



Software Process Improvement – Targeted Optimization The SmartPhase Methodology

By Jamal Moustafaev, BBA, MBA, PMP
President & Principal Consultant
Thinktank Consulting

*However beautiful the strategy, you should occasionally look at the results.
Sir Winston Churchill*

Introduction

Popularity of SPI

Software Process Improvement (SPI) is enjoying a renewed interest lately. The topic has become more popular not only in North America, but also in Europe and some Asian countries as well. Judging by articles in the software development media, SPI has become more mainstream in the last 3-5 years with more and more companies starting to show interest in this process.

A lot has been written about software process improvement in the last twenty years. Several great models have been developed to support and aid companies willing to improve their software delivery. These include CMM, ISO/IEC 15504, BOOTSTRAP, Trillium, The V-Model and others.

Success of SPI

There have been a lot of reports about the success of SPI initiatives in Europe and North America. For example, a study conducted as early as 1994 provides the following information about the costs and benefits of CMM implementation:

- \$490-\$2,004 invested per software engineer per year
- 9%-67% annual increase in productivity
- 15%-23% annual reduction in cycle time
- 16%-94% annual reduction in field error reports
- Return on investment (ROI) ranging from 300% to 780%

Furthermore, Sami Zahran in his book “Software Process Improvement” provides several examples of successful SPI programs in several European countries and USA. These include companies like Motorola, US Air Force Logistics Command, Hughes Aircraft, Raytheon (all US), Ingegneria Informatica SPA, ENEL SPA, Computer-Logic SA (all Europe).

Issues with SPI

However, despite the evidence of tremendous successes of SPI initiatives there are still problems with SPI acceptance in the industry. According to some researchers 2/3 of SPI programs have failed. Others report that there are serious commitment issues arising in many SPI initiatives.

Recent survey of 300 CEOs by the Society for Information Management indicates that “IT/Business alignment” is a number one issue for C-level executives while “Reengineering business processes” is considered to be priority number ten.

Very frequently senior management cites lack of resources and time while explaining their reluctance to take on large SPI initiatives.

My personal interactions with executives of various companies in US and Canada have frequently yielded the following comments with respect to SPI programs:

- I can't justify the SPI investment with my bosses
- How do I know that CMM Level 3 is better for us than CMM Level 2? What is the value of such shift for my company?
- When do I stop with process improvement? Literature I read tells me that I have to improve continuously until I get to Level 5, but that is cost-prohibitive. We are not NASA, we are building insurance claims processing software.

All these facts allow for a possibility that while the SPI models are ultimately viable, the utilization of these methodologies in practice could be flawed.

Examination of a Classical SPI Model

Traditional SPI model

In order to understand these issues one has to revisit the classical SPI process. The SPI “classical school” suggests the following process improvement approach:

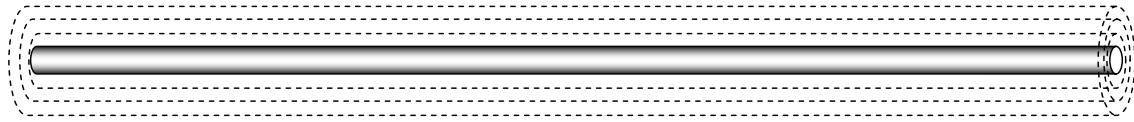
1. Conduct pre-assessment by understanding business goals of the company
2. Select an SPI roadmap (e.g. CMM, BOOTSTRAP, etc.)
3. Conduct an assessment of the development process in a company
4. Based on the results of your assessment establish a list of process improvement actions
5. Create an SPI plan
6. Execute the plan
7. Restart the cycle in order to continuously improve your processes

Basically if we use a pipeline metaphor, company's initial development process can be compared to a narrow pipe with a very small “throughput” capacity (see Figure 1).

Note: for illustrative purposes let's agree that in this paper “throughput” means both amount of work that can be done and the quality of work. In other words, the larger the diameter of the pipe, the more code of higher quality our technical team can produce

In each improvement cycle we improve the process by replacing the entire pipe with that of a larger diameter, thus progressing through maturity levels. According to Step 7 of the model we shouldn't stop until we achieve the largest diameter possible (e.g. CMM Level 5).

Figure 1



Questions and Observations

The examination of the classical SPI model and some of the above-mentioned issues warrants, the following set of questions:

Question 1 - Since the primary drive for software process assessment has not come from software development industry, but rather from acquirers of large, critical software-intensive – notably in telecom and defense industries, is it really surprising that current models of SPI are not entirely reflective of software business needs?

Question 2 - “Software process improvement is not a destination, it is a journey”. Once the organization goes through a complete cycle of SPI, it should begin the next one and repeat this continuously. Is this really the case? Does this mean that every company has to be at, say, CMM Level 5? After all, industry knows a lot of examples of successful software development companies operating quite effectively at levels well below, say, CMM level 5.

Question 3 - By the same token, can the company reach its “sweet spot” of process maturity if it has fulfilled the requirements of a certain level only partially?

Question 4 - Staged process improvement models (e.g. CMM) require the firm to complete the prerequisites of the model completely to achieve a certain stage; i.e. if the company fulfills 90% of level 3 requirements it is still considered to be at level 2. Doesn't this “all or nothing” approach act as a deterrent for companies' executives who are usually reluctant to invest in risky mega-projects? After all the financial utility theory tells us that a rational risk-averse individual will always choose Option B when provided with the following gambling (investment) options:

Option A:	Option B:
50% chance of success – payoff \$1000	50% chance of success – payoff \$10
50% chance of failure – loss of \$1000	50% chance of failure – loss of \$10

Question 5 - Classical SPI school tells us that monetary ROI arguments are somewhat, if not frequently, deficient when applied to the SPI justification. Some of the arguments used in this discussion are:

- Sound management and engineering practices should not have to be ROI issues;
- ROI arguments ignore intangible benefits
- ROI arguments are often used as a stalling tactic; they are a symptom of non-commitment

Return on investment, net present value and internal rate of return in various forms are one of the most trusted and reliable methods used by the business people and company shareholders to assess feasibility of new investments. The importance of financial modeling in software development was pioneered by Barry Boehm in his book “Software Engineering Economics” and

more recently reiterated by other authors. Since the mere existence of information technology and software development sectors is conditional upon generation of economic profits, can we outright dismiss the ROI arguments preferred by the business people?

Business Aligned SPI – The SmartPhase Methodology

Principles for the new model

The new proposed methodology is based on using existing roadmaps (e.g. CMM, BOOTSTRAP, ISO/IEC 15504) but utilizes them differently, through a gradual, targeted, business-aligned and value-adding method.

Principle 1 - We accept the fact that the optimal point of software development maturity is not necessarily at the highest level attainable in any of the SPI roadmaps or models. Each company has its own “sweet spot” of the technological maturity that is dictated strictly by economic factors. In other words, if we are adhering to one of the main principles of economics, we should be investing in the software improvement as long as the “marginal revenue” from our investment is larger than “marginal cost” of the investment itself.

Principle 2 - Furthermore the optimal spot can be achieved even if the company has fulfilled the requirements of any given level only partially.

Principle 3 - ROI (IRR, NPV) combined with risk should be the ultimate measures of the feasibility of the SPI process. Since every business is measured in terms of a combination of risk and return factors, we can achieve our best alignment with business only if we accept this framework.

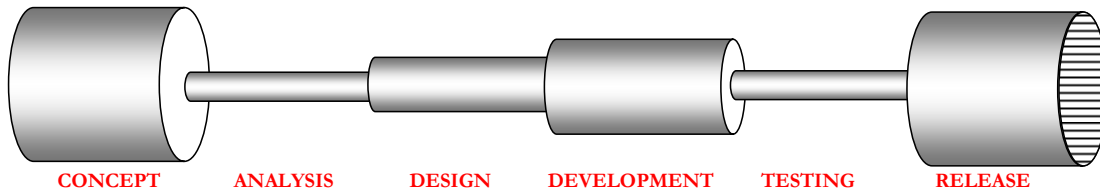
Principle 4 - We allow business to directly influence the content and structure of our SPI roadmaps by examining business issues outlined by the business team, mapping these issues to technological root causes and prioritizing them.

Principle 5 - It is entirely possible that different process areas associated with software development at a given company can be at different maturity levels (e.g. development and testing can be at relatively sophisticated levels, while requirements gathering and management can be at an “embryonic” stage)

SmartPhase Model Description

For the analysis of the *SmartPhase* model we have to return to the pipeline metaphor where the software development process is compared to a pipeline consisting of several pipes of different diameters (see Figure 2). In our model the diameter of the pipe reflects the operational effectiveness (maturity) of the IT/software development process. (Please note, that Figure 2 represents a simplified model of the software development process purely for illustrational purposes).

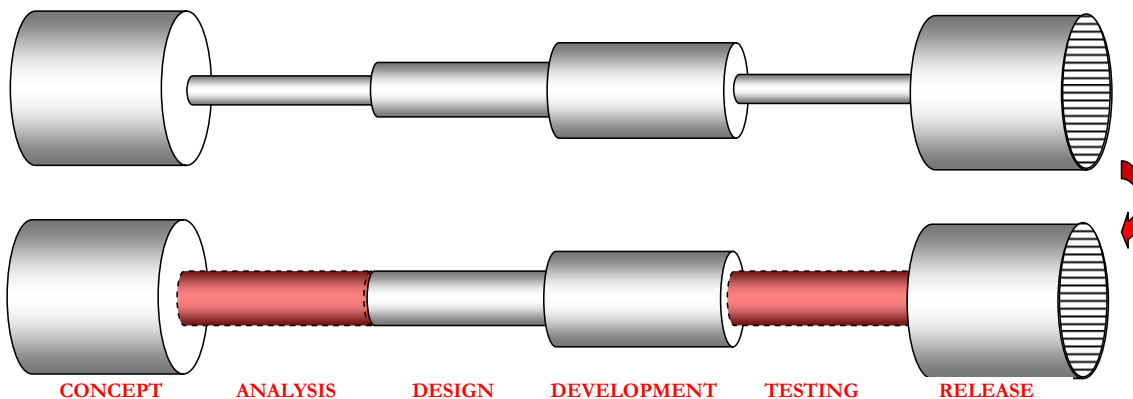
Figure 2



The *SmartPhase* model consists of the following steps:

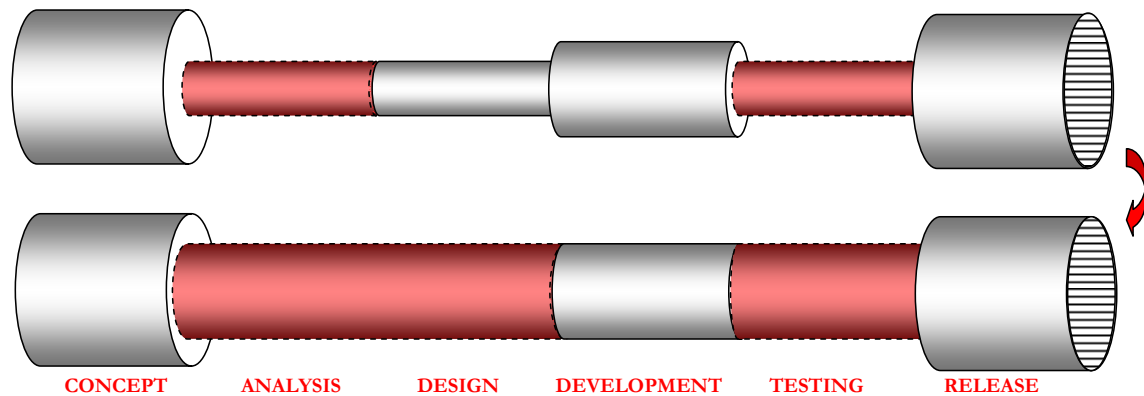
1. Familiarize yourself with company's strategic business goals.
2. Interview company executives, product management, sales & marketing people and, if possible, customers to find out the business issues (e.g. cost overruns, late projects, project overloads, low product quality, etc.) associated with the work of software development or IT department. If possible, rank these issues in the order of importance to the business.
3. Interview representatives from all areas of software development department (i.e. project managers, functional managers, business analysts, developers, testers, architects, etc.) to determine the "technical" root causes of the business problems (for example low product quality can be attributed to poor requirements engineering practices). This step should also include a review of project documentation and sometimes even participation in company projects as an observer.
4. Combine the information obtained from Steps 1, 2 and 3 and the general knowledge of best practices in software development process to analyze, confirm and prioritize the root causes of the business problems in the order of importance (i.e. create a mapping).
5. Discuss your findings with company executives and agree on which areas and how many of them will be targeted in the first phase of the SPI project. Create ROI projections for the initiative. Also agree on whether SPI initiative will apply to the entire IT organization, specific department or a pilot project.
6. Develop an SPI project plan and identify all the relevant metrics to gather **before** and **after** the SPI project. Gather the "before" metrics.
7. Implement the SPI plan by "replacing the narrowest pipes" in the software development pipeline (see Figure 3)

Figure 3



8. Collect the “after” metrics, compare them with the “before” statistics and validate the ROI projections
9. Decide whether there is a necessity to continue with the next phase of SPI plan. In general if the ROI on the initiative has been low or negative, there is a possibility that a company has reached its optimal software development maturity level (see Figure 4)

Figure 4



“Smart Phase” Case Study

Here is an example of how this approach works in real life. A software process improvement consulting firm was approached by the management of a relatively small software development company that was looking for some help with their issues.

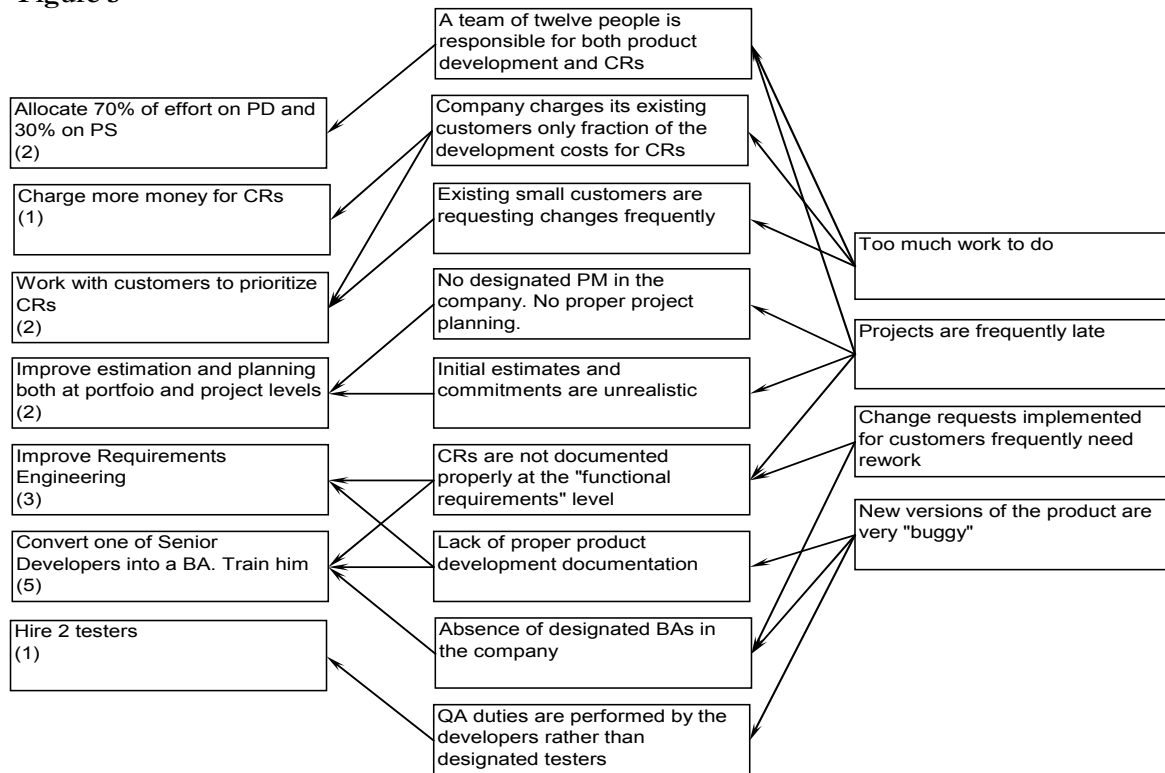
Step 1 – The consultants took some time to familiarize themselves with the company’s strategic goals. Basically it was a small firm developing software for telecommunication companies in Asia. At the time they had approximately ten to fifteen small clients that they sold their product to and were obligated to support these clients to generate cash flows to support the company and pay the bills. At the same time the firm was selected to participate in a tender for a very large contract with one of the larger players in the Asian telecommunications market. The management felt that they had a very good chance of winning this contract providing they can produce a major upgrade to their product.

Step 2 - The issues mentioned by the executives of the software house included: excessive workloads, projects (both from Professional Services and Product Development areas) frequently being late, Change Requests implemented for customers frequently required rework and “buggy” upgrades to the existing software product.

Step 3 – The consultants conducted in-depth interviews of the technical personnel including software development managers, product manager, CTO, developers and testers to try to identify the root causes of the problems mentioned earlier.

Step 4 - A mapping of the problems, root causes and potential solutions was created in order to facilitate the understanding of the entire process (see Figure 5).

Figure 5



Note: Numbers in brackets in the “Solutions” rectangles mean how many of the original problems the recommendation was related to. Therefore the higher the number, the larger the expected positive impact of the recommendation on the business operations.

Step 5 - The results of our findings were presented to the executive team. After a prolonged discussion both sides agreed that while all of the recommendations are valid and useful, the company has enough resources to:

- Designate a Business Analyst and train him
- Develop and adopt a set of Requirements Engineering templates
- Dedicate 70% of the company effort on product development and 30% of the effort on the change request implementation (professional services)
- Work with customers on prioritizing and thus decreasing the number of change requests.

The following activities were deemed necessary, but were postponed to the second phase of the software process improvement program:

- Charging more money for CRs
- Improvements in project planning and estimation
- Hiring of new testers

Step 6 – The following metrics were deemed to be important during the first phase of the software process improvement program:

- Amount of new software (i.e. product development) produced (measured in number of features and/or lines of code per release) – before and after the SPI program

- Average effort (man-hours) per feature – before and after the SPI program
- Number of CRs processed per month – before and after the SPI program
- Average effort (man-hours) per CR – before and after the SPI program
- Number of defects reported by the customers per CR – before and after the SPI program
- Number of defects detected by testers per release – before and after the SPI program
- Average effort (man-hours) per customer-reported defect – before and after the SPI program
- Average effort (man-hours) per tester-reported defect – before and after the SPI program

All the “before” metrics were collected when possible or estimated if no historical data was available.

Step 7 – The SPI plan was implemented over the next three months. The activities undertaken included:

- Designation of a senior developer who performed a business analyst role on a part-time basis as a full-time business analyst.
- Training of the business analyst and the product manager in the areas of Requirements Elicitation, Requirements Management, Use Cases, creation of Vision and Scope, Software Requirements Specifications and Detailed Design documents
- Development, fine-tuning and acceptance of the templates for Vision and Scope, Software Requirements Specifications and Detailed Design documents
- Dissemination of the process to the entire team (several one-day courses)
- Implementation of a basic time-tracking software (timesheets)
- Proper resource allocation inline with the decision to split the company efforts according to a 70/30 formula. A team of seven technical resources was allocated to product development and a team of four resources – to professional services.
- Customer support resource and the business analyst were also trained in the areas of negotiations and change request prioritization.

Step 8 – All the “after metrics were collected in the course of the next year and the return on investment for the entire program was calculated in the following manner:

$$ROI = \frac{CS_F + CS_{CR} + CS_{DC} + CS_{DT}}{TC_{CF} + TC_{INT}}$$

Where

- CS_F - Cost Savings on Features
- CS_{CR} - Cost Savings per Change Request
- CS_{DC} - Cost Savings per Defects found by Customers
- CS_{DT} - Cost Savings per Defects found by Testers
- TC_{CF} - Total Improvement Cost (Consulting Firm)
- TC_{INT} – Total Improvement Cost (Internal)

$$CS_F = (AEF_{OLD} - AEF_{NEW}) \times NOF_{OLD} \times \$/Hour$$

Where:

- CS_F – Cost Savings on Features
- AEF_{OLD} - Average Effort per Feature (Old)
- AEF_{NEW} - Average Effort per Feature (New)
- NOF_{OLD} - Number of Features produced per year (Old)

\$/Hour – Labor Rate per Hour

$$CS_{CR} = (AECR_{OLD} - AECR_{NEW}) \times NOCR_{OLD} \times \$/Hour$$

Where

CS_{CR} - Cost Savings on Change Requests
 $AECR_{OLD}$ - Average Effort per Change Request (Old)
 $AECR_{NEW}$ - Average Effort per Change Request (New)
 $NOCR_{OLD}$ - Number of Change Requests produced per year (Old)
\$/Hour – Labor Rate per Hour

$$CS_{DC} = [TNDC_{OLD} \times AEDC_{OLD} - TNDC_{NEW} \times AEDC_{NEW}] \times \$/Hour$$

Where

CS_{DC} – Cost Savings on Defects found by Customers
 $TNDC_{OLD}$ – Total Number of Defects found by Customers (Old)
 $AEDC_{OLD}$ – Average Effort per Defect found by Customers (Old)
 $TNDC_{NEW}$ – Total Number of Defects found by Customers (New)
 $AEDC_{NEW}$ - Average Effort per Defect found by Customers (New)
\$/Hour – Labor Rate per Hour

$$CS_{DT} = [TNDT_{OLD} \times AEDT_{OLD} - TNDT_{NEW} \times AEDT_{NEW}] \times \$/Hour$$

Where

CS_{DT} – Cost Savings on Defects found by Testers
 $TNDT_{OLD}$ – Total Number of Defects found by Testers (Old)
 $AEDT_{OLD}$ – Average Effort per Defect found by Testers (Old)
 $TNDT_{NEW}$ – Total Number of Defects found by Testers (New)
 $AEDT_{NEW}$ - Average Effort per Defect found by Testers (New)
\$/Hour – Labor Rate per Hour

Since the ROI on this exercise was positive, the management decided to continue with the second round of process improvement by concentrating on project management, additional investment in testing and changing the prices charged for change requests.

Note: The type of historical data to collect and the ROI calculations tend to vary considerably from company to company due to difference in the nature of the business (e.g. IT shop vs. product development company), types of problems (e.g. late projects vs. unhappy customers) and company structure (e.g. traditional vs. projectized).

Conclusion

As was shown above, the real value of this process comes from concentration on the weakest links in your software development process rather than spreading your money and manpower all over the place (“80/20 rule”). This guarantees you that the ROI on process improvement investments is maximized to its full potential. In addition SmartPhase can be characterized by the following attributes:

- *SmartPhase* is completely scalable – it can be applied to an entire organization, select department or even one pilot project

- Phased approach allows the company to stop at any point of time once the management feels that the optimal stage is reached; it allows the company to better zero in on their software development maturity “sweet spot”
- *SmartPhase* is much cheaper than the traditional methods
- *SmartPhase* implementation is much faster than the traditional processes
- Rather than measuring the progress with respect to generic goals, *SmartPhase* allows to align IT processes with the strategic business goals of the company