TOTAL QUALITY MANAGEMENT, VOL. 14, NO. 5, 2003, 571–590

A longitudinal study of software process management in Taiwan's top companies

ELDON Y. LI[1], HOUN-GEE CHEN[2] & TIEN-SHENG LEE[3]
[1]*California Polytechnic State University, San Luis Obispo, USA;* [2]*National Chung Cheng University, Min-Hsiung, Taiwan, Republic of China &* [3]*The Chinese University of Hong Kong, Shatin, Hong Kong, People's Republic of China*

ABSTRACT *Managing the software development process has been an important issue in the IT field. Since the mid-1980s, the Software Engineering Institute (SEI) has been exploring effective ways of managing the software development process. In 1987, SEI proposed a software process maturity framework (Humphrey & Sweet, 1987). Later in 1991, it further announced the capability maturity model (CMM) (Paulk* et al., *1991). Since then, there has been a rapidly growing number of software organizations adopting the CMM as a framework for internal improvement initiatives, including commercial organizations. This study surveys the IT organizations of the top 1000 business companies in Taiwan. It explores the status of software process management (SPM) in these companies and compares the findings with a similar study done in 1996. It further compares the SPM status with Japanese and US data reported in the literature. Finally, it identifies the weaknesses of the surveyed companies and recommends corrective actions for improving their software processes.*

Introduction

Software and its related products are a significant portion of the information technology (IT) budget in a company today (Keil, 1995; Yourdon, 1993). The importance of software quality can never be overemphasized. To this end, the US Department of Defense contracted Carnegie Mellon University in December 1984 to set up and operate the Software Engineering Institute (SEI) for the purpose of advancing the practice of software engineering. Since then, the SEI has been promoting the evolution of software engineering from an ad hoc and error-prone process to a discipline that is well managed and supported by technology. In 1987, SEI proposed a software process maturity framework (Humphrey & Sweet, 1987). Four years later, they announced the first version of Capability Maturity Model (CMM) (Paulk *et al.*, 1991). Later, in 1993, the latest version of the model CMM 1.1 (Paulk *et al.*, 1993), was released. Table 1 shows the characteristics of the five CMM levels.

Since the emergence of the software process maturity framework in 1987, IT organizations around the world have been taking note of this promising concept. The IT organizations in Taiwan joined the bandwagon in 1996 when the Bureau of Industry at the Ministry of Economic Affairs commissioned the Software Industry Productivity Task Force (SIPTF) to

Correspondence: E. Y. Li, Orfalea College of Business, California Polytechnic State University, San Luis Obispo, California 93407, USA. E-mail: eli@calpoly.edu.

ISSN 1478-3363 print/ISSN 1478-3371 online/03/050571-20
DOI: 10.1080/1478336032000053591

Table 1. *The characteristics of CMM levels*

CMM Level	Characteristics	Key challenges	Key process areas
1. Initial	(Ad hoc) Project chaotic	• Project management • Project planning • Configuration management • Software quality assurance	• Ad hoc processes
2. Repeatable	(Intuitive) Process dependent on individuals	• Training • Technical practices • Process focus	• Requirements management • Software subcontract management • Software project tracking & oversight • Software project planning • Software quality assurance • Software configuration management
3. Defined	(Qualitative) Process defined and institutionalized	• Process measurement • Process analysis • Quantitative quality plans	• Peer reviews • Intergroup coordination • Software product engineering • Integrated software management • Training program • Organization process definition • Organization process focus
4. Managed	(Quantitative) Measured process	• Changing technology • Problem analysis • Problem prevention	• Software quality management • Quantitative process management
5. Optimizing	Improvement fed back into process	• Still human intensive process • Maintain organization at optimizing level	• Process change management • Technology change management • Defect prevention

Source: Adapted from Humphrey, *et al.* (1991b) and Paulk *et al.* (1993).

increase the institutionalization of CMM practice in Taiwan's software industry. The SIPTF adapted SEI's model and created its own Software Development Capability Measurement Model (SDCMM). It further developed and announced its Software Development Capability Measurement Handbook (Software Industry Productivity Task Force, 1997) in January 1997. The latest version of this handbook (Software Industry Productivity Task Force, 1998), Version 2, was released in July 1998.

Studies have shown that software process improvement can significantly improve software quality and productivity (Harter *et al.*, 2000; Herbsleb *et al.*, 1994, 1997). The original intent of the CMM was to serve as a tool for the US Department of Defense to benchmark the software process management (SPM) infrastructure of its software contractors. Nevertheless, it may be used to diagnose the software development capability of the IT department/group of a business company and to identify process improvement strategies for improving software product quality (Olson *et al.*, 1989). For the purpose of benchmarking and diagnosing, SEI has developed and been applying five methods (Barbour, 1996; Humphrey *et al.*, 1991a, Kitson & Masters, 1992; Olson *et al.*, 1989). A description of each method is given in Table 2. Based on the outcome of the assessment, SEI places an organization into one of the five CMM levels. According to the DOD's policy, a software company must achieve a maturity

Table 2. *SEI assessment methods*

Method	Description
Assessment Tutorial	An assessment tutorial gives professionals from various organizations an introduction to process management concepts, assessment techniques, and the SEI assessment methodology. At the end of the tutorials, they supply demographic data on themselves and their organizations and complete an assessment questionnaire based on their experience with a project (Humphrey *et al.*, 1991a; Kitson & Masters, 1992).
SEI-Assisted Assessment	This assessment is typically conducted on site by a team of 4 to 6 SEI professionals and 1 to 3 professionals from the organizational being assessed. The team receives SEI training prior to conducting the assessment and typically 4 to 6 projects are examined during an assessment. The objective is to facilitate improvement of the organization's software process. Therefore, any validation of questionnaire responses (e.g. requesting substantiating documents) is directed to those affecting progress to the next higher level of software process maturity (Humphrey *et al.*, 1991a; Kitson & Masters, 1992).
Self-Assessment	A self-assessment is similar to an SEI-assisted assessment, but is conducted with little or no direct SEI involvement. The assessment team is trained by the SEI prior to conducting the assessment. It consists of software professionals mainly from the organization being assessed, with possibly one or two SEI assessment coaches (Humphrey *et al.*, 1991a; Kitson & Masters, 1992).
SEI-Licensed Vendor Assessment	This assessment is similar to a self-assessment, with a commercial vendor playing the coaching role of the SEI. The vendor must be trained and licensed by the SEI and it, in turn, trains the assessment team to do the assessment. The team is composed of software professionals primarily from the organization being assessed and at least one vendor professional who has been qualified by the SEI (Humphrey *et al.*, 1991a; Kitson & Masters, 1992).
Capability Evaluation	A software capability evaluation (SCE) is conducted as part of the DOD's software acquisition process. It provides information about an organization's software engineering capabilities for an acquisition agency and may be used by commercial organizations to evaluate software sub-conractors (Olson *et al.*, 1989). The SCE method evaluates not the technical production processes, such as requirements analysis, specification, and design, but instead the management of these processes along with other key processes. These include the support for organizational management, project management, and product building operations. The team typically consists of four to six trained and experienced people from the sponsoring organization and conducts interviews and document reviews to sample and analyse information about the development organization's implementation of the software processes (Barbour, 1996). This information is then considered along with other relevant information in the source selection decision. Therefore, validation of maturity questionnaire responses is a greater consideration here than it is in other types of assessment (Humphrey *et al.*, 1991a).

level of three or higher in order to participate in the bidding process of the DOD's software contract.

There are many individual success stories (Anonymous, 2000, 2001a, b; Business and High Tech Editors, 2000, 2001; Humphrey *et al.*, 1991b) reported in the news but only a few studies reviewed the overall status of an industry. Those available are mostly about the software industry (Herbsleb *et al.*, 1997; Humphrey *et al.*, 1989, 1991a; Kitson & Masters, 1992). Although Humphrey *et al.* (1991a) reported data about some Japanese business companies, the only study to date about the status of business industries as a whole was Li *et al.* (2002) for Taiwan's top 1000 companies. Since then, no other similar study has been reported and none has provided a longitudinal analysis of the growth of SPM practices in business industries. This study attempts to fill this void.

The purpose of this study is threefold. First is to take the pulse of the SPM infrastructure of the Taiwan's top business companies and gain an insight into their strengths and weaknesses. This would be very helpful for the Taiwan government and industries to prioritize their actions and allocate their resources effectively for improving their national SPM status. The second purpose is to compare the results of this study with those of Li *et al.* (2002) conducted in 1996. Because the subject groups surveyed by both studies are identical, this allows us to examine the improvement progress of SPM practices in Taiwan's top 1000 companies. The third purpose is to compare the results of this study with those of Japan and US software organizations reported by Humphrey *et al.* (1991a). This helps us identify the strengths and weaknesses of the responding companies and to recommend remedial actions for software process managers in Taiwan.

Literature review

Humphrey *et al.* (1991a) reported three sets of assessment data, two from the US and one from Japan. The participants in the US were Department of Defense (DoD) organizations, DoD contractors and commercial organizations. These included 55 projects from 10 organizations that participated in SEI-assessments and 113 projects from over 70 organizations that participated in assessment tutorials. The former data set (55 projects) was provided by Humphrey, *et al.* (1989). In contrast, the participants in Japan were from over 88 software organizations in six Japanese companies. These included many business-application programming groups, a few communications and military suppliers, and two computer manufacturers. Through the assessment tutorials, 196 projects were assessed. In these three data sets, each data point is one set of yes–no responses to the maturity questionnaire regarding a specific software project. The authors found that the US software industry, in general, was ahead of its Japanese counterpart, perhaps due to the stringent requirements that the DoD put on its software contractors.

Kitson & Masters (1992) conducted software process assessments on 296 projects in 59 organizations during February 1987 through March 1991. Among the organizations assessed, 23% were commercial firms, 51% were DOD contractors, 9% military services, 8% federal agencies, and 11% others. They found that 85% of the key practices implemented in this sample to be in level 2 or 3 categories. The five most implemented key practice areas were software product engineering, software project planning, organization process definition, project tracking and oversight, and training programmes. The five least implemented ones were process change management, defect prevention, subcontract management, quality management, and peer reviews.

Herbsleb *et al.* (1997) surveyed 167 organizations and received 138 usable questionnaires. The purpose of their study was to address three questions: 'How long does it take, how much does it cost, and how will it benefit an organization to move up a maturity level?', 'What are the factors that influence the success and failure of CMM-based software process improvement (SPI)?' and 'Is the CMM an appropriate framework for guiding improvements in a way that can be understood and applied to the full variety of software organizations?'. All surveyed organizations were at the level-one to level-three, Initial level to Defined level. The authors found that most organizations do not think that CMM-based SPI was counterproductive

(96%), neglected non-CMM issues (90%), or made an organization rigid and bureaucratic (84%). Both of the aforementioned studies (Herbsleb *et al.*, 1997; Kitson & Masters, 1992), did not report the implementation frequency of any key practice. Their data points are not useful for our research purposes, thus will not be analysed in this study.

Li *et al.* (2002) surveyed the top 1000 Taiwanese business companies in 1996 and reported their CMM key practices based on 138 respondents. They compared their findings with those of Humphrey *et al.* (1991a) in order to identify the weaknesses of the companies. They concluded that SPM in Taiwan's top companies is still in its infancy stage. These companies lagged behind their counterparts in Japan and the US in many key practices even almost a decade later.

Research method

The subjects

The subjects for this study were the top 1000 companies listed in a recent *Directory of Large Corporations in Taiwan*, published by China Credit Information Service, Taipei, Taiwan. The sample included 622 manufacturing and 378 service companies. The CMM questionnaire was sent twice to the IT executive in each of the sampled companies. The executive was implored to direct the questionnaire to someone who has expert knowledge about the software development practices in the company. Our questionnaire also indicated that additional experts should be consulted if a single individual could not answer all the questions. The first wave of mailing results in 68 response questionnaires. Four weeks later, the second wave of mailing went out to the non-responding companies and attracted 40 more respondents. At the end of the tenth week, we received in total 108 responding questionnaires. However, seven contained excessive missing or inconsistent data and were excluded from the study, giving 101 usable questionnaires and a 10.1% response rate. The characteristics of the participating companies are listed in Table 3.

The questionnaire

For comparison purposes, this study adopted the questionnaire used by Li *et al.* (2002). The questionnaire contains 89 questions while the original questionnaire measuring the CMM level contained only 85 questions (Humphrey & Sweet, 1987). The additional four questions were developed by Li *et al.* in order to eliminate the ambiguity and insufficiency of the original questionnaire. Three original questions were modified into two questions each and one new question was added to the questionnaire. Table 4 exhibits the distribution of questions in each assessment class. The respondent was asked to check each 'yes' box only if the key practice in question has been a standard practice in his/her organization.

Procedure

Under the circumstance that a company is being certified for the eligibility of contracting a DOD software project, an SEI-trained team from the sponsoring organization must conduct CMM benchmarking on various software projects via site visits. This formal procedure eliminates the Hawthorn effect (Roethlisberger & Dickson, 1939) encountered in the self-reporting survey in an assessment tutorial. Nonetheless, the assessment tutorial method is an effective and less-expensive way of collecting large data samples. In this study, we employed the approach used by Li *et al.*, in which they adapted the assessment tutorial method into a

Table 3. *The profile of participating companies (N = 101*)*

Category	Classification	*N**	1% of total
Type of industry	Service	36	35.6
	Manufacturing	65	64.4
Annual sales (US$)	Below 50 million	3	3.0
	50–99 million	25	24.8
	100–499 million	54	53.5
	50–999 million	12	11.9
	1 billion or above	7	6.9
Number of company employees	50 persons or less	2	2.0
	51–100 persons	2	2.0
	101–500 persons	37	36.6
	501–1000 persons	25	24.8
	1001 persons or more	35	34.7
Experience in software development	0–2 years	16	16.8
	3–5 years	50	52.6
	6–10 years	26	27.4
	11 years or more	3	3.2
Number of information system employees	Below 5 persons	24	23.8
	5–10 persons	35	34.7
	11–50 persons	29	28.7
	51 persons or more	13	12.9
Number of software development employees	1–5 persons	40	42.6
	6–10 persons	18	19.1
	11–20 persons	14	14.9
	21–50 persons	10	10.6
	51 persons or more	12	12.8

*Due to missing responses, the total sample size of each category might not equal 101.

Table 4. Distribution of CMM assessment questions in this study

Assessment category	Assessment class	Level 2	Level 3	Level 4	Level 5	Row total
Organization and resource management	Organizational structure	4	3	0	0	7
	Resources, personnel, and training	1	4	0	0	5
	Technology management	1	1	2	1	
Software engineering processes and its management	Documented standards and procedures	10 (1)*	7	2	0	19 (1)
	Process metrics	13 (1)	3	6 (2)	0	22 (3)
	Data management and analysis	0	0	6	3	9
	Process control	6	14	2	0	22
	Column total	35 (2)	32	18 (2)	4	89 (4)

*Parenthesized number indicates the number of additional questions created by this study. The number preceding the parenthesized number indicates the total number of questions used in this study.

mail survey method and replaced the required tutorial session with a self-paced tutorial document. This document was included in the mailing along with the survey questionnaire. We expect the data provided by our participants should reflect closely the actual SPM status of the companies for the same reasons as Li *et al.* identified. Further validation of the data is reported in the next section.

Data validation

The first step to ensure the validity of the data is to examine non-response bias and data representativeness. For the non-response bias, the usable data collected from the first-wave of mailing and those from the second wave were tested. No significant difference was found at the 95% confidence level, indicating the absence of a bias. Subsequently, the data representativeness was examined by testing the differences in demographic distributions between the population (1000 companies) and the usable sample (101 respondents). No significant difference was found (at the 95% confidence level) in terms of company size (including annual sales and number of employees) and industry type. All these results support the quality and the representativeness of the response data.

Results and discussion

The top companies in Taiwan have implemented, on average, 43.7% of the 89 key practices of CMM Compared with 35.4% in 1996, reported by Li *et al.* (2002), this is a 23% increase. Figure 1 exhibits the profile of key practices achieved by the top companies in Taiwan, while

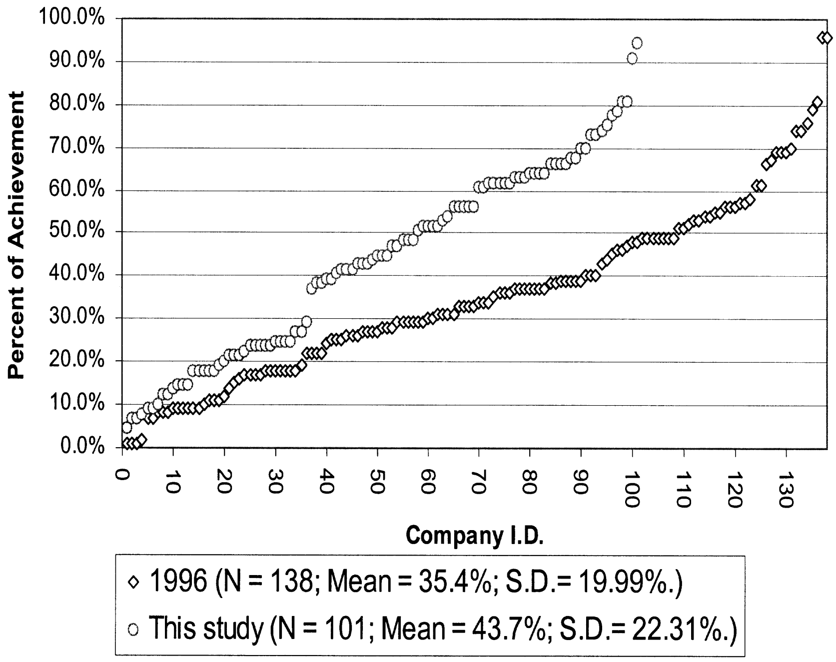

Figure 1. *A profile of the percentage of key-practice achievement.*

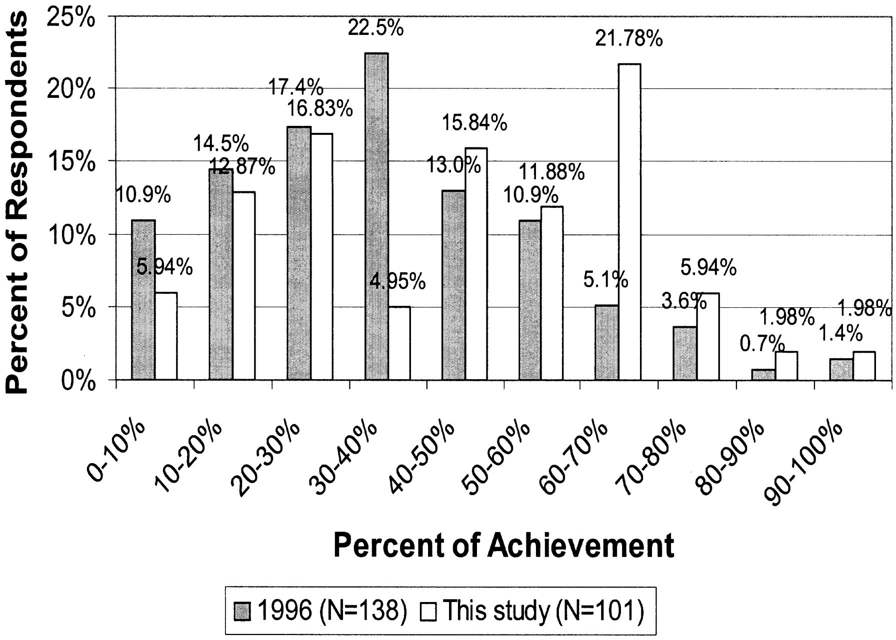

Figure 2. *Distribution of key-practice achievement levels.*

Fig. 2 shows the frequency distribution of the percentage of achievement among the respondents. According to Fig. 1, there were two companies that achieved over 90% of the key practices in both studies. Moreover, this study has a larger standard deviation (22.31%) than the 1996 survey (19.99%), as shown by the steeper slope of the plot for this study. That is, the differences in the percentage of key practices implemented among the sampled companies are higher in this study than in the 1996 survey. The trend of having a steeper slope is expected to continue as long as the existing companies are improving their achievement in SPM practices while more companies are joining the CMM bandwagon. As time passes, we expect these sampled companies will eventually display a dichotomy pattern on the plot, having one group of high achievers and another group of low achievers with a large gap between the two groups. In fact, the plot for this study has already exhibited a sign of such a pattern. The rationale is that the companies who know what to do and can afford to do it would eventually become the top achievers. Those who do not know what to do and cannot afford to do it, would be the low achievers and remain in the status quo.

The distribution in Fig. 2 reveals that only 9.8% of companies achieved 70% or more of key practices (up from 5.7% in 1996). The largest group (21.8%) of companies achieved between 60% and 70%, while the largest group in 1996 was between 30% and 40%. Almost half of the companies (43.57%) carried out more than 50% of the key practices, up from 21.74%.

Table 5 indicates the five most and the five least implemented key practices. The first four most implemented practices are the same in both studies while the fifth key practice is now 'Q1.1.7: Is there a software engineering process group function?' (66.3%). Nonetheless, practices Q1.1.1 and Q1.1.2 remain very frequent (52.5% and 51.5% respectively). On the other side, the three least implemented practices have changed places; Q2.4.21, Q2.2.4.2s

Table 5. *The most and the least implemented key practices*

Level	Question ID and description		1996 survey	This study	Difference
Most implemented practices:					
3	Q1.2.1.	Does each software developer have a private computer-supported workstation/terminal?	**92.3%**	**90.1%**	−2.2%
3	Q2.1.11.	Are code maintainability standards applied?	**70.8%**	**81.2%**	10.4%
3@	Q2.1.1.	Does the software organization use a standardized and documented software development process on each project?	**72.3%**	**77.2%**	4.9%
2	Q2.1.9.	Are coding standards applied to each software development project?	**75.4%**	**71.3%**	−4.1%
3@	Q1.1.7.	Is there a software engineering process group function?	*41.5%*	**66.3%**	24.8%
2	Q1.1.1.	For each project involving software development, is there a designated software manager?	**63.8%**	*52.50%*	−11.3%
2	Q1.1.2.	Does the project software manager report directly to the project (or project development) manager?	**63.8%**	*51.5%*	−12.3%
Least implemented practices:					
2@	Q2.1.14.x.	Is a formal procedure used to make estimates of software complexity?	**8.5%**	**9.9%**	1.4%
2@	Q2.2.4.1s.	Are statistics on software code errors gathered?	**7.7%**	**16.8%**	9.1%
3@	Q2.2.3.	Are statistics on software design errors gathered?	*11.50%*	**16.8%**	5.3%
4@	Q2.3.8.	Is review efficiency analyzed for each project?	*17.7%*	**17.8%**	0.1%
2@	Q2.4.7.	Do software development first-line managers sign off on their schedules and cost estimates?	*16.9%*	**18.8%**	1.9%
3@	Q2.4.21.	Is there a mechanism for assuring the adequacy of regression testing?	**8.5%**	*20.8%*	12.3%
2@	Q2.2.4.2s.	Are statistics on software test errors gathered?	**9.2%**	*22.8%*	13.6%
2	Q2.2.1.	Are software staffing profiles maintained of actual staffing versus planned staffing?	**9.2%**	*23.8%*	14.6%

and Q2.2.1 are replaced by Q2.2.3, Q2.3.8 and Q2.4.7. The former three practices remain in sixth to eighth places of the least frequent practices.

There are 12 key practices that show decreased implementation frequencies, as listed in Table 6. The largest decrease (−14.8%) belongs to the formal procedures for producing software development schedules. It appears that change control, error analysis, and management structure are also less concerned in these companies today.

On the positive side, 77 (86.5%) of the 89 key practices in this survey have gained more implementation in the companies. Table 7 shows the 43 key practices that have gained over 10% increase. Among them, seven have gained over 20%. These seven practices are concerned primarily with software testing (Q2.2.6.2s, Q2.2.14 and Q2.4.20), process management (Q1.1.7 and Q1.3.2), configuration management (Q2.2.9), and code review (Q.2.2.13.2s). The Pearson's and Spearman's correlations between the data from 1996 survey and this study are 0.854 and 0.825, respectively. Both correlations are significant at the $p < 0.001$ level. The paired-samples t-tst between the two sets of data is significant at the $p < 0.001$ level (t-statistics = 9.463) and the Wilcoxon's sign rank test is significant at the $p < 0.001$ level (z-statistics = 6.728). Both tests support the fact that the number of companies implementing the SPM practices in Taiwan today is significantly higher than in 1996.

Table 6. *The decreasingly implemented key practices*

Level	Question ID and description		1996 survey	This study	Difference
2@	Q2.1.15.	Is a formal procedure used to produce software development schedules?	55.4%	40.6%	−14.8%
3@	Q2.4.13.	Is a mechanism used for controlling changes to the software design?	57.7%	43.6%	−14.1%
2	Q1.1.2.	Does the project software manager report directly to the project (or project (or project development) manager?	63.8%	51.5%	−12.3%
4@	Q2.3.4.	Are analyses of errors conducted to determine their process related causes?	44.6%	32.7%	−11.9%
2	Q1.1.1.	For each project involving software development, is there a designated software manager?	63.8%	52.5%	−11.3%
3	Q1.1.5.	Is software system engineering represented on the system design team?	52.3%	47.5%	−4.8%
2	Q2.1.9.	Are coding standards applied to each software development project?	75.4%	71.3%	−4.1%
2	Q2.1.5.	Is there a mechanism for ensuring that software subcontractors, if any, follow a disciplined software development process?	44.6%	41.6%	−3.0%
2@	Q1.1.3.	Does the Software Quality Assurance (SQA) function have a management reporting channel separate from the software development project management?	40.0%	37.6%	−2.4%
3	Q1.2.1.	Does each software developer have a private computer-supported workstation/terminal?	92.3%	90.1%	−2.2%
2@	Q2.4.9.	Is a mechanism used for controlling changes to the software requirements?	58.5%	56.4%	−2.1%
2	Q2.1.17.	Is a mechanism used for ensuring that the software requirement?	44.6%	43.6%	−1.0%

Comparing with Japan and the US

A review of the literature reveals that Humphrey *et al.* (1991a) is the only study reporting the individual statuses of level-two and level-three critical practices in Japan and the US. Although their participants are not all business companies, their data are useful for us to identify the weaknesses of SPM in our participating companies because the SPM key practices are essentially the same in every organization. The rationale for this comparison is that, if significantly more organizations in Japan and the US could perform a key practice back in 1991 than those in Taiwan now, this practice gap would be wider today and it would be a weakness of Taiwanese companies not to keep up with their Japanese and US counterparts regarding this practice (Li *et al.*, 2002). A scrutiny of Figs 3 and 4 reveals that Taiwanese companies are still behind in the same six key practices found in the 1996 survey, though the percentages of achievement are much higher today.

(1) Gathering statistics on software design errors (Item Q2.2.3, 16.8%; up from 11.5% in 1996).
(2) Gathering statistics on software code errors (Item Q2.2.4.1s, 16.8%; up from 7.7%).
(3) Gathering statistics on software test errors (Item Q2.2.4.2s, 22.8%; up from 9.2%).
(4) Empowring software development first-line managers to sign off on their schedules and cost estimates (Item Q2.4.7, 18.8%; up from 16.9%).
(5) Conducting software code reviews (Item Q2.4.16, 36.6%; up from 20.8%).
(6) Conducting internal software design reviews (Item Q2.4.12, 42.6%; up from 23.1%).

Table 7. *The most increasingly implemented key practices*

Level	Question ID	and description	1996 survey	This study	Differ- ence
3@	Q1.1.7.	Is there a software engineering process group function?	41.5%	66.3%	24.8%
4@	Q2.2.6.2s.	Are test errors projected and compared to actuals?	23.8%	48.5%	24.7%
4@	Q2.2.14.	Is test coverage measured and recorded for each phase of functional testing?	31.5%	55.4%	23.9%
2	Q2.4.20.	Is there a mechanism for ensuring that regression testing is routinely performed?	15.4%	38.6%	23.2%
2	Q2.2.9.	Are profiles maintained of actual versus planned software units integrated, over time?	12.3%	33.7%	21.4%
3	Q1.3.2.	Is a mechanism used for evaluating technologies used by the organization versus those externally available?	19.2%	39.6%	20.4%
4@	Q2.2.13.2s.	Is code review coverage measured and recorded?	22.3%	42.6%	20.3%
3@	Q2.4.12.	Are internal software design reviews conducted?	23.1%	42.6%	19.5%
3	Q2.4.22.	Are formal test case reviews conducted?	28.5%	47.5%	19.0%
4@	Q2.3.1.	Has a managed and controlled process database been established for process metrics data across all projects?	18.5%	36.6%	18.1%
2@	Q2.1.3.	Is a formal procedure used in the management review of each software development prior to making contractual commitments?	43.8%	61.4%	17.6%
3	Q2.1.2.	Does the standard software development process documentation describe the use of tools and techniques?	40.0%	57.4%	17.4%
2@	Q1.1.6.	Is there a software configuration control function for each project that involves software development?	46.2%	63.4%	17.2%
2	Q2.2.8.	Are profiles maintained of actual versus planned software units completing unit testing, over time?	18.5%	35.6%	17.1%
4	Q2.3.9.	Is software productivity analysed for major process steps?	17.7%	34.7%	17.0%
3	Q2.4.8.	Is a mechanism used for ensuring traceability between the software requirements and top-level design?	26.9%	43.6%	16.7%
4@	Q2.1.13.	Are code review standards applied?	26.2%	42.6%	16.4%
2	Q1.2.2.	Is there a required training programme for all newly appointed development managers designed to familiarize them with software project management?	40.0%	56.4%	16.4%
3	Q2.4.15.	Are formal records maintained of unit (module) development progress?	34.6%	50.5%	15.9%
4	Q1.3.3.	Is a mechanism used for deciding when to insert new technology into the development process?	27.7%	43.6%	15.9%
3@	Q2.4.16.	Are software code reviews conducted?	20.8%	36.6%	15.8%
4@	Q2.2.6.1s.	Are code errors projected and compared to actuals?	19.2%	34.7%	15.5%
2	Q2.2.11.	Are target computer throughput utilization estimates and actuals tracked?	32.3%	47.5%	15.2%
4	Q2.1.12.	Are internal design review standards applied?	28.5%	43.6%	15.1%
4@	Q2.2.5.	Are design errors projected and compared to actuals?	23.8%	38.6%	14.8%
3	Q2.1.10.	Are standards applied to the preparation of unit test cases?	30.8%	45.5%	14.7%
2	Q2.2.1.	Are software staffing profiles maintained of actual staffing versus planned staffing?	9.2%	23.8%	14.6%
3	Q2.4.3.	Is a mechanism used for identifying and resolving system engineering issues that affect software?	26.2%	40.6%	14.4%
4@	Q1.3.4.	Is a mechanism used for managing and supporting the introduction of new technologies?	32.3%	46.5%	14.2%
2	Q2.1.7.	For each project, are independent audits conducted for each step of the software development process?	28.5%	42.6%	14.1%
3	Q2.1.8.	Is a mechanism used for assessing existing designs and code for reuse in new applications?	34.6%	48.5%	13.9%

Table 7. *Continued*

Level	Question ID and description		1996 survey	This study	Differ-ence
2@	Q2.2.4.2s.	Are statistics on software test errors gathered?	9.2%	22.8%	13.6%
4@	Q2.3.2.	Are the review data gathered during design reviews analysed?	43.1%	56.4%	13.3%
3@	Q2.2.15.	Are the action items resulting from design reviews tracked to closure?	43.8%	56.4%	12.6%
3@	Q2.2.17.	Are the action items resulting from code reviews tracked to closure?	39.2%	51.5%	12.3%
2	Q2.2.7.	Are profiles maintained of actual versus planned software units designed, over time?	11.5%	23.8%	12.3%
3@	Q2.4.21.	Is there a mechanism for ensuring the adequacy of regression testing?	8.5%	20.8%	12.3%
2@	Q2.4.1.	Does senior management have a mechanism for regular review of the status of software the development projects?	26.9%	38.6%	11.7%
3	Q1.2.4.	Is there required software engineering training programme for first-line supervisors of software development?	20.0%	31.7%	11.7%
3	Q2.4.14.	Is a mechanism used for ensuring traceability between the software detailed design and the code?	29.2%	40.6%	11.4%
2	Q2.2.1.0.	Are target computer memory utilization estimates and actuals tracked?	50.0%	61.4%	11.4%
3	Q2.1.18.	Are man–machine interface standards applied to each appropriate software development project?	43.8%	54.5%	10.7%
3	Q2.1.11.	Are code maintainability standards applied?	70.8%	81.2%	10.4%

All these are critical practices and Items 1, 2 and 4 belong to the least implemented practices in this study while Items 1 and 3 are among the least implemented in the 1996 survey. The first three items are all the practices that pertain to gathering error statistics. This indicates that Taiwanese business companies have not utilized much of the error statistics to predict errors, prevent errors, and develop training programmes, etc. Item 4 reveals that first-line managers are less likely to be accountable for the schedules and cost estimates in Taiwan. Although this weakness may or may not affect software quality, it is directly related to the efficacy of project management and at some point in time, may results in unresponsiveness to schedule sliding and cost overrun. Finally, Items 5 and 6 (code and design reviews) are the two technical reviews most critical to software quality. It implies that Taiwan's companies are lagging far behind the other two countries in manual quality assurance activities. These activities normally can catch most of the errors before the project proceeds into the code compilation and testing phase in the software development process.

Conclusions and recommendations

Taiwan's business industries have made great strides in SPM practices since 1996. They have improved most of their SPM key practices significantly. This study surveys the software organizations in Taiwan's top 1000 companies and reveals the overall status of each SPM practice achievement. This overall status provides each organization with a yardstick against which it could measure itself and identify its own strengths and weaknesses. Based on the results of this study, today's top companies in Taiwan have several weaknesses compared with the statuses of their Japanese and American counterparts over a decade ago. Consequently, we may conclude that the software quality and productivity of these companies would be

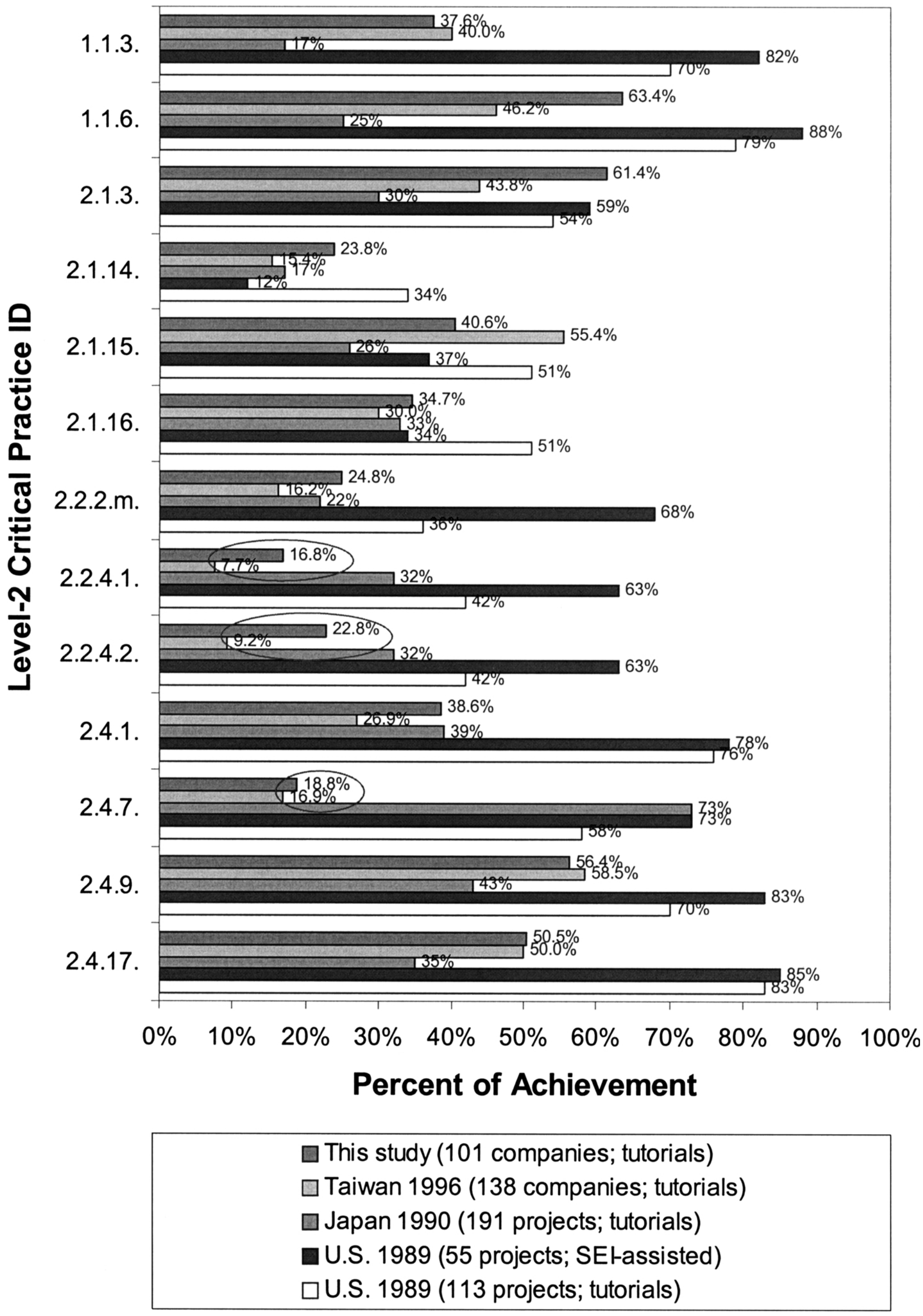

Figure 3. *Comparison of Level-2 critical practices.*

much lower than those in Japan and the US today. In addition, several conclusions and recommendations may be drawn:

First, the percentages of key practices implemented by Taiwanese business companies are increasing. The average percentage of achievement has risen from 35.4% to 43.7%. While only

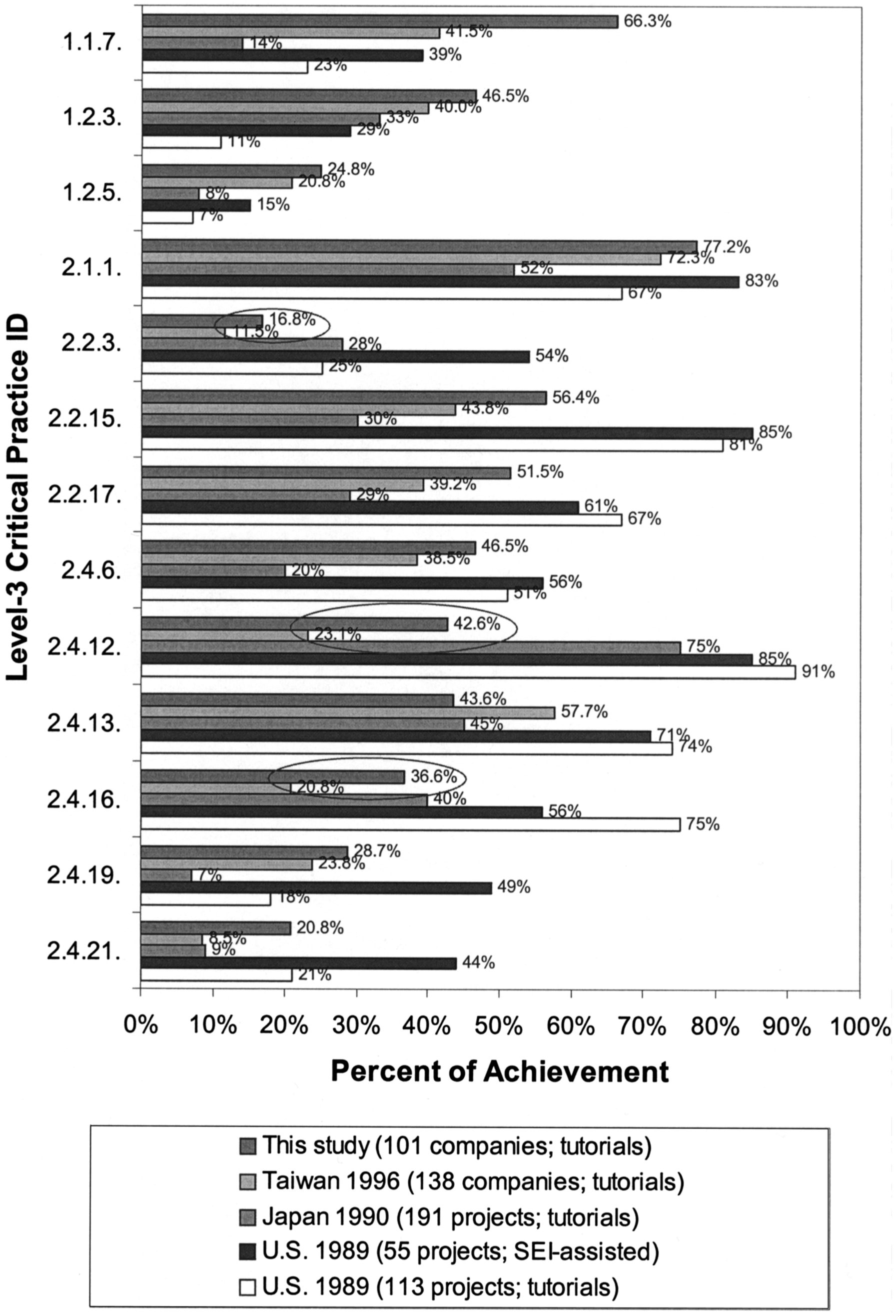

Figure 4. *Comparison of Level-3 critical practices.*

21.7% of these companies implemented half or more of the 89 key practices in 1996, today this percentage has increased to 43.6%. This indicates that the management in more and more companies have realized the importance of the software process in ensuring quality and improving productivity. Without quality, productivity means nothing. It is the quality that attracts the user to use the software products. 'Buggy' software is considered a defect and a waste, and will never be accepted by an educated user. Therefore, every software organization should manage its software process and produce quality software. In order to do so effectively, the management must provide adequate resource support to carry out the CMM key practices.

Second, most (77 of 89) key practices showed increased implementation today, compared with 1996. While 43 key practices increased by 10.4% to 24.8%, seven increased by at least 20%. This dramatic achievement is probably due to the effort of the Software Industry Productivity Task Force in promoting its SDCMM to business industries since early 1997. Nevertheless, the institutionalization of SPM practices in industries could have been sped up had the software engineering curricula of Taiwanese universities been improved to emphasize software quality and processes management techniques. Only through the educational process can the concept of CMM become prevalent in every software organization.

Third, Taiwan's companies have significantly improved most of their major weaknesses found in the 1996 survey. All of these weaknesses have now reached at least 16% of achievement in key practices, except using a formal procedure for estimating software complexity (9.9% achievement). Software complexity involved computational complexity (Rabin, 1977) and psychological complexity (Curtis *et al.*, 1979). While computational complexity measures the efficiency of alternative algorithmic solutions, the psychological complexity assesses human performance on programming tasks. Estimates of programme complexity enable project managers to pinpoint error-prone software modules and allocate more resources to minimize error occurrences. However, most complexity metrics are very difficult to derive, except the control-flow complexity metric (McCabe, 1976). The control-flow complexity metric is easy to understand and calculate. It is also the only one that lends itself to determining the minimum test set for programme testing (Li, 1987). Since the computational complexity in business companies is usually low, we strongly recommend using the control-flow metric as a basis of estimating the complexity of a software project.

Fourth, compared to Japanese and US data reported in 1991, Taiwan's companies are still behind in the same six key practices found by the 1996 survey. These six practices continue to be the major weaknesses of Taiwan's top business companies. They are related to a lack of empowerment, error statistics and technical reviews. Empowerment to first-line managers is a catalyst for continuous process improvement. As the first-line managers virtually live with their staff, they know best what their staff want, need and are good at, thus eliminating communication gaps. According to total quality management (TQM) principle, empowerment should happen not only to the lower-level managers, but also to all staff in the software project (Li *et al.*, 2000). Everyone should be accountable for his or her own work. Everyone must be committed to improving continually his or her own work and to suggesting improvements to organizational processes. In fact, SPM and TQM are two sides of a coin. While SPM is the heart, TQM is the soul of a software organization. Without both, the organization would never be able to produce quality software. Therefore, regardless of how many CMM practices a software organization has institutionalized, the management should ensure that the TQM concept and methods have been instilled into the organization before implementing the SPM process. Otherwise, it is impossible for an organization to produce quality software even if it is at the maturity (optimizing) stage of CMM.

Finally, Taiwanese business companies rarely gathered error statistics, including code errors, test errors, and design errors. Therefore, they did not have the ability to make error prediction and to prevent the errors from happening, not to mention learning how to reduce human errors in design, code, and test activities. As for design reviews and code reviews, these are effective methods for error prevention and quality assurance. Taiwan's companies must allocate adequate time and effort to these activities. Most important of all is the education and training programmes a company must provide to its software development employees in order to ensure that everyone knows how to do his or her job productively. Without such programmes, capable employees are difficult to come by and software productivity is only a dream that can never be fulfilled.

References

ANONYMOUS (2000) Lockheed Martin management and data systems earns a world class rating for its software engineering processes, *PR Newswire*, New York, 4 December, p. 1.

ANONYMOUS (2001a) Keane and Gateway achieve SEI CMM level 3 compliance, *PR Newswire*, New York, 15 January, p. 1.

ANONYMOUS (2001b) Veridian information solutions group obtains CMM(R) level 2 rating with process Max(R)2, *PR Newswire*, New York, 11 January, p. 1.

BARBOUR, R. (1996) Software capability evaluation version 3.0 implementation guide for supplier selection, Technical Report CMU/SEI-95-TR-012, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA 15213.

BUSINSS & HIGH TECH EDITORS (2000) Questra achieves CMM level 2 compliance; repeatable software processes ensures higher quality development for clients, *Business Wire*, New York, 5 September, p. 1.

BUSINESS & HIGH TECH EDITORS (2001) Tanning technology India awarded SEI SW-CMM level 3 certification, *Business Wire*, New York, 8 January, p. 1.

CURTIS, B., SHEPPARD, S.B., MILLIMAN, P., BORST, M.A. & LOVE, T. (1979) Measuring the psychological complexity of software maintenance tasks with the Halstead and McCabe metrics, *IEEE Transactions on Software Engineering*, 5(2), pp. 96–104.

HARTER, D.E., KRISHNAN, M.S. & SLAUGHTER, S.A. *(2000) Effects of process maturity on quality cycle time, and effort in software product development, Management Science*, 40(4), pp. 451–466.

HERBSLEB, J., CARLETON, A., ROZUM, J., SIEGEL, J. & ZUBROW, D. (1994) Benefits of CMM-based software process improvement: executive summary of initial results, Technical Report CMU/SEI-94-SR-013, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA 15213, September.

HERBSLEB, J., ZUBROW, D., GOLDENSON, D., HAYES, W. & PAULK, M. (1997) Software quality and the capability maturity model, *Communications of the ACM*, 40(6), pp. 30–40.

HUMPHREY, W.S., KITSON, D.H. & KASSE, T.C. (1989) The state of software engineering practice: a preliminary report, Technical Report CMU/SEI-89-TR-1, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA 15213.

HUMPHREY, W.S., KITSON, D.H. & GALE, J. (1991a) *A comparison of US and Japanese software process maturity, Technical Report CMU/SEI-91-TR-27, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA 15213.*

HUMPHREY, W.S., SNYDER, T.R. & WILLIS, R.R. *(1991b) Software process improvement at Hughes Aircraft, IEEE Software*, 8(4), pp. 11–23.

HUMPHREY, W.S. & SWEET, W.L. (1987) A method for assessing the software engineering capability of contractors, Technical Report CMU/SEI-87-TR-23, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA 15213.

KEIL, M. (1995) Pulling the plug: software project management and the problem of project escalation, *MIS Quarterly*, 19(4), pp. 421–447.

KITSON, D. & MASTERS, S. (1992) An analysis of SEI software process assessment results 1987–1991, Technical Report CMU/SEI-92-TR-24, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA 15213.

LI, E.Y. (1987) On the cyclomatic metric of program complexity, *Quality Data Processing* 1(3), pp. 15–23.

LI, E.Y., CHEN, H.G. & CHEUNG, W.M. (2000) Total quality management in software development process, *The Journal of Quality Assurance Institute*, 14(1), pp. 4–6 & 35–41.

LI, E.Y., CHEN, H.G. & LEE, T.S. (2002) *Software process management of top companies in Taiwan: a comparative study, Total Quality Management*, 13(5), pp. 701–713.

MCCABE, T.J. (1976) A complexity measure, *IEEE Transactions on Software Engineering*, 2(4), pp. 308–320.

OLSON, T.G., HUMPHREY, W.S. & KITSON, D.H. (1989) Conducting SEI-assisted software process assessments, Technical Report CMU/SEI-89-TR-7, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA 15213.

PAULK, M., CURTIS, B., AVERILL, E., BAMBERGER, J., KASSE, T., KONRAD, M., PERDUE, J., WEBER, C. & WITHEY, J. (1991) Capability maturity model for software, Technical Report CMU/SEI-91-TR-024, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA 15213.

PAULK, M., CURTIS, B., CHRISSIS, M. & WEBER, C. (1993) The capability maturity model for software, version 1.1, Technical Report CMU/SEI-93-TR-024, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA 15213.
RABIN, M.O. (1977) Complexity of computation, *Communications of the ACM*, 20, pp. 625–633.
ROETHLISBERGER, F.J. & DICKSON, W.J. (1939) *Management and the Worker; An Account of A Research Program Conducted by the Western Electric Company, Hawthorne Works, Chicago*. With the assistance and collaboration of Harold A. Wright. Cambridge, Mass. (Harvard University Press).
SOFTWARE INDUSTRY PRODUCTIVITY TASK FORCE (1997) *Software Development Capability Measurement Handbook* (in Chinese), First edition, China Information Software Association, Taiwan, January.
SOFTWARE INDUSTRY PRODUCTIVITY TASK FORCE (1998) *Software Development Capability Measurement Handbook* (in Chinese), Second edition, China Information Software Association, Taiwan, July.
YOURDON, E. (1993) *Decline and Fall of the American Programmer* (Prentice-Hall).

Appendix: Percent of respondents achieving each key practice

Level	Question ID and description		1996 survey	This study	Difference
1.1. Organizational structure					
2	Q1.1.1.	For each project involving software development, is there a designated software manager?	63.8%	52.5%	−11.3%
2	Q1.1.2.	Does the project software manager report directly to the project (or project development) manager?	63.8%	51.5%	−12.3%
2@	Q1.1.3.	Does the Software Quality Assurance (SQA) function have a management reporting channel separate from the software development project management?	40.0%	37.6%	−2.4%
3	Q1.1.4.	Is there a designated individual or team responsible for the control of software interfaces?	63.1%	63.4%	0.3%
3	Q1.1.5.	Is software system engineering represented on the system design team?	52.3%	47.5%	−4.8%
2@	Q1.1.6.	Is there a software configuration control function for each project that involves software development?	46.2%	63.4%	17.2%
3@	Q1.1.7.	Is there a software engineering *process group* function?	41.5%	66.3%	24.8%
1.2. Resources, personnel, and training					
3	Q1.2.1.	Does each software developer have a private computer-supported workstation/terminal?	92.3%	90.1%	−2.2%
2	Q1.2.2.	Is there a required training programme for all newly appointed development managers designed to familiarize them with software project management?	40.0%	56.4%	16.4%
3@	Q1.2.3.	Is there a required software engineering training programme for software developers?	40.0%	46.5%	6.5%
3	Q1.2.4.	Is there a required software engineering training programme for first-line supervisors of software development?	20.0%	31.7%	11.7%
3@	Q1.2.5.	Is a formal training programme required for design and code *review leaders*?	20.8%	24.8%	4.0%
1.3. Technology management					
2	Q1.3.1.	Is a *mechanism* used for maintaining awareness of the state-of-the-art in software engineering technology?	33.8%	42.6%	8.8%
3	Q1.3.2.	Is a *mechanism* used for evaluating technologies used by the organization versus those externally available?	19.2%	39.6%	20.4%
4	Q1.3.3.	Is a *mechanism* used for deciding when to insert new technology into the development process?	27.7%	43.6%	15.9%
4@	Q1.3.4.	Is a *mechanism* used for managing and supporting the introduction of new technologies?	32.3%	46.5%	14.2%
5@	Q1.3.5.	Is a *mechanism* used for identifying and replacing obsolete technologies?	29.2%	36.6%	7.4%

Level	Question ID and description		1996 survey	This study	Differ-ence
2.1. Documented standards and procedures					
3@	Q2.1.1.	Does the software organization use a standardized and documented software development *process* on each project?	72.3%	77.2%	4.9%
3	Q2.1.2.	Does the standard software development *process* documentation describe the use of tools and techniques?	40.0%	57.4%	17.4%
2@	Q2.1.3.	Is a *formal procedure* used in the management review of each software development prior to making contractual commitments?	43.8%	61.4%	17.6%
2	Q2.1.4.	Is a *formal procedure* used to ensure periodic management review of the status of each software development project?	37.7%	42.6%	4.9%
2	Q2.1.5.	Is there a *mechanism* for ensuring that software subcontractors, if any, follow a disciplined software development process?	44.6%	41.6%	−3.0%
3	Q2.1.6.	Are *standards* used for the content of software development files/folders?	63.1%	65.3%	2.2%
2	Q2.1.7.	For each project, are independent audits conducted for each step of the software development *process*?	28.5%	42.6%	14.1%
3	Q2.1.8.	Is a *mechanism* used for assessing existing designs and code for reuse in new applications?	34.6%	48.5%	13.9%
2	Q2.1.9.	Are coding *standards* applied to each software development project?	75.4%	71.3%	−4.1%
3	Q2.1.10.	Are *standards* applied to the preparation of unit test cases?	30.8%	45.5%	14.7%
3	Q2.1.11.	Are code maintainability *standards* applied?	70.8%	81.2%	10.4%
4	Q2.1.12.	Are internal design review *standards* applied?	28.5%	43.6%	15.1%
4@	Q2.1.13.	Are code review *standards* applied?	26.2%	42.6%	16.4%
2@	Q2.1.14.	Is a *formal procedure* used to make estimates of software size?	15.4%	23.8%	8.4%
2@	Q2.1.14.*x*.	Is a *formal procedure* used to make estimates of software complexity?	8.5%	9.9%	1.4%
2@	Q2.1.15.	Is a formal procedure used to produce software development schedules?	55.4%	40.6%	−14.8%
2@	Q2.1.16.	Are *formal procedures* applied to estimating software development cost?	30.0%	34.7%	4.7%
2	Q2.1.17.	Is a *mechanism* used for ensuring that the software design teams understand each software requirement?	44.6%	43.6%	−1.0%
3	Q2.1.18.	Are man–machine interface *standards* applied to each appropriate software development project?	43.8%	54.5%	10.7%
2.2. Process metrics					
2	Q2.2.1.	Are software staffing profiles maintained of actual staffing versus planned staffing?	9.2%	23.8%	14.6%
2@	Q2.2.2.*m*.	Are profiles of software metrics (size or complexity) maintained over time for each software confignration item?	16.2%	24.8%	8.6%
3@	Q2.2.3.	Are statistics on software design errors gathered?	11.5%	16.8%	5.3%
2@	Q2.2.4.1*s*.	Are statistics on software code errors gathered?	7.7%	16.8%	9.1%
2@	Q2.2.4.2*s*.	Are statistics on software test errors gathered?	9.2%	22.8%	13.6%
4@	Q2.2.5.	Are design errors projected and compared to actuals?	23.8%	38.6%	14.8%
4@	Q2.2.6.1*s*.	Are code errors projected and compared to actuals?	19.2%	34.7%	15.5%
4@	Q2.2.6.2*s*.	Are test errors projected and compared to actuals?	23.8%	48.5%	24.7%
2	Q2.2.7.	Are profiles maintained of actual versus planned software units designed, over time?	11.5%	23.8%	12.3%

Level	Question ID and description		1996 survey	This study	Difference
2	Q2.2.8.	Are profiles maintained of actual versus planned software units completing unit testing, over time?	18.5%	35.6%	17.1%
2	Q2.2.9.	Are profiles maintained of actual versus planned software units integrated, over time?	12.3%	33.7%	21.4%
2	Q2.2.10.	Are target computer memory utilization estimates and actuals tracked?	50.0%	61.4%	11.4%
2	Q2.2.11.	Are target computer throughput utilization estimates and actuals tracked?	32.3%	47.5%	15.2%
2	Q2.2.12.	Is target computer I/O channel utilization tracked?	39.2%	44.6%	5.4%
4@	Q2.2.13.1*s*.	Is design *review coverage* measured and recorded?	37.7%	46.5%	8.8%
4@	Q2.2.13.2*s*.	Is code *review coverage* measured and recorded?	22.3%	42.6%	20.3%
4@	Q2.2.14.	Is *test coverage* measured and recorded for each phase of functional testing?	31.5%	55.4%	23.9%
3@	Q2.2.15.	Are the action items resulting from design reviews tracked to closure?	43.8%	56.4%	12.6%
2	Q2.2.16.	Are software trouble reports resulting from testing tracked to closure?	52.3%	55.4%	3.1%
3@	Q2.2.17.	Are the action items resulting from code reviews tracked to closure?	39.2%	51.5%	12.3%
2	Q2.2.18.	Is test progress tracked by deliverable software component and compared to the plan?	45.4%	51.5%	6.1%
2	Q2.2.19.	Are profiles maintained of software build/release content versus time?	32.3%	32.7%	0.4%
2.3. Data management and analysis					
4@	Q2.3.1.	Has a managed and controlled *process database* been established for process metrics data across all projects?	18.5%	36.6%	18.1%
4@	Q2.3.2.	Are the *review data* gathered during design reviews analysed?	43.1%	56.4%	13.3%
4@	Q2.3.3.	Are the error data from code reviews and tests analysed to determine the likely distribution and characteristics of the errors remaining in the product?	30.8%	32.7%	1.9%
4@	Q2.3.4.	Are analyses of errors conducted to determine their *process* related causes?	44.6%	32.7%	−11.9%
5@	Q2.3.5.	Is a *mechanism* used for error cause analysis?	38.5%	45.5%	7.0%
5@	Q2.3.6.	Are the error causes reviewed to determine the *process* changes required to prevent them?	56.2%	65.3%	9.1%
5@	Q2.3.7.	Is a *mechanism* used for initiating error prevention actions?	53.1%	55.4%	2.3%
4@	Q2.3.8.	Is *review efficiency* analysed for each project?	17.7%	17.8%	0.1%
4	Q2.3.9.	Is software productivity analysed for major *process* steps?	17.7%	34.7%	17.0%
2.4. Process control					
2@	Q2.4.1.	Does senior management have a *mechanism* for the regular review of the status of software development projects?	26.9%	38.6%	11.7%
4@	Q2.4.2.	Is a *mechanism* used for periodically assessing the software engineering *process* and implementing indicated improvements?	36.9%	43.6%	6.7%
3	Q2.4.3.	Is a *mechanism* used for identifying and resolving system engineering issues that affect software?	26.2%	40.6%	14.4%
3	Q2.4.4.	Is a *mechanism* used for independently calling integration and test issues to the attention of the project manager?	27.7%	32.7%	5.0%
2	Q2.4.5.	Is a *mechanism* used for regular technical interchanges with the customer?	30.8%	33.7%	2.9%

Level	Question ID and description		1996 survey	This study	Differ-ence
3@	Q2.4.6.	Is a *mechanism* used for ensuring compliance with the software engineering *standards*?	38.5%	46.5%	8.0%
2@	Q2.4.7.	Do software development first-line managers sign off on their schedules and cost estimates?	16.9%	18.8%	1.9%
3	Q2.4.8.	Is a *mechanism* used for ensuring traceability between the software requirements and top-level design?	26.9%	43.6%	16.7%
2@	Q2.4.9.	Is a *mechanism* used for controlling changes to the software requirements?	58.5%	56.4%	−2.1%
4	Q2.4.10.	Is there a formal management *process* for determining if the prototyping of software functions is an appropriate part of the design process?	24.6%	28.7%	4.1%
3	Q2.4.1 1.	Is a *mechanism* used for ensuring traceability between the software top-level and detailed designs?	33.1%	41.6%	8.5%
3@	Q2.4.12.	Are internal software design reviews conducted?	23.1%	42.6%	19.5%
3@	Q2.4.13.	Is a *mechanism* used for controlling changes to the software design?	57.7%	43.6%	−14.1%
3	Q2.4.14.	Is a *mechanism* used for ensuring traceability between the software detailed design and the code?	29.2%	40.6%	11.4%
3	Q2.4.15.	Are formal records maintained of unit (module) development progress?	34.6%	50.5%	15.9%
3@	Q2.4.16.	Are software code reviews conducted?	20.8%	36.6%	15.8%
2@	Q2.4.17.	Is a *mechanism* used for controlling changes to the code? (Who can make changes and under which circumstances?)	50.0%	50.5%	0.5%
3	Q2.4.18.	Is a *mechanism* used for configuration management of the software tools used in the development process?	43.8%	53.5%	9.7%
3@	Q2.4.19.	Is a *mechanism* used for verifying that the samples examined by Software Quality Assurance are truly representative of the work performed?	23.8%	28.7%	4.9%
2	Q2.4.20.	Is there a *mechanism* for assuring that regression testing is routinely performed?	15.4%	38.6%	23.2%
3@	Q2.4.21.	Is there a *mechanism* for assuring the adequacy of regression testing?	8.5%	20.8%	12.3%
3	Q2.4.22.	Are formal test case reviews conducted?	28.5%	47.5%	19.0%

@ indicates a 'critical' key practice.
m indicates the description of this key practice was modified by the author.
x indicates this key practice was created by the author.
s indicates the description of this key practice was split into two key practices by the author.