Florida. He obtained his PhD in information systems at the University of Cincinnati. His research interests include IS project management and IS personnel management. He has published over 70 referred articles in journals such as *IEEE Transaction on System, Man, & Cybernetics*, *Decision Support Systems*, *IEEE Transactions on Engineering Management*, *Decision Sciences*, *Journal of Management Information Systems* (*JMIS*), *Communications of the ACM*, *Journal of the Association for Information Systems* (*JAIS*), *Information & Management*, *Journal of Systems & Software*, *Data Base*, *International Journal of Project Management*, and *Project Management Journal*. He is a member of IEEE, ACM, AIS, and DSI.

Dr. Gary Klein is the couger professor of information systems at the University of Colorado in Colorado Springs. He obtained his PhD in management science from Purdue University. Before that time, he served with the company now known as Accenture in Kansas City and was director of the Information Systems Department for a regional financial institution. His research interests include project management, system development, and mathematical modeling with over 90 academic publications in these areas. In addition to being an active participant in international conferences, he has made professional presentations on Decision Support Systems in the US and Japan. He is a member of the Institute of Electrical and Electronic Engineers, the Association for Computing Machinery, the Society of Competitive Intelligence Professionals, the Decision Science Institute, and the Project Management Institute.

Dr. Hsin-Ginn Hwang is the department head of information management at the National Chung Cheng University, Taiwan. He received his PhD in information management from the University of Texas at Arlington. His research interests include group decision-making, hospital information systems, and IS project management.

Jack Shi-Ming Huang received his PhD degree at the School of Computing and Information Systems, University of Sunderland, UK. He is currently an associate professor of information management and a director for the Center of e-Manufacturing and e-Commerce at National Chung Cheng University, Taiwan. Before joining the university faculty, he was a head and associate professor at the Department of Information Management, Tatung University, Taiwan. He has published three books and over 50 papers in the fields of information systems and has acted as a consultant for a variety of Taiwan government departments, software companies and commercial companies.

Shin-Yuan Hung is an associate professor in the Department of Information Management at National Chung Cheng University. He received his PhD in management information systems from National Sun Yat-sen University. His current research interests include executive information systems, group support systems, electronic commerce, and knowledge management.