Benefits of CMM-Based Software Process Improvement: Executive Summary of Initial Results

James Herbsleb
Anita Carleton
James Rozum
Jane Siegel
David Zubrow

September 1994

SPECIAL REPORT
CMU/SEI-94-SR-013

Special Report
CMU/SEI-94-SR-13
ESC-SR-94-013
September 1994

Benefits of CMM-Based Software Process Improvement: Executive Summary of Initial Results

James Herbsleb

Anita Carleton

James Rozum

Jane Siegel

David Zubrow

Empirical Methods Project
Software Process Measurement Project

This report was prepared for the

SEI Joint Program Office
HQ ESC/DIB
5 Eglin Street
Hanscom AFB, MA 01731-2116

The ideas and findings in this report should not be construed as an official DoD position. It is published in the interest of scientific and technical information exchange.

FOR THE COMMANDER

Mario Moya

Mario Moya, Maj, USAF
SEI Joint Program Office

This work is sponsored by the U.S. Department of Defense. The Software Engineering Institute is a federally funded research and development center sponsored by the U.S. Department of Defense.

For information about purchasing paper copies of SEI reports, please visit the publications portion of our Web site (http://www.sei.cmu.edu/publications/pubweb.html).

Table of Contents

List of Figures

List of Tables

Acknowledgments

Many people were involved in the production of this technical report. First, we would like to thank all of those who provided us with data and patiently answered our questions about software process improvement in their organizations (see Table 1):

Edward Weller, Ron Radice	Bull HN
Constance Ahara, Debbie De Toma, Jim Perry	GTE Government Systems
Sue Stetak	Hewlett Packard
Bob Rova, Ken Shumate, Ron Willis, Thomas Winfield	Hughes Aircraft Co.
Ted Keller	Loral Federal Systems (formerly IBM Federal Systems Company)
Barbara Bankeroff	Lockheed Sanders
John Pellegrin, Richard Stenglein, Bob Yacobellis	Motorola
Leitha Purcell	Northrop
Peter Burrows, Harvey Wohlwend	Schlumberger
Henry Gueldner	Siemens Stromberg-Carlson
Marie Silverthorn, Mary Vorgert, Bob Stoddard	Texas Instruments
Kelley Butler	United States Air Force Oklahoma City Air Logistics Center
Brenda Zettervall	United States Navy Fleet Combat Direction Systems Support Activity

Table 1. Organizations and People Who Provided Data

We would also like to thank Jim Armitage, Jack Harding, and Bob Park for their helpful comments on earlier versions of the ideas contained in this report. We also appreciate the help of Julia Allen and Ron Radice who reviewed several drafts of this report and provided numerous helpful comments along the way. Finally, we would like to thank Suzanne Couturiaux for her skillful help with the technical editing.

To the Reader

This special report is an executive summary of CMU/SEI-94-TR-13 [Herbsleb 94]. It is intended to provide the reader with an overview of some initial results of the effects of software process improvement efforts in 13 organizations. The full technical report is intended primarily for software practitioners, members of software engineering process groups, and software managers interested in understanding the business case for investing in software process improvement. Both reports assume a familiarity with the capability maturity model for software (CMM) [Paulk 93a, 93b]. The following paragraph is intended to guide readers to the parts of this executive summary that are of most interest to them.

Readers interested in a single table that summarizes all of our aggregated results may refer to Table 2 on page 12. On page 13 is a brief summary of the results contained in the five case studies in the technical report. Readers wishing to review our overall conclusions and the next steps we anticipate in this line of work may go directly to page 15.

Benefits of CMM-Based Software Process Improvement: Executive Summary of Initial Results

Abstract. Data from 13 organizations were collected and analyzed to obtain information on the results of software process improvement efforts that were based on the capability maturity model (CMM). We report the cost and business value of improvement efforts, as well as the yearly improvement in productivity, early defect detection, time to market, and post-release defect reports. Case studies of improvement efforts and results in 5 organizations are summarized. We end with conclusions about the results of software process improvement (SPI) efforts.

1. Background

1.1 Motivation for this Empirical Study

Process improvement within software organizations is gaining momentum. Judging by the Software Engineering Institute (SEI) process assessment database, the number of organizations initiating software process improvement programs has increased rapidly in the last several years (see Figure 1). Clearly, there is a widespread perception that software process improvement is a worthwhile investment.

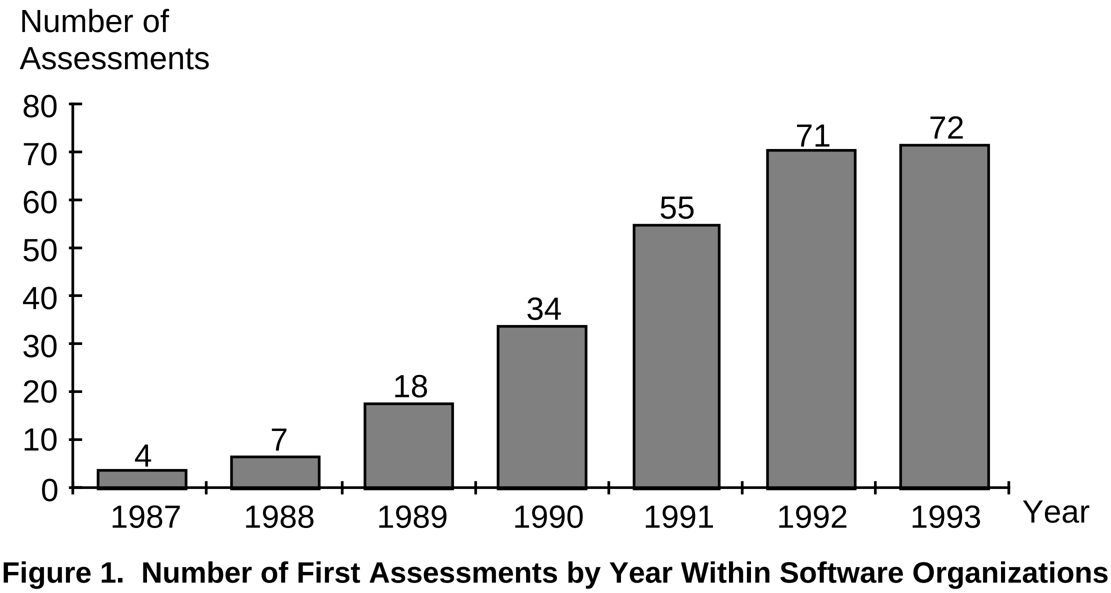

Figure 1. Number of First Assessments by Year Within Software Organizations

One indication of the results of these efforts is the increased process maturity of the organizations undertaking them. The capability maturity model (CMM) for software [Paulk 93a, Paulk 93b] provides one widely adopted evolutionary view of the capabilities of software development organizations. According to this model, organizations progress through five stages as their processes evolve from an initial, often chaotic, state to become first repeatable, then defined, managed, and finally optimizing.

1.2 Data Collection

Initial study of the benefits of SPI. We asked 20 organizations who expressed initial interest to provide readily available quantitative and qualitative data relevant to an assessment of the results of their SPI efforts. Specifically, we asked for data about

- Organizational characteristics, e.g.,
    - size of organization, and
    - type of software.
- Software process improvement efforts, e.g.,
    - assessments,
    - action plans, and
    - metrics program.
- Impact of SPI on, e.g.,
    - productivity, quality, time to market;
    - morale, turnover; and
    - ability to stay on schedule, within budget.

After further discussions, resolution of confidentiality concerns, and evaluation of data available, 13 organizations, listed below, were able to provide data appropriate for this study. They represent a wide range of process maturity levels, and include Department of Defense contractors, commercial companies, and government organizations. The range of application areas is also diverse, including telecommunications, embedded real-time systems, information systems, and operating systems.

- Bull HN
- GTE Government Systems
- Hewlett Packard
- Hughes Aircraft Co.
- Loral Federal Systems (formerly IBM Federal Systems Company)
- Lockheed Sanders
- Motorola
- Northrop
- Schlumberger
- Siemens Stromberg-Carlson
- Texas Instruments
- United States Air Force Oklahoma City Air Logistics Center
- United States Navy Fleet Combat Direction Systems Support Activity

Not all companies provided all types of data, and the data we received varied considerably in level of detail. Fortunately, most organizations were able to report data over multiple years, which is important in helping us to address the question of how long SPI continues to pay off.

To ensure reasonable quality for this study, we established several criteria for the inclusion of data. First, we focused on software process improvement efforts based on the capability maturity model. Since this model is closely associated with the SEI, and this is the data we are most frequently asked for, we decided to make this our first priority. Second, we included only data that had an interpretation that was fairly clear to us. To obtain our current results, we scoured several thousand pages of documents of many kinds, including presentation slides, memos, internal reports, newsletters, action plans, and so on. We often followed up examination of documents with phone calls for clarification and requests for more documents. In spite of this, there remained a few cases where we did not believe we had sufficient understanding of some potential data points to include them in this study. Third, we looked for some indication that the data point was valid. Examples of things that would satisfy this criterion are a description of the organization metrics program, detail in the data itself which seemed to indicate care was taken in its collection and handling, or a description of how these particular data were collected. These do not guarantee accuracy and precision, of course, but we regard them as reasonable reassurances appropriate for the type of data available to us.

There are several caveats to bear in mind when reading this report. They represent important limitations in how these data can be interpreted, as well as representing important challenges for us to overcome in future studies. First, there may be what we are calling *masked tradeoffs* in the data (see [Kaplan 92]). In other words, we typically have an incomplete set of data from each participating organization, so we do not know if the unreported measures paint as favorable a picture as the reported ones. It is possible, for example, that a (reported) reduction in time to market is offset by an (unreported) decrease in quality.

It is also possible that activities other than SPI may have contributed to the results – companies were doing many things over the time period when process improvement activities were underway. They may be gaining experience in a domain, building simpler applications, adding new personnel, using new tools, adopting new methods and technologies, and so on. We asked for data that specifically showed the effects of SPI, but we do not know the extent to which other factors influenced the results.

Finally, we do not know how typical these results are since our participants are companies with good SPI experience, and the reported data come from projects and divisions that have been successful in realizing change. For this reason, we think the results are best thought of as showing what is possible for companies engaged in a serious software process improvement effort in a supportive environment.

2. Improvement Activities

The organizations in this study were engaged in a wide range of software process improvement activities. All of the organizations in this study had action plans and were working on the key process areas appropriate to their maturity levels. We will not try to enumerate all the activities here, but rather will just summarize those that were most frequent.

Virtually all of the organizations formed a software engineering process group (SEPG) as the organizational focus of the process improvement efforts. Many had both corporate-level SEPGs and local or site SEPGs. In addition, many organizations created other special purpose groups such as a management steering team or software mentor team to oversee, coordinate, and prioritize improvement efforts. Several organizations also had software process and technology task forces to investigate technologies (e.g., technology working group, Software Technology Center) or to address particular process issues (e.g., software quality engineering).

Training was an important element of the improvement activities in nearly every organization. Although we do not have complete and systematic information on all of the training programs, the most frequently offered courses appear to be project management, peer reviews, and instruction in the local development process and methods. Some courses were offered by third parties, while other organizations handled all training internally.

Another very common activity for organizations at all maturity levels was some form of peer reviews. Most of the organizations conducted code inspections; many also conducted design inspections; and several were pioneering requirements inspections. All who commented about or collected data on peer reviews were firmly convinced of their value (e.g., [Weller 93]).

A few organizations had established forums for the exchange of ideas and for coordinating efforts among groups. The more expensive type of forum is an in-house conference or workshop where SEPGs across the corporation get together to learn from each others' experiences (e.g., [Wohlwend 93]). A more modest, but apparently effective alternative is a newsletter that informs and publicizes accomplishments (e.g., [Lipke 92]).

Several organizations were changing their processes not only for process improvement per se but also to enable the insertion of new technology. The most frequently mentioned was reuse, based either on a library or on a generic architecture. In these cases, there were many comments on the extensive process changes required to incorporate these technologies.

3. Initial Results

The results are in five categories:

- cost of the process improvement effort,
- productivity,
- calendar time,
- quality, and
- business value (often loosely referred to as return on investment).

These measures were defined and collected in different ways in different organizations. For this reason, the absolute levels are not particularly meaningful. For example, if you do not know what constitutes a defect, and you don't know what counts as a line of code, the absolute number of, say, defects per thousand lines of code is not very meaningful. On the other hand, it is reasonable to assume that there is some consistency within an organization in how they define and collect these data. For this reason, the most informative measure is probably change within an organization over time, and that is, in general, what we report here. The number we use for productivity, calendar time, and quality is *gain per year* (for increasing quantities, *reduction per year* for decreasing quantities), which we compute much like a (nominal) compound interest rate. For example, tripling productivity over a period of 3 years is approximately a 44% gain per year for each of the 3 years.

In the figures that follow, organizations are labeled with alphabetical identifiers to protect confidential data. Organizations labeled with different letters in various charts might or might not be the same organization. Again, this is to protect the confidentiality of the organizations, so that it is not possible to identify an organization from a published figure, then find additional, confidential information about that organization.

As mentioned in the preceding paragraph, some of these data have been published before, while other data points have not, to our knowledge, been previously made public. Of the 24 data points reported here, 8 have been previously published while the remaining 16 have not.

Cost of the process improvement effort. Figure 2 shows the amount spent, per software engineer, by each organization on software process improvement activities. Interestingly, the largest organization and the smallest organization spent the two highest dollar amounts per software engineer. These two organizations also had the two highest business value figures (see the discussion of business value data).

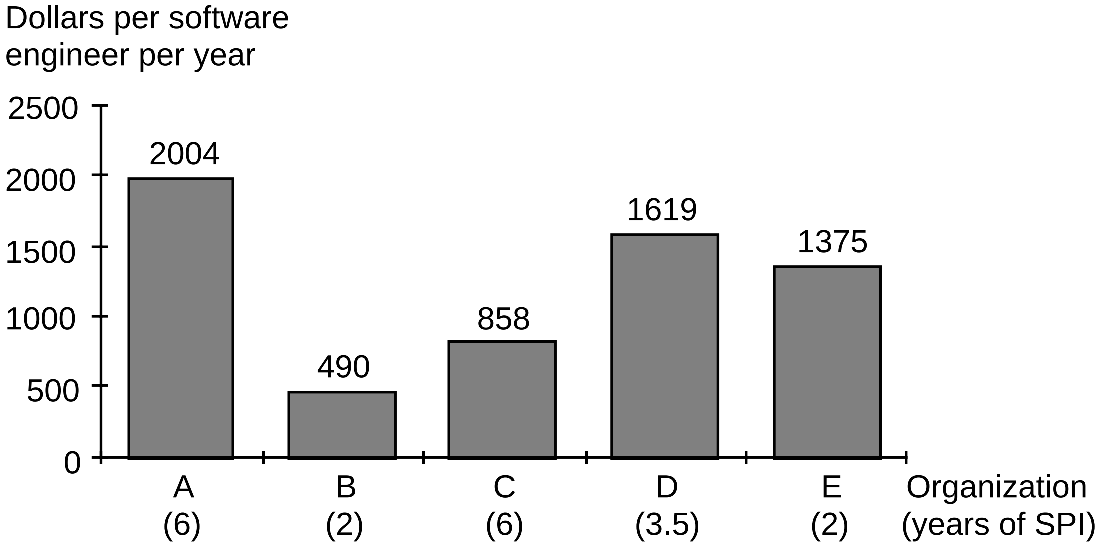

Figure 2. Dollars per Software Engineer per Year Spent on SPI

Productivity. The productivity measures we report here are all variations of lines of code (LOC) per unit of time. These results, recorded as productivity gain per year, are shown in Figure 3.

The largest gain, in organization G, is based mainly on a comparison of two projects, one that did not participate in SPI and one that did. These two projects developed very similar applications and had substantial overlap in staff. The most important factor in the superior performance of the second project appeared to be requirements elicitation and management. The second largest gain, organization H, represents a process improvement effort which included a reuse program, with tools and environment to support development with reuse. The reused code is counted in the productivity figure; if it were excluded, the figure would still be an impressive 28% gain per year. It is impossible to separate process improvement gains from reuse gains, since the reuse would not have been possible without extensive improvements in the software process. The other two data points also represent substantial gains maintained over significant periods.

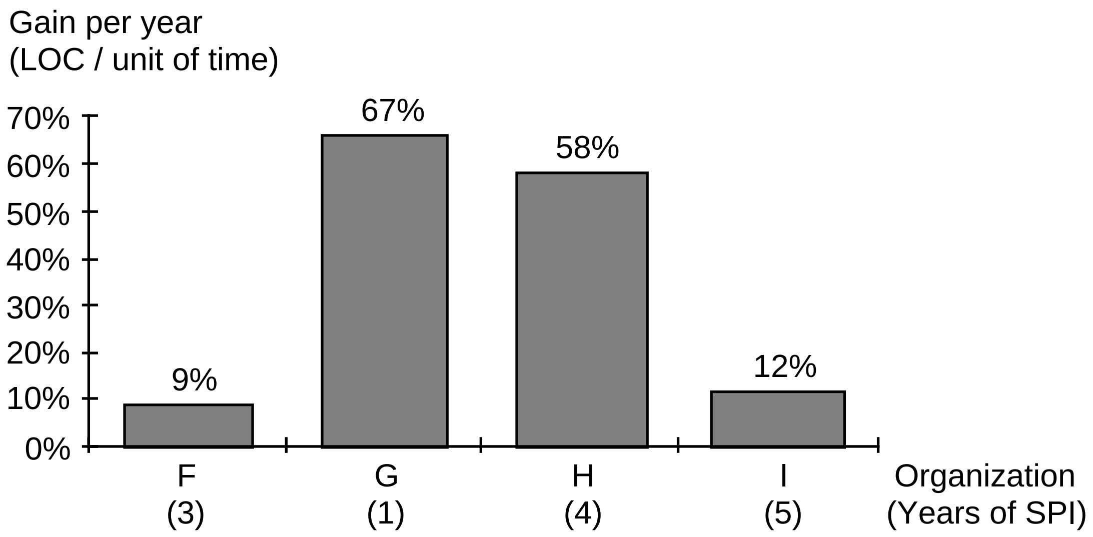

Figure 3. Gain per Year in Productivity

Another form of productivity for which we have data is the early detection of defects. Figure 4 shows the yearly increases in early detection of defects for three organizations. For organization J, all of the software systems on which this figure is based have gone through the entire life cycle and are now out of service. The gains represented on the chart for organization J are gains in the proportion of total life-cycle defects that were found pretest. These data are taken from a number of releases of a large application, each with substantial new and modified code. All are now out of service, so the total life-cycle defects are known. The figures for K and L, on the other hand, are based on software still in service. These figures simply represent increases in the number of defects found pretest. All of these gains represent improvement over time through the use of inspections. Since defects caught early are much cheaper to fix, these changes produce substantial savings in the cost of rework.

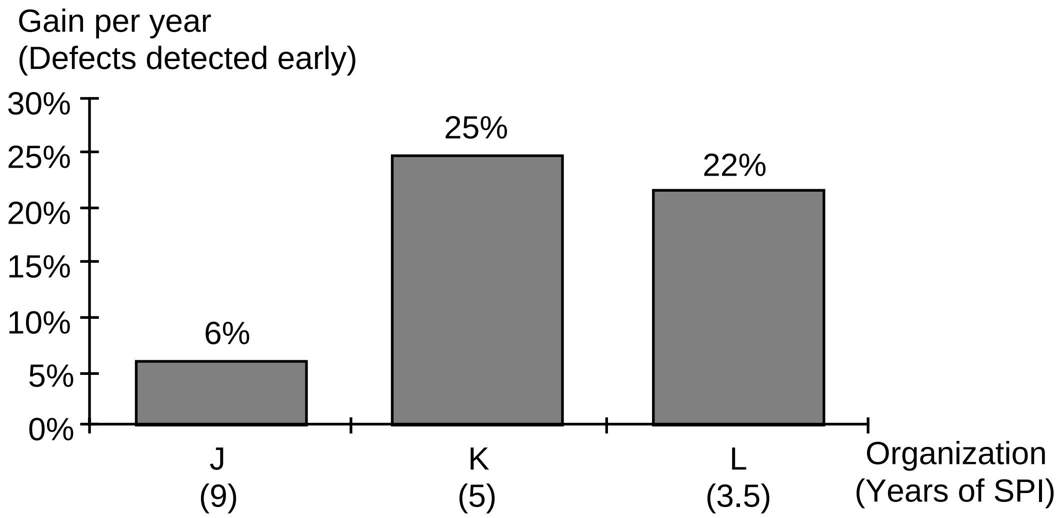

Figure 4. Gain per Year in Early Detection of Defects

Calendar Time. Figure 5 shows the reduction in calendar time to develop software products experienced by two organizations. Obviously, these are substantial gains, and represent the potential of very significant competitive advantage. Both figures are for development covering the entire life cycle. As one would expect, given the enormous gain for organization N, there are several factors contributing to reduction in time to market in this organization. The product was embedded software in a product line, and the process improvement effort included reuse as an inseparable element. The time to market actually suffered for the first year or so as the reusable components were being developed. After two-three years, the time to market dropped very rapidly, to generate these enormous gains relative to the pre-SPI (and pre-reuse) baseline.

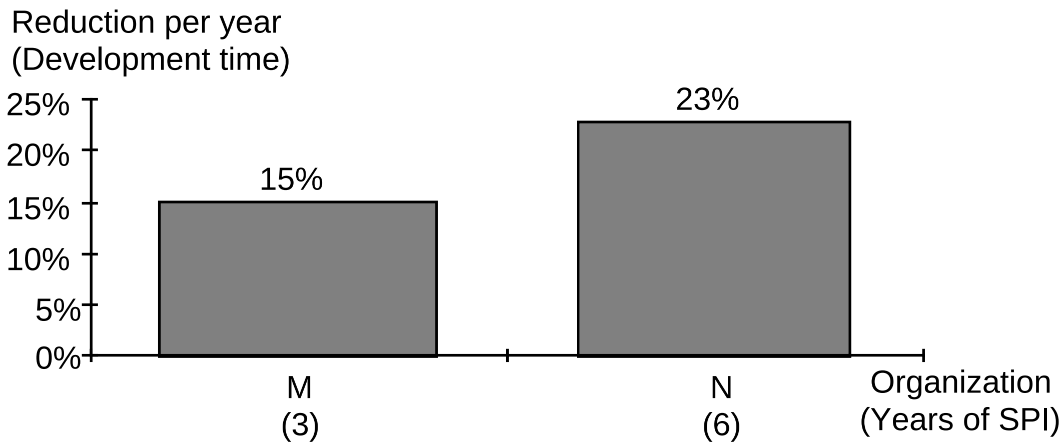

Figure 5. Reduction per Year in Calendar Time to Develop Software Systems

Quality. The most common measure of quality among the data submitted to us was the number of post-release defect reports. Figure 6 shows the yearly reduction in this measure. Most organizations were engaged in improvements aimed at lowering defect injection rates as well as improving detection. Two organizations, Q and R, had astonishing results within a very short time frame. The figure for Q is probably influenced by a major release near the beginning of the period, with much less new code over the remainder. The gain for R was very substantial as well, and was in part the result of a relatively high defect density baseline.

Organization P is remarkable for the period over which it sustained a very sizable reduction of 39% per year. The rates for P represent successive releases, with substantial amounts of new and modified code, all of which have gone through their entire life cycle. The last release had no defects reported in the new and modified code. The 39% rate of reduction, over a 9-year period, represents 2 orders of magnitude reduction in the post-release rate of error reports. The other 2 organizations, S and T, also sustained substantial gains over a significant period.

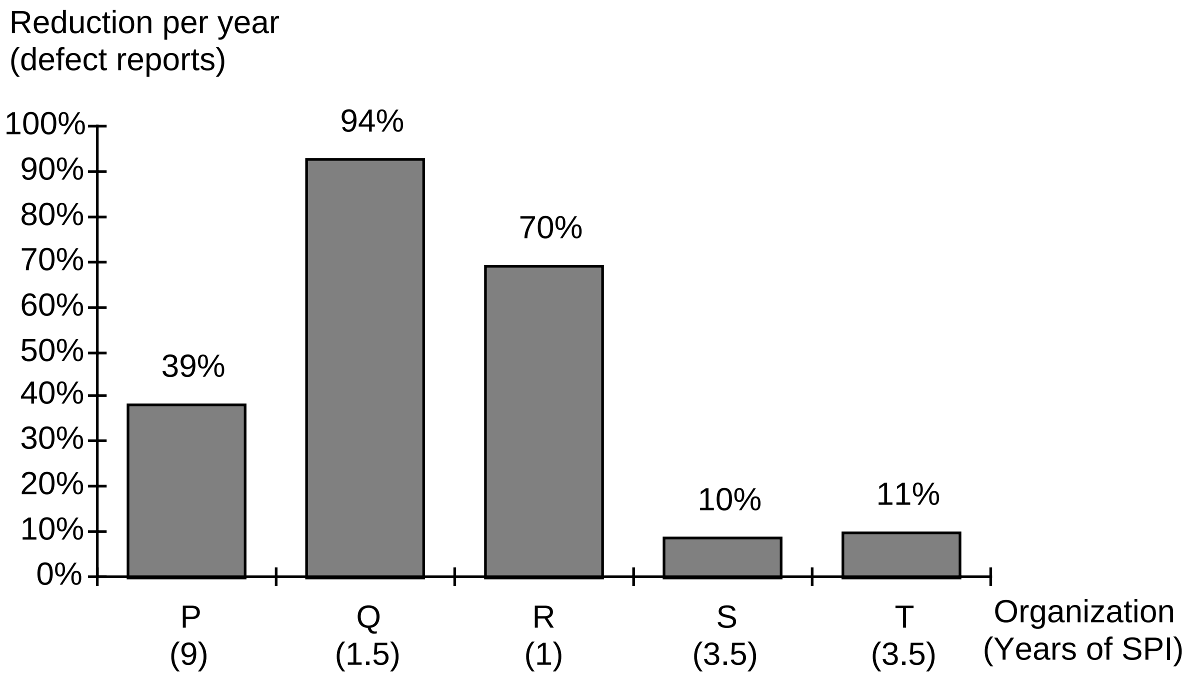

Figure 6. Reduction per Year in Post-Release Defect Reports

Business Value. The bottom-line figure of most interest to many practitioners and managers is the value returned on each dollar invested. This is often referred to rather loosely as the "return on investment" (ROI). The numbers that we report in Figure 7 are the ratio of measured benefits to measured costs.

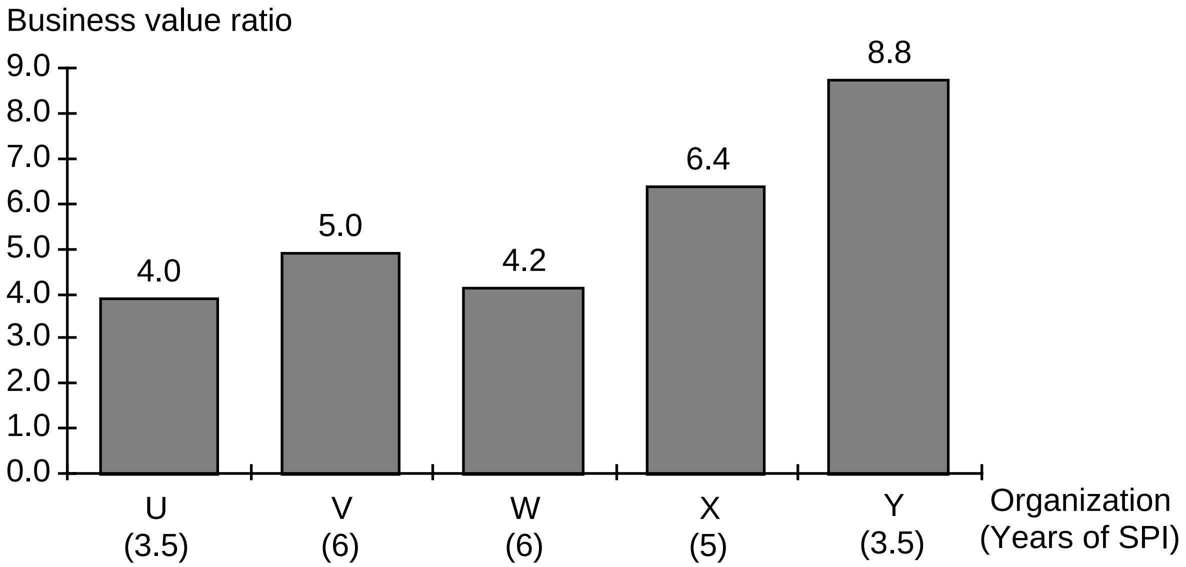

Figure 7. Business Value Ratio of SPI Efforts

This number, despite its importance to many in the community, is probably the most difficult of the numbers to interpret because of variations in how costs and benefits were estimated and allocated to the improvement effort (see [Rozum 93]). In general the business savings figures were calculated as follows. Benefits typically included savings from productivity gains and savings from fewer defects. Less frequently, savings from earlier detection were also included. The benefits generally did not include the value of enhanced competitive position, as a result, for example, of increased quality and shortened time to market. The costs of SPI generally included the cost of the SEPG, assessments, and training, but did not include indirect costs such as incidental staff time to put new processes into place.

As Figure 7 shows, the results are very impressive. There are very few investments one can make that return a business value five or six times their cost. But we wish to stress once again that we do not know how typical these results are. We take them to be indicative of what is possible when improvement is planned and executed well and takes place in a favorable environment. The case studies in the SEI technical report [Herbsleb 94] provide "lessons learned" which summarize many of the factors these organizations believe are important in getting these kinds of results.

Summary of results. In the table below, each category of data is presented with the range and median of data values that were reported, as well as the number of data points from which each type of data was calculated.

Measure **Category of SPI**	**Range**	**Median**	**Number of** **Data Points**
Years engaged in SPI	1 - 9	3.5	24
Yearly cost of SPI per software engineer	$490 - $2004	$1375	5
Productivity gain per year	9% - 67%	35%	4
Early defect detection gain per year	6% - 25%	22%	3
Yearly reduction in time to market	15% - 23%	19%	2
Yearly reduction in post-release defect reports	10% - 94%	39%	5
Business value (savings/cost of SPI)	4.0 - 8.8	5.0	5

Table 2. Summary of the Overall Results

4. Summary of Case Studies

We reviewed case studies [Herbsleb 94] of five organizations that differ markedly in type, application domain, and approach to SPI. It is worth reflecting on that diversity for a moment. The organizations are commercial, DoD contractor, and military. Application domains include operating systems, embedded real-time, information systems, and test program sets. Each has its own approach to SPI, including

- An early emphasis on analysis and use of inspection data, within a wider-reaching SPI effort (Bull HN).
- A long history of process definition, data collection, and quality management (Hughes).
- Corporate-level assessment and consulting resources made available to many software organizations (Schlumberger and Texas Instruments).
- Many small, grass-roots improvement efforts coming together under an organization-wide structure to meet organizational goals (Tinker).

There was a similar diversity in the ways that organizations measured their progress. Some looked only at a few key measures, while others are collecting many. Some have a baseline going back a number of years, while others have begun to collect process measures only recently. Differing business strategies, of course, dictate a focus on different measures.

What we find remarkable about this collection of case studies is that, despite all these differences, each of these organizations was able to use the CMM as the basis for substantial measurable improvement. They are not merely reaching higher maturity levels, but more importantly, they are making progress toward their business objectives.

5. Conclusions

It is clear, on the basis of the data reported here and elsewhere [Dion 92, Humphrey 91, Lipke 92, Wohlwend 93] that software process improvement can pay off. Diverse organizations with very different software products have approached software process improvement in a variety of ways within the context of the CMM, and each has realized substantial benefit. The rough calculations of business value ratios indicate that the returns can be very substantial. We are continuing to solicit data of the sort reported here to further test this conclusion.

We are convinced, however, that it is time to move beyond the basic question of whether process improvement *can* pay off. We need to learn how, when, why, and for whom process improvement pays off. This includes both an empirical examination of the assumptions underlying models of process improvement such as the CMM, as well as an analysis of enablers and inhibitors not accounted for in models. We need to achieve an understanding of the critical factors that cause success and failure.

Achieving this understanding, however, is not easy. No single organization is likely to have the range of experience necessary to identify these critical factors. To address these questions, the SEI is investigating new mechanisms, such as a data-sharing consortium of organizations, to provide the essential data and some resources. The problems seem well-suited to a consortium, because a group of organizations working together can accomplish things impossible for any individual member. We are actively working on this and other approaches to understanding the results of SPI as this paper goes to press.

For organizations undertaking SPI efforts, establishing an effective measurement program should be a very high priority. If the program is to gain and maintain the support of management, the leaders of the improvement effort must be able to make the business case. As we discussed above, in order to do this, the organization must develop measures that are closely tied to business strategy, then be able to show they improve over time. This implies, of course, that there is a baseline from pre-SPI projects to provide a background against which change can be detected. Often, of course, this will not be the case, but if data collection begins very early in the improvement effort, this will generally be adequate.

In addition to making the business case for management, collecting and analyzing data is important for guiding the process improvement effort. Some of the early initiatives will succeed and others will not. Effective measurement will allow the organization to make this distinction and propagate what works. Measurement and data analysis are essential components of any software process improvement program.

References

[Dion 93] Dion, Raymond. “Process Improvement and the Corporate Balance Sheet.” IEEE Software (July 1993): 28-35.

[Fenton 93] Fenton, Norman. “How Effective Are Software Engineering Methods?” *Journal of Systems and Software* 22 (1993): 141-146.

[Herbsleb 94] Herbsleb, James D., Anita D. Carleton, James A. Rozum, Jane Siegel, and David Zubrow. *Benefits of CMM-Based Software Process Improvement: Initial Results* (CMU/SEI-94-TR-13, ADA 283848). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 1994.

[Humphrey 91] Humphrey, Watts S., Terry R. Snyder, and Ronald R. Willis. “Software Process Improvement at Hughes Aircraft.” *IEEE Software* (July, 1991): 11-23.

[Kaplan 92] Kaplan, Robert S. and David P. Norton. “The Balanced Scorecard – Measures that Drive Performance.” *Harvard Business Review* (January-February 1992): 71-79.

[Lipke 92] Lipke, Walter H., and Kelley L. Butler. “Software Process Improvement: A Success Story.” *Crosstalk* (November 1992): 29-31.

[Paulk 93a] Paulk, Mark C., Bill Curtis, Mary Beth Chrissis, and Charles V. Weber. *Capability Maturity Model for Software*, *Version* 1.1 (CMU/SEI-93-TR-24, ADA263403). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, February 1993.

[Paulk 93b] Paulk, Mark C., Charles V. Weber, Suzanne M. Garcia, Mary Beth Chrissis, and Marilyn W. Bush. *Key Practices of the Capability Maturity Model*, *Version* 1.1 (CMU/SEI-93-TR-25, ADA263432). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, February 1993.

[Rozum 93] Rozum, James R. *Concepts on Measuring the Benefits of Software Process Improvements* (CMU/SEI-93-TR-09, ADA 266994). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, June 1993.

[Weller 93] Weller, Edward F. “Lessons from Three Years of Inspection Data.” *IEEE Software* (September 1993): 38-45.

[Wohlwend 93] Wohlwend, Harvey and Susan Rosenbaum. “Software Improvements in an International Company.” *Proceedings of the 15th Annual International Conference on Software Engineering* (1993): 212-220.