

Accumulating the Body of Evidence for The Payoff of Software Process Improvement

Herb Krasner, President, Krasner Consulting

1. WHAT IS SOFTWARE PROCESS IMPROVEMENT (SPI) ?

Improvements in the software process have been going on for several decades. Under the rubric of software engineering, the primary thrust has been better discipline, methods and automated technology to support software development. SPI guided by organizational process maturity principles has emerged in the U.S. in the last 12 years, the charge being led by the SEI (Humphrey, 1989), and now internationally by the emerging ISO SPICE initiative.

Achieving a mature process establishes a project management and engineering foundation for quantitative control of the software process, which becomes the basis for continuous process improvement. An organization with a mature process will take full responsibility for executing its planned commitments.

In a fully mature software organization, the following holds:

- Quality is defined and therefore predictable,
- Costs and schedules are predictable and normally met,
- Processes are defined and under statistical control,
- Roles and responsibilities are clear - interdisciplinary communications are good,
- Software measurement discipline is practiced,
- Success rides on organizational capability, and individual talent flourishes within that,
- Technology that supports the process is used effectively,
- Staff development practices for software talent growth are established and effective,
- Corporate success factors recognize "core competency" of software as important and software strategy is aligned with the business strategy,
- Management and staff are committed to total quality and continuous improvement, and results are obvious.

The primary mechanism for achieving maturity and bottom line results in a specific organization is a focused, structured, and institutionalized program of continuous software process improvement. This requires the cyclic application of a model-based improvement method. In addition to a well defined set of improvement objectives, such a method may use one or more of several popular goal oriented models for guiding the improvement program.

A successful systematic SPI program requires:

- Well defined objectives - some of which are measurable and measured,
- A method for catalyzing and institutionalizing the improvement program in an organizational setting,
- One or more goal/maturity models for guidance,
- Best practice examples and benchmarks to draw from,
- An organizational commitment to action in the form of an improvement road map that is defined, resourced and followed,
- Expertise in process diagnosis, culture change tactics, process problem solving, etc.
- A set of champions/change agents that can sponsor, commit to and effectively implement a planned improvement program.

An SPI road map will focus on targeted organizational improvement areas at all organizational levels. It is assumed that some form of projects are being done, where a project could be anything from a one

person custom product variation effort to a large multi-team contract systems development. It is simply a matter of defining a scope of work to be managed as a project unit.

The following areas are typically addressed: within projects focusing on better project planning and control, across projects focusing on cross project coordination/learning, and at the business unit/company level focusing on TQM issues. A foundation software measurement program supports all areas.

1. Better project planning and execution - The basic benefits of improvements in this area are better predictability and control of projects, and the improved ability to recognize off track or out-of-control projects earlier. Typical improvements made include:

- a rigorous planning and tracking framework is established for projects,
- changes to requirements and associated objects are identified and managed,
- responsibility is taken for planned commitments (individual, team, project).

2. Better cross project learning, and coordination - The basic benefits of improvements in this area are improved people portability and better organizational communication, learning and efficiencies. Typical improvements include:

- a common process framework is defined and reused,
- educational vehicles deliver common training needs,
- teamwork is enhanced via structured techniques,
- focused responsibility for process improvement is established,
- process-based technology usage is facilitated,
- lessons learned, and planning data are collected and passed from old to new projects,
- common standards and system architectures are created to facilitate reuse.

3. Foundation software measurement program - The benefit of improvement in this area is the creation of a factual basis for dealing with issues in the other three areas. Measurement provides quantitative feedback that reinforces and even accelerates improvements. Typical improvements include:

- customer satisfaction, software quality, software resource consumption, and schedule delivery performance are all measured,
- software rework, cost of conformance to requirements, problem resolution effectiveness are all measured,
- team effectiveness is measured,
- the effectiveness of implemented improvements is measured.

4. Focus on total quality and continuous improvement - The basic benefits of improvements in this area are common values, context and lexicon, and a culture that encourages constant improvements and values staff and the results of previous developments as assets to the organization. Typical improvements include:

- customer satisfaction feedback is aligned with internal directions,
- people are treated as assets - hiring and career development is aligned with core competency requirements and project needs,
- leadership and empowerment is facilitated by the establishment of a quality oriented culture,
- a multidisciplinary systems design process is integrated with the software process (in those organizations that deliver more than software),
- a continuous improvement culture is established.

Specific improvement objectives and a fully elaborated road map will point to specific processes to be improved within a given organization. The highest leverage areas for improvement are best identified by the results of rigorous empirical evaluation of recurring software problems (e.g. Curtis, Krasner and Iscoe, 1987).

2. WHAT IS PAYOFF AND WHY THE NEED FOR DATA ?

By *payoff* we mean the reward/result/benefits for doing improvements, usually in quantitative terms (e.g. dollars). In some cases, the non-quantifiable payoffs (e.g. pride in work, company reputation) may be even more important. If the costs of doing SPI are viewed as an investment, then the payoff is expressed in a temporally-shifted, return-on-investment (ROI) model. The observable payoff is delayed in time, sometimes by as much as several years, due to the complexities of deployment, institutionalization and culture change. This delay can stress the attention span of short term, delivery oriented, crisis driven, middle managers whose support is necessary for SPI to flourish.

The payoff that will be of most interest to a specific organization will depend on that organization's business objectives for doing SPI.

Payoff data is needed for at least three reasons:

- data from external sources is needed to justify the beginnings of an SPI program until significant internal results can be demonstrated,
- data from internal sources is needed to close the feedback loop on the impact of internal improvements, which then helps to sustain the program,
- to validate process maturity based approaches that may have started on good faith but are now being squeezed by financial conditions.

The latter will continue to be true until the pursuit of SEI Maturity Level X becomes less of a process hygiene factor and more of a certification requirement for doing business (as in the ISO community).

The costs of SPI are usually more obvious, since they consume identifiable resources. These costs can be easily measured if flexible effort collection mechanisms are already in place, and if SPI work can be separated from business-as-usual activities. The magnitude of typical costs for SPI depend on organizational size, type and the scope of changes made. A guideline cited by the SEI states that SPI costs should be 3-5% of an organization's total software development costs. Sometimes, these costs are deployed throughout an organization, and not necessarily concentrated within a software engineering process group (SEPG). In an organization that has SPI embedded in its culture, the distinction between improvements and normal discipline may be blurred. Since this paper is primarily about payoff, the discussion of the costs of SPI will be treated in the context of overall ROI analysis.

3. PREVIOUS REPORTS OF SPI PAYOFFS

Prior to the crystallization of the process maturity movement by the SEI in the late 1980s, quantitative payoffs for specific process improvements were reported in a haphazard fashion in a wide variety of places. For example, the payoff for using a formal inspection technique within a business unit setting has been reported in several different conference proceedings, in books, in refereed journals, and in various newsletters. The same is true for various software engineering methods and tools.

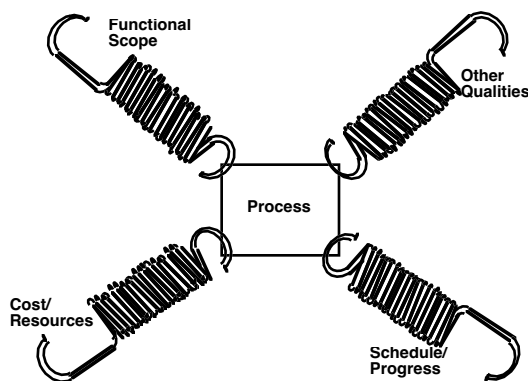
The earliest report of the potential payoff from a holistic maturity-based process improvement program came when Krasner, 1990 reported on the results of a set of formal SEI assessments in 5 major divisions of Lockheed (these assessments included a snapshot of 28 large projects). Data collected showed that projects at higher levels of maturity were 3 to 5 times more productive than Level 1 projects. A tentative correlation was hypothesized between CMM levels and other key project performance factors (quality, cycle time, effort, etc.). This early report received considerable attention until 1991-92, when reports of the Hughes, Raytheon and HP programs were published.

At about the same time, the NASA SEL (McGarry, 1990) and the IBM Houston Shuttle program (Rhone, 1990) started to publish reports of the payoffs resulting from their improvement programs.

In 1993-96, along with the case studies presented here, numerous other reports of the results of systematic SPI programs have been published, including: Schlumberger, IBM Toronto, Litton Data Systems, Rockwell, Oklahoma Air Logistics Center and the TI Systems Group. These are all summarized in Table 1 and are included in the reference section at the end of this paper. In each report, the notion of payoff is defined and measured differently, and typically reflects improvement in at least one dimension of the software development project challenge model shown in Figure 1.

In this model, the challenge of software development is represented as the "push-pull" attempt to control and manage the four major project outcome factors: cost, schedule, scope and quality; and one internal factor: the process. The outcome factors are viewed as "springs" that are dynamically compressed or decompressed during the life of a project. The process is viewed as the rubber band holding those springs together, and whose shape must change as changes occur in the manipulation of each factor.

Figure 1 - Software Development Project Challenge Model



The payoff metrics that have been reported in published sources include:

- cost metrics such as: ratio of actual vs. planned cost of work performed, engineering hours saved, productivity increases, non-conformance/rework cost reduction, and reuse increases,
- quality metrics such as: defect density or rate reduction, % defects discovered by customer reduced, early defect detection rate improvements, complexity/size growth control, and operational reliability improvements,
- time based metrics such as: cycle time reduction, on time delivery rates improved, schedule slippage rates reduced, and requirements validation cycle time decreased,
- process metrics such as: process maturity level increases, and improvements in defect removal effectiveness.

Summary Reports of Industry Wide Payoff

In mid 1993, Broadman and Johnson (1994) undertook a research project for the U.S.A.F. to investigate the existence of ROI data for SEI CMM-based SPI initiatives. The study included an analysis of 20 questionnaires, 22 interviews and a literature search that turned up 100 sources. They found that even though many definitions of ROI existed, numerous reports of quantified results existed in areas such as: productivity, quality, cost, schedule, and effort. They also noted that the number of metrics collected increased with maturity level; for example, twice the number of metrics on average were collected by Level 3 organizations over Level 1 organizations. ROI data was empirically linked to the CMM in the areas of productivity, quality and schedule.

In early 1994, the Empirical Methods Project at the SEI began a study to investigate the results of CMM-based software process improvement efforts. Survey data was collected from 13 leading organizations representing DoD contractors, commercial companies, and government organizations. This included: Bellcore, Bull, GTE, HP, Hughes, Loral FSD, Lockheed, Motorola, Northrup, TI,

Siemens, and Tinker AFB. The study attempted to identify SPI impact in the areas of: cost, productivity, schedule, quality and ROI. Sparse data was reported in each area, but the data generally showed substantial gains in productivity, time to market, quality, reduced rework, and ROI (Herbsleb, 1994). In Herbsleb and Zubrow, 1994, the SEI described their study results of the 13 leading SPI organizations surveyed.

Their summarized results indicated that:

1. average number of years engaged in their SPI program was 3.5,
2. average cost of SPI per software professional per year was \$1375,
3. annual productivity gain was 37% (based on 4 reports),
4. annual reduction in time to market was 19% (based on 2 reports),
5. annual reduction in post delivery defect reports was 45% (based on 5 reports),
6. average ROI for SPI was 5.7 to 1 (based on 4 reports).

In future studies, the SEI may attempt to empirically validate the CMM. Programs based on models other than the CMM have not received the attention the SEI approach has. Results are therefore sparsely found in the open literature.

Even though SPI Payoff data is being reported, these reports have not generally included results in the areas of better cross project learning, organizational coordination, and total quality improvement. Such payoff reports might eventually include such things as: reputation for excellence, competitiveness, timeliness to market, organizational learning efficiency, process architecture effectiveness, and customer satisfaction rates. The current lack of such payoff information may be due to the fact that there are not many high maturity organizations, and those that do exist either don't have such data yet, or will not report on it.

4. CASE HISTORIES OF SPI PAYOFF

There is still considerable demand for data on successful SPI programs, as the SPI movement proliferates to new sectors and companies. Therefore, several short case studies of successful SPI programs serve as valuable examples.

A number of successful organizational SPI programs have reported the payoffs from their efforts in the open literature. In the following section, we highlight a representative set of programs, so that the potential for payoff in specific situations may be seen. These organizations may be considered as SPI benchmarks for the industry - they were chosen because their payoffs have been reasonably well documented. We briefly describe the SPI programs of:

1. NASA SEL,
2. IBM (now Loral) Federal Systems- Space Shuttle Program,
3. Hewlett Packard Corporation,
4. Raytheon Equipment Division, and
5. Motorola India Electronics Laboratory

These organizations have had SPI initiatives for at least 5-10 years. In these cases, the payoffs took several years to become observable through measurements.

NASA SEL

The NASA Software Engineering Laboratory (SEL) represents a consortium involving NASA Goddard Space Flight Dynamics Division, the Computer Science Department of the U. of Maryland, and Computer Sciences Corporation's Software Engineering Operation. This lab houses software developed and maintained for flight dynamics (e.g. orbit, mission analysis) which is characterized as scientific/mathematical with tight delivery constraints imposed by launch schedules. They have had an ongoing SPI program for 18 years, during which many experiments, case studies and trials have

been performed. Their SPI approach is characterized as domain-specific, product-based, and Quality Improvement Paradigm (QIP) based. They have been evaluating changes and fine tuning life cycle processes, evaluating significant changes to technology/methods (e.g. Cleanroom), and providing support to the development organizations to help them with process definition, modeling, etc.

The results they have achieved are within the context of a dynamic environment in which the complexity of systems doubled, the number of requirements doubled, and the code size tripled. In spite of that, they report the following payoffs:

- development error rates were reduced by 75% (from 4.5 to 1/KSLOC),
- the cost per SLOC decreased,
- predictability of cost, schedule and quality improved,
- reuse went from 20% to 79% on similar systems,
- overall reduction of 55% in total software costs was realized.

See Heller and Page, 1993, McGarry, 1993, Basili and Scott, 1994 and Krasner, 1995 for more information. In 1994, the NASA SEL organization was awarded the first IEEE Computer Society Award for Software Process Achievement. A presentation on their accomplishments was given by Frank McGarry at the August, 1994 SEI Software Engineering Symposium, and has been described in a technical report from the SEI (McGarry, et. al., 1994).

IBM Federal Systems Company (then Loral and now Lockheed Martin) - Space Shuttle Program

This organization has been developing on-board and ground support software for NASA shuttle missions for almost 18 years. Approximately 300 software professionals work in this organization that has produced a base system of almost 10M SLOC. The current version of the system has .5 M SLOC on-board software, and about 1.7 M SLOC of ground support software. In response to an almost continuous flight schedule, they delivered a new release every 4-8 months (every 12 months during the last few years), with each new version consisting of from 10-53 KSLOC new code. The program's quality goal is to produce error free software. Their SPI program consists of: performing assessments, defining their process (using ETVX), formal process training, adopting advanced software engineering methods (Waterfall+Spiral), conducting rigorous in-process inspections, performing defect cause analysis and prevention, and utilizing specialized testing methods and tools. The following payoffs from their program have been cited:

- ability to predict costs within 10%,
- only 1 deadline missed in 15 years,
- learned the relative cost of fixing defects ranged from 1x during inspection to 13x during system test and 92x during operation,
- productivity 180-200 SLOC/MM; maintenance cost of ~ \$20/SLOC (1/2 of the maintenance costs of other IBM software shops),
- early error detection (%) went from 48 to 95% from 1982-1993,
- reconfiguration time (weeks) went from 11 weeks to 5 weeks from 1982 - 1985,
- product error rate (defects/KSLOC) went from 2.0 to .01 from 1982 to 1993.

In 1989 they were evaluated by a NASA-led, SEI-trained team which determined that they had many Level 5 characteristics - the first organization to be recognized as such. For more information see Krasner, 1994, Paulk, et. al, 1995, Humphrey, 1991; Gomez, 1994; and Billings, et al, 1994.

Hewlett Packard Corporation

Hewlett Packard is in many businesses ranging from printers, medical systems, computer workstations, etc. In the mid 80's, they established a corporate software quality goal of achieving a 10X reduction in software defects. Even though software size and complexity was growing exponentially in many product lines they have achieved significant results. Their program focused on improvements in: process, systems definition and design, configuration management, inspections, maintenance, and reuse. The payoffs from their program have been cited as:

- corporate aggregate defects rate went from 1/KSLOC to .1/KSLOC from 1988 to 1993 (10X objective achieved),
 - inspections in wide-spread use are saving 5 hours per inspection on average (saved \$20 million in last year alone due to inspections),
 - average time to fix a defect was cut in half in one business unit.
 - reduced time to market by 5X over 5 year period in another business unit.
- All of these were achieved in spite of large growth in software size and complexity.

For more information see Platt, 1993, Grady, 1992 and Grady and Van Slack, 1994.

Raytheon Equipment Division - Software Systems Laboratory (now RES)

The Raytheon Equipment Division Software Systems Laboratory (SSL) began their SPI initiative in earnest in August of 1988 after an extensive planning phase that began in 1987. This report discusses the results from 1988 until 1994 when their organization was merged with two others into the newly created Raytheon Electronic Systems (RES) Software Engineering Laboratory. The SSL was about 600 software engineers working on defense electronics systems for government customers during the span of the period reported here.

Their SPI goals included the improved predictability of software development, for which they wanted quantitative measures of improvements made. To address this challenge, they selected an approach for measuring ROI based on the business goal of reducing the amount of rework involved in developing software. This Cost of Software Quality approach was adapted from Crosby, 1984. They later supplemented this approach with the analysis of productivity on projects, and cost at completion as compared to original budget (CAC/Budget) to measure predictability of project performance. Defect density analysis was used to measure overall software product quality. They continue to use these four basic measures on their projects today: cost of quality, productivity, predictability, and product quality, to monitor the impact of their SPI initiative.

Using a tailored CMM-based approach, SSL (and subsequently RES) have made steady process maturity progress. They cite the following achievements:

- SSL made SEI level 3 in late 1991,
- all new projects are required to operate at Level 4, which they reached in 1995
- cost/benefit demonstration and full institutionalization of level 4 behavior is being pursued
- a goal to achieve Level 5 has been set
- their defined process has been institutionalized, and subsequently transferred to other Raytheon organizations in the U.S. and beyond
- in 1995, they received the IEEE Computer Society Software Process Achievement Award - only the second organization in the U.S. to do so

The annual investment into the SPI initiative was at a level of about 1 million dollars per year. Since there were about 600 software engineers in the SSL, this amounted to about \$1666 per software engineer per year, invested into SPI efforts. Over the lifetime of the initiative (1988-94) the following effects were measured:

- rework was reduced from 40% of development cost down to about 10%
- productivity of the development staff increased by a factor of 170%
- predictability of project budget and schedule has been reduced from about 1.41 (40% overrun on average) to a range of +/- 3%
- product quality (defect density) went from about 17 TRs/KDSI to 4 TRs/KDSI (TRs are software trouble reports, KDSI means thousands of delivered source instructions)

In 1990 the ROI was 7.7 to 1. This is based on a total savings of \$4.48 million utilizing a total investment of \$.58 million that year. The savings coming from reduced rework measured on projects. By 1994, 18 completed projects were represented in their ROI database. In addition (not included in the ROI), in 1991 they received a \$9.6 million schedule incentive award for bringing one of their major

projects in 6% under budget. . That award alone more than paid for the SPI initiative (1 million X 7 years = 7 million).

They cite the following primary reasons for their successful SPI initiative:

- vision and active commitment from management
- sponsorship and support
- improvements clearly and continually demonstrated business benefits to projects
- careful consideration of the culture, and how to change it
- run from within the ranks of the software organization leading to ownership and empowerment

Specific leverage points were identified as the major contributors to their sustained SPI growth, these focal points were:

- system and requirements definition practices
- inspections
- integration and qualification testing
- development planning and management controls
- training
- pathfinding

For more detailed information about the SSL SPI program see Haley, et al, 1995.

Motorola India Electronics, LTD (MIEL)

MIEL was established as a wholly owned subsidiary of Motorola in Bangalore, India in 1991 with 22 staff members in order to develop software for internal Motorola engineering use in a variety of application domains. This was part of a company initiative to create effective, off-shore software factories. MIEL had the opportunity to create their process from a clean sheet a paper based on software factory principles using the CMM as a guideline. Therefore they created MIEL with the intent of initially being a Level 3 organization with a well defined process, and a firm commitment to institutionalized common practices. They recruited staff who were willing to follow their process, and with a relatively high turnover rate, this was critical to long term success. This approach served as a model for other Motorola international software factories. By 1995, MIEL had grown to about 250 software engineers working in several major product lines.

Process innovation was not their SPI objective, with most of their practices based on commonly accepted standards (e.g. IEEE Software Engineering Standards Collection). Their processes are highly integrated. They did create and follow their process religiously, keeping all 18 CMM KPAs in their vision as they evolved. Their biggest hurdle was in going from level 1 to level 2. The rest of their growth was deemed (by them) to be a logical conclusion of the initial transformation. They were assessed (by trained Motorola assessors) at SEI Level 5 in December of 1993. Subsequent visits to MIEL by SEI luminaries acknowledged the existence of high process maturity characteristics.

The improvement results MIEL achieved over the 1991-1994 period were:

- delivered about 2 million LOC over this period,
- productivity of more than 43 Non-Comment Lines Of Code (NCLOCs) per staff-day,
- productivity increased about 3.5X as they went from Level 3 to Level 5,
- post release quality of better than 2 defects per 930 KLOCs (50% with no known defects),
- in-process quality of less than 1.1 defects per KNLOCs,
- less than 3.75% rework due to in-process faults
- cost of error detection and correction less than 17% of total costs,
- estimation accuracy on project schedule and effort better than 90%, and
- cycle time was reduced by 40% in a 12 month period.

These quantitative results are based on 42 data elements that MIEL regularly collects from their projects. Their causal analysis showed that most defects resulted from process non-compliance. They

encourage process improvement changes but insist that all changes be justified with data as well as be controlled (piloted first, deployed after success is demonstrated).

The CMM aspects that had the most early impact on their overall results were, in priority order:

- product engineering,
- peer reviews, and
- configuration management.

Regarding peer reviews, inspection was one of the biggest contributors to quality, and was a strong support technique for the cleanroom approach. Inspection was also an excellent technique for collecting software metrics. If a review becomes formal, it is an excellent point for data collection. They also maintain a risk repository for all identified risks, and make decisions based on overall risk levels. They define quality as compliance with implicit and explicit requirements of the customer. Recognizing that changing requirements is the prerogative of the customer and will happen, so when requirements change the plans must change accordingly.

MIEL's approach to implementing SPI was:

1. develop a first draft of their development process,
2. ensure that the organization structure was "in synch" with the actual process as practiced (not just to satisfy the CMM),
3. assign key persons to all key process areas [comment: they did not address only one level at a time],
4. select pioneers,
5. conduct integrated induction training (42 hours of training of which process, tools and techniques were taught by the managers of the organization),
6. require total compliance with process,
7. demonstrate the benefits early,
8. improve the process based on experience, and
9. grow capabilities incrementally by iterating on this approach

As a result of their process development, MIEL believes the following are some of their "best practices"

1. Integrated induction training;
2. Career development plan in which each engineer works in each of the functional areas (e.g., engineering, QA, systems, test, CM, project management) before they can attain their highest job rank;
3. Cost of quality measurement system (that supports the claim that it costs less to detect and correct a defect in the stage in which it is created than at a later stage) and its use to evaluate piloted changes;
4. Phase and project post mortems;
5. Re-use;
6. Data modeling, including the number of hours of testing necessary to reach 6 sigma given test time and an estimate of latent defect density;
7. Estimation techniques;
8. Senior management reviews which include managers and software professionals; and
9. Senior management review of all customer commitments.

For more information about MIEL's SPI program see IEEE Software, March, 1994, page 92, Srikant Inamdar (1994), and Curtis and Statz (1996).

Summary of case histories

The common themes that have been observed in these successful SPI programs are:

1. they are all in SPI for the long term;
2. they focus on software quality and related project performance issues;
3. they all have intensive measurements programs;

4. they all focus on the improvement of processes that lead to measured improvement toward business-based objectives; and
5. they all exhibit the sustained commitment needed to institute the changes.

The tables that follow contains a brief summary of other known cases of reported SPI payoff that were not highlighted in the above section. The reader interested in becoming an expert may wish to acquire and read the cited references.

Table 1 - Summary of US Organizational SPI Payoffs - pre 1996

Organization	Payoff Summary	References
Hughes	predictability improved as measured by cost performance index which went from .94 to .97 in 3 years, \$2 Million annual reduction in cost overruns	Humphrey, W., Snyder, T. and Willis, R. (1991)
Lockheed	Assessment project survey concluded that CMM Level 3 projects are 3-5 times more productive than Level 1 projects	Krasner, H. (1990); McConnell, S. (1993) (unattributed)
SEMATECH Equipment Supplier	Operational software reliability of process tool improved by 48X	Krasner, H. And Ziehe, T. (1995)
Litton Data Systems	76% less defects encountered in integration - inspections benefit	Dixon, S. (1994)
USAF Oklahoma City Air Logistics Center	SPI ROI of 6.35X	Lipke, W. and Butler, K. (1992)
IBM Toronto Lab	10X reduction in delivered defect rates, productivity up by 240%, rework reduced by 80%	Schwarz, J. (1993)
Rockwell	for 2 major projects: 625% improvement in post release defect reports, 97% pre-release defects detected, improvements in cost/schedule/quality performance indices, and award fees up to 93%	Selfridge, W. (1994)
Schlumberger	4X reduction in betatest bugs, ISO certification, early deliveries of products, open defect rate dropping as size and complexity increases	Wohlwend, H. and Rosenbaum, S. (1993); Lloyd, P. (1994)
Texas Instruments - Systems Group	60% productivity improvement over 2 years, 10X reduction in delivered defect rate over 3 years, 12% annual cycle time reduction	Hudec, J. and Suddarth, G. (1996)
Procace Corporation	cycle time reduction of 4.3X over 18 month period	Sudlow, B. (1994)
Computer Sciences Corp.	predictability improved, error rates reduced by 65%, cost per SLOC reduced slightly in spite of dramatic increases in size and complexity required	Heller, G. and Page, G.(1993)
USAF survey	positive corrolation was determined between increasing CMM levels (1-3) and cost and schedule performance on 13 previous large USAF ASC/ESC contracts	Lawlis, P., Flowe, R. and Thordahl, J. (1995)

Table 2 - recently reported payoff summaries in the US - 1996-97

Organization	Payoff Summary	References
Boeing Info. Systems	project estimates within 20% using historical data, CPK 38% better, defect containment effectiveness at 80%, cycle time improved 36%, staff support needs down 62%, staff size reduced 31%, customer satisfaction score up 10%, \$5.5 M saved in 1996 alone (1992-1996 results)	Vu, J., 1997
Boeing STS	customer satisfaction rated excellent, pre-release defect containment effectiveness at 99%, 31% reduction in rework-inspections benefit, employee satisfaction level from mean of 5.7 to 8.3, operational systems performance close to bullseye, level 5 process injected into new programs	Yamamura, G. and Wigle, G., 1997
Bellcore	defects 10X lower than industry average, customer satisfaction rates improved from 60 to 91% over 4 years, achieved 9 hr. cutover to add 888 to 800 system with no reported defects	Bellcore Press Release, Feb. 5, 1997
HP SESD	3X3 SPI program, 1 year benefits include: cycle time reduced by 33%, major open defects reduced from 4.6 to 1.6, fewer missed deadlines, ROI - 9:1	Lowe and Cox, 1996
Harris ISD DPL	2.5X productivity gain over norm, 90% defect rate reduction, cycle time down to 6-9 months	Robeson, D., Davidson, S. and Bearden, L., 1997
Motorola	3X productivity improvement, 3X cycle time reduction, 7X quality improvement, results from '92-'96 representing 85% of all products & released software, 75% of product development orgs. are >= level 3	Major, J., 1996
Motorola GED	On 34 current programs compared to baseline - each CMM level increases quality by 2X, significant decreases in cycle time as higher levels reached (2-7X), productivity increases of 2-3X at highest levels of maturity, 6.77X SPI ROI	Diaz, M. and Sligo, J., 1997
SAIC Health Tech.	50% improvement in customer satisfaction, 71% reduction in error rate, 12% annual improvement in developer productivity, production rate up 30%	Lane, J. and Zubrow, D., 1997

Table 3 - International payoff summaries - 1997

Organization	Payoff Summary	References
Siemens	cycle time cut in half in OEN & EWSD projects, 90% reduction in released defects, new process acceptance rate of 94%	Mobrin, J. and Wasterlid, 1997
Ericsson	product faults reduced in all phases, 60% reduction in operational faults over 4 years since 1993, delivery delays significantly reduced	Volker, A. and Wackerbarth, G., 1997
Thomson-CSF	CPI improved 17% in 2 years, SPI improved drastically, 12% cost reduction, reduced the cost of pre-test defect correction 4X, ROI of 3.6:1	des Rochettes, G.,1997

5. MEASURING PAYOFF

Payoff that can be quantified in dollars is relatively easy to show if you collect certain data, and are patient enough to observe the long term trends as improvements become institutionalized.

A simple model of payoff that can be used is:

$$\text{Payoff}(t+i) = [\text{old costs}(t) - \text{new costs}(t+i)] - \text{cost of improvements}(i-t),$$

$$\text{C/B ratio}(t+i) = [\text{old costs}(t) - \text{new costs}(t+i)] / \text{cost of improvements}(i-t),$$

where: t is a point in time when performance costs were baselined over a selected set of projects,
 $t+i$ is a new point in time when performance costs are remeasured on a current selected set of projects,
 $i-t$ is the interval of time in which improvement investments were spent, and new practices were learned.

This model can be used at the SPI program level or at the specific improvement level and yields a first-order approximation for payoff. This assumes that the benefits of SPI will show up in reduced costs of performance. This simple model also neglects the time shifting of the widespread impact of the improvements from the point at which the new behaviors are introduced, which could be as much as up to 2 years in a large organization. A standard pro-forma analysis can determine the break even point for the SPI program.

There are many issues that are routinely faced in building an organization specific cost-benefit argument for justifying an SPI program. A standard SPI ROI method would be useful, but does not currently exist. *How to* guidance is needed on how to create an ROI argument using an ROI model and case examples for gaining support and commitment from management for the initiation of an SPI program that requires upfront investment capital, and a multiyear timeframe prior to seeing the benefits. Gathering and analyzing data is stressed on the cost of sub-par performance aspects as: terminated projects, missed deliveries, schedule overruns, defect fixing, rework, customer complaints, growth and complexity trends. Presenting the case to management is done in terms of: current performance results, improvement opportunities, benchmarks from other companies, SPI action plans, and measurement approaches for evaluating SPI impact. This helps to establish a quantitative basis for making the SPI decision within an organization that likely has not had any meaningful data on software project performance in the past. Cost benefit determination requires establishing a baseline level of performance to compare against. Deciding what projects go into the baseline(s) and how often to rebaseline are key choices to be made, and should be done in conjunction with the

assessment-improvement cycle in mind. Improvement results indicators need to be intelligently interpreted because periodic baseline figures are developed from a set of projects that may have different: scopes, timeframes, domains, levels of complexity, organizational teams. In small organizations, a major project failure may skew the overall improvement results when normalized.

SPI costs are sometimes straightforward to compute, especially when resources are consumed or effort is spent on doing something differently than before. A simple metric is the additional \$\$ spent per software staff member on SPI. However, in some cultures, improvement is a natural part of doing the work, and can be difficult to separately compute. Benefits can be much harder to compute, but can be made visible if the measurement program is well defined. To the extent that all benefits can somehow be computed in dollars, we can show an overall cost benefit ratio. Aspects such as improved job satisfaction are not easily quantified by \$\$ (except in turnover statistics and the cost of expertise replacement - which is very seldom measured).

Improvements in average project performance relative to an established baseline, and precisely measured, and relevant to the improvement objective that was established, will be the most compelling and easiest to demonstrate.

Example ROI models that have been used to date often include measurements on:

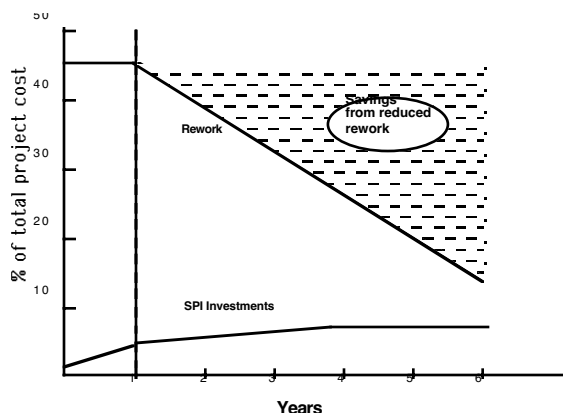
1. cost of software quality
2. cost of high reliability (Keller, 1992)
3. delivered defect rate trends (Keller, 1992)
4. cost and schedule performance indices (Lawlis et al, 1995)
5. productivity growth trends
6. cycle time reduction trends

Measuring performance costs can get quite complicated. Dion, 1993 makes use of one possible cost model for collecting this kind of information. Curtis and Statz, 1996 provided needed advice on the construction of specific cost-benefit models for SPI programs. In my opinion, in order to determine a true reflection of SPI payoff, improvements in at least process and product quality should be measured. Other measurements will depend upon the SPI program's specific goals (e.g. cycle time reduction, predictability improvements, etc.)

Process Quality

One way that process quality can be directly measured is by examining the amount of rework that occurs on projects. Rework percentages for immature organizations have been reported to be in the range of 40-60% of total software effort. Since rework takes away resources from new functionality development, it can be directly tied to business measures such as customer needs or requirements satisfaction.

Figure 2 - Process Quality Measurement



If project and improvement costs are collected and analyzed in typical work breakdown structure (WBS) categories with rework and SPI investments added, then rework % can be plotted over time as in the above Figure 2. A payoff is indicated by computing the savings accrued from reduced rework over time, less the cost of the improvement investments. An excellent example of this can be found in Dion, 1993.

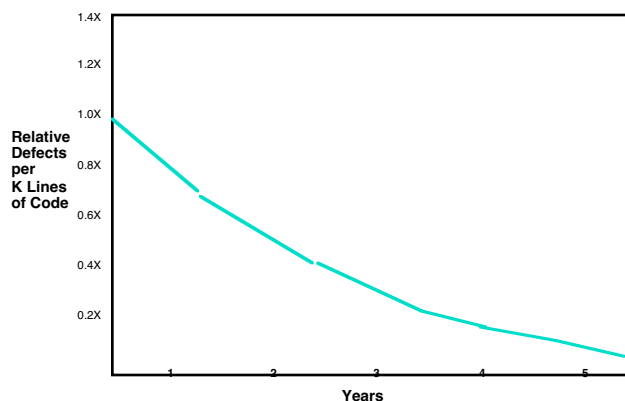
Measurements of process quality can be directly tied to specific process improvement implementations.

Product Quality

Process improvement for the sake of process improvement alone will not be sustainable in the long run. The changes must show up as improvements in the products or systems being developed. One simple measure of product quality that is frequently used is delivered defect density. The average trend over a number of projects within an organization tells us much about whether institutionalized process improvements are paying off over time.

Figure 3 is an extrapolation based on the experiences of IBM, HP, and Motorola. It shows how average defect density can be tracked over a multi-year SPI program, once it is baselined. An excellent example of this can be found in Platt, 1993.

Figure 3 - Product Quality Trends

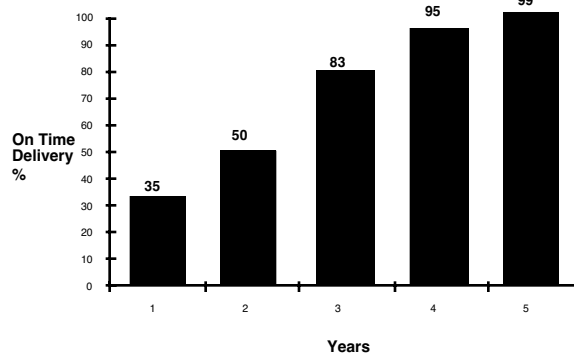


Underlying this simple chart is an organization wide data collection and analysis scheme. This scheme precisely defines for that organization the meaning of "software defect" and "line of code". Analysis procedures describe legal variations and how data is combined to form overall results. In some cases, an organization wide defect database and tracking system are in place that facilitate uniformity. Product quality measures can be directly tied to customer and requirements satisfaction results, thus making SPI a part of achieving business objectives.

Project Predictability

Project predictability can be measured in several ways by examining the trends in: actual versus planned progress, on time delivery rates, and schedule slippage rate. Figure 4 shows an example of the improvement in the percentage of projects that deliver their software products on time over a 5 year period. An average of actual costs vs. budgeted cost of work performed can be similarly plotted. As an organization's ability to more accurately plan their work gets better, these metrics get better. See Humphrey, et al (1991) and Wohlwend and Rosenbaum, 1993 for good examples of this.

Figure 4 - On time product delivery trends



Other measures of payoff

Other important areas of payoff have not yet been reported, and there are numerous qualitative benefits of SPI that are recognizable but not yet measurable. Training and other human resource development efforts create skill depth and breadth. We have not yet seen reports of SPI payoff in this area. Customer satisfaction is measurable, and is often measured for fielded products, yet we have not seen reports of SPI payoff in this area.

6. CONCLUSIONS

Many companies in many different business sectors are reporting successful SPI programs. They report ROI figures between 5 to 1 and 9 to 1, with approximately a two year lag between the investments and the observable benefits - but these ROI figures have different underlying measures. The amount of lag usually depends on organization size, type and scope of improvements attempted.

Several notable systematic SPI programs have emerged which may serve as models for the remainder of the U.S. software industry. These have been summarized here. However, many formal SPI programs which attempted to start in the late '80's faltered soon after a formal assessment was performed (2/3 reportedly died), are now in decline or perhaps will soon fail. This may be happening because of: a flawed strategy, a lack of commitment, lack of follow through, not measuring improvements, or lack of crisp SPI objectives that may not have been tied to business objectives. Newer SPI programs may be more likely to succeed because they can learn from the mistakes of the pioneers, and explicitly represent that process maturity is a means to an end. For example, Krasner and Ziehe, 1995 describe the results of the SEMATECH SPI initiative, which has caused improved operational reliability of the software embedded in semiconductor manufacturing equipment, using CMM-based SPI methods.

A defined SPI payoff function must tie business, engineering/development and personnel improvement objectives together in a meaningful way. SPI works best when payoff is demonstrated in all three of these areas simultaneously (Win, Win, Win). A payoff function which demonstrates the pursuit of higher quality software seems to do that. The following high level payoff impact areas should be considered when setting SPI program objectives:

- improving profitability, market share, time to market, competitiveness, productivity, competence, etc.,
- delivering ever-improving software quality to customers/users,
- maximizing the overall predictability, productivity and effectiveness of the software development process.
- reducing the amount of crisis-driven chaos by managing growth and change in scope, size and complexity,
- increasing the pride in workmanship, use of new skills learned and personal empowerment in decision making.

Management indicators within this payoff function would include measures of: product quality, process quality (rework, productivity), project predictability, skill base growth, and customer satisfaction.

Some of the biggest payoffs of SPI are expressible in human terms, not dollars. They might involve: better job satisfaction, pride in work, an increased ability to attract, retain and grow experts that will innovate, company reputation for excellence, etc.

For organizations that aren't fully mature yet, the existing data suggests that there is no reason not to start a well focused, well designed SPI program. Crisp SPI objectives should be tied to corporate business success factors. Executive sponsorship and demonstrated commitment by management will help motivate the program. Measurement discipline is a key to success - quantitative baselines must be established and then measured against. A focus on improved software quality seems to accelerate an SPI program, and should be targeted. Starting quickly with specific improvements that demonstrate early results raises the level of confidence in the program. Periodic re-appraisals/process audits reinforce and stimulate the program. Remember to be patient, since payoff measures resulting from culture changes may not be visible for years. Setting realistic expectations of upper management are important if the SPI program is to be perceived as credible. This article is intended to be useful for that purpose, but not to suggest what specific improvements should be undertaken by any specific organization.

The question of how to get started with an SPI program is often asked. The advice usually given is to start small, demonstrate a success and then build momentum. Improving the basic understanding of software phenomena in small, low maturity organizations is achievable within a few months (Krasner, 1994c). A formal systematic SPI program should then be pursued in expanding cycles that consist of: establishing sponsorship for the program, baselining project performance, process maturity and current "as is" process definition, setting improvement goals, formulating and executing an improvement road map, measuring progress, adjusting the program as needed and continuing until a "world class" software shop is achieved.

But by any means - get going - the competition may be getting ahead.

The preponderance of data from companies that are successful at SPI suggests the possibility of an underlying theory that relates level of organizational process maturity and expected level of project performance (reflected in delivered quality, timeliness, cost, productivity, and/or rework). A speculative causal model has been developed that attempts to explain the relationship between the key factors that induce organizational change that then shows up in measurable organizational indicators (Krasner, 1994b).

7. REFERENCES

1. Basili, V. and Green, S. (1994), Software Process Evolution at the SEL, IEEE Software, Vol. 11, No. 4, July, 1994
2. Bellcore, 1997 - in Bellcore Press Release, Feb. 5, 1997, and SEPG97 BOF
3. Billings, C., Clifton, J., Kolkhorst, B., Lee, E., and Wingert, W. (1994), Journey to a Mature Software Process, IBM Systems Journal, vol 33, No. 1, 1994, pp. 46-61
4. Broadman, J. and Johnson, D. (1994), Measurement Programs: Does One Size Fit All, at the 8th Annual Conference on Improving Productivity in System Development, Scottsdale, AZ, Feb., 1994
5. CMM - Two publications that document the Capability Maturity Model for Software are: M. Paulk, B. Curtis, M. Chrissis, and C. Weber, *Capability Maturity Model for Software, Version 1.1*, Software Engineering Institute, Pittsburgh, PA, CMU/SEI-93-TR-024; and M. Paulk, C. Weber, S. Garcia, M. Chrissis, and M. Bush, *Key Practices of the Capability Maturity Model, Version 1.1*, Software Engineering Institute, Pittsburgh, PA, CMU/SEI-93-TR-025
6. Curtis, B. and Statz, J. (1996) Building the Cost-Benefit Case for SPI, 1996 National SEPG Conference Tutorial, Atlantic City, NJ, May 23, 1996

7. Curtis, B., Krasner, H. and Iscoe, N. (1988), A field study of the software design process for large systems. *Communications of the ACM* 31(11): 1268–1287.
8. des Rochettes, G., Five Years Experience in SPI:Lessons Learned, 2nd European SEPG Conference, June, 1997
9. Diaz, M. and Sligo, J., How SPI Helped Motorola, *IEEE Software*, Vol. 15, No. 5, September, 1997, pp 75-81
10. Dion, R. (1993) Process Improvement and the Corporate Balance Sheet, *IEEE Software*, 10(4), pp. 28-35, July, 1993
11. Dion, R. (1991) Elements of a Software Process Improvement Program, *IEEE Software*, July, 1992 (originally reported at the 2nd SEPG Workshop, Nov. 1990)
12. Dixon, S. (1994), Software Process Improvement Lessons Learned at Litton Data Systems, at the 6th National SEPG Conference, Dallas, TX, April, 1994
13. Gomez, E. (1994), Lessons Learned and Observations of a Level 5 Organization: The Space Shuttle Project, presentation to the Austin SPIN, February, 17, 1994, Austin, TX. Summarized in *Software Quality Matters*, Vol. 2, No. 2, Summer 1994, Newsletter of the UT Software Quality Institute, Austin, TX
14. Grady, R. (1992) *Practical Software Metrics for Project Management and Process Improvement*. Prentice-Hall, Englewood Cliffs, NJ, 1992
15. Grady, R. and Van Slack, T. (1994), Key Lessons in Achieving Widespread Inspection Use, *IEEE Software*, Vol. 11, No. 4, July, 1994
16. Haley, T., Ireland, B., Wojtaszek, E., Nash, D., and Dion, R. (1995), Raytheon Electronic Systems Experience in Software Process Improvement, CMU/SEI-95-TR-017, Carnegie Mellon University, SEI, November, 1995
17. Heller, G. and Page, G., Impact of a Process Improvement Program in a Production Software Environment:Are We Any Better?, in *Society for Software Quality Journal*, Nov. 93, pp. 1-8, (previously presented at the NASA SEL Workshop, Nov. 90)
18. Herbsleb, J. (1994), Results of Software Process Improvement Efforts: Work in Progress, at the 6th National SEPG Meeting, April, 1994, Dallas, TX
19. Herbsleb, J. and Zubrow, D. (1994) , Software Process Improvement: An Analysis of Assessment Data and Outcomes, in the *Proceedings of the SEI Software Engineering Symposium*, also Technical Report CMU/SEI 94-TR-13, August, 1994, Pittsburgh, PA
20. Hudec, J. and Suddarth, G. (1996), Experiences in Implementing Quantitative Process Management, in the *proceedings of the 1996 National SEPG Conference*, Atlantic City, NJ, May, 1996
21. Humphrey, W. (1989), *Managing the Software Process*, Addison-Wesley, 1989
22. Humphrey, W. (1991), Hardware and Software, in *Texas Instruments - Technical Journal*, Vol. 8, No. 3, May-June, 1991, pp. 5-12, Dallas, TX
23. Humphrey, W., Snyder, T. and Willis, R. (1991) Software Process Improvement at Hughes Aircraft, *IEEE Software*, 8(4), pp. 11-23, July, 1991
24. Krasner, H. (1990), The Payoff of Software Process Maturity, presented at the 2nd National SEPG Conference, Nov., 1990, Washington, DC; also republished (unattributed) in (a) Pietrasanta, A. (1991), A Strategy for Software Process Improvement, at the 9th Pacific Northwest Software Quality Conference, Oct. 1991, and (b) McConnell, S. in *Software Development Magazine*, Vol. 1, No. 1, pp. 51-57, July, 1993.
25. Krasner, H. (1994), *A Case History of the Space Shuttle Onboard Systems Project*, SEMATECH Technology Transfer #94092551A-TR, October 31, 1994
26. Krasner, H. (1995), *A Case History of Software Process Improvements at the NASA Software Engineering Laboratory*, SEMATECH Technology Transfer #94122662A-TR, January 31, 1995
27. Krasner, H. (1994a), Better Living Through Software Quality and Process Improvement, in the *Proceedings of the Achieving Quality Software IV Conference*, pp. 198-206, January, 1994, San Diego, CA
28. Krasner, H. (1994b), The State of the Union of Software Quality in the U.S., Keynote Speech - ASQC 4th International Conference on Software Quality, Tysons Corner, VA, October 3, 1994
29. Krasner, H. (1994c), SPI Improvement Report to SEMATECH, KC TR-ST-94.4

30. Krasner, H. And Ziehe, T. (1995), Lessons Learned From The Semiconductor Industry Initiative for Improving Software Process, Quality, and Reliability, in the Proceedings of the 1st World Congress for Software Quality, June, 1995, San Francisco, CA
31. Lane, J. and Zubrow, D., Integrating Measurement with Improvement, Proceedings of 19th ICSE, May, 1997
32. Lawlis, P., Flowe, R. and Thordahl, J. (1995), A Correlational Study of the CMM and Software Development Performance, *Crosstalk*, Vol. 8, No. 9, September, 1995, pp 21-25
33. Lipke, W. and Butler, K. (1992) Software Process Improvement: A Success Story, *Crosstalk*, No. 38, pp. 29-39, published by the U.S.A.F. Software Technology Support Center, Hill AFB, UT, November 1992
34. Lloyd, P. (1994) ROI from Continuous SPI in an International Company, proceedings of the 5th International Conference on Software Quality, October, 1995, Austin, TX
35. Lowe and Cox, Implementing the CMM for Software Development, HP Journal, August, 1996
36. Major, J. The Software Challenge: The Next Imperative, Keynote Speech, National SEPG Conference, May 22, 1996, Atlantic City, NJ
37. Majors, J. (1994), Driving for Software Excellence: On The Road With Motorola, keynote speech at the 6th National SEPG Conference, Dallas, TX, April, 1994
38. McGarry, F. (1993), presented at the NASA SEL Workshop, Dec. 1993, NASA Goddard Space Flight Center, Greenbelt, MD
39. McGarry, F., et al (1994), Software Process Improvement at the NASA SEL, CMU/SEI 94-TR-22, December, 1994
40. Mobrin, J. and Wasterlid, A., The Improvement Engine of the ESSI, 2nd European SEPG Conference, June, 1997
41. NASA SEL (1976-1993), Proceedings of the NASA/SEL Software Engineering Workshop, from NASA GSFC, Greenbelt, MD
42. Paulk, M., Weber, C., Curtis, B. and Chrissis, M. (1995), The Capability Maturity Model: Guidelines for Improving the Software Process, ISBN 0-201-54664-7, SEI Series on Software Engineering, Addison-Wesley Publishing Company, 1995
43. Platt, L. (1993), Keynote address at the SEI Software Engineering Symposium, August 24, 1993, Pittsburgh, PA
44. Rhone, K. (1990), The IBM Space Shuttle Program, presentation at the 2nd National SEPG Conference, November, 1990, Washington, DC
45. Robeson, D., Davidson, S. and Bearden, L., Evolution of a Software Engineering Factory, *Crosstalk*, September, 1997
46. Schwarz, J. (1993), presentation at the 10th IEEE Conference on Software Maintenance, Sept. 1993, Montreal, Canada, reported by Hicks, M. and Card, D., in *IEEE Software*, Vol. 11, No. 1, Jan. 1994, pp. 114-115
47. Selfridge, W. (1994) Process Improvement at Rockwell - Experiences, SEPG Process, PAL, Recognition of ROI, at the 6th National SEPG Conference, Dallas, TX, April, 1994
48. Srikant Inamdar (1994), The SEI Certification Process at Motorola India Electronics Ltd. (MIEL), published paper in the National Seminar on Software Quality Assurance, Bangalore, India, August 4-6, 1994, the Institution of Electronics and Telecommunications Engineers (IETE)
49. Sudlow, B. (1994), *Software Development Magazine*, December, 1994, pp 37-40
50. Volker, A. and Wackerbarth, G., Competence in Software and Engineering, 2nd European SEPG Conference, June, 1997
51. Vu, J., Software Process Improvement Journey, 2nd European SEPG Conference, June, 1997
52. Wohlwend, H. and Rosenbaum, S. (1993) Software Improvements in an International Company, Proceedings of the Fifteenth International Conference on Software Engineering, Washington, DC, IEEE Computer Society
53. Yamamura, G. and Wigle, G., SEI CMM Level 5: For the Right Reasons, *Crosstalk*, August, 1997 (and related articles in Sept.&Oct. issues)