The DoD SoftwareTech NEWS

WWW.SOFTWARETECHNEWS.COM

November 2002 Vol.5, No.4

Special Double Sized Issue

Return On Investment from Software Process Improvement

Unclassified and Unlimited Distribution

http://iac.dtic.mil/dacs/

About the DoD Software Tech News

Article Reproduction

Images and information presented in these articles may be reproduced as long as the following message is noted:

"This article was originally printed in the *DoD Software Tech News*, Vol. 5, No. 4. Requests for copies of the referenced newsletter may be submitted to the following address:

Lon R. Dean, Editor
Data & Analysis Center for Software
P.O. Box 1400
Rome, NY 13442-1400

Phone: 800-214-7921
Fax: 315-334-4964
E-mail: news-editor@dacs.dtic.mil

An archive of past newsletters is available at www.dacs.dtic.mil/awareness/newsletters/."

In addition to this print message, we ask that you send us three copies of any document that references any article appearing in the *DoD Software Tech News*.

About This Publication:

The *DoD Software Tech News* is published quarterly by the Data & Analysis Center for Software (DACS). The DACS is a DoD sponsored Information Analysis Center (IAC), administratively managed by the Defense Technical Information Center (DTIC) under the Defense Information Systems Agency (DISA). The DACS is technically managed by Air Force Research Laboratory, Rome, NY and operated by ITT Industries, Advanced Engineering and Sciences Division.

Data & Analysis Center for Software
http://iac.dtic.mil/dacs

DACS
P.O. Box 1400
Rome, NY 13442-1400

Phone: 800-214-7921
Fax: 315-334-4964
E-mail: dacs@dtic.mil
URL: http://iac.dtic.mil/dacs

Cover Design by
Steffen Publishing

To Subscribe to this Publication Contact:

Phone: 800-214-7921
Fax: 315-334-4964

E-mail: news-editor@dacs.dtic.mil
Web: www.dacs.dtic.mil/forms/regform.shtml

Distribution Statement:

unclassified and unlimited

Tech Views: Return-On-Investment: The Return Comes Later, The Investment is Now

Background

In my 20+ years of experience of software development and dealing with executives of software companies, there are three things that I've observed:

1. Executives are motivated by their compensation and bonuses (as are most of us). Executive's compensation is tightly tied to their organization's performance this year, not next years or the year after. Any proposed improvement that requires an investment in which the return does not materialize on the bottom line for several years is viewed negatively compared to proposals with more immediate returns.
2. Executives have more proposed investments than they can fund. Any software process improvement proposal needs to stand out positively compared to the alternatives.
3. Technically oriented people do not know how to justify investing other people's money. We, as technical people, know intuitively that software process improvement makes sense. We know it will improve the quality of our and our fellow software developer's software and will produce a product that we will all be proud of. So why don't the executives see that it is intuitively obvious that these improvements make sense? Why do I have to go through this exercise?

The above observations suggest to me that we in the software industry need to be able to deploy process improvement in such a way that returns are observed sooner than later. Second, the returns have to be huge and obvious - they can not be subtle. We have to be able to say to senior management something like "If we make this software weigh less by using modern and more reliable software materials, we will save thousands of dollars per copy for every software ounce removed." And third, technically simple Return-On-Investment (ROI) tools and models are needed for building the business case for any proposed technical improvement.

Obviously we are a long ways from being able to do any of this - software is still a very intangible entity to many. But I believe that we need to begin the dialog between the technical, management, and business people to come to a common understanding and appreciation of process improvement in the software industry.

Since I published my "A Business Case for Software Process Improvement" report (www.thedacs.com/techs/roispi2/), many people have expressed interest in additional Return-On-Investment (ROI) from software process improvement (SPI) data. A general problem is the lack of publicly available data on the benefits from SPI that people can apply to their own situation. A limited amount of data is available in the open literature, as noted by the authors of articles in this issue.

However most of the data is very sensitive, proprietary information and thus will never end up in the open literature.

In This Issue

This particular issue of the DACS Software Tech News is an attempt to begin the dialog between technical and business people on understanding ROI from SPI. We also want to raise the technical communities awareness of cost benefit analysis as it relates to SPI so that technical people can begin to justify improvement to their management.

The article "Benchmarking the ROI from SPI" describes some additional positive results of SPI from real benchmark data. This article highlights the need for additional data I alluded to above.

The article "The Definitive Paper: Quantifying the Benefits of Process Improvement" presents exciting results from a new case study of SPI benefits. It discusses the benefits realized from the use of product lines and systematic reuse. The point about the sensitivity of data I discussed earlier is apparent from this article.

by Tom McGibbon, DACS Director

The article "Making Investment Decisions for Software Process Improvement" provides a framework for evaluating, from an ROI perspective, process improvement proposals. It demonstrates that we need to understand the business person's perspective.

"How to Estimate ROI for Inspections, PSPsm, TSPsm, SW-CMM_, ISO 9000, and CMMIsm" shows that the benefits of process improvement are very impressive. It provides a step-by-step "How-To" for ROI and benefit calculation.

Please pass a copy of this newsletter on to your senior management and business people. This is not a newsletter just for the technical folks.

We at the DACS look forward to your feedback on this important software business issue. Please provide any comments through the DACS website comment form at http/www.thedacs.com/forms/mailform.html or contact me.

Author Contact Information

Tom McGibbon
DACS Director
775 Daedalian Dr
Rome, NY 13441

Phone: 800.214.7921
E-mail: tom.mcgibbon@itt.com.

Other ROI Resources In This Issue

- **DACS Data Call for ROI data from Software Process Improvement (SPI) projects.** On page 17 of this newsletter there is a data call from the DACS. The DACS has included a ROI from SPI data collection survey. Anyone participating in this survey will gain access to a searchable database DACS ROI from SPI database. In addition all participants will receive a FREE copy of the DACS CD-ROM entitled "ROI from SPI Products". This excellent resource sells for $50 and includes the DACS ROI database, a working spreadsheet model that allows users to model the cost benefits achievable through SPI, and two technical reports. This first report is *A Business Case for Software Process Improvement (Revised): Measuring Return-On-Investment from Software Engineering and Management*. This report examines the details necessary to demonstrate the benefits of improved Software Management using SPI techniques from business, profit and loss, and senior management perspectives. The second report, *Cost Benefit Analysis for Software Process Improvement,* examines a cross-section of popular SPI methods and approaches, prioritizing them by their costs and benefits.

Benchmarking the ROI for Software Process Improvement (SPI)

Some new thoughts on an old problem….

1. Introduction

Examining the measures needed for handling Software Process Improvement (SPI) programs, we find several types of information needs:

- Ways to determine readiness for a SPI program (identifying risks and the level of threat to SPI)
- Information to track and manage the SPI program effectively as a project or set of projects (similar to progress and process measures for other types of projects)
- Information to justify the program (at the start, and throughout the program)

The third category of needs is often handled by building a business case to show a return on investment (ROI). Measures used in the business case need to show an ROI in terms of the business concerns of the organization. When an organization has quantified business goals, perhaps organized in a balanced scorecard, determining an appropriate measure for the ROI on SPI is relatively easy to do. When the organization does not already have such a quantitative understanding of its business goals, an ROI argument may need to leverage industry data using a benchmarking approach like the ones described here.

2. Business Cases and ROI

When generating a business case for any project, we consider the reasons for doing a project and try to quantify the benefits expected from that project. Benefits from a SPI project might include:

- New revenues from new market or product capabilities
- More revenue from additional business due to improved customer satisfaction
- More revenue from additional business due to improved product quality
- More revenue from current product (lines) because of faster cycle time, introducing new features more quickly to current or new customers
- Reduced costs from reductions in rework
- Reduced costs of operations
- Reduced costs (or additional revenue) from increased productivity

We then collect best estimates of our investments (costs), which for SPI projects tend to be

- Effort (labor) invested in performing the project work
- Travel and administrative costs
- Training
- Specialty services, such as assessments and other consulting support

Using this information, a classic ROI analysis would be computed as shown in Formula 1. ROI Capital

In many projects, including most SPI projects, the computation generally used is illustrated in Formula 2. ROI from SPI.

Return on Invested Capital = Earnings before Interest and Taxes/Average Invested Capital

Formula 1. ROI Capital

Return on SPI Program = (Program Benefits – Program Cost)/Program Cost

Formula 2. ROI from SPI

3. Challenges with Demonstrating ROI for SPI

For an organization that can measure its costs and knows its current baseline values for benefits, building the business case and showing an ROI is feasible. However, for an organization without such baseline data, establishing an ROI argument for building a SPI program is considerably more difficult, and the organization may try to use experiences of other organizations to justify their program.

There is a great deal of anecdotal evidence from SPI programs that have succeeded over the last several decades, with information such as:

- ROI values reported from 4 to 70 x cost (median about 5 to 1)
- Various benefits based on organization goals: productivity, defect levels, cost, schedule attainment, effort spent, customer satisfaction, staff attitude
- Costs per individual in the organization varying from $200 to $2500

Deciding which of these values to use when justifying a particular SPI program is problematic, since it is

difficult to tell which relates well to the type of work being done in this organization. Which can truly be treated as an industry benchmark value?

4. Benchmarks for ROI

To use benchmark measurement data to justify a SPI program, the measures must be applicable across industries and across organizations of different types and sizes. Which of the SPI benefits cited in business cases are reasonable subjects for ROI industry benchmarks? Which of the cost categories are comparable? That is, which can be normalized in a way to allow us to compare current organizational performance to that of other organizations? Table 1 identifies candidate benefits and costs and how well they appear to work as benchmark items.

It appears that improvements to measures of productivity may be a useful candidate for an industry benchmark to justify and to explain the ROI of a SPI program. Others that appear reasonable are measures of product quality and savings in operations costs.

Recent work on updates to the COCOMO model also suggest that productivity is a good measure of the impact of SPI [1]. A new scale factor (PMAT) appears in the COCOMO II model, showing the benefit of process maturity on an estimate of effort for a software project. Based on an analysis of 161 data points in the COCOMO II database, Boehm and his team found a statistically significant correlation between improvements in productivity and reductions in software project effort. Table 2 shows how going from level 2 to level 3 in CMM-based process maturity affected the productivity of teams working on different-sized systems. While not a benchmark in a strict sense, this set of data provides a good indicator of the value of improving an organization's processes.

Other sources of benchmark data on productivity include the publicly available ISBSG collection of function point data, and the work of a number of consulting organizations that provide benchmark services. (Several of these are described in the September-October, 2001 issue of *IEEE Software*, which is a focus issue on benchmarking.)

One such industry benchmark is the Application Development (AD) Benchmark done by Gartner Inc. The AD benchmark allows an organization to compare itself to other information technology organizations throughout the industry or within its own industry segment, looking at data about how it builds and maintains software systems. As of early 2002, the Gartner database included information from

- 43,700 development projects, 44,616,000 Function Points
- 55,700 supported applications, 124,588,000 Function Points
- all major technologies, languages, databases
- project data from ~1991

article continued on page 8

Benchmarking the ROI for SPI continued

Table 1: Candidate Benchmark Items

Benefit	Yes/No	Comment
Increased revenue	No	Values vary broadly by market type and product type; revenue/employee is a normalized measure, but not easily compared outside a given domain
Reduced cycle time	No	Interactions with processes and tools used, as well as specific type of product, make this difficult to compare
Reduced cost of operations	No/Yes	Basis of costs vary so widely by type of industry and culture, that comparison is very difficult. For comparable operations, such as corporate IT spending, there are usable benchmarks.
Level of quality	Yes/No	If the quality level is established through a standard test of product performance, a recognized level of quality can be assigned. Otherwise, the methods used by different organizations and their different users are unlikely to be comparable.
Reduced rework	Yes/No	Percent of effort spent can be normalized, although the categories of effort that contribute to a rework figure can vary widely across organizations because of their processes and cultures
Increased productivity	Yes	Level of productivity (using function points to measure amount of work) works well
Cost	**Yes/No**	**Comment**
Costs of the program	Yes	Total costs (labor, training, specialty services, tools, travel, etc.) are quite comparable, and they can be normalized by number of people in the organization benefiting from the SPI program

Table 2. Benefits of Process Maturity Productivity in COCOMO II

Project Type	Typical Size	% Productivity Improvement
Small	10 KSLOC*	4%
Medium	100 KSLOC	7%
Large	2000 KSLCO	11%

* Note: KSLOC = thousands of source lines of code

The AD benchmark gathers size and effort (labor) data at the application and/or project level, allowing for analysis of technical and performance data at a low level. Productivity figures for a given organization can be calculated at the application or project level, or productivity data can be aggregated at a higher level.

The benchmark also asks respondents to identify their generic life cycle as one of waterfall or prototyping, and to rate their development process rigor as one of:

- Loose: informally followed; little or no documentation
- Moderate: checkpoints at major phase boundaries; responsibility rests with project managers; little or no external oversight
- Rigorous: extensive documentation; independent oversight/ quality assurance/process management tools often used

The levels of rigor do not map directly to process maturity levels, but they do indicate increasing levels of process discipline. This benchmark evolves with market needs and is being updated to ask more detailed questions of life cycles and to gather CMM-based process maturity information where it exists.

Meanwhile, it is interesting to examine the data in the current database for relationships between productivity and process maturity, to see what support there might be for an ROI for SPI.

Using the data from the last two years, Figure 1 shows the productivity in function points developed per full time equivalent developer (FP/FTE) for the different types of lifecycle, by level of rigor. It also shows the aggregate across type of life cycle. The aggregate data across lifecycle types shows that productivity rises as process rigor increases. The prototyping lifecycle supports this trend, with the greatest increase when going from loose to moderate rigor. The waterfall cycle, however, shows the lowest productivity at the moderate level of rigor. This pattern appears counterintuitive, in light of other industry information.

Looking at the waterfall data further, removing outliers, the pattern changes a bit, with the moderate and rigorous levels of rigor having about the same productivity, as shown in Figure 2. While this tempers the apparent anomaly, it still leaves a question of why such a lower productivity at moderate or rigorous level, when compared to the loose level of rigor. (As a reminder, the data used here is not process maturity data, but a general characterization of process rigor.)

Another study may help us understand why such a negative trend in productivity appears in the Gartner data. A study done by Harter, Krishnan, and Slaughter [2] shows a similar effect. In a study of 30 software products (a COBOL MRP system) built by a large IT firm over a period of 12 years (1984-1996), they found that increases in process maturity were associated with increases in development effort (that is, decreases in productivity). They also found that the increases in process maturity were associated with improvements in product quality. Investigating the interaction of these changes, they found that for the full product life cycle, the impact of improved quality outweighed the decreased development productivity, because of its positive effect on the long term maintenance work. Thus, there were reductions in overall cycle time and effort because of the improvements in product quality. The diagrams in Figure 3 show some of the relationships found in their study.

Thus, we conjecture that the data in the Gartner database also portrays this same reduction in productivity during development, as more process rigor is applied. However, the Gartner data does not include product quality data with which we can investigate the impact of improved rigor on product quality - or on the long-term life cycle impact of that product quality. To get this information will require other modifications to the AD benchmark. That work is underway.

5. Future of Benchmarking for ROI

If organizations are to use industry data to make their ROI arguments, we clearly need to have access to benchmark data that is easily

article continued on page 10

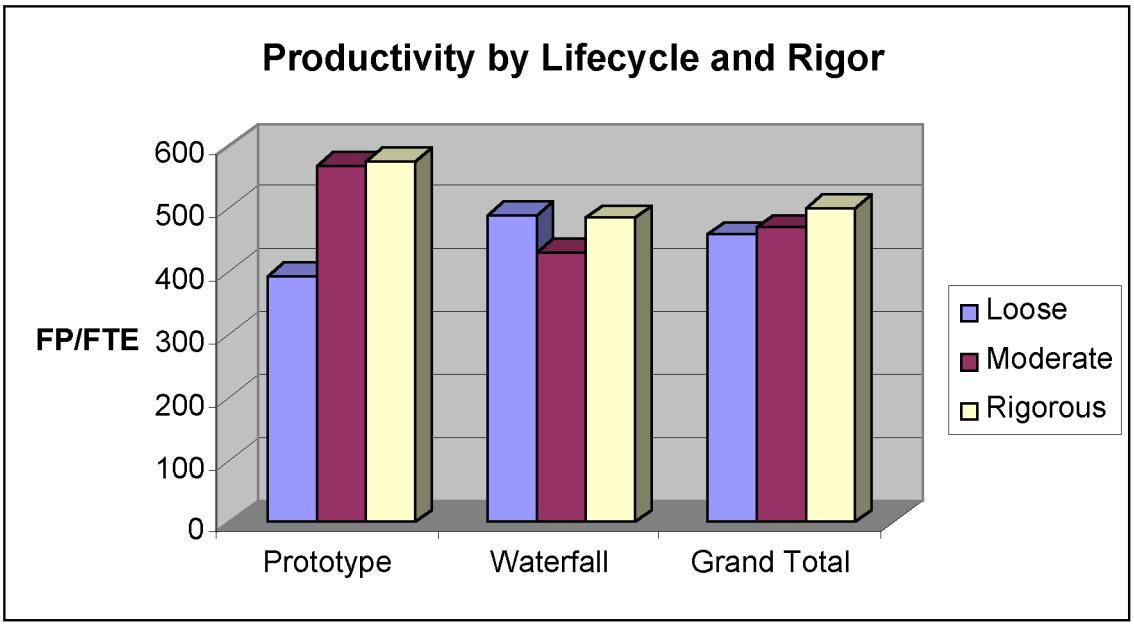


Figure 1: Productivity by Lifecycle and Rigor

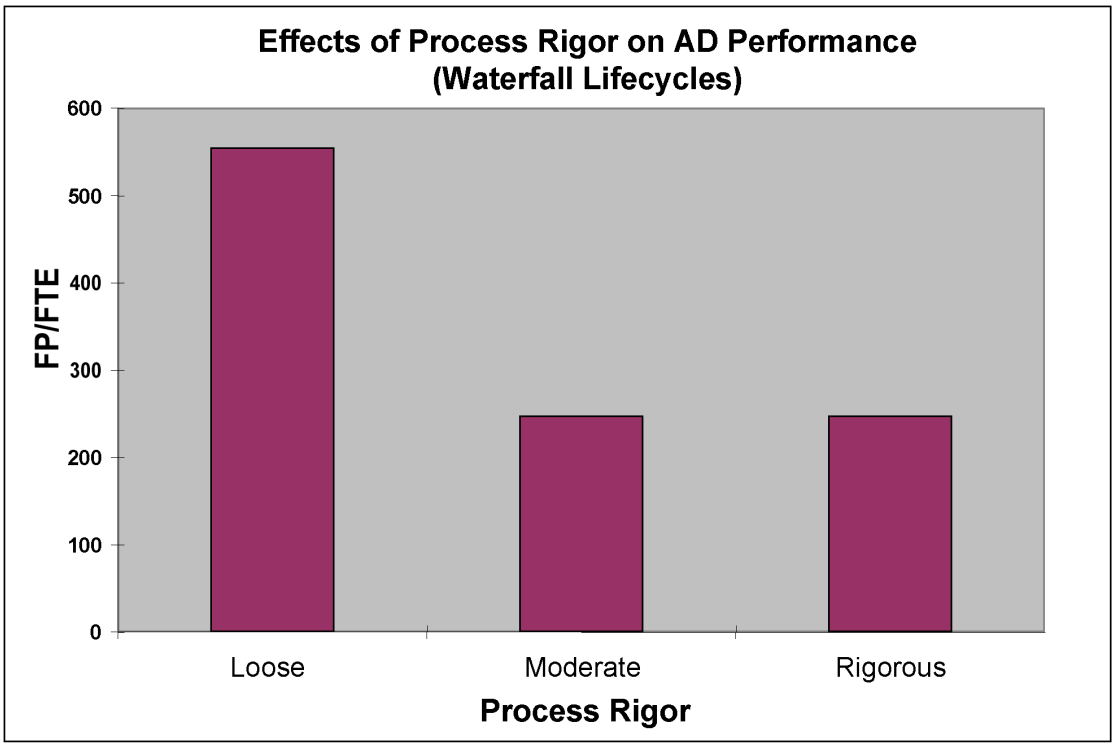


Figure 2 – Effects of Rigor with Waterfall Cycle

comparable across many types of organizations doing their work in different ways. Today's benchmarks provide the hint that productivity data is useful, but needs to be considered along with measures of product quality. As indicated in Table 1, quality data will need to be provided in a consistent way to be useful in a benchmark. Research is needed in how to collect consistent product quality data from customers, as well as from review and testing activities of development organizations.

Are there other measures that could be used, perhaps measures already in common benchmarks? We invite your comments and suggestions, as we continue our search.

About the Authors

Robert Solon is Research Director for Application Development (AD) Measurement with Gartner, Inc. He is responsible for the full range of Gartner' AD

measurement services worldwide. Prior to joining Gartner, Bob served quality assurance and compliance with Roche Diagnostics and as a software project manager for Keane Inc., He also served in several staff and development positions with the software organization of the Defense Finance and Accounting Service.

Mr. Solon holds a BA in computer science and political science from Capital University, and an MBA from Anderson University.

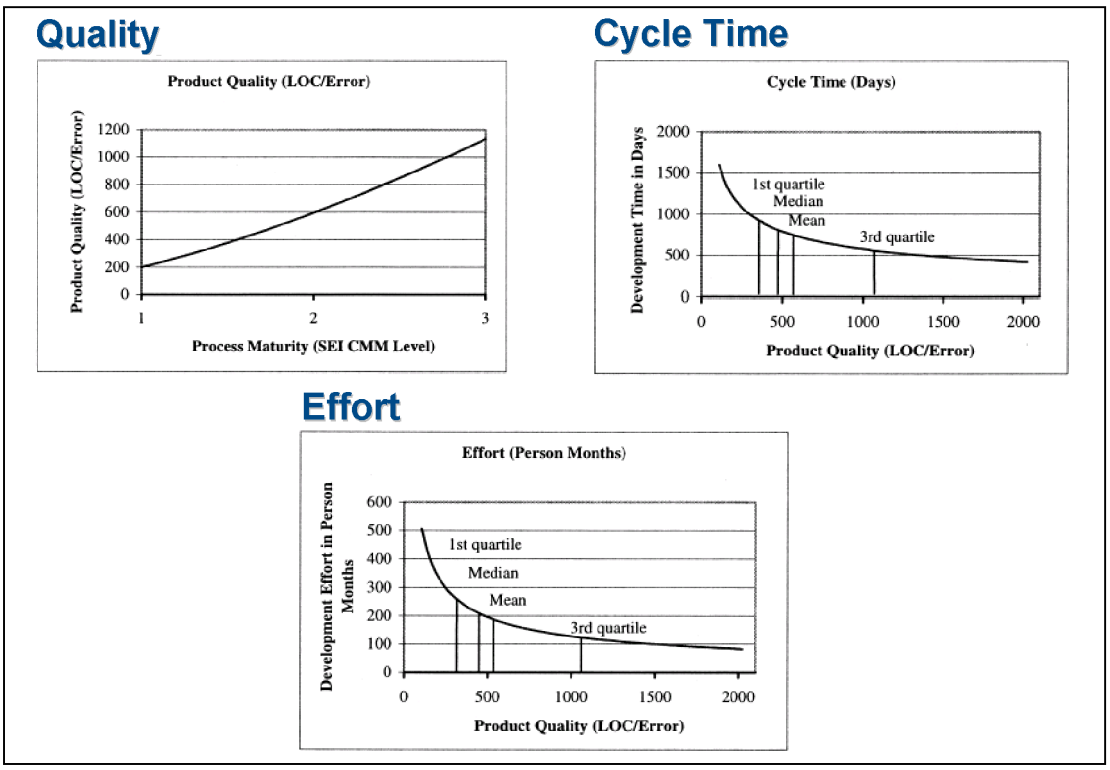


Figure 3 - Process Maturity Affects Quality, Effort, Cycle Time

Dr. Joyce Statz is Vice President of Knowledge Management at TeraQuest, where she helps employees and client organizations with process improvement programs She also coordinates development of product and service offerings of TeraQuest.

She has 15 years of experience software systems at Texas Instruments. Prior to that, she taught computer science at Bowling Green State University. She is a founder of the Software Quality Institute of The University of Texas at Austin (SQI).

Author Contact Information

Robert F. Solon, Jr.
Gartner, Inc.
3836 North Drexel Avenue
Indianapolis, IN 46226

PH: (317) 237-4039
Fax: (317) 237-4072
E-mail: robert.solon@gartner.com
URL: www.gartner.com

Joyce Statz
TeraQuest Metrics, Inc.
12885 Research Blvd, Suite 207
Austin, TX 78750

Phone: (512) 219-9152
Fax : (512) 219-0587
E-mail: statz@teraquest.com
URL: www.teraquest.com

References

1. Boehm, Barry W., et.al. Software Estimation with COCOMO II. Upper Saddle River, NJ: Prentice-Hall, 2000, p.67.
2. Harter, Donald E. , Mayuram S. Krishnan, and Sandra A. Slaughter. "Effects of Process Maturity on Quality, Cycle Time,and Effort in Software Product Development. Management Science, vol. 46, No. 4, April, 2000, p. 451-466.

The Definitive Paper: Quantifying the Benefits of Software Process Improvement

Summary: This paper summarizes the tangible and intangible benefits that Northrop Grumman Electronics Systems has reaped from our process improvement program. It puts facts and figures in the public domain to help others make the business case for process improvement. This is an actual case study that shows that investing in process improvement pays off.

1. Introduction

Northrop Grumman Electronics Systems (ES) initiated its process improvement program in the late 1980's. We were forced to do it because it was a customer requirement (i.e., these were not forced but were instead recognized as an implied requirement). At the time, there was lots of resistance to change. We were rated a level 2 in 1987 and level 3 in 1989 by an in-house team with observers from the Software Engineering Institute present during the assessment. As an early adopter, we went through all the trials and tribulations that you normally read about in the case studies that have appeared since that time.

We stayed at level 3 for almost a decade. Investment in process improvement was refocused onto ISO compliance and management viewed the business requirement as having been met. We were winning contracts and performing well. However, in 1996 our General Manager resurrected process improvement when a customer survey identified software as an area needing improvement. In response, an aggressive program to be re-certified a level 3 and reach level 4 was mounted.

2. Environment

Northrop Grumman ES builds some of the best sensors in the world. Our radars fly in the F16 and F22 fighters. We build air traffic control systems and outfit eyes in the sky like the Defense Meteorological Support Program. We have a professional workforce of more than 600 software engineers working hundreds of embedded system projects that provide the intelligence for such systems. Our programs are large and small, new and old. The software we produce is complex, life-critical and runs in real-time.

Engineering management has run hot and cold over the years when it comes to process improvement. However, we have had solid support from our executive staff since 1996. Like most in the industry, we have formed a process group and used it to write processes and put them into practice. Unlike most, we have staffed our process group with 30 year veterans, senior management and technical personnel and some that came back from retirement, to lead projects through the transition. Use of people who know and are respected by those in the organization is deemed one of our critical success factors.

Costs for our software process improvement program have averaged about $2 million annually. This budget covers the process group, training and the transition activities. Our philosophy has been and remains one of partnership with projects. We let our projects tailor the process to the specifics of the application. The process group budget provides charge numbers for key people to participate in working groups, process authorship and reviews. Projects fund the remainder of the activities including training, deployment and tailoring.

3. Process Improvement Strategy

Our goal with our process improvement program was to put processes in place that made a difference. We were not concerned with process for the purpose of process. Instead, we wanted to generate our products quicker, better and cheaper than our competition. We've succeeded because we've tied process improvement to business goals. The strategy that we initiated in 1996 revolved around achieving these goals. Our two primary objectives were:

- **Accelerate Productivity Gains** - Our investment strategy for productivity was reoriented to process so we could accelerate gains made through improvement in management practices.
- **Move to the use of Product Lines, Architecture and Systematic Reuse** – We were convinced that developing avionics product lines that permit us to systematically reuse software from project-to-project based upon a reference architecture was the way to go.

Prior to 1996, we had pursued these two strategies piece-meal. In 1996, we developed a process improvement plan that focused our attention and investment dollars on making these four things happen in a planned and systematic manner. This plan has yielded both tangible benefits that justify the investment and justify process improvement based upon its returns.

4. Tangible Benefits

Most of the information we've seen in the literature about process improvement has harped on the benefits without putting numbers around them[1,2]. While useful, such discussions don't help the community make a strong business case for process improvement. Because we have developed such numbers to convince our internal critics that process improvement pays dividends, we want to share them with the community. However, we must do so in such a way that we don't let our competition know our actual costs.

4.1 Accelerating Productivity Gains through Process

The average gain in productivity that we have experienced during the past 5 years as we have moved from Level 3 to 4 is approximately 20 percent annually. During our static years, our nominal gain was 10 percent annually. We can thereby conclude that we have accelerated our gain by 10 percent a year based upon a strategy that was heavily Software Process Improvement based. Such acceleration results in a cost avoidance averaging $25 million annually over a five-year investment time span based upon the analysis in Table 1. It should be noted that we assumed no gain during the first year of the investment strategy. We also assumed a static workforce. Both of these assumptions simplify the analysis and make the results very conservative.

The non-discounted ROI due to productivity improvements alone is calculated as follows:

ROI = ($125.1M - $10M)/$10M = 1251% or 250% annually

Of course, this is not a true number. However, it does illustrate the benefits that we have accrued which are more than the numbers provided in this paper suggest.

4.2 Movement to Product Lines, Architecture and Systematic Reuse

The hardest part of our strategy to implement was moving to architecture-based avionics software product lines. The reason behind this is that the SW-CMM and CMMI offer little structure for

article continued on page 14

Table 1 – Dollar Savings Attributed to Accelerating Productivity from 10 to 20% Annually

	Year 1	Year 2	Year 3	Year 4	Year 5
Current productivity (SLOC/staff-month) (10% nominal gain)	105	116	127	140	154
Accelerated gain (20%)		126	151	181	218
Additional number of SLOCs that can be generated via acceleration assuming 600 engineers		72,000	172,800	295,200	460,800
Cost avoidance ($125/SLOC)		$9.0 million	$21.6 million	$36.9 million	$57.6 million
Cumulative cost avoidance		$9.0 million			

Note: For competitive reasons, we have used a base productivity of 105 SLOC/staff-month as the basis of our analysis. This is not our current productivity. The cost of $125/SLOC assumed is also not our actual cost/SLOC. These numbers are industry averages taken from a productivity report that represents the cost for the military airborne domain [3]. These numbers are conservative and used to illustrate the benefits.

initiatives in this area. For the most part we were on our own to develop processes in this area. Because we are a defense contractor, we also have many restrictions that make it difficult to share software developed for one project on another. Sharing is not something that our customers encourage or provide us financial incentives to do. But, we wanted to leverage our previous work to be more competitive. Therefore, we took the risk and moved ahead paving the ground for others to follow.

Northrop Grumman ES has been pursuing systematic reuse for over a decade on Internal Research & Development and technology research projects. We completed a domain analysis and developed a radar system architecture, both hardware and software, that facilitates reuse at the system level in the mid-1990's. Our goal was to deploy this architecture using product line management concepts by making it part of the processes our engineers used to do their work [4]. In response, our engineers incorporated reuse provisions into our processes as they were developed or updated for Level 4.

The benefits attributed to systematic reuse are many and substantial. Reuse saves money and time by making big jobs smaller. Table 2 illustrates this savings using an example. It shows how exploiting an existing architecture that is maintained using product line management concepts can reduce the effective size in SLOC's of a typical job in half. While these numbers are hypothetical, they are in the ballpark for a real system that has many more modes.

The benefits of cutting the size in half can be quantified using a simple cost model like COCOMO II [5]. Using the model with its nominal settings for cost drivers for the example summarized in Table 2 results in the effort and duration estimates in Table 3. Both nominal and shortest development time options are estimated. The only cost driver varied was the Process Maturity (PMAT). It was set to reflect a Level 4 organization.

The example in Table 3 illustrates the benefits associated with reuse. It suggests that about half the cost (e.g., about $5 million) and as much as a year can be saved through systematic reuse for this basic radar. In reality, our radar systems are much bigger and more complex than what is in the Table. We estimate that our cost saving exceeds $5 million on each new radar using the reference architecture we have developed with and the infrastructure we have introduced. Multiply this savings across the four product lines that we have developed and we estimate we will realize at least a savings of $20 million annually. However, there are increased costs associated with maintaining our architecture and with designing assets for reuse. The cost/benefits that result as a product of this fourth prong of our initiative are summarized in Table 4 across all of our product lines:

The ROI associated with this part of our strategy alone is therefore computed using the guidance in the excellent reference on software business cases as follows [7]:

ROI = $19.2M/$800K = 240% or 48% a year across our 5 year planning horizon

Again, this is not the true number. We have in reality been able to realize larger gains through reuse than we at first anticipated.

Table 2 – Size of Application with/without Reuse

Application	Without Reuse	With Reuse
Executive	10,000	500
Radar Scheduler	30,000	0
Radar Mode 1 – Search	50,000	10,000
Radar Mode 2 – Precision Track	50,000	25,000
BIT/FIT (hardware specific)	60,000	60,000
TOTAL	**200,000**	**95,500**

Table 3 – Effort and Duration Estimates with and without Reuse

	Without Reuse	With Reuse
Nominal Development Time (months)	30	23.4
Nominal Effort (staff-months)	845.3	383.7
Shortest Development Time (months)	22.5	17.6
Shortest Development Time Effort (staff-months)	1208.7	548.7

In summary, our successes in process improvement and architectural reuse strategies have exceeded our initial expectations even considering that process improvement can't take credit for all of the productivity gains (i.e., our move towards the use of COTS products, improved tool sets, and increased training also contribute directly to the bottom line). The business case is clear and is being extended to encompass all enterprise disciplines, systems engineering, hardware design, program management, and business operations.

Table 4 – Cost/Benefits Associated with Product Lines, Architectures and Systematic Reuse

Non-recurring costs		Tangible benefits	
• Domain engineering	Completed on IR&D	• Cost avoidance	$20 million
• Reusable assets	Project funded		
• Infrastructure	Done by process development group	Intangible benefits • Deliver 12 months earlier than the norm	
Recurring costs		• 10 times reduction in errors upon delivery [6]	
• Architecture maintenance	$200K	• Architecture stable, proven and can be demonstrated	
• Asset maintenance	500K		
• Process updates	100K	• Scheduling algorithms for the radar can be optimized and improved each time a new radar is built	
TOTAL COSTS	**$800K**	**TOTAL BENEFITS**	**$20 million**

Note: This analysis assumes that the non-recurring costs are treated as sunk costs.

Conclusion

Based on the tangible benefits accrued, Northrop Grumman ES has become a true believer in process improvement. Either accelerated productivity or move to product lines alone would have justified our investments. However, the real scorecard happens monthly at our internal financial reviews. Before our process initiative, we used to spend hours explaining why many of our software projects had problems delivering acceptable products on schedule and within budget. Today, life is easier. We run our software organization like a business. Few of our projects are in trouble. Yes, there are still challenges that we must address. But, we aren't scolded any longer for being the problem on the project. Other organizations are now taking our place in the hot seat.

While the numbers we presented are fictitious, the facts aren't. Because we have the improvement data, we can justify our investments. That's why our initiatives have been funded and why we are well on the path to achieving level 5 hopefully later this year.

article continued on page 16

About the Authors

Donald J. Reifer is a consultant with Reifer Consultants, Inc. in Torrance, California. He specializes in change management.

He has over 30 years of experience managing large software projects and putting software technology to work in Fortune 500 firms. From 1993 to 1995, he was Chief of the Ada Joint Project Office, Technical Advisor to the Center for Software and chief of the DoD Software Reuse Initiative under an Intergovernmental Personnel Act assignment with the Defense Information Systems Agency. Reifer is currently helping clients insert product line and component-based software engineering technologies into their software operations.

Doug Walters is a senior software executive with Northrop Grumman ES. Starting as a programmer over 42 years ago, Mr. Walters has managed many major software-intensive projects, the Northrop Grumman ES Software Engineering Operations, and is involved in the major strategic initiatives involving software engineering and enterprise-wide process improvement.

Recently, he managed Northrop Grumman ES software efforts on SBIRS Low PDRR. Prior to that, he managed a large real-time radar system software organization. Mr. Walters is currently developing improvement strategies for Northrop as part of a greybeard team.

Al Chatmon manages the Northrop Grumman ES process group. He is currently leading their push to level 5. Previously, Mr. Chatmon managed real-time avionics projects and led process improvement efforts for projects like the F22 radar. Mr. Chatmon has been instrumental in Northrop Grumman ES successes in achieving SE CMM Level 4 and ISO TickIT compliance. Before joining Northrop, Mr. Chatmon retired as an officer with the U.S. Air Force where he managed several large software projects primarily in the avionics area.

Authors Contact Information

Donald J. Reifer
Reifer Consultants, Inc.
P.O. Box 4046
Torrance, CA 90510
Phone: 310-530-4493
E-mail: d.reifer@ieee.org

Doug Walters
Northrop Grumman ES
P.O. Box 746
Baltimore, MD 21203
Phone: 410-765-5305
E-mail: charles_d_walters@mail.northgrum.com

Al Chatmon
Northrop Grumman ES
P.O. Box 746
Baltimore, MD 21203
Phone: 410-765-4217
E-mail: albert_i_chatmon@mail.northgrum.com

References

1. Cross, Steve: "Reflections on the Process Revolution," Tutorial Software Management (6th Edition), IEEE Computer Society, New York (2002) 77-90.
2. Haley, Thomas J.: "Software Process Improvement at Raytheon," Software, Vol. 13, No. 6, IEEE Computer Society, New York (1996) 33-41.
3. Reifer, Donald J.: "Let the Numbers do the Talking," Crosstalk, Vol. 15, No. 3 (2002) 4-8.
4. Reifer, Donald J.: *Practical Software Reuse*, John Wiley & Sons, New York (1997).
5. Boehm, B. W., Abts, C., Brown, A. W., Chulani, S., Clark, B., Horowitz, E., Madachy, R. Reifer, D. and Steece, B.: Software Cost Estimation with COCOMO II, Prentice-Hall, New York (2000).
6. Poulin, Jeffrey: Measuring Software Reuse, Addison-Wesley, New York (1997).
7. Reifer, Donald J.: Making the Software Business Case: Improvement by the Numbers, Addison-Wesley, New York (2001).

Data & Analysis Center for Software (DACS) Annoucement

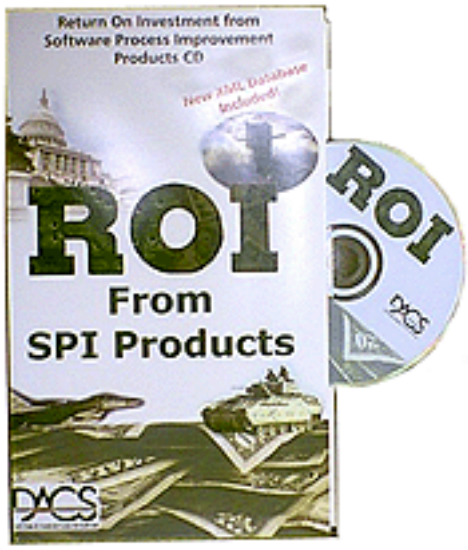
DACS DATA CALL

The DACS is asking for data to update its Return-On-Investment (ROI) from Software Process Improvement (SPI) database

The DACS offers this valuable product as a thank you for your participation. ▼

▼ Fold Here ▼

On the next page the DACS has included a ROI from SPI Data Collection Survey. Anyone participating in this survey will gain access to the searchable DACS ROI from SPI database. In addition to this all participants will receive a FREE copy of the DACS CD-ROM entitled "ROI from SPI Products" shown here. This product is described in detail on page 5 of this newsletter.



▼ Fold Here ▼

NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES

BUSINESS REPLY MAIL
FIRST-CLASS MAIL PERMIT NO. 120 ROME NY
POSTAGE WILL BE PAID BY ADDRESSEE

DATA & ANALYSIS CENTER FOR SOFTWARE
775 DAEDALIAN DR
ROME NY 13449-0139

DACS Software Process Improvement (SPI) Data Collection Survey

1. What is the current Software Process Improvement technique used for which you have data?

❑ Cleanroom ❑ PSP/TSP ❑ CMM ❑ ISO ❑ Reuse ❑ Formal Inspection ❑ Other (Describe below):

_____

2. Briefly describe what technique was used prior to the SPI technique identified above (e.g., "ad hoc", "CMM Level 2 going to Level 3", "0% Reuse going to 60% Reuse", etc.):

_____

_____

3. How many years has your organization used the SPI technique identified in Question 1? _____ yrs.

4. For each applicable attribute listed below, use percent, dollars, time, etc., to describe the positive or negative results obtained using the SPI technique identified in Question 1. Also, identify the year(s) for which the data is calculated (e.g., '99-'01), the formula used to calculate the attribute value, and provide any relevant comments. If necessary, formula and comments can be included on a separate sheet.
Ex.: Defect Reduction = 10% = [(Old # Defects – New # Defects)/(Old # Defects)]*100

Attribute	Value & Units	Year(s)	Formula	Comments
1. Defect Reduction				
2. Reliability Improvement				
3. Productivity Improvement				
4. Schedule Reduction				
5. Cycle Time Improvement				
6. Reduction in Employee Turnover				
7. Market Share Improvement				
8. Profitability Increase				
9. Cost Incentive Benefits				
10. Rework Reduction				
11. Maintenance Reduction				
12. Return on Investment (ROI)				

All information below must be filled out before data can be used:

Name: _____

Company: _____

Address: _____

City: _____ State: _____ Zip+4: _____

Phone: _____ Fax: _____ E-mail: _____

❑ Please send me a Nondisclosure Agreement before I submit data

Making Investment Decisions for Software Process Improvement

Daniel T. Fetzer, Science Applications International Corporation (SAIC)

1. Introduction

Software managers are under intense pressure to improve the quality of their products, accuracy of their plans and budgets, and to reduce the cost of software projects. Such gains can only come by process improvements. Yet few software managers are able to quantify the short and long term benefits of proposed software process improvements. One of the impediments that prevent many organizations from implementing needed improvements is the failure of software managers to make an effective business case to obtain the start-up funding. Process improvements typically require up-front investments to design, deploy, and support new processes, as well as investments in training, technology, tools, and cultural change.

A *Process Improvement Proposal (PIP)* can be defined as any proposed initiative to introduce technology or process changes into a software organization for the purpose of improving the quality or productivity of its software development process. A proposal could be broad in scope such as adopting a new development methodology or embarking on advancing a certain level in the Capability Maturity Model (CMM) framework [1]. A more narrowly defined proposal could involve modifying a testing process or adopting a development tool.

PIPs can be viewed as potential investment alternatives, since the costs will be more immediate and the projected benefits will be long term. The investment is the total cost involved in implementing and maintaining a process improvement. The expected higher *profit* would result from a combination of factors: a reduction in life cycle cost to develop and support the software, higher market share and sales from reduced time to market, or a higher demand and sale price from producing a superior quality product.

2. A Decision Support Framework

We have developed a general framework and prototype tool to support evaluating Software Process Improvements on the basis of their economic desirability [1]. This framework is based on the standard principles of Cost-Benefit Analysis (CBA) [3] which include:

1. Use of discounted cash flow analysis to account for the time value of cost and schedule impacts
2. Use of life-cycle cost-benefit analysis
3. Adoption of with-without rather than the before-after perspective in comparing alternatives
4. Use of net-present value as the single best financial criterion in aggregating costs and benefits over time
5. Use of corporate opportunity cost of capital as the appropriate discount rate in discounted cash flow calculations.

Our framework includes an effect taxonomy for classifying the cost-benefit effects associated with Software Process Improvement (SPI). The top-level categories for this taxonomy include:

1. Implementation and Support
2. Production Effects
3. Quality Effects
4. Cycle Time
5. Customer / Market Effects

For each new kind of SPI considered, we would construct a template to identify the set of cost-benefit effects for the process improvement. For each cost-benefit effect we provide quantification functions and parameters based upon the best available industry data or models for estimating the effects for a given development organization or environment. Figure 1. provides an example profile of how two hypothetical PIPs might be evaluated within this framework.

3. Evaluating Process Improvements Proposals

We can apply a systematic procedure for evaluating process improvement proposals as follows:

1. Recognize a problem or opportunity
2. Identify alternative Process Improvements Proposals (PIPs)
3. Determine opportunity cost (discount rate)
4. Determine the time horizon
5. Analyze each proposal
   a. Identify the cost-benefit effects

article continued on page 20

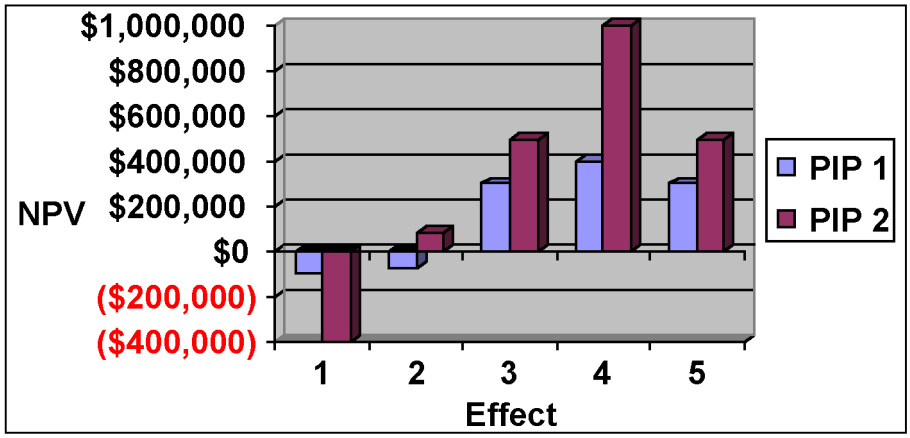


Figure 1. Example Profile of Two Process Improvement Proposals

*Note: NPV is the abbreviation for Net Present Value

b. Estimate and quantify cash flows for each effect

c. Summarize the NPV* for the PIP

6. Compare alternatives.

3.1 Recognize a Problem or Opportunity

The starting point for an organization to consider a process improvement is first to recognize a problem or an opportunity for improvement. The recognition of problems in a process immature organization may occur after much damage has already been done. One or more failed software projects due to some combination of missed deadlines, blown budgets, and unacceptable software quality will prompt management to seek solutions. In more mature organizations, management is more proactive in making problems and opportunities visible and in taking corrective actions at the first signs of trouble. In the most mature organizations, a continuous improvement paradigm is part of the culture. Performance is monitored against planned performance targets and deviations may trigger causal-analysis studies with follow-up corrective actions. Metrics programs and improvement models such as the Goal-Question-Metric paradigm [4] can be invaluable for making problems visible. Problems and opportunities should be defined in terms of the negative effects to be eliminated (e.g., cost overruns, missed schedules) or the positive effects to be improved (e.g., increase business, customer satisfaction).

3.2 Identify Alternative Process Improvement Proposals (PIPs)

Once the problems and opportunities have been well-defined, one or more process improvement proposals can be identified to solve the problems and realize the needed gains. Management can use the framework to identify potential solutions. The effects that they would like to change can be compared to the framework to identify PIPs that address those effects.

3.3 Determine the Opportunity Cost (Discount Rate)

The discount rate is a critical parameter in the Net Present Value (NPV) calculation since it can affect whether a single proposal has a NPV > 0 or change the ratings among several proposals. Higher rates penalize proposals with benefits occurring farther in the future. Within a private business, the discount rate should already be established by top management based on the cost of capital for the business.

3.4 Determine the Time Horizon

The time horizon will depend on the type of improvement and the environment where it is to be implemented. A 5-10 year horizon is usually sufficient. Since future benefits are discounted, benefits are greatly diminished beyond 5-10 years and would have minimal impact on the decision.

3.5 Analyze Each Proposal

3.5.1 Identify the Cost-Benefit Effects

Identifying cost and benefit effects is one of the most important steps in conducting a Cost-Benefit Analysis (CBA). Examples of software process costs include extra time to

perform a new process step, consulting fees, training materials, and the cost of tools to support the improvement. In general costs are relatively immediate, certain, and tangible. Benefits often take the form of cost avoidance such as reduced rework, error reduction, improved quality, time savings, reduced time to market, and improved process control. Less tangible benefits often cited in the literature for software process improvements include "improved customer satisfaction" leading to higher future sales and customer retention. Benefits are often more long-term, uncertain and less tangible than costs. The costs and benefits for a process improvement can be identified from the literature and from considering the impacts it will have within a particular environment.

Our decision support framework provides significant assistance to the decision maker by identifying the potential cost-benefit effects an organization may expect to receive from a PIP. A decision analyst may choose to add or subtract from the provided list of potential effects.

3.5.2 Estimate and Quantify Cash Flows for Each Effect

The most critical and most difficult aspect of conducting a cost-benefit analysis is quantifying the costs and benefits and determining the time periods the costs and benefits will be realized. The main difficulty is the unavoidable fact that the analyst is faced with forecasting the future. However, as much as possible, it is still important to quantify these impacts. As technology-economist Peter Sassone has stated: "Only through quantification is the aggregation of effects and the analysis of trade-offs generally possible." [2]

The uncertainty and risk associated with the estimates can be handled through sensitivity analysis, adding a risk premium in future years, or providing ranges of values for some of the parameters.

The estimator should only be concerned with marginal cost-benefit flows (i.e., cash flow differences from the baseline scenario). The organization's historical data as well as data and estimates from the literature can be useful for estimating cost-benefit effects.

The framework allows the decision maker to estimate cash flows by time periods in the future for each effect. Estimation models and default parameters help facilitate the estimation process, but allow the estimator the flexibility to override values as needed.

3.5.3 Calculate the NPV Metric for the Proposal

Net Present Value (NPV) is a method for discounting projected costs or benefits which will occur in the future. NPV is considered to be a superior financial criterion for comparing two alternative proposals. NPV recognizes the time value of money and allows us to reduce a stream of costs and benefits to a single number which we can use to evaluate a single alternative or to compare multiple alternatives. The formula for the NPV is

$$NPV = \sum_{t=0}^{n} \frac{B_t - C_t}{(1+r)^t}$$

where

- Bt is the monetary value of benefits received at time t,
- Ct is the cost incurred at time t,
- r is the discount rate,
- n is the time horizon for the decision (or the life of the project), and
- t is the time in units such as years or months.

3.6 Compare Alternatives

There are three mutually exclusive forms a CBA decision problem may take:

1. Evaluate whether or not to implement a single proposal

   For this situation the proposal is worth pursuing if the NPV is greater than zero.

2. Choose a single proposal to implement from among several alternatives

   Select the proposal with the maximum NPV greater than zero.

3. Select a set of proposals to implement from a larger set of possibilities

   For this form of the decision problem, one must first determine whether the proposals are independent and if the proposals are subject to a capital constraint, which limits the initial expenditures that can be spent on the selected set of proposals. A

article continued on page 22

proposal is independent of other proposals if the NPV of a proposal is not affected by whether or not the other proposals are implemented. If proposals are dependent, then one must form all possible subsets of combinations of proposals and evaluate the NPV of each combination. That is:

IF proposals are independent

THEN

IF Capital Constraint

THEN rank by ROI > 1

ELSE rank by NPV > 0

END IF

ELSE (proposals are dependent)

IF capital constraint

THEN find feasible sets maximize NPV

ELSE find possible sets maximize NPV

END IF

END IF

The NPV evaluation provides a structured way for the decision maker to understand the trade-offs and to compare alternatives. Of course the decision maker should not blindly select the option with the highest or maximum NPV. The assumptions behind the calculations, the discount rates, and estimating procedures should be thoroughly reviewed and understood.

4. Summary

To test our cost-benefit framework, we have constructed and validated templates for using Emerald, a software risk assessment tool [5], and for four key Cleanroom technologies: sequence-based specification, functional verification, incremental development, and statistical testing [6].

The value of performing a cost-benefit analysis is in improving how decisions are made for implementing and sustaining improvement efforts. The impact of potential improvements is difficult for software managers to assess and even more difficult for sponsors to understand. Intense schedule pressure often leads to a focus on short-term gains (e.g., writing code with insufficient design, poor architecture, and no code reviews) at the expense of long-term losses (e.g., extended testing cycle, high rework, unacceptable defect levels). To a certain extent the failure to implement and sustain improved practices is caused by uninformed (or out of control) management failing to understand their long-term costs and benefits. Our framework and prototype is designed to help remedy this situation by making it easier to evaluate and visualize the economic impact of improved practices.

About The Author

Dan Fetzer received his Ph.D. in Computer Science from the University of Tennessee in 2000. He has over 20 years experience in software engineering and information technology. Currently, he is a Senior Software Process Consultant with Science Applications International Corporation (SAIC).

Author Contact Information

Daniel T. Fetzer
Science Applications International Corporation (SAIC)
301 Laboratory Road
P. O. Box 2501
Oak Ridge, TN 37831

Phone: 1-865-425-4034
Fax: 1-865-425-4121
E-mail: fetzerda@saic.com

References

[1] Paulk, M. C., C. V. Weber, B. Curtis, M. B. Chrissis, The Capability Maturity Model: Guidelines for Improving the Software Process, Addison-Wesley, Reading, MA, 1995.

[2] Fetzer, D. T., Cost-Benefit Analysis for Software Process Improvement, Ph.D. Dissertation, University of Tennessee, 2000.

[3] Sassone, Peter G., William A. Schaffer, Cost-Benefit Analysis: A Handbook, Academic Press, New York, 1978.

[4] Birk, A., P. Derks, R. van Solingen, J. Jarvinen, Business Impact, Benefit, and Cost of Applying GQM in Industry: An In-Depth, Long-Term Investigation at Schlumberger RPS, Fraunhofer Institute for Experimental Software Engineering, Germany, August 1998, IESE-Report 040.98/E

[5] Hudepohl, John, Emerald: Software Metrics and Models on the Desktop, IEEE Software, vol. 13, no. 5, September 1996, pp. 56-60.

[6] Prowell, S. J., C. Trammell, R. Linger, J. Poore, Cleanroom Software Engineering: Technology and Process, Addison-Wesley, 1999.

How to Estimate ROI for Inspections, PSP[sm], TSP[sm], SW-CMM®, ISO 9000, and CMMI[sm]

by David F. Rico, Independent Consultant

1. Introduction

Return-On-Investment (ROI) is the quantification of the financial return of an investment. In more technical terms, ROI is the actual value developed by comparing program costs to benefits, measuring the magnitude of benefits relative to costs, the net benefit after expending some level of resources, or profit computed by dividing net income by assets used.

This article shows software managers and engineers how to estimate ROI early, quickly, and accurately by applying practical top-down methods for rapidly producing estimates of ROI for popular approaches to Software Process Improvement (SPI) (and is based on Rico [1]). These approaches include: Inspections, Personal Software Process[sm] (PSP), Team Software Processs[sm] (TSP), Software Capability Maturity Model® (SW-CMM), ISO 9001, and Capability Maturity Model Integration[sm] (CMMI).

2. Model

While, one can spend months and years analyzing the literature and searching for relevant approaches to defining and estimating ROI, Phillips [2] provides one-stop shopping on this seemingly futile journey. Phillips defines the basic model for estimating ROI, as well as a complete process for applying these simple equations in a professional manner. Phillips' ROI model consists of two basic equations:

- Benefit/Cost Ratio (B/CR): B/CR is a simple process of dividing the benefits of SPI by the costs of SPI.
- Return on Investment (ROI%): The ROI% equation is similar to the B/CR equation, except that the costs of SPI are subtracted from the benefits of SPI before dividing by the costs.

3. Examples

This section provides simple and relatively accurate examples of how to apply Phillips' basic equations for estimating the ROI of six major approaches to SPI .

B/CR calculations for each of the methods described here are shown in Figure 1.

Phillips' B/CR and ROI% equations will be applied to benefit data from Rico [3] as well as other sources of SPI data. The six approaches to SPI are:

- Inspection: The software inspection process is a highly structured and facilitated group meeting to objectively identify the maximum number of software defects with the purpose of improving software quality.
- PSP: The PSP is a training curriculum to teach simple, but powerful techniques in software project management and quality management.
- TSP: The TSP is an extension of PSP, which introduces group software project management techniques versus the individual focus taught by PSP.

article continued on page 24

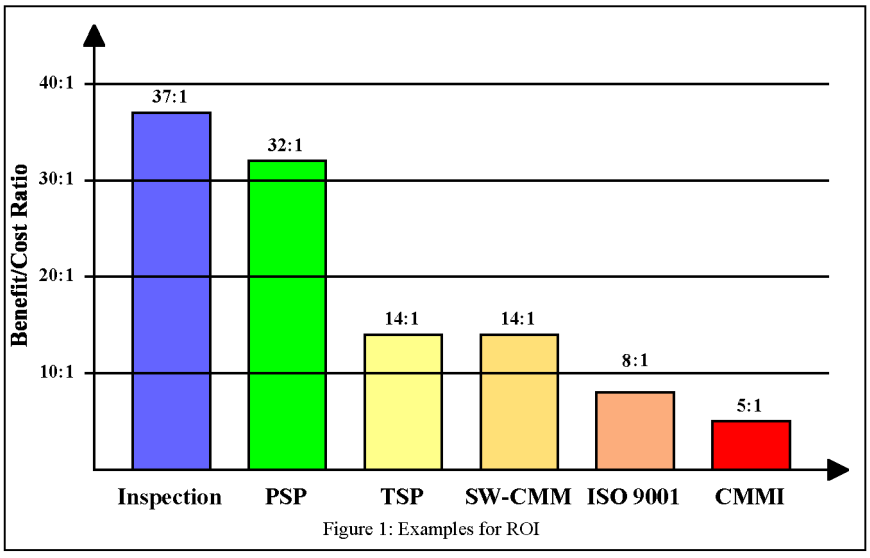


Figure 1. B/CR Calculations

Footnotes

sm Personal Software Process, PSP, Team Software Process, TSP, Capability Maturity Model Integration, and CMMI are service marks of Carnegie Mellon University.

® Capability Maturity Model and Software CMM are registered in the U.S. Patent and Trademark Office.

- SW-CMM: The SW-CMM is a supplier selection model created by the U.S. DoD to evaluate and select software contractors that practice minimum software project management techniques.
- ISO 9001: ISO 9001, like the SW-CMM, is a supplier selection model created by the European Union to evaluate, identify, and select suppliers that practice minimum quality management techniques.
- CMMI: The CMMI, which is the newest version of SW-CMM, is also a supplier selection model created by the U.S. DoD to evaluate and select systems engineering contractors that practice minimum systems engineering management techniques.

4. Inspection

Let's examine the dynamics of Inspection cost, benefit, and ROI analysis using Phillips' equations for B/CR and ROI%.

- Training Cost: Let's begin by modeling the training costs for implementing Inspections on a four-person project. The average market price for Inspection training is about $410 per person. The average length of time for Inspection training is three days or 24 business hours. At a minimum cost of $100 per hour, training time comes to $2,400. Add $410 to $2,400 for a total of $2,810 per person for Inspection training. Multiply $2,810 by four people and that comes to $11,240 to train four people to perform Inspections.
- Implementation Cost: Now let's examine the cost of implementing Inspections by our four trained inspectors. Let's assume the project will develop 10,000 software source lines of code (SLOC), which is not unlikely for a web project in modern times. (Inspections of requirements, designs, and tests drive the Inspection costs even higher, but are omitted for simplicity's sake.) At an Inspection rate of 240 SLOC per meeting, that comes to approximately 41.67 meetings. Since each Inspection run requires about 17 hours for planning, overviews, preparation, meetings, rework, and follow-up, we then multiply 41.67 by 17 for a total of 708.33 hours. Once again, at $100 per hour, that comes to $70,833 for our four trained inspectors to perform Inspections on 10,000 SLOC. _ Total Cost: So, we add the training cost of $11,240 to the implementation cost of $70,833, and we arrive at a total cost of $82,073 for four trained inspectors to Inspect 10,000 SLOC.
- Total Life Cycle Benefits: The estimated maintenance hours for 10,000 SLOC after our four trained inspectors perform their Inspections are 11,806. The estimated maintenance hours for 10,000 SLOC with no Inspections are 41,800. So, our four trained inspectors have saved 29,994 maintenance hours on their very first implementation of Inspections. Multiply 29,994 by $100 and the estimated savings are an impressive $2,999,400.
- B/CR: (B/CR is benefits divided by costs.) Therefore, divide $2,999,400 by $82,073 and the B/CR for Inspections is 37:1.
- ROI%: (ROI% is benefits less costs divided by costs times 100.) Therefore, first subtract the $82,073 in Inspection costs from the $2,999,400 in Inspection benefits and divide the results by the $82,073 in Inspection costs and multiply by 100 for an impressive ROI% of 3,555%.

5. PSP

Now, let's examine the dynamics of PSP cost, benefit, and ROI analysis using Phillips' equations for B/CR and ROI%.

- Training Cost: Let's begin by modeling the training costs for implementing PSP on a four-person project. The Software Engineering Institute's (SEI's) price for PSP training is $5,000 per person. The costs of the airline, hotels, meals, and parking are about $5,400 for two weeks. The length of time for PSP training is 10 days or 80 business hours. Each hour of classroom time requires approximately one hour of non-classroom time for a total of 80 more hours. At a minimum cost of $100 per hour, training time comes to $16,000. Add $5,000, $5,400, and $16,000 for a total of $26,400 per person for PSP training. Multiply

$26,400 by four people and that comes to $105,600 to train four people to perform PSP.

- Implementation Cost: Now let's examine the cost of implementing PSP by our four PSP-trained engineers. Let's assume the project will develop 10,000 software source lines of code (SLOC), once again, which is not unlikely for a web project in modern times. At an average productivity rate of 25 SLOC per hour, that comes to approximately 400 hours. At $100 per hour, that comes to $40,000 for our four PSP-trained engineers to produce 10,000 SLOC using PSP.
- Total Cost: So, we add the training cost of $105,600 to the implementation cost of $40,000, and we arrive at a total cost of $145,600 for four PSP-trained engineers to produce 10,000 SLOC using PSP.
- Total Life Cycle Benefits: The estimated maintenance hours for 10,000 SLOC after our four PSP-trained engineers apply PSP are zero. The estimated maintenance hours for 10,000 SLOC without PSP are 41,800. So, our four PSP-trained engineers have saved 41,800 maintenance hours on their very first application of PSP. Typical software development hours for 10,000 SLOC are 5,088. However, software development hours with PSP are only 242, for an additional savings of 4,846 hours. Add 41,800 maintenance hours saved to 4,846 development hours saved for a total of 46,646 saved software maintenance and development hours. Multiply 46,646 by $100 and the estimated savings are an impressive $4,664,600.
- B/CR: (B/CR is benefits divided by costs.) Therefore, divide $4,664,600 by $145,600 and the B/CR for PSP is 32:1.
- ROI%: (The formula for ROI% is benefits less costs divided by costs times 100.) Therefore, first subtract the $145,600 in PSP costs from the $4,664,600 in PSP benefits, divide the results by the $145,600 in costs, and multiply by 100 for an impressive ROI% of 3,104%.

6. TSP

Now, let's examine the dynamics of TSP cost, benefit, and ROI analysis using Phillips' equations for B/CR and ROI%.

- Training Cost: Let's begin by modeling the training costs for implementing TSP on a four-person project. The SEI's price for TSP training is $4,000 per person. The costs of the airline, hotels, meals, and parking are about $2,700 for one week. The length of time for TSP training is 5 days or 40 business hours. At a minimum cost of $100 per hour, training time comes to $4,000. Add $4,000, $2,700, and $4,000 for a total of $10,700 per person for TSP-specific training. Add the $26,400 for PSP training to the $10,700 for TSP training and the total overall TSP costs come to $37,100 per person. Multiply $37,100 by four people and that comes to a budget-busting $148,400 to train four people to use TSP.
- Implementation Cost: Now let's examine the cost of implementing TSP by our four TSP-trained engineers. Let's assume the project will develop 10,000 software source lines of code (SLOC), once again, which is not unlikely for a web project. At an average productivity rate of 6.12 SLOC per hour, that comes to approximately 1,634 hours. At $100 per hour, that comes to $163,400 for our four TSP-trained engineers to produce 10,000 SLOC using TSP. (See Humphrey [4] for an in-depth analysis of TSP metrics, models, effort, and costs.)
- Total Cost: So, we add the training cost of $148,400 to the implementation cost of $163,400, and arrive at a total cost of $311,800 for four TSP-trained engineers to produce 10,000 SLOC using TSP.
- Total Life Cycle Benefits: The estimated maintenance hours for 10,000 SLOC after our four TSP-trained engineers apply TSP are zero. The estimated maintenance hours for 10,000 SLOC without TSP are 41,800. So, our four TSP-trained engineers have saved 41,800 maintenance hours on their very first application of TSP. Typical software development hours for 10,000 SLOC are 5,088. However, software development hours with TSP are only 1,634,

article continued on page 26

for an additional savings of 3,454 hours. Add 41,800 maintenance hours saved to 3,454 development hours saved for a total of 45,254 saved software maintenance and development hours. Multiply 45,254 by $100 and the estimated savings are an impressive $4,525,400.

- B/CR: (B/CR is benefits divided by costs.) Therefore, divide $4,525,400 by $311,800 and the B/CR for TSP is 14:1.
- ROI%: (ROI% is benefits less costs divided by costs times 100.) Therefore, first subtract the $311,800 in TSP costs from the $4,525,400 in TSP benefits and divide the results by the $311,800 in TSP costs and multiply by 100 for an impressive ROI% of 1,351%.

7. SW-CMM

Now, let's examine the dynamics of SW-CMM Level 2 and 3 cost, benefit, and ROI analysis using Phillips' equations for B/CR and ROI%.

- Deployment Cost (Level 2): Let's begin by modeling the deployment costs for implementing SW-CMM for four projects as a representative sample of a software producing organization. Rico [5] makes the following estimates: 66 hours for 6 policies, 264 hours for 24 procedures, 512 hours for 32 documents, 304 hours for 76 work authorizations, 464 hours for 116 records, 544 hours for 136 reports, and 304 hours for 76 meeting minutes. The total deployment hours for SW-CMM Level 2 are 2,458. Multiply 2,458 by $100 and that comes to $245,800.
- Deployment Cost (Level 3): Rico [5] makes the following estimates: 77 hours for 7 policies, 154 hours for 14 procedures, 1,280 hours for 80 documents, 176 hours for 44 work authorizations, 592 hours for 148 records, 336 hours for 84 reports, and 192 hours for 48 meeting minutes. The total deployment hours for SW-CMM Level 3 are 2,807. Multiply 2,807 by $100 and that comes to $280,700.
- Assessment Preparation Costs: Let's estimate four projects of five people in 13 indoctrination courses at 2 hours each which totals 520 hours. Let's similarly estimate four projects of five people in 13 response-conditioning courses at 2 hours, each which also totals 520 hours. Finally, let's estimate four projects of five people in one 40 hour mock assessment or two 20 hour mock assessments for total of 800 hours. Now, let's add 520 indoctrination hours, 520 response conditioning hours, and 800 mock assessment hours for a total of 1,840 hours. Finally, let's multiply 1,840 by $100 for a total of $184,000 in assessment preparation costs.
- Total Deployment Costs: Combine $245,800, $280,700, and $184,000 for a total SW-CMM Level 2 and 3 deployment cost of $710,500.
- Assessment Cost: The SEI estimates that an assessment requires up to 3,208 hours of internal labor (not including the assessors effort). However, for our four projects of five people let's estimate 62 hours for planning, 234 hours for preparation, 646 hours for the appraisal itself, and 57 hours of follow-up which totals 1,000 hours. (This doesn't include the assessor's time, and the SEI estimates over three times more internal effort.) So, now multiply 1,000 by $100 for a total labor cost of $100,000 plus $40,000 in assessment fees for a total assessment cost of $140,000.
- Total SW-CMM Cost: Adding the $710,500 in total deployment costs to the $140,000 in assessment costs results in a total SW-CMM cost of $850,500.
- Total Life Cycle Benefits: Let's assume each of our four projects also build 10,000 SLOC software products. Let's also assume that each of our four projects apply Inspections to satisfy their SW-CMM Level 3 goals. Now, we're ready to begin estimating the benefits. Let's assume each of our four projects saves an average of 27,867 maintenance hours by performing Inspections for total maintenance savings of 111,466 hours. Now, let's assume our productivity doubles at SW-CMM Level 3 as reported by Diaz [6], which results in a per project savings of 2,544 hours for a total of 10,176 development hours saved. Add the 111,466 hours in maintenance savings to the

10,176 hours in development savings for a total of 121,642 hours saved at SW-CMM Level 3. Multiply 121,642 by $100 to arrive at an estimated savings of $12,164,200.

- B/CR: (B/CR is benefits divided by costs.) Therefore, divide $12,164,200 by $850,500 and the B/CR for SW-CMM is 14:1.
- ROI%: (ROI% is benefits less costs divided by costs times 100.) Therefore, first subtract the $850,500 in SW-CMM costs from the $12,164,200 in SW-CMM benefits and divide the results by the $850,500 in costs and multiply by 100 for an impressive ROI% of 1,330%.

8. ISO 9001

Now, let's examine the dynamics of ISO 9001 cost, benefit, and ROI analysis using Phillips' equations for B/CR and ROI%.

- Deployment Costs: Let's begin by modeling the costs for ISO 9001 in a 20-person software organization. El Emam's [7] cost model results in 2,184 hours to prepare for ISO 9001 registration. Multiply 2,184 by $100 and that comes to $218,396.
- Assessment Costs: Let's estimate four projects of five people at 32 hours each which totals 640 hours to prepare for the assessment. Multiply 640 by $100 for a total of $64,000 in assessment preparation costs. Add a $48,000 assessment fee to the $64,000 assessment preparation cost for a total assessment cost of $112,000.
- Total Deployment Costs: Combine $218,396 and $112,000 for a total ISO 9001 deployment cost of $330,396 for ISO 9001 registration.
- Total Life Cycle Benefits: Let's assume each of our four projects also build 10,000 SLOC software products. Now, we're ready to begin estimating the benefits. Let's assume each of our four projects has a 15% increase in maintenance savings, which is consistent with ISO 9001 experiences. Multiply 41,800 maintenance hours by 15% for 6,270 maintenance hours saved per project. Multiply 6,270 by 4 for a total maintenance savings of 25,080 hours. Now, let's assume each of our four projects has a 13% increase in productivity, which is consistent with ISO 9001 experience. Multiply 5,088 development hours by 13% for 661 development hours saved per project. Multiply 661 by 4 for a total development savings of 2,646 hours. Now, add the 25,080 maintenance hours saved to the 2,644 development hours saved for a total of 27,726 total maintenance and development hours saved. Finally multiply the 27,726 maintenance and development hours saved by $100 for a total of $2,772,600 in savings by using ISO 9001.
- B/CR: (The formula for B/CR is benefits divided by costs.) Therefore, divide $2,772,600 by $330,396 and the B/CR for ISO 9001 is 8:1.
- ROI%: (The formula for ROI% is benefits less costs divided by costs times 100.) Therefore, first subtract the $330,396 in ISO 9001 costs from the $2,772,600 in ISO 9001 benefits and divide the results by the $330,396 in ISO 9001 costs and multiply by 100 for an impressive ROI% of 739%.

9. CMMI

Now, let's examine the dynamics of CMMI cost, benefit, and ROI analysis using Phillips' [2] equations for B/CR and ROI%.

- CMMI Policies and Procedures: Let's begin by modeling the costs for implementing CMMI policies and procedures for four projects as a representative sample of a systems engineering organization. Rico [8] makes the following estimates: CMMI Level 2 requires 2,091 hours to develop 56 policies and procedures and CMMI Level 3 requires 3,771 hours to develop 101 policies and procedures. So, 5,862 hours are required to develop CMMI Level 2 and 3 policies and procedures. Multiply 5,862 by $100 and that comes to $586,200. Half of this is software engineering, which amounts to $293,100.
- CMMI Evidence of Use: Rico [8] also makes the following estimates: CMMI Level 2 requires 10,304 hours to develop 138 products for four systems engineering projects and CMMI Level 3 requires 20,533 hours to develop 275 products for these

article continued on page 28

projects. So, 30,837 hours are required to develop CMMI Level 2 and 3 products. Multiply 30,837 by $100 and that comes to $3,083,700. Half of this is software engineering, which amounts to $1,541,850. _ CMMI Implementation Costs: Now add $293,100 for CMMI Level 2 and 3 policies and procedures and $1,541,850 for CMMI Level 2 and 3 products for four projects, which is $1,834,950 for software engineering.

- Assessment Preparation Costs: Let's estimate four projects of ten people in 20 indoctrination courses at 2 hours each which totals 1,600 hours. Let's similarly estimate four projects of ten people in 20 response conditioning courses at 2 hours each, which also totals 1,600 hours. Finally, let's estimate four projects of ten people in one 40 hour mock assessment or two 20 hour mock assessments for total of 1,600 hours. Now, let's add 1,600 indoctrination hours, 1,600 response conditioning hours, and 1,600 mock assessment hours for a total of 4,800 hours. Finally, let's multiply 4,800 by $100 for a total of $480,000 in assessment preparation costs. Half is software engineering, which amounts to $240,000.
- Total Deployment Costs: Combine $1,834,950 and $240,000 for a total CMMI Level 2 and 3 deployment cost of $2,074,950 for software engineering.
- Assessment Cost: For our four projects of five people, let's estimate 636 hours for the plan and prepare for appraisal stage. Let's estimate 1,018 hours for the conduct appraisal stage. And, let's estimate 106 hours for the report results stage. This totals to 1,760 hours. Multiply 1,760 by $100 for an internal labor estimate of $176,000. Add an assessment fee of $64,615 for a total assessment cost of $240,615. (Assessment costs were based on labor distributions from Carnegie Mellon University [9].)
- Total CMMI Cost: Adding the $2,074,950 in total deployment costs to the $240,615 in assessment costs gives a total CMMI cost of $2,315,565.
- Total Life Cycle Benefits: Let's assume each of our four projects also build 10,000 SLOC software products. Let's also assume that each of our four projects apply Inspections to satisfy their CMMI Level 3 goals. Now, we're ready to begin estimating the benefits. Let's assume each of our four projects saves an average of 27,867 maintenance hours by performing Inspections for total maintenance savings of 111,466 hours. Now, let's assume our productivity doubles at CMMI Level 3 as with the SW-CMM, which results in a per project savings of 2,544 hours for a total of 10,176 development hours saved. Add the 111,466 hours in maintenance savings to the 10,176 hours in development savings for a total of 121,642 hours saved at CMMI Level 3. Multiply 121,642 by $100 to arrive at an estimated savings of $12,164,200.
- B/CR: (B/CR is benefits divided by costs.) Therefore, divide $12,164,200 by $2,315,565 and the B/CR for CMMI is 5:1.
- ROI%: (ROI% is benefits less costs divided by costs times 100.) Therefore, first subtract the $2,315,565 in CMMI costs from the $12,164,200 in CMMI benefits and divide the results by the $2,315,565 in CMMI costs and multiply by 100 for an impressive ROI% of 425%.

10. Recommendations

This is an important part of the article. It is one of discovery, reflection, and future direction:

- Pinpoint High-ROI Factors: It's unnecessary to identify every cost and benefit factor when producing early, top-down estimates of ROI. The law of diminishing returns applies. There are only a few significant drivers of costs and benefits.
- Target High-ROI Approaches: This article is sufficient to point out approaches to SPI which yield the greatest benefits at the least possible cost. And, it reminds the reader that the best approaches are yet to come.
- Minimize Cost Incurrence: Choose low-cost, low-risk approaches to SPI. Using low-cost solutions to SPI guarantees successful, early returns.

article continued on page 31

Software Tech News Subscriber Survey

1. Which volume of the Software Tech News did you receive? STN 5:4 - ROI from SPI

2. When did you receive the newsletter? *(month/year)* _____

3. How satisfied were you with the CONTENT of the newsletter? (Article Quality)

❑ Very Satisfied ❑ Satisfied ❑ Neither Satisfied nor Dissatisfied ❑ Dissatisfied ❑ Very Dissatisfied

4. How satisfied were you with the APPEARANCE of the newsletter?

❑ Very Satisfied ❑ Satisfied ❑ Neither Satisfied nor Dissatisfied ❑ Dissatisfied ❑ Very Dissatisfied

5. How satisfied were you with the OVERALL QUALITY of the newsletter?

❑ Very Satisfied ❑ Satisfied ❑ Neither Satisfied nor Dissatisfied ❑ Dissatisfied ❑ Very Dissatisfied

6. How satisfied were you with the ACCURACY of the address on the newsletter?

❑ Very Satisfied ❑ Satisfied ❑ Neither Satisfied nor Dissatisfied ❑ Dissatisfied ❑ Very Dissatisfied

7. Approximately how much of the newsletter do you read?

❑ The entire issue ❑ Most of the content ❑ About half the content ❑ Briefly Skimmed ❑ Didn't read

8. Would you read this newsletter in an E-mail Newsletter format?

❑ Definitely ❑ Probably ❑ Not Sure ❑ Probably Not ❑ Definitely Not

9. How did you request the product or service?

❑ Phone Call ❑ E-mail ❑ DACS Website ❑ Subscription Form Other _____

10. Would you recommend the DoD Software Tech News to a Colleague?

❑ Definitely ❑ Probably ❑ Not Sure ❑ Probably Not ❑ Definitely Not

11. What topics would you like to see this newsletter devoted to, that we have *not* yet covered?

_____

Comments (Optional)

_____

_____

Contact Information (Optional)

Name: Position/Title:

Organization: Office Symbol:

Address:

City: State: Zip Code:

Country: E-mail:

Telephone: Fax:

Organization Type: ❑ Air Force ❑ Army ❑ Navy ❑ Other DoD _____

❑ Commercial ❑ Non-Profit ❑ Non-US ❑ US Government ❑ FFR&D ❑ Other _____

The first 50 people to send in a completed survey will receive a FREE DACS Information Pack CD.

This informative CD-ROM contains 2 past newsletters in PDF format. The topics of these newsletters are Software Reliability and High Performance Computing. In addition to these newsletter there are DACS brochures explaining our available services and the DACS Product and Services Catalog. This unique-shaped CD-ROM plays in your computer's regular CD drive. (Windows only)

▼ Fold Here ▼

Data & Analysis Center for Software (DACS)

http://iac.dtic.mil/dacs/

▼ Fold Here ▼

NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES

BUSINESS REPLY MAIL
FIRST-CLASS MAIL PERMIT NO. 120 ROME NY
POSTAGE WILL BE PAID BY ADDRESSEE

DATA & ANALYSIS CENTER FOR SOFTWARE
775 DAEDALIAN DR
ROME NY 13449-0139

- Avoid Cost-Intensive Approaches: This article sufficiently exposes the approaches to SPI which are sure to drain your organization's assets.
- Avoid Training-Intensive Approaches: Training-intensive approaches are generally unsuccessful in the marketplace because of their great expense, immense difficulty, and lack of sufficient tools for deployment beyond the classroom.
- Look for Low-Cost Automated Solutions: The future of SPI isn't in large bureaucratic and manually intensive approaches to SPI. The future is in low-cost, non-invasive automated tools that perform complex tasks in spite of us.
- Use Professional Methods for Analyzing ROI: This article guides readers toward relevant methods in ROI analysis and estimation. However, even the process of ROI is subject to low-cost automation. Look for low cost automation to ROI embedded in web-based project management tools.

About the Author

David F. Rico is a SPI consultant specializing in cost and benefit analysis. As an SPI strategist he consults in cost, benefit, and Return-on-Investment (ROI) economic analyses of Software Engineering standards, design, deployment, and measurement of software cost, quality, and reliability metrics and models, and design of measurement-intensive Software Quality Management Systems (SQMS).

His work includes; helping to design a $250M Software Engineering Toolset and the spacecraft software for NASA's $20B space station in the 1980s, performing graduate studies under SEI Level 5 space shuttle managers, aiding a $40B Japanese corporation to design a CMM self assessment tool in 1993, and designing a software cost model for 37 kinds of U.S. Navy fighter aircraft, helping to reengineer 36 logistics depots for America's largest foreign military customer. Rico also played key roles in the design of U.S. military intelligence satellites, and has supported 15 Software Engineering Process Groups (SEPGs) over the last decade.

Rico has over 19 years of Software Engineering experience. He has been an international keynote speaker, published numerous articles, and holds a B.S. in Computer Science and Master's Degree in Software Engineering.

Author Contact Information

David F. Rico
Phone: (443) 756-7118

E-mail: dave@davidfrico.com
URL: http://davidfrico.com

References

[1] Rico, D. F. (2002). Software Process Improvement: Modeling Return on Investment (ROI) [WWW document]. URL http://davidfrico.com/dacs02 pdf.htm

[2] Phillips, J. J. (1997). *Return-On-Investment in Training and Performance Improvement Programs*. Houston, TX: Gulf Publishing Company.

[3] Rico, D. F. (2000). Using Cost Benefit Analyses to Develop Software Process Improvement (SPI) Strategies [DACS Technical Report] (Contract Number SP0700-98-D-4000). Rome, NY: Air Force Research Laboratory/Information Directorate (AFRL/ IF), Data and Analysis Center for Software (DACS). http://www.dacs.dtic.mil/techs/abstracts/rico.html

[4] Humphrey, W.S. (2000). The Team Software Process [sm] (TSP). (CMU/SEI-2000-TR-023). Pittsburgh, PA: SEI.

[5] Rico, D. F. (n.d./2001). SEI level 2 thru 5: Cost model [WWW document]. URL http://davidfrico.com/sw-cmm-costpdf.htm

[6] Diaz, M., & Sligo, J. (1997). How Software Process Improvement Helped Motorola. *IEEE Software*, 14 (5), 75-81.

[7] El Emam, K., & Briand, L. C. (1997). Costs and Benefits of Software Process Improvement (IESE-Report 047.97/E). Kaiserslautern, Germany: University of Kaiserslautern, Fraunhofer-Institute for Experimental Software Engineering.

[8] Rico, D. F. (n.d./2001). Capability Maturity Model Integration: CMMI Introductory Overview [WWW document]. URL http://davidfrico.com/cmmipdf.htm

[9] Carnegie Mellon University (2001). Standard CMMISM Appraisal Method for Process Improvement (SCAMPISM), Version 1.1: Method definition document CMU/SEI-2001-HB-001. Pittsburgh, PA: Software Engineering Institute (SEI).

STN Vol. 5, No. 4

In This Issue

Data & Analysis Center for Software
P.O. Box 1400
Rome, NY 13442-1400

Return Service Requested

PRSRT STD
U.S. Postage
PAID
Permit #566
UTICA, NY