TSP SYMPOSIUM 2008
HIGH-PERFORMANCE PROCESS

3rd Annual Software Engineering Institute Team Software Process Symposium

September 22-25, 2008 • Hilton Phoenix East/Mesa, Mesa, Arizona

Software Engineering Institute | Carnegie Mellon

www.sei.cmu.edu/tsp/symposium.html

Software Engineering Institute | Carnegie Mellon

Team Software Process Symposium Proceedings

September 22-25, 2008 • Hilton Phoenix East/Mesa, Mesa, Arizona

Program Committee 2

Research Areas

Getting the Functional out of Dysfunctional Teams 3
Lana Cagle and Mark Femal, Naval Oceanographic Office

Results of the Software Process Improvement Efforts of the Early Adopters in NAVAIR 4.0 11
David C.H. Saint-Amand and Bradley Hodgins

PSP in High Maturity Settings 25
Adham Shaher, Dr. Amr Kamel

TOWA's TSP Initiative: The Ambition to Succeed 33
Gerardo López, Yuri Ontibón, and Miguel Serrano

Uses of Monte Carlo Simulation for TSPSM Teams 45
Robert Stoddard, Software Engineering Institute
David Tuma, Tuma Solutions, Inc.
James Van Buren, The Charles Stark Draper Laboratory
David R. Webb, 309th Software Maintenance Group, Hill Air Force Base

Editor's Note

The experiences and ideas presented in these papers are those of the authors, and not necessarily of the SEI. Most individuals and teams customize the TSP and the PSP to best fit their own needs. The processes were designed to be flexible and support customization. As always, use judgment before following advice from others. Take special caution when modifying basic principles of the PSP and the TSP. Examples include the use of personal data for anything besides personal improvement and providing team status.

Program Committee

Robert Cannon
Software Engineering Institute, Visiting Scientist

Anita Carleton
Software Engineering Institute

Tim Chick
Software Engineering Institute

Tom Hilburn
Embry Riddle University

Jodie Nesta
Software Engineering Institute

James Over
Software Engineering Institute

David Ratnaraj
AIS

David Saint Amand
NAVAIR

Rafael Salazar
Tec de Monterrey

Jim Van Buren
STSC

Getting the Functional out of Dysfunctional Teams

Lana Cagle and Mark Femal, Naval Oceanographic Office

1 Introduction

The Team Software Process (TSP) has been used in the Systems Integration Division of the Naval Oceanographic Office (NAVOCEANO) since the year 2000. Its methods of analysis, techniques for reporting, and tracking measures have been employed with varying degrees of success on NAVOCEANO projects. The defined roles, tools for measuring against a plan, scripts to guide meetings, and status reporting mechanisms ensure teams can improve quality and measure results effectively. However, in the case of the Data Services Project (DSP), team interaction was often perceived to be painful, stressful, and unrewarding. This breakdown of team cohesiveness stems from not synthesizing team management best practices with the structure and framework that the TSP provides.

This paper focuses on identifying and inhibiting the dysfunctions often found in teams with an emphasis on the learning received from the DSP. Constructive exercises for improving those negative characteristics were planned and implemented. These changes were then measured using TSP benchmarks and team surveys. Although team building should always be considered a continuous activity, our interim results are shared and discussed in this paper.

The first section of this paper describes the relevant background material needed to understand the project. The second section quantifies the negative characteristics found on the team. The third section describes what steps were taken to counteract those negative behaviors and some of the metrics used to gauge their effectiveness. Finally, we present a summary of our findings and discuss the future changes planned for the division.

1.1 Background

The DSP ensued as a result of coordination meetings among agencies in the United States Government on Joint Meteorological and Oceanographic Standards for data exchange [Washburn and Morris 2005]. Initial planning for these standards started in 1995. Much negotiation occurred between subject matter experts, data modelers, analysts, program managers, and application developers in defining the constructs and semantics of how data might be stored and used. The development and implementation efforts using these standards began in 2003, with several formal TSP project launches starting in 2005.

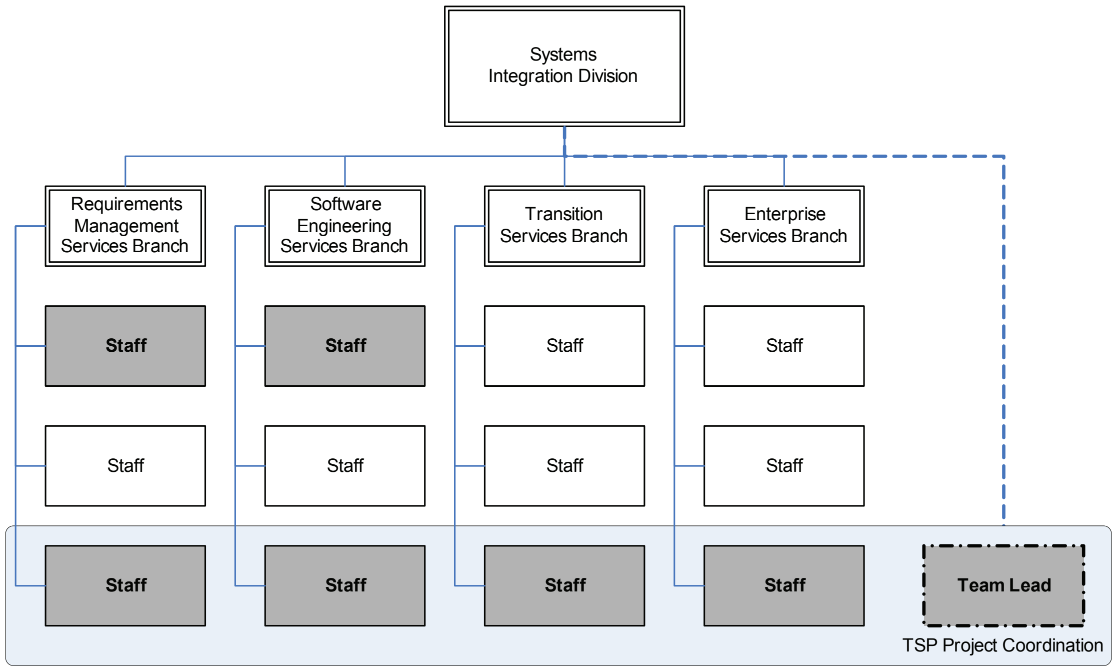
It is important to recognize that several fundamental technologies employed on the project were completely new to the division. These technologies include Service-Oriented Architecture, Web Services Technologies, Web Application Clustering, Distributed Processing, and Multi-Platform Development. A number of additional toolkits and techniques were employed that exceeded the current knowledge and experience base of team members. Many of the previous software development efforts in the division were focused on stand-alone PCs. In addition, adequate training and preparation were not employed for all team members.

1.2 Team Composition

The initial project team consisted of about six software engineers (five contractors and one government software employee) as well as the project manager. This effort was driven largely from one branch in the division with approximately fifteen subject matter experts pulled in from across the office (with very diverse backgrounds and no PSP or TSP training). As the project progressed and matured, additional resources were added from across the division. The project scope and areas of responsibility continued to expand until the project was separated into two distinct projects based on competing goals and resources.

One newly formed project team focused on machine-to-machine integration and the other on human-to-machine interaction. This separation reduced the administrative overhead of one large team (nearly thirty members) and made the skills needed to accomplish the development easier to manage across the separate projects. For the purposes of measurement, this paper focuses on the implementation of the machine-to-machine requests with an emphasis on the last two years of effort. However, many of the negative behaviors discussed in the next section have persisted since project inception.

Figure 1: Systems Integration Division Structure and TSP Project Organization.


Systems Integration Division
Requirements Management Services Branch
Software Engineering Services Branch
Transition Services Branch
Enterprise Services Branch
Staff
Team Lead
TSP Project Coordination

2 The Basis for Negative Team Behaviors

Throughout the history of the project, unproductive team behaviors have persisted. Exacerbating the lack of teamwork, the quantity and pace of change is high. Early in the timeline, considerable interpersonal friction occurred due to immature components created for the prototype. In addition, the algorithm scope and complexity was not articulated well in concise documentation, and many assumptions were made. There was also a propensity not to involve team members in activities where organization structure and job responsibility dictated. As the project team progressed and the constituents changed (leadership and member makeup), the prevailing negative behaviors continued.

The organization structure of the Systems Integration Division contributed to the dysfunctions of the DSP. The division is divided into branches based on responsibility and similar technical backgrounds. This arrangement provided unique challenges to the DSP. A typical division project team consists largely of two groups. One group performs requirements engineering and software testing (Oceanographers and Meteorologists). The other group contains software developers that do design and implementation (Computer Scientists and Mathematicians). The project manager comes from any branch and is accountable for goals, schedule, and budget. The branch heads are responsible for the entire set of projects. Conflicting goals (i.e., priorities and direction) often ensue between the project manager and the branch heads as well as between branch heads themselves. As seen in Figure 1, the technical staff reports administratively (e.g., performance reviews) to their respective branch heads. Thus, real unity on project teams is challenging, and shared responsibility hinders timely decisions.

The problems associated with this organization structure are corroborated in two assessments. A Software Risk Evaluation (SRE) done by Software Engineering Institute (SEI) in 1993 revealed a significant finding that perceived boundaries between the branches were high. In addition, an employee survey in August 2006 showed teamwork and cooperation in the division scored low (2.2 on a scale of 1 to 5). Specific comments cited from this survey include:

- Branches do not communicate or respect each other.
- Management does not seem to care about the rift between the branches.
- People throughout the Systems Integration Division do not cooperate effectively with one another.
- Branches are competitive and hostile towards each other.

As discussed in the previous section, the DSP was initiated primarily in one branch. Initial development was carried out by contractors that were not TSP and PSP trained. Long-term planning on how to fully transition this software into the division was not done, which resulted in confusion and resentment as the transition occurred. Software coding was often done before requirements were documented. System functionality had to be inferred from existing Joint Standards that were at times ambiguous and not well understood. Low team morale and performance problems (Table 1) were seen when the project was brought under TSP and expanded to include representation from other branches.

Table 1: Performance data for DSP early releases prior to Team Building.

Mo/Yr Completion	Version	% Cost Error	% Schedule Error	% Features	ST Defect Density
Jul-07	1.1	201	239	100	2.5
Jan-08	1.2	14	75	91	0.12

In these first few iterations, there were both performance and process problems. The postmortems for both releases resulted in 48 Process Improvement Proposals (PIPs) from all areas of the development lifecycle: planning, requirements, design, implementation, testing, and support.

The coach and others observed many unproductive team member behaviors during launches, team meetings, postmortem meetings, and at the water cooler as shown in Table 2. The DSP team members avoided conflict, and the organization team structure prohibited direct reporting relationships. Thus, no one challenged these unproductive behaviors. An environment of artificial harmony was perpetuated both internally within the project team and externally to management.

Based on these unproductive behaviors, a series of team-building exercises was planned in conjunction with the functional deliverables scheduled for the next software release. Within the project launch, it was agreed that spending several hours each month working on team building would not detract from delivering required functionality in the timeframe that features were needed. The team-building knowledge base had to be built, and Patrick Lencioni's book, *The Five Dysfunctions of a Team* was used as a starting point to highlight unproductive behaviors. This book is discussed briefly in the next section.

Table 2: Observed Unproductive Team Member Behaviors prior to Team Building.

Behavior	Resultant Effect
Lack of Participation	• Not volunteering for role manager positions during launches. • Not recording defect data. • Not meeting commitments with no ill personnel effects. • Not volunteering to record meeting minutes.
Negative Body Language	• Sitting separately from the team. • Eye-rolling and background whispering.
Bad Attitudes	• Not properly recording process data • Why do we have to fill in the blank? *...do this quality manager role, review this code, etc.*
Avoiding Conflict and Productive Discourse	• Not challenging each other's technical approach. • Culture of *them* versus *us* at the branch level. • Wanting to throw requirements or software "over the fence" (lack of reviews, documentation, etc.).
Disrespectful Comments	• Talking about a person or branch rather than directly to them in addressing conflicts. • Stereotyping and ridicule.

3 Team-Building Knowledge Foundation

There are numerous resources available on team building. After acquiring team building training that included reading *The Five Dysfunctions of a Team*, by Patrick Lencioni, recognizing some underlying issues became easier, and each source offered many possible answers. According to Lencioni, successful teams have the following attributes:

1. Trust
2. No fear of conflict
3. Commitment
4. Accountability
5. Attention to results

In addition to these attributes, during instructor-led training [PH Associates 2008] these team characteristics were mapped to the four stages of team development [Tuckman 1965] and are depicted in Figure 2. As can be noted, the various phases of team development align well with the dysfunctions. In other words, there is an expected path of progression for work groups as they proceed through the team life cycle and, in turn, through addressing each of the dysfunctions. The TSP helps achieve many of the aforementioned team attributes, and a discussion of each now follows.

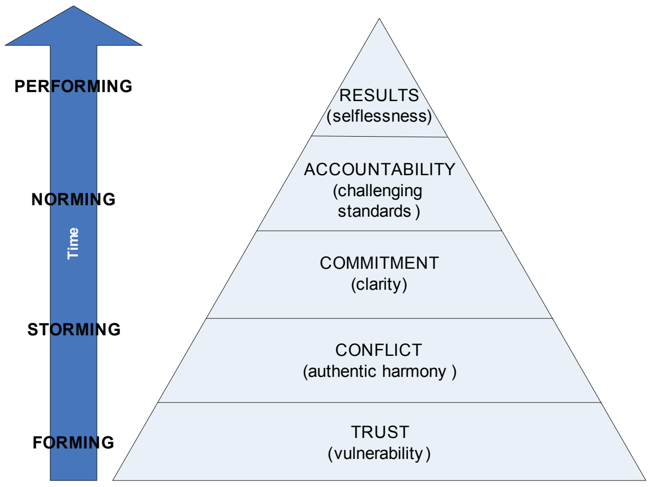


Figure 2: Four Stages of Team Development coupled with the Five Dysfunctions of a Team.

3.1 Trust

Openness and trust are desired outcomes of the TSP launch or the team-forming stage. To achieve this, team members must openly admit their strengths and weaknesses and make it acceptable for others to provide feedback. In order to accept criticism from others, a certain level of trust must be present. Building trust may mean making yourself vulnerable with the confidence it will not be used against you. These vulnerabilities can include skill deficiencies, mistakes, failures, and requests for help. This level of trust is achieved if other team members have good intentions. A lack of trust inhibits individuals from opening up to one another. Lencioni is not talking about developing deep, personal friendships, but establishing enough trust so that all input is valued and decisions are a collective, productive effort. A lack of trust coincides with a lack of cooperation. This results in poorly developed work and little to no satisfaction towards the finished product.

Groups lacking trust are not willing to challenge one another. Groups that avoid debate are not nearly as effective as those that embrace it. Debate, when carried out properly, is a very efficient means of combining many different ideas into one structurally sound, centralized idea. Teams have to feel free to challenge one another. If a member is not in agreement about an issue and puts forth no effort to explain why, the issue remains unresolved. To resolve an issue, all sides of the argument need to be presented and considered in order to investigate options and select the best possible outcome. The absence of trust is related to the fear of conflict.

3.2 Conflict

Conflict is inevitable in relationships, and healthy conflict enables you to produce the best possible solution in the shortest amount of time. Conflict is often seen when one or more individuals are not getting what they want and are motivated by individual, rather than team goals. *Storming* is normal, but staying stuck in conflict will prevent the team from performing at its highest level. The DSP clearly had the preference of avoiding conflict or being indirect about confronting issues. The first step in addressing this is to acknowledge that conflict can be productive and should not be avoided. Issues have to be confronted and done so in a collaborative manner (i.e., behavior that is both highly assertive and cooperative).

3.3 Commitment

Commitment is all about buy-in and ensuring that all team members are rowing the boat in the same direction. Lencioni showed very well that consensus is not needed to get buy-in; most team members just want to know their input is understood and considered. Basically, individuals feel disenfranchised if they do not feel their voice is heard. If an idea is presented, it is beneficial for team members to respectfully point out their differences; otherwise, everyone thinks there is a de facto resolution, and there is no need to explore other options.

The TSP scripts are effective in enabling commitment. At the end of the launch, each individual has made a commitment to the team with regard to goals, scheduling, and project expectations. Status of personal commitments is reviewed in team meetings. It is publicly clarified what needs to be achieved and who needs to deliver what. The weekly meetings enable this information to be kept in the open so that it can not be ignored.

3.4 Accountability

Avoidance of accountability is the fourth dysfunction and ultimately leads to resentment among team members when everyone is not held to the same performance standards. Willingness of team members to point out, in a respectful manner, unproductive behaviors in a public forum aids accountability. Many team members will feel uncomfortable doing this. However, the data tracked via the TSP enables us to quickly assess if we are meeting commitments using the standards of performance set forth during the launch.

If a team does not hold its members accountable, the members are more likely to turn their attention to their own needs versus the collective needs of the team. Performing teams have a strong attention to results with a solid commitment to accomplishing the goal. There is acceptance by the team of its challenges, its strengths/weaknesses, and its assumptions. Team members appreciate the diversity among its members, and ultimately, the end result is a high level of achievement.

3.5 Results

The culmination of addressing all the dysfunctions is improved results. Team members must accept the greater good rather than personal agendas. By addressing the previous dysfunctions, the team leader can make improvements based upon the TSP measured results. As many other factors are involved in a project's results, isolating and measuring these team characteristics can be challenging. However, as discussed in the next section, mechanisms can be employed on a project to help address these common team attributes.

4 Team-Building Activities

Since some of the problems related to the way this project was initiated are attributed to organizational issues, addressing team work at the project level alone is not enough. A separate initiative is needed to address fundamental team issues at the management level.

4.1 Project Level

One of the first tasks performed was to conduct a team assessment from *The Five Dysfunctions of a Team*. Not surprisingly, the assessment indicated that all five dysfunctions were or could be a problem. The data for this assessment is shown in Table 3.

- A score of 3 to 5 is probably an indication that the dysfunction needs to be addressed.
- A score of 6 or 7 indicates that the dysfunction could be a problem.
- A score of 8 or 9 is a probable indication that the dysfunction is not a problem for your team.

Table 3: Team Assessment Results

Dysfunction	Score	Interpretation
Absence of Trust	5	Needs to be addressed
Fear of Conflict	5	Needs to be addressed
Lack of Commitment	6	
Avoidance of Accountability	4	Needs to be addressed
Inattention to Results	6	

As can be seen in Table 3, avoidance of accountability scored the lowest followed closely by absence of trust and fear of conflict; therefore, the initial plan was to address these three dysfunctions first. We rationalized the TSP could address some aspects related to commitment and attention to results.

To ensure the DSP would commit to team development, team-building tasks were added to the project plan during the launch. All team members were given a copy of Lencioni's book to read for later discussion. Overall feedback from this exercise was positive, and it helped promote productive conflict resolution. It also helped the team realize the importance of providing and receiving feedback. With everyone committing to this work during the launch, no scheduling conflicts or resistance was encountered.

Next, the team established ground rules to operate by and to hopefully, guide productive confrontation. These ground rules included:

- No interruptions (one person speaks at a time)
- Respect team members and their contributions to the team
- Be timely, prepared, and ready to participate at meetings
- Handle disagreement with tact
- Share information; do not intentionally withhold information
- Each team member holds each other jointly accountable for inappropriate behavior
- Be helpful to other team members (especially new members)
- Let team members know when you are behind
- Try to resolve issues within the team prior to elevating it to management
- Assume good intentions
- Seek valid information
- Manage yourself

A suggested exercise in Lencioni's book called for a *miner of conflict* (MOC), whose role is to stay with the conflict until it is resolved. For instance, an attendance issue arose at the launch. Several team members discussed it among themselves rather than in the group. Becoming aware of the disagreement, the launch coach took on the role as the MOC to bring the situation out in the open, so that all viewpoints could be heard. The issue was resolved quickly without undermining trust and wasting valuable time and emotional energy.

Another independent activity performed by the group was to utilize the Thomas-Kilmann Conflict Mode Instrument [TKI 1974]. This tool uses the responses gathered in a personal assessment to map behavior into one of five conflict management styles. Responses are characterized according to whether they are unassertive to assertive and uncooperative to cooperative. Totals are then calculated to indicate whether a person has a predisposition in a conflict to be competing, avoiding, compromising, collaborating, or accommodating. Once someone understands the different styles, suggested methods and scenarios for each style can then be analyzed and adapted in different scenarios (i.e., weighing time limits and other tradeoffs). Feedback from this exercise was positive; however, results were not shared publicly within the group.

The final activity planned within the DSP included an analysis of the Johari Window [Luft and Ingham 1955]. This activity is based on a cognitive psychological tool created by Joseph Luft and Harry Ingham in 1955 in the United States. This was used to promote self disclosure on the team by making some information known only to self known to others. This is an exercise meant to establish more trust in team dynamics and to make people understand there are blind spots (not known to self, but known to others) based on feedback from others on personality traits.

4.2 Management Level

Managers have also read Lencioni's book, and a workshop is planned to establish performance standards for the division. The team development work that is underway for this group is outside the scope of this paper and is its own separate subject and focus.

5 Performance Results

Measuring the effectiveness of team building is challenging. An easy empirical measure is the observed interaction between team members. Do members of the team function together more cohesively? Are those factors that led to negative behaviors still occurring? Is productive debate occurring within the team? In addition to these observed team traits, one can measure whether members meet their TSP commitments (quality, effort, timeliness, etc.). Since there is no single measurement that provides a direct correlation between team-building efforts and its effectiveness, assessments and surveys also have an important role.

The latest DSP data are presented in Table 4. Key areas may be contrasted with values from Table 1 in Section 2. The DSP team is predicted to deliver seven weeks late. However, this delay is primarily due to system stability problems that are causing testing delays. In addition, training a new resource to manage application configurations and installation is occurring. To date, the *% Features* is expected to rise due to additions scheduled by the Configuration Control Board. Despite these additions, software coding has been on schedule. The *ST Defect Density* cannot be calculated until system testing is completed.

Table 4: Latest Performance Data for the DSP.

Mo/Yr Completion	Version	% Cost Error	% Schedule Error (predicted)	% Features	ST Defect Density
To Date Jul-08	1.3	13	29	116 ↑	?

Some of the initially observed negative team behaviors have also changed. For instance, the team recently expressed discontent on the way status meetings were conducted. The fact that the team brought this up publicly should be viewed in a positive manner. In the past, issues such as this would have been voiced outside of the meeting. Based on the public discussion, the team lead made some changes, and a follow-up evaluation was done. After these changes were made, the next meeting rated a (7) on a scale of one (1)-Negative/Ineffective to nine (9)-Positive/Productive. Such public discussion facilitates everyone's belief that their opinion matters and that they can enact positive change.

A team effectiveness questionnaire was also given during Week 16 of the project to assess the team-building effort. The meeting evaluation form and the team effectiveness questionnaire came from the team-building training provided by PH Associates. Scores on this questionnaire were ranked by each team member from 1 (deficient) to 7 (exceptional). Final results showed the DSP Team scored more favorably than the division on 11 of the 15 questions posed to the team (total responses in the range 5 to 7).

A summary of the strengths and weaknesses follows:

Strengths:

- The team communicates easily and frequently.
- Each member understands each other's roles and skills.
- Team members understand the team's purpose and express it in the same way.
- All three of the following skill categories are either actually or potentially represented across the team's membership:
  - functional/technical
  - problem-solving/decision making
  - interpersonal
- Our work approach is concrete, clear, and agreed upon by everybody on the team.
- Our team develops and rewards collaboration.
- Planning is a critical element in our team process, and all members participate in it.

Weaknesses:

- Members are willing to spend the time to help themselves and others learn and develop functional/technical, problem-solving/decision making and interpersonal skills.
- If interpersonal issues arise, we confront, not overlook them.
- We handle conflicts constructively.
- We consider conflicts normal and deal with them openly and immediately.
- Our team's trust level is high.
- Team members are individually and collectively accountable for the team's results.

The survey was inconclusive regarding the following two statements:

- The team's purpose is broader and deeper than just near-term goals.
- Members provide feedback to each other consistently and effectively.

Although defect data recording is better than previous releases, improvement is still needed. The quality manager is more active this release in reviewing defect data (rather than waiting until post-mortem). Some negative behaviors and body language still persist. However, many of the disrespectful comments have stopped, and the team has done well to adhere to the ground rules listed in Section 4. In the past, some team members were not as productive in meeting their commitments and managing their task hours. In general, everyone is achieving their scheduled task hours and meeting their commitments.

6 Summary and Future Work

Additional work is being done within our division to improve the standards needed to attain certain positions within the TSP Team. For instance, job qualification requirements (JQR) will feed into performance reviews and training requirements. Positions will be classified in one of three categories: Team Lead, Role Manager, and Team Member. Each of these roles will have prerequisite training objectives that will address team management skills and behaviors. Additional one or two day focus workshops are also planned to provide additional emphasis on the topic.

Teamwork development can stop at any stage in the software life cycle and can even regress. Continual work is required to develop groups into *norming* and *performing* teams. The TSP is an enabling framework to support commitment to goals, accountability to performance standards, and attention to results. However, fundamental team dysfunctions such as lack of trust weaken a group's ability to achieve productive results. Self awareness and team training may be needed to improve results.

Team management problems may be complex (i.e., political, technical, etc.), and there is not one solution for all teams. However, realizing team dysfunctions exist is the first step towards implementing a plan for improvement. Teams exist to address complex problems and tap into a wider knowledge and experience base. Unfortunately, much like choosing your relatives, you cannot handpick your coworkers. Group members are mostly inherited, and there may be someone with whom you would rather not sit down to dinner.

References

[Lencioni 2002]
Lencioni, Patrick
The Five Dysfunctions of a Team: a leadership fable.
San Francisco, CA: Jossey-Bass, 2002.

[Luft and Ingham 1955]
Luft, J. and Ingham, H.
The Johari Window, a Graphic Model of Interpersonal Awareness. Proceedings of the Western Training Laboratory in Group Development, Los Angeles, CA: UCLA 1955.

[PH Associates 2008]
Sampson, Chuck of PH Associates,
Meridian, MS 39393
Conflict Management and Team Building Training.

[TKI 1974]
Thomas-Kilmann Conflict MODE Instrument (TKI)
Mountain View, CA: Xicom and CPP, Inc., 1974.

[Tuckman 1965]
Tuckman, Bruce
Developmental Sequence in Small Groups.
Psychological Bulletin (1965), 63, pp. 384-399.

[Washburn and Morris 2005]
Washburn, Peter and Morris, Terry
NAVOCEANO Web Services, CHIPS Magazine,
January 2005.

Authors

Lana Cagle
Lana Cagle is a Physical Scientist with the Systems Integration Division of the Naval Oceanographic Office, Stennis Space Center, Mississippi. She is currently team lead of the Process Improvement Program, and is an authorized Personal Software Process Instructor and a Team Software Process Launch Coach. Ms. Cagle is also a certified Lean Six Sigma Green Belt.

Lana Cagle
Naval Oceanographic Office
Code N64Q
Stennis Space Center, MS 39522-5001
(228) 688-4743
lana.cagle@navy.mil

Mark Femal
Mark Femal is a Computer Scientist with the Systems Integration Division of the Naval Oceanographic Office, Stennis Space Center, Mississippi. He received his M.S. in Computer Science from North Carolina State University and B.S. from the University of Wisconsin at Oshkosh. He currently serves as a division technical liaison for web services and SOA.

Mark Femal
Naval Oceanographic Office
Code N642
Stennis Space Center, MS 39522-5001
(228) 688-4451
mark.femal@navy.mil

Results of the Software Process Improvement Efforts of the Early Adopters in NAVAIR 4.0

David C.H. Saint-Amand and Bradley Hodgins, NAVAIR

1.1 Introduction

This paper is an overview of the Software Process Improvement (SPI) efforts of Naval Air Systems Command's (NAVAIR) Air 4.0, Research and Engineering Competency. It is an extract from NAVAIR Technical Paper 8642 (Saint-Amand and Hodgins, 2007) which has been approved for public release; distribution unlimited. This version has been approved for public release under NAVAIR Public Release number 08-0132.

NAVAIR 4.0 provides life-cycle systems development along with operations and maintenance support for the aircraft and weapons of the U.S. Navy and Marine Corps. NAVAIR 4.0 is distributed across the U.S.: Patuxent River, Maryland; Lakehurst, New Jersey; Orlando, Florida; and China Lake and Point Mugu, California (Figure 1). While there are NAVAIR facilities located in Italy and Japan, this report will focus on the 24 discrete software engineering teams located within the U.S., and specifically on the SPI efforts of the six teams that were early SPI adopters.

The early SPI adopters mentioned above were the software development teams within the AV-8B Joint System Support Activity (JSSA); the E-2C, EA-6B, P-3C, and Tactical Aircraft Electronic Warfare (TACAIR EW) Software Support Activities (SSAs); and the F/A-18 Software Development Task Team (SWDTT). These teams ranged in size from less than 10 to more than 70 NAVAIR software engineers and support contractors. They were organized under the Director of the Engineering Division of the Research and Engineering Group, Code 4.0, of NAVAIR Weapons Division (WD).

1.2 Background

Over the last several decades NAVAIR, the parent organization of Code 4.0, Research and Engineering Group, has experienced tightening budgets, decreasing labor pools (Figure 2), increasing software complexity (Figure 3), and, finally, the demands of the Global War on Terrorism (GWT). Throughout this period, NAVAIR has worked to meet the challenge of accomplishing its mission while procuring the new aircraft necessary to meet its future obligations to the Fleet.

To meet this challenge, process improvement efforts were initiated throughout NAVAIR, in all business areas, including administration, contracting, support, and software development. These initiatives began to take shape for Code 4.0 in 1998 when AV-8B joined F/A-18 in the pursuit of process improvement. Between April and September 2002, NAVAIR issued a set of five formal instructions as guidance on process improvement for software acquisition, development, and life cycle maintenance. One of these instructions, NAVAIRINST 5234.2, was based in part on Code 4.0's research into process improvement tools and techniques (Wall 2005, page 9).

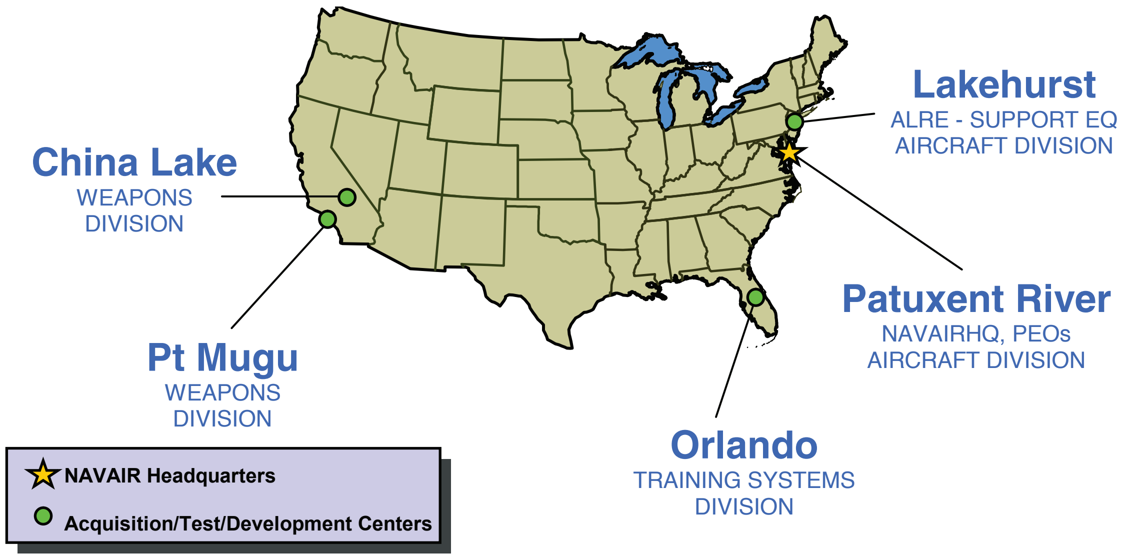


Figure 1. The Locations of NAVAIR Facilities Within the U.S

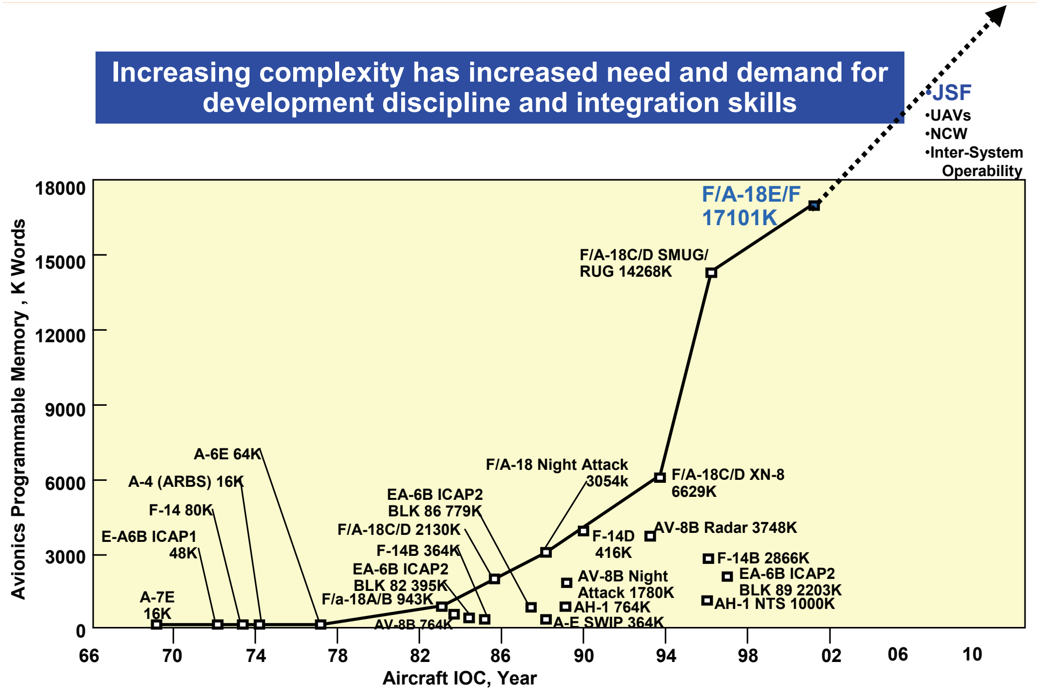


FIGURE 2. Manning History and Forecast.

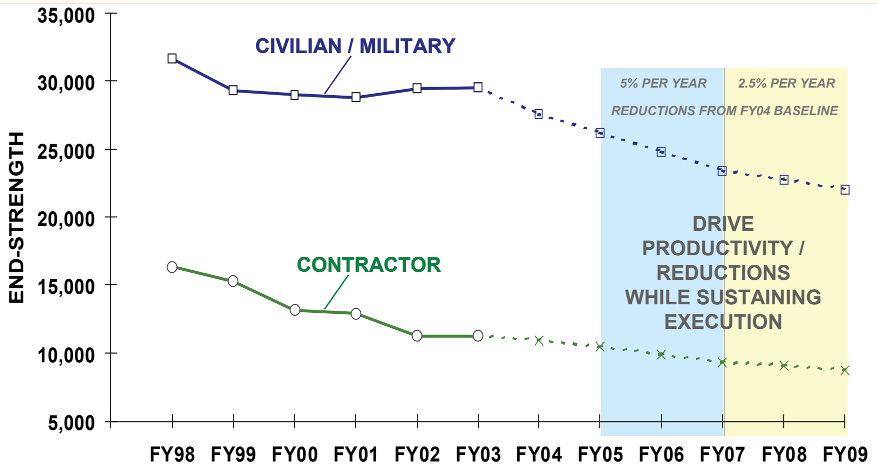


FIGURE 3. Aviation Software Complexity.

In December 2002 NAVAIR 4.0's voluntary effort was bolstered by the passage of the U.S. Federal Government statute, Public Law 107-314, the Bob Stump National Defense Authorization Act for Fiscal Year 2003. Section 804 specifies that software acquisition programs must meet the following requirements:

- Shall have a documented process for planning, requirements development and management, project management and oversight, and risk management.
- Shall have a metrics for performance measurement and continual process improvement.
- Shall have a process to ensure adherence to established process, and requirements related to software acquisition.

In this environment, the goals of the SSAs were to improve the maturity of the software development processes, to realize cost savings, and to deliver higher quality products to the Fleet—in essence, to meet the challenges of their missions while at the same time meeting NAVAIR's stated organizational goals, as stated in 2005 (See sidebar).

1.3 The Software Process Improvement Tool Set

The initial SPI efforts of the individual SSAs were not coordinated across NAVAIR 4.0. Each SSA acted as an independent entity within the overall effort, starting at different times and selecting SPI tool sets specific to the needs of their individual organizations (Table 1).

TABLE 1. The SPI Tool Sets of the NAVAIR Early Adopters.

Organization	Process improvement tools					
	CMM	CMMI	TSP	EVMS	HPO	TSPm
AV-8B	✓		✓	✓	✓	
E-2	✓		✓			✓
EA-6B	✓	✓			✓	
P-3C	✓	✓	✓		✓	✓
TACAIR EW	✓	✓			✓	
F/A-18 SWDTT	✓	✓	✓	✓	✓	

1.3.1 Capability Maturity Model (CMM)

A Carnegie Mellon University, Software Engineering Institute (SEI) framework for incremental process improvement, CMM consists of best practices that cover the complete product life cycle, starting with defining requirements and continuing on through maintenance of the delivered products. The CMM is broken into five levels of maturity, each with a discrete set of practices that characterize an organization operating at that level. It is a useful tool for appraising organizational maturity and for guiding incremental process improvement efforts. CMM is sometimes used interchangeably with Software-CMM (SW-CMM).

1.3.2 Capability Maturity Model Integration (CMMI)

This is SEI's follow-on model, which replaces several process improvement tools, including the CMM. CMM is the baseline for the CMMI, but expanded to include System Engineering (SE) and generalized to accommodate a wider variety of business models. It encourages organizations to focus process improvement efforts based on one or more specific areas of their business, instead of requiring one all-encompassing process improvement effort. In this way, organizations may pursue process improvement in only those areas they deem most urgently in need of process improvement.

1.3.3 Earned Value Management System (EVMS)

This is a tool for program management that allows visibility into both technical issues and cost and schedule progress on projects and contracts. Analysis using EVMS can provide an early warning for issues with a project or contract, from as early as the point at which 15% of that effort has been completed. In order to use EVMS, program management must ensure that their effort is fully defined from the beginning, including a bottom-up plan. In this plan, each discrete task will have an associated value that corresponds to a percentage of the total work effort. This will allow measurement of the bottom-up plan for the entire period of performance. The data collected provide a way to show actual performance improvement, and become the basis of modeling predictable performance for future projects. EVMS is one of the CMMI's best practices for project planning.

1.3.4 High Performance Organization (HPO)

HPO is an organizational improvement plan that is guided by the Diagnostic/Change Model for Building High-Performance Organizations as developed by the Common-wealth Centers for High Performance Organizations (CCHPO). It is commonly referred to as the Diagnostic/Change Model. The work is typically initiated with a workshop ("Teamway") designed to help a group develop the skills required to improve their performance by continually using the Diagnostic/Change Model. Conducted with intact work groups, the 3- to 5-day workshop is tailored to each group's needs, concerns, and issues.

1.3.5 Personal Software Process (PSP)

The PSP is a SEI methodology for developing high quality software. It is based on the practices that are characteristic of CMM and CMMI Maturity Level 5 organizations. It uses standards, methods, scripts, measures, and forms to provide a highly structured and disciplined framework for individual software developers to use in their daily software development efforts. The measurements collected are used to improve the quality of the products developed.

1.3.6 Team Software Process (TSP)

The TSP is based on the concepts and practices of the PSP and is the methodology through which PSP may be applied to team-based, software development efforts. All software engineers participating in a TSP team are required to be PSP trained. TSP and CMMI are complementary, and they work best when introduced into an organization at the same time (Wall 2005, page 5). The data collected from these teams are used to set performance and quality objectives for the organization.

1.3.7 Team Software Process for Multiple Teams (TSPm)

TSPm is an SEI prototype methodology derived from the TSP that is intended to facilitate the application of TSP principles in situations where there are multiple TSP teams engaged in developing sub-units of software for a common product.

1.4 The SPI Journeys of the Early Adopters

1.4.1 AV-8B Joint System/Software Support Activity

The AV-8B JSSA started its SPI process in 1998. At that time, AV-8B's active project ("Project A"), was estimated to be 9 months behind schedule (a 17.6% schedule overrun) and $49 million over budget (a 28.3% cost overrun). Determined to address the root causes of this disappointing performance, AV-8B created an independent review team to inspect the Project A software development effort. The team completed the review and recommended that AV-8B pursue process improvement. To focus that pursuit, AV-8B determined that their top-level SPI goals would be to implement EVMS for tracking project cost and schedule, and to implement HPO concepts in order to develop a more mature software development process. The plan also called for obtaining a Department of Defense (DOD) EVMS certification.

AV-8B's progress was swift. EVMS training began in October 1998 and AV-8B's Software Engineering Process Group (SEPG^{SM}) was formed in March 2000. This was followed by TSP training in October 2000 and HPO training in January 2001. By May 2001 AV-8B had been assessed at CMM Level 2 and by October 2001 they received their DOD EVMS certification, the second organization in the Federal Government to be certified. In September 2002, AV-8B commissioned a SW-CMM CMM-Based Appraisal for Internal Process Improvement (CBA IPI). The assessment concluded that AV-8B had achieved SEI CMM Level 4. Given a start date of March 2000 for CMM, AV-8B managed to reach CMM Level 4 in 2½ years. (The SEI average for progressing from CMM Level 1 to Level 4 is 5½ years (Wall 2007, pages 3-4).)

The AV-8B team attributed their rapid advancement through the CMM levels to the use of the TSP and a culture of process improvement. TSP provides a quickly implemented, flexible process framework. The guidance contained in the SEI Technical Report, Relating the Team Software Process to the SW-CMM (Davis 2002), helped the AV-8B SEPG focus and prioritize its efforts. The technical report proved a valuable tool in applying process improvement techniques in the most efficient manner, shortening what might otherwise have been a long learning curve. Furthermore, TSP distributed the process improvement responsibilities across the project teams, so that process changes originated from within the development teams. This increased the entire AV-8B team's commitment to those changes. The AV-8B JSSA's Leader, Dwayne Heinsma, added "The recipe for accelerating AV-8B's climb up the software maturity ladder centered around identifying champions and using process discipline as an enabler. These champions included a Personal Software Process/ Team Software Process champion leading the software team; an organizational process champion leading the development and the institutionalization of organizational standards; senior managers championing the overall effort and removing roadblocks (establishing both TSP and an Earned Value Management as the standard way of doing business at the JSSA); and, most importantly, an excellent team of software engineers, systems engineers, and product integrity support members that made it all happen."

TSP and EVMS improved AV-8B's cost and schedule performance as well. Once EVMS was put into use, schedule and cost variances were brought down to within 10%; the introduction of TSP brought them even lower (Table 2).

AV-8B's excellent record in cost and schedule estimation continues to this day. Figure 4 is an Earned Value Chart for AV-8B's Project F Mission Systems Computer (MSC) Project Software Development Team. The chart was generated by the same TSP tool that the team uses to enter and track their project plan and performance data. As can be seen in Figure 4, after a short delay at the start, the project is now on track with their original plan.

TABLE 2. AV-8B Schedule and Cost Variances Related to EVMS and TSP.

Project	Date	Schedule variance	Cost variance	Used EVMS?	Used TSP?
Project A	At 7/98	17.6% overrun	28.3% overrun	No	No
Project B	Complete 4/02	50.0% overrun	300.0% overrun	No	No
Project C	Complete 5/04	5.0% overrun	8.1% overrun	Yes	No
Project D	As of 7/04	0.5% overrun	1.5% overrun	Yes	Yes
Project E	As of 5/04	1.1% overrun	6.9% overrun	Yes	Yes

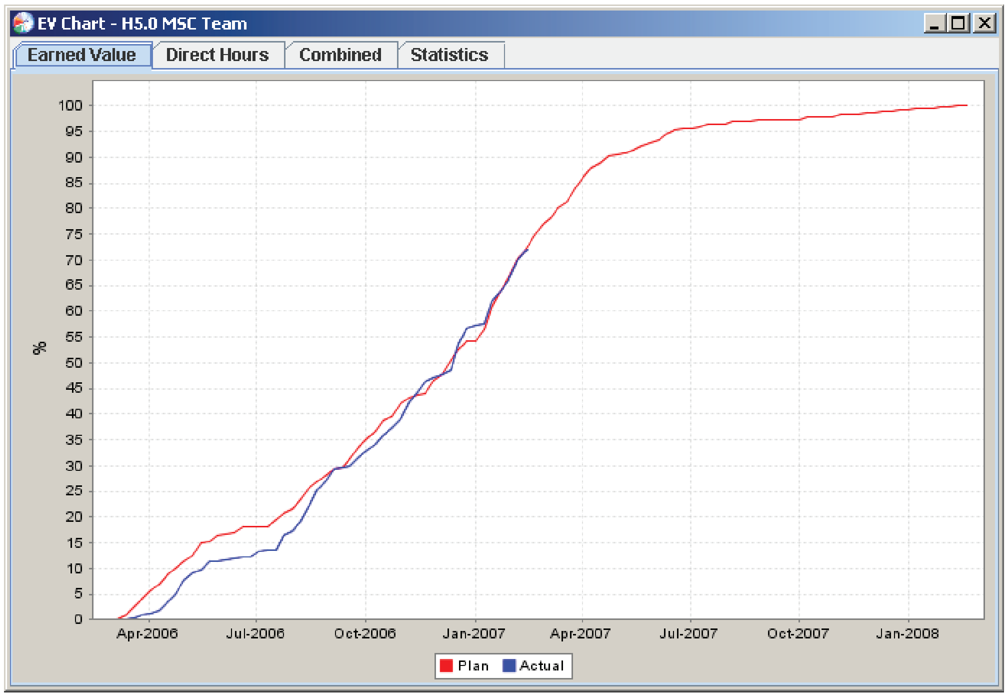


Figure 4. Earned Value Chart From the Project F MSC Software Team.

The TSP, while also helping to drive down schedule variance, was instrumental in bringing about a significant reduction in the defect density of the final software products (see Table 3). A 48% decrease in defect density, measured in defects per 1,000 lines of code, occurred between two projects, B and D. The same software engineers were responsible for both development efforts, but Project B used neither EVMS nor TSP, while Project D used both (Table 2). In another illustration of the improvement in quality, a 21% reduction in defect density occurred between Projects C and D. While both of these projects were using EVMS to manage their costs and schedule, only Project D used TSP.

To calculate their return on investment (ROI) for TSP, AV-8B compared the defect data from two projects: B and D. Project B was a pre-TSP project that had a defect density of 1.13 defects per KSLOC. Project D was the first TSP project and it had a defect density of 0.59 defects per KSLOC. Table 4 shows the ROI for TSP from savings derived from the avoidance of rework. A hypothetical cost for Project D without TSP is calculated to give an indication of what the cost could have been.

TABLE 3. AV-8B Defect Densities Related to TSP.

S/W development projects	**Date completed**	**S/W defects during V&V**	**KSLOC**	**S/W defects per KSLOC**	**Used TSP?**
Project B	4/02	36	32	1.13	No
Project C	5/04	66	89	0.74	No
Project D	7/04	260	443	0.59	Yes
S/W maintenance projects	**Date completed**	**STR defects during system test**	**STRs resolved**	**STR defects per 10 STRs resolved**	**Used TSP?**
Project E S/W Cycle 1	3/04	10	88	1.13	Yes
Project G S/W Cycle 1	9/04	2	40	0.50	Yes

TABLE 4. AV-8B Return on Investment in TSP After One Project.

	Product size (KSLOC)	Defect density (defects/KSLOC)	Number of defects	Cost of addressing defect	Total cost for addressing all defects
Pre-TSP performance baseline					
Project B (pre-TSP)		1.13			
Hypothetical cost of addressing defects for a non-TSP Project D					
Hypothetical Project D cost	443	1.13	501	$8,330	$4,169,831
Actual cost of addressing defects for the Project D TSP					
Project D (TSP)	443	0.59	261	$8,330	$2,177,169
Cost savings from the avoidance of rework					
Cost savings from reduced defect density					$1,992,662
AV-8B's cost of TSP training and support					$225,300
ROI from cost savings from the avoidance of rework					$1,767,362

AV-8B saved $1.992 million through the avoidance of rework. Even subtracting AV 8B's expense for initiating TSP, the investment was more than returned.

1.4.2 E-2C Software Support Activity

The E-2C project office in Patuxent, Maryland, initiated its SPI effort in 2000 with an initial goal of "providing the Fleet with quality products that are both affordable and available when they are needed." E-2C intended to achieve an SEI CMM Level 2 rating over the course of a 6- to 12-month project. After reviewing the available SPI tools, the team adopted TSP. It was decided that the use of TSP would commence with the start of the next major project. During the planning for that project the scope of work proved to be larger than the current work force could handle. In 2002, while searching for a solution to this challenge, E-2C discovered an opportunity.

In 2001, the F-14D SSA at Point Mugu, California, was using SEI CMM and had begun training for (and using) TSP in its final major software release to the Fleet. Although the program was scheduled to be phased out after the completion of that project, F-14D management considered TSP training to be an excellent investment to make in the project engineers. They set the goal of achieving a CMM Level 2 rating and proceeded. E-2C learned that the F-14D program would be closed in mid 2003, and that the project engineers would become available for work elsewhere. The F-14D engineers had the training and disciplined software processes that E-2C was seeking, so E-2C approached F-14D management with the idea of folding those engineers into the E-2C at the conclusion of the F-14D program. The F-14D managers agreed.

E-2C launched its first TSP project in May 2003 and was formally assessed at CMM Level 2 in June. In July 2003 it incorporated the F-14D engineers into E-2C and launched a second TSP project at Point Mugu. After making some progress on the project, E-2C re-launched, replacing TSP with TSPm. E-2C found that TSPm was effective for organizing and managing both large and distributed teams. It was also effective in bringing together groups with different backgrounds, by giving them a common language and process. Table 5 shows the performance of three of E-2C's early TSP projects.

E-2C is currently transitioning from CMM to CMMI.

TABLE 5. Schedule Performance of Three Early E-2C TSP Projects.

Project name	Planned length, weeks	Actual length, weeks	Schedule variance
SCS-04 ACIS	33	32	3% under
SCS-04 MC	28	38	36% overrun
SCS-05 SIAP Phase I	29	34	17% overrun

1.4.3 EA-6B Software Support Activity

The EA-6B SSA employs over 200 people in support of the development, enhancement, and maintenance of almost 8 million source lines of code (SLOC). EA-6B began its SPI process in October 2000 with the acquisition of Pragma's processMax® software. A process improvement lead was assigned in May 2001 and a Process Steering Committee was established in September 2001. EA-6B's SPI toolset included the CMM and HPO, but the initial focus was the implementation of an organizational level process via the processMax software tool. Mini-assessments were conducted in May and October 2003 and in February 2004. These preparations paid off when the EA-6B SSA achieved CMM Level 3 in its first official appraisal in September 2004.

One of the most significant payoffs for the EA-6B SSA occurred during the EA-6B ICAP III Block 2 project. The payoff was a substantial reduction in the number of Operational Flight Program (OFP) defects discovered per 100 system test hours. The rate of discovery of defects is a standard NAVAIR product maturity measure used in Operational Test Readiness Reviews (OTRRs). The goal of the EA-6B ICAP III Block 2 project was to have a rate of discovery of no more than 12 defects per 100 hours. However in the last quarter of 2004, EA-6B noted that their OFP defect discovery rate went from the desired rate of 12 to more than 20 per 100 hours. Their ICAP III team reviewed the data and in early 2005 used the results of their analysis to modify their software peer review process to enhance its effectiveness. The changes that the ICAP III team made allowed them to discover and correct a greater number of defects prior to releasing the next OFP Build. The result was a reduction in the rate of discovery to 6 per 100 hours, surpassing their original quality goal.

The EA-6B SSA was also able to deliver software intensive products ahead of schedule. SPI helped the Airborne Electronic Attack (AEA) Unique Planning Component (UPC) project maintain and surpass their planned software development and delivery schedule. Maintaining these delivery commitments was critical to the success of the prime contractor development activities.

A number of other SPI initiatives resulted in cost and schedule savings for the EA-6B Weapon System Support Laboratory (WSSL). When added together, these additional annual savings come to 1,231 labor hours (two-thirds of a work-year). These initiatives include:

- Upgrading the WSSL Discrepancy Reporting (WDR) process to be CMMI compliant and utilizing Lean Six Sigma concepts to reduce work-in-progress. The new process provided web-based access for submission and tracking of WDRs. This resulted in a reduction of manual labor for input/updates from 4 hours per week to 1 hour, an estimated annual savings of 156 hours. Metrics reporting is now automated rather than manual. A new process for testing and closing WDRs reduced work-in-progress by 50% in the first year, an estimated savings in labor of 650 hours per year.
- Documenting and improving the laboratory engineering drawing and simulation Configuration Management (CM) process to be CMMI compliant. The estimated savings in labor was 425 hours per year.

1.4.4 P-3C Software Support Activity

P-3C began its process improvement initiative in April 2001 with the formation of an HPO leadership team. Their initial goal was to implement HPO concepts within the organization to develop more mature software development processes. P-3C then decided to add the CMMI and the TSP to its tool set. A SEPG was formed in February 2002 and the first TSP launch was conducted in May 2002. In March 2003, after performing a comprehensive gap analysis, the organization transitioned from CMMI to the CMM. While the value of CMMI was recognized, CMM would allow a quicker pace (with an earlier "win" providing encouragement to the team). In May 2004, within 27 months of forming their SEPG, P-3C achieved CMM Level 4. As with AV-8B, P-3C achieved this in less than half of the 5½ years normally expected (Wall 2007, pages 3-4).

In August 2004 P-3C performed a CMMI gap analysis and transitioned from CMM back to the CMMI. In October 2005, 17 months after achieving CMM Level 4, P-3C completed a Standard CMMI Assessment Method for Process Improvement (SCAMPI) B. While a SCAMPI B does not result in an official CMMI rating, the results of the SCAMPI^{SM} B indicated that P-3C was operating close to CMMI-Software Level 5. One interesting finding from the P-3C SCAMPI B Appraisal Findings Report was that "TSP/PSP implementation has provided a rich data source upon which to build, compare, and begin statistical management of selected processes" (NAVAIR 2005).

P-3C had progressed rapidly through the CMM levels, but what sort of return on investment did they see for TSP? Assuming all other things to be equal, it is useful to focus on the savings realized through the increased quality that TSP brings (i.e., savings from the avoidance of rework). To do that, two projects were compared: a P-3C non-TSP project (for the performance baseline) and P-3C's first TSP project. Table 6 lists some basic performance data for these projects. It includes a hypothetical cost for the TSP project based on the non-TSP defect density. This will give an indication of what it might have cost the project to repair defects in "Unit" test had it not been a TSP project. In this example, P-3C refers to defects as Software Problem Reports (SPRs).

TABLE 6. P-3C Performance Data Comparison for Non-TSP and TSP Projects.

	KSLOC added/ changed	Defect density/ KSLOC	Number of SPRs	Average SPR fix cost	Total SPR fix cost
Pre-TSP performance baseline					
Non-TSP Project	27.8	4.6	128	$8,432	$1,078,284
Hypothetical cost of addressing defects had TSP project not used TSP					
Hypothetical Project	38.3	4.6	176	$8,432	$1,484,032
Actual cost of addressing defects for the TSP project					
TSP Project	38.3	0.6	23	$8,432	$193,936
Cost savings from the avoidance of rework					
Cost savings from reduced defect density					$1,290,096
P-3C cost of TSP training and support					$311,247
ROI from cost savings from the avoidance of rework					$978,849

The number of SPRs generated during Unit testing of the TSP project was seven times lower than the non-TSP project because the developers were able to identify defects early in the development process, rather than having to "test" them out. As a result, the total cost of removing the Unit test defects was significantly lower than the non-TSP project, even though that project was actually larger in terms of total KSLOC.

When the Unit test defect removal cost of the TSP project is compared to what that cost might have been had TSP not been used, the savings by avoiding rework were nearly $1.3 million. P-3C invested $311,247 into the training, setup, and support for TSP. Subtracting those costs from the savings gives an ROI of $978,849. P-3C's investment in TSP was more than returned with their first TSP project.

1.4.5 TACAIR EW Software Support Team

TACAIR SSA provides post-deployment, mission critical software and systems engineering, integration, and testing for TACAIR Strike and Assault aircraft. TACAIR SSA began pursuing SPI in March 1999. A process guide was developed and published in October 2000 and several HPO sessions were held. In August 2002 TACAIR SSA held a formal assessment and, due to some minor deficiencies in the area of software quality assurance, just missed obtaining a CMM Level 2 rating. Applying the lessons from that assessment, by August 2004 TACAIR SSA was assessed at CMM Level 3. In 2005 it was using the results of a SCAMPI C assessment to help formulate a transition from CMM to CMMI.

1.4.6 F/A-18 SWDTT

The F/A-18 SWDTT is a 70 member sub-team of the F/A-18 Advanced Weapons Lab (AWL). It was among the earliest adopters of process improvement as a way to reduce cost and improve quality. F/A-18 SWDTT has pursued SPI since the early 1990s and in December 1997 achieved a CMM Level 3 rating. In November 2000 there was a setback when the SPI lead left during a heavy turnover in personnel (104 personnel within 1½ years). When the SPI Lead position was filled in November 2001, F/A-18 SWDTT realized that any institutionalization of its process improvement had been lost. The decision was made to reorganize and restart the SPI initiative. F/A-18 SWDTT developed a plan based on the results of a CBA IPI assessment conducted in April 2003. As part of that plan, TSP was selected for use within the team and a TSP launch was conducted. F/A-18 SWDTT improved its time tracking tools and established a web site to provide information and resources to support team process improvement. F/A-18 SWDTT also generated a "Five Step Model" for the organization (Juarez 2006):

1. Focus on familiarization, re-education, and training.
   - Understand that these need to be continuous.
   - Update and present training/orientation packages.
2. Recognize process compliance.
   - Observe: communicate what has been observed.
   - Recognize: brief team members on metrics that are gathered and used.
   - Make it cultural: talk about it.
3. Complete a Self-Assessment and Gap Analysis.
   - "Health check" for process.
   - Next steps.
4. Translate the process changes into something meaningful to the engineering staff.
   - Previous process changes had left the engineering staff feeling uninvolved.
   - Time saving.
   - Effort saving.
5. Collect and use your metrics.
   - Improve cost estimates.
   - Improve schedules.
   - Improve product quality.
   - "Steps" must be concurrent.
   - "Steps" must be sustained.
   - Process Improvement should be a project.

F/A-18 SWDTT conducted a SCAMPI B in October 2004 and used the results of that assessment to prepare for a SCAMPI A. The effort was successful and in March 2005, F/A-18 SWDTT was assessed at CMM Level 5, the first in the Navy.

Following the Level 5 assessment, F/A-18 SWDTT set new goals that included assisting all the appropriate AWL teams in the entire F/A-18 AWL to achieve CMMI Level 3 maturity. If successful, this would encompass the development, enhancement, and maintenance of over 10 million SLOC.

1.4.7 Overall

Overall, the SSAs made steady SPI progress and delivered significant achievements (see Figure 5). It is important to note again the rapid progression of several of the SSAs through the CMM levels. Using TSP, AV-8B and P-3C were able to achieve CMM Level 4 in less than 3 years: March 2000 to September 2003 for AV-8B, and February 2002 to May 2004 for P-3C. The SEI average for progressing from CMM Level 1 to Level 4 is 5½ years (Wall 2007). The colored zones in Figure 5 represent recommended CMM goals that were set in the February 1999 NAVAIR Team Software Strategic Plan. The lines connecting the milestones are there only to group certain information.

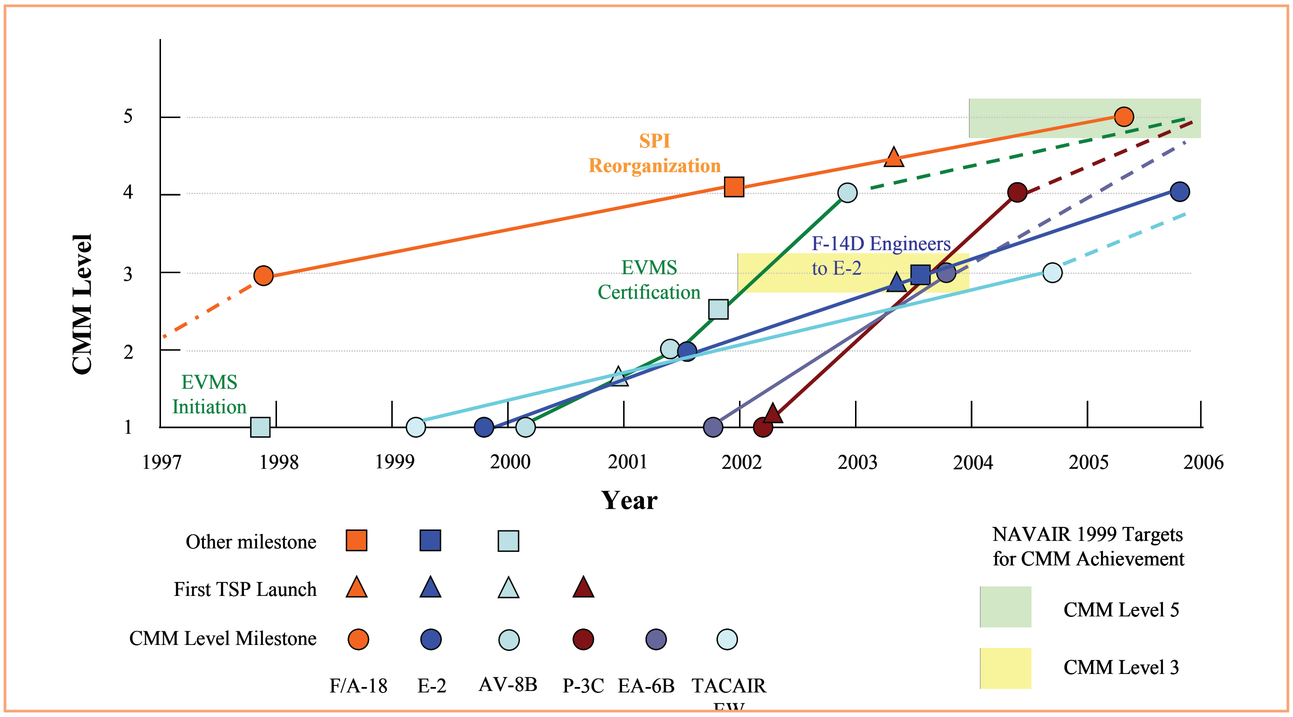


FIGURE 5. Time Line of SAA CMM Progress and Related Milestones.

1.5 The Future of Process Improvement within NAVAIR

NAVAIR 4.0's SPI efforts have been successful in developing mature organizations and in obtaining an excellent ROI. The early SPI adopters are meeting their missions, producing higher quality products, and generating significant cost savings. Their success stories have inspired other SSAs within NAVAIR 4.0 to adopt SPI. Fifteen of the 18 SSAs that were not among the early adopters of SPI are now pursuing SPI in some form. Figure 6 is an Earned Value Chart generated by the TSP tool of the Anti-Radiation Missile (ARM) UPC 0.4 Software Development Team documenting that these new adopters are experiencing the same performance improvement as the early adopters.

SPIKE Program Manager Mr. Steven D. Felix said of their recent PI efforts ". . . TSP was a major contributor to the success of our project. Most processes assume large teams and huge budgets, and because of this are of no value to small projects like SPIKE. Never have I seen a process that was so scaleable that it was useful to a team as small as SPIKE. The metrics collected are tuned for our project. When a metric did not show any value, we could stop collecting it and figure out what to collect that did make sense. TSP has been so successful that the SPIKE project has adapted it to our hardware design process."

As these new adopters continue to make progress with SPI, the recurring savings will allow NAVAIR to redirect even more funds to the Fleet for procurement of critical needs, including new aircraft.

As part of the effort to promote process improvement amongst the SSAs, NAVAIR 4.0 created the NAVAIR Software/Systems Support Center (NSSC), an organization tasked with providing assistance and guidance with model-based and process-based performance improvement. In pursuit of this mission, the NSSC:

- Developed an internal appraisal method based on the SEI Appraisal Requirements for CMMI (ARC) document to baseline SPI efforts across NAVAIR.
- Works to expand the number of SSAs within NAVAIR 4.0 that are pursuing SPI.
- Sponsors the NAVAIR Software Process Improvement Community of Practice (SPI CoP) quarterly conferences. Representatives from all NAVAIR 4.0 SEPGs attend these conferences. SPI CoPs are a forum for exchanging SPI histories, best practices, techniques, tools, and lessons learned.
- Sponsors the NAVAIR TSP CoP monthly meetings. TSP coaches and instructors who support NAVAIR 4.0's TSP teams attend these meetings. They coordinate future events, share best practices, and keep each other aware of the status of ongoing efforts.
- Works with the SEI to introduce and pilot TPI projects, a modified TSP for acquisition and SE process improvement. This has resulted in the creation of new courses, including the 2-day SE focused course, Team Member Training.
- Provides SEI-authorized training in CMMI and coaching in PSP/TSP/TPI.

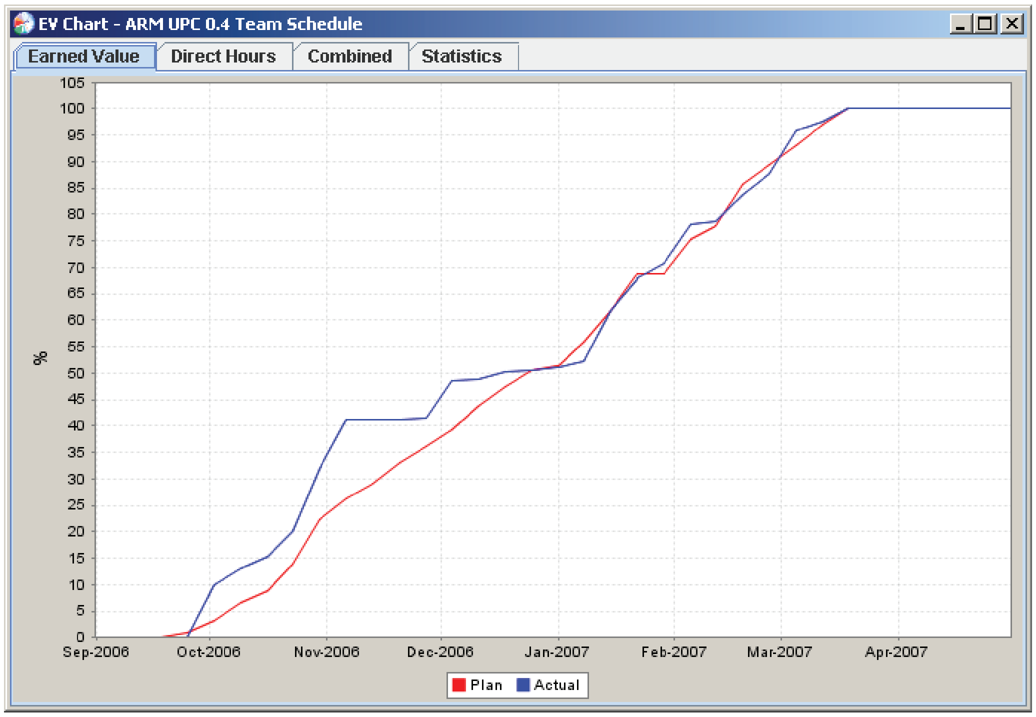


FIGURE 6. Earned Value Chart of the ARM UPC 0.4 Software Team.

The charter of the NSSC also calls for reinforcing and aligning NAVAIR 4.0's SPI initiatives with NAVAIR's general process improvement initiatives, such as the AIRSpeed lean six-sigma project. While the efforts of NAVAIR AIRSpeed do not fall under the scope of this document, AIRSpeed is described here as an important part of the overall process improvement environment. NAVAIR AIRSpeed lean six-sigma was selected as the Naval Aviation Enterprise (NAE)-wide mechanism for reducing costs and improving productivity and customer satisfaction. AIRSpeed utilizes a structured, problem solving methodology called DMAIC (Define, Measure, Analyze, Improve, Control), widely used in business. DMAIC leads project teams through the logical steps from problem definition to problem resolution. Each phase has a specific set of requirements to be met before moving on to the next phase of the project. A summary of steps are as follows:

- Define the problem.
- Measure the baseline performance of the process being transformed.
- Analyze the process to determine a prioritized list of root causes for any process performance shortfalls.
- Improve the target process by designing innovative solutions to resolve the identified root causes.
- Control the process to ensure that the improved process continues to deliver the expected results.

AIRSpeed solicits recommendations on process change from all NAVAIR personnel and contractors. The responsibility for following up on those recommendations rests with specially trained personnel (Black Belts).

As part of the effort to institutionalize lean six-sigma, NAVAIR organized and held an annual NAVAIR lean six-sigma symposium. The Navy, looking for a way to spread lean six-sigma throughout the organization, saw the success of NAVAIR's event and had NAVAIR establish the first Navy-wide, lean six-sigma symposium in May 2007.

Not content with just experiencing the tangible advantages of process improvement, NAVAIR 4.0 is devoted to spreading SPI throughout the Navy, the Federal Government, industry, and beyond. NAVAIR 4.0 is a co-sponsor of DOD's Crosstalk magazine; has been the sponsor of various conferences, workshops, and panel discussions; and has published numerous articles on SPI. NAVAIR 4.0 personnel participate in the international TSP Users Group (TUG) conferences, with one NAVAIR employee holding the office of TUG Conference Chairman for 2 years; the annual National Defense Industrial Association (NDIA) CMMI-User's Conferences; and the annual SEI SEPG Conferences. This effort on behalf of SPI has been noticed by the wider community, which gave a NAVAIR employee the 2007 SEI Member Representative award.

1.6 Conclusions

NAVAIR 4.0 had recognized the need for and was pursuing SPI even before the U.S. Government entered into the issue with Public Law 107-314. NAVAIR 4.0's dedicated interest in SPI resulted in:

- The first CMM Level 5 rating in the Navy.
- The second EVMS certification in the Federal Government.
- Two teams achieving CMM Level 4 in less than half the normal time.
- Defect density reductions ranging from 22 to 48%.
- Cost and schedule variances reduced to within 10%.

SPI also paid impressive ROIs. AV-8B and P-3C invested a combined total of $536,000 into the adoption of TSP and saw a combined gross savings from their first TSP efforts (through the avoidance of rework) of an estimated $3.283 million. This gave a net ROI of approximately $2.746 million. EA-6B, using the WDR process and lean six sigma concepts, saved at least $116,000.

The successes that NAVAIR 4.0 has enjoyed from SPI and its culture of process improvement have helped to ensure the continued pursuit of and advancement of SPI within the organization. A belief in the real value of SPI to the Navy and to the Government has created a NAVAIR 4.0-wide software development community with a desire to spread SPI outside its own ranks. Government bureaucracies are wary of change, and many will actively resist change unless they are provided with concrete examples of its advantages. This resistance often begins with the individual workers and extends up through middle management. When a Government organization recognizes the need for change, and the personnel throughout that organization actively seek it out, a fundamental shift in paradigms has taken place. NAVAIR 4.0 demonstrated this dramatic change in thinking through the widespread adoption and institutionalization of CMMI and TSP. It is an operational example of the concrete advantages of pursuing SPI: higher quality products, at lower cost, while maintaining the mission.

In January 2003, Darrell Maxwell, at that time Department Head of the NAVAIR Systems Engineering Department, made the following statement after reviewing NAVAIR 4.0's organization and the strides that had been made in just 3½ years of SPI efforts: "In February 1999 we at NAVAIR set out to make notable achievements in software process improvement across the organization. It was just good business. It is now January 2003, and we have not only met our goals, but in some cases achieved even more than we planned."

1.7 Recommendations

A complete discussion of the introduction of process improvement into an organization is not within the scope of this document. However there are some basic concepts and resources that will help. Process improvement can be a difficult undertaking, and if it is not pursued in a systematic fashion, it will be much more difficult. SEI identified eight key concepts for introducing process improvement into an organization, with the focus on introducing CMMI (http://www.sei.cmu.edu/cmmi/adoption/cmmi-start.html, 2007). The concepts are summarized here.

1. Secure Sponsorship and Funding. First, ensure that the process improvement effort has both a senior management sponsor and funding.
2. Take Core Training. Understand the basic concepts of the tools and methodologies that will be used in the process improvement effort.
3. Prepare the Organization for Change. Treat process improvement as a project. Establish business reasons and goals for the effort. Create a case and rationale for undertaking this change. Identify the expected costs and benefits for everyone. Plan for and manage the human side of change.
4. Form an Engineering Process Group (EPG). The EPG should coordinate the process improvement activities across the organization.
5. Know Where You Are. Survey the organization and compare their processes to those of the tools and methodologies that will be used in the process improvement effort. This will serve as a baseline for the effort.
6. Know Where You Are Going. Determine the overall process improvement goals for the organization. Prioritize the areas to be addressed and then, using the baseline for the organization as the starting place, plan the path to achieve those goals. Track the organization's progress against the plan.
7. Communicate and Coordinate. Promote and practice honest and open communication. The plan should be shared with all affected parties. Comments and concerns should be taken seriously.
8. Track Your Progress. Monitor the progress of the organization through the plan, making adjustments as needed.

1.8 References

[Saint-Amand 2007]
Naval Air Systems Command. *Results of the Software Process Improvement Efforts of the Early Adopters in NAVAIR 4.0*, by David Saint-Amand, and Bradley Hodgins. NAVAIR Systems/Software Support Center (NSSC), 2007. (NAVAIR Technical Paper 8642 UNCLASSIFIED.)

[Wall 2005]
Software Engineering Institute. *Case Study: Accelerating Process Improvement by Integrating the TSP and CMMI*, by Daniel S. Wall, James McHale, and Marsha Pomeroy-Huff. Pittsburgh, Pennsylvania, Carnegie Mellon University, SEI, 2005. (CMU/SEI-SR-012, report UNCLASSIFIED.)

[Wall 2007]
Software Engineering Institute. *Case Study: Accelerating Process Improvement by Integrating the TSP and CMMI*, by Daniel S. Wall, James McHale, and Marsha Pomeroy-Huff. Pittsburgh, Pennsylvania, Carnegie Mellon University, SEI, 2007. (CMU/SEI-2007-TR-013, report UNCLASSIFIED.)

[Davis 2002]
Software Engineering Institute. *Relating the Team Software Process to the SW-CMM*, by Noopur Davis and Jim McHale. Pittsburgh, Pennsylvania, Carnegie Mellon University, SEI, 2002. (CMU/SEI-2002-TR-008, report UNCLASSIFIED.)

[NAVAIR 2005]
Naval Air Systems Command. *SCAMPI^{SM} B Appraisal Findings*, by P-3C Software Support, Multi Mission Patrol, Mission Area Team. Naval Air Station Patuxent River, Maryland, NAVAIR, 14 November 14 2005. (Report UNCLASSIFIED.)

[Juarez 2006]
Naval Air Warfare Center Weapons Division. *F/A-18 Software Development Task Team (SWDTT) Lessons Learned on Achieving a CMM Level 5,* by Sharon Juarez. China Lake, California, NAWCWD, 2006. (Presentation UNCLASSIFIED.)

Sidebar Material

NAVAIR ORGANIZATIONAL GOALS

In 2005, NAVAIR's Organizational Goals were stated on the NAVAIR web site.

1. To Balance Current and Future Readiness. We need to ensure that we provide our Naval aviators with the right products to fight the Global War on Terrorism and other future conflicts.
2. To Reduce Our Costs of Doing Business. We need to pursue actual cost reductions, not so-called "savings" or "avoidance." We need to return resources to recapitalize our Fleet for the future. We must continue to introduce best business practices and to remove any barriers to getting our job done.
3. To Improve Agility. It is essential that we make rapid decisions in support of emerging Fleet requirements in order to continue to provide value to the nation. We must reinvigorate a solid chain of command that values responsibility and accountability in its leadership.
4. To Ensure Alignment. We have come a long way with aligning ourselves internally. Now we must ensure that we are fully aligned, internally and externally, with the Chief of Naval Operation's (CNO) transformation initiatives.
5. To Implement Fleet-Driven Metrics. Single Fleet-driven metrics will ensure that we directly contribute to the Naval Aviation Enterprise.

ACRONYMS

AEA	Airborne Electronic Attack
AFCEA	Armed Forces Communications and Electronics Association
ARBS	Angle Rate Bombing System
ARC	Appraisal Requirements for CMMI
ARM	Anti-Radiation Missile
AVJMPS	AV-8B Joint Mission Planning System
AWL	Advanced Weapons Lab
BLK	Block
CBA IPI	CMM-Based Appraisal for Internal Process Improvement
CCHPO	Commonwealth Centers for High Performance Organizations
CM	Configuration Management
CMMI	Capability Maturity Model Integration
CNO	Chief of Naval Operations
DMAIC	Define, Measure, Analyze, Improve, Control
DOD	Department of Defense
DoN CIO	Department of the Navy Chief Information Officer
EPG	Engineering Process Group
EV	Earned Value
EVMS	Earned Value Management System
EW	Electronic Warfare
GWT	Global War on Terrorism
HPO	High Performance Organization
ICAP	Initial Capability
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
IEPR	Independent Expert Program Review
IOC	Initial Operational Capability
ISO	International Organization for Standardization
JSF	Joint Strike Fighter
JSSA	Joint System Support Activity
KSLOC	One thousand source lines of code
LOE	Level of Effort
MAIS	Major Automated Information System
MDAP	[U.S. DOD] Major Defense Acquisition Program
MR	Management Reserves
MSC	Mission Systems Computer
NAE	Naval Aviation Enterprise
NAVAIR	Naval Air Systems Command
NCW	Network-Centric Warfare
NDIA	National Defense Industrial Association
NSSC	NAVAIR Software/Systems Support Center
NTS	Night Targeting System
OFP	Operational Flight Program

OTRR	Operational Test Readiness Review
P3R	People, Process, and Product Resource [group]
PI	Process Improvement
PSP	Personal Software Process
ROI	Return on Investment
RUG	Radar Upgrade
SCAMPI	Standard CMMI Assessment Method for Process Improvement
SE	System Engineering
SEI	[Carnegie Mellon University] Software Engineering Institute, Pittsburgh, Pennsylvania
SEPG	Software Engineering Process Group
SLOC	Source Lines of Code
SMUG	Stores Management Upgrade
SPI	Software Process Improvement
SPI CoP	Software Process Improvement Community of Practice
SPR	Software Problem Report
SSA	Software Support Activity
STR	System Trouble Report
S/W	Software
SW-CMM	Software-CMM
SWDTT	Software Development Task Team
SWIP	Software Improvement Program
TACAIR	Tactical Aircraft
TPI	Team Process Integration
TSP CoP	Team Software Process Community of Practice
TSPm	Team Software Process for Multiple Teams
TUG	TSP Users Group
UAV	Unmanned Aerial Vehicle
UPC	Unique Planning Component
V&V	Verification and Validation
WBS	Work Breakdown Structure
WD	Weapons Division
WDR	WSSL Discrepancy Reporting
WSSL	Weapon System Support Laboratory

Authors

Brad Hodgins

Brad Hodgins is a SEI-Authorized TSP Coach, SEI-Certified PSP/TSP Instructor, and SEI-Certified Software Developer for the NAVAIR System/Software Support Center (NSSC) of the Engineering Division of the Naval Air Systems Command (NAVAIR) located at China Lake, California.

Hodgins has been with NAVAIR for 24 years. He has over 20 years experience as a software engineer developing simulation and avionics software. He has been applying PSP/TSP for over seven years and has coached over 30 launches.

Hodgins earned a BS in Computer Science from California State University, Chico.

David Saint-Amand

Mr. David Saint-Amand is an SEI-Authorized PSP Instructor and Naval Air Systems Command (NAVAIR) TSP Coach at the Process Resource Team. He has supported the Navy's mission as a Defense Department Contractor off and on since 1980. His previous positions include Naval Operations Research Analyst, Configuration Management Board Chair, Help Desk manager, Instructor, and Liaison.

Prior to his move to full-time support of the Navy in 1994, Mr. Saint-Amand was a Geologist and Seismic Safety Consultant.

Saint-Amand holds a B.A. in Geology from the University of California at Santa Barbara.

PSP in High Maturity Settings

Adham Shaher, Dr. Amr Kamel

1 Introduction

In their journey to achieve CMMI Maturity Level 5, organizations begin to focus on innovations that would improve its current processes in a significant manner and help in achieving its business objectives. Following the Organization Innovation and Deployment (OID) framework [SEI CMMI 00], the organization started to investigate the PSP as a consistent framework for process improvement. Surveys were conducted across the industry aiming at realizing the benefits and implications of applying the PSP in real-world settings. It showed that PSP had drastically improved the quality of the products, achieving zero delivered defects for 17 projects [Ferguson 97], while reducing the duration of system and acceptance testing [Noopur 05]. However, there were some challenges we faced; (1) none of the studies covered implementing the PSP within a CMMI L5 organization (2) reported results recommend coupling the PSP /TSP and not to implement PSP alone [Wall 05], [Karina 05], [Humphrey 98].

The decision was to start piloting for the PSP without the TSP – due to funding limitations - and evaluate its results against the organization objectives. In addition, the PSP pilot should be scoped and to focus on the most important improvement objective in the organization, which is improving the quality of deliverables in terms of early defect removal, and delivered defects.

Based on this, the pilot scope was defined as follows:

- Incorporating the practices suggested by the PSP Quality Management Process within the life cycle of the project,
- Monitoring and evaluating the impact of these practices on the quality of work products and product of the pilot, and
- Comparing the pilot project performance to the organization's quality baselines.

The pilot results showed that 97.4% of defects were detected before delivery, with 75% of known defects were removed before testing phase. Both Design DRE (*Defect Removal Efficiency)* and Code DRE has improved by more than 25% compared to the organization baselines, achieving the organization improvement objective set for both metrics. There was a minor improvement in overall Defect Density (0.016692 Defect / KLOC) and this could be attributed to the minor changes in the design process. There was no improvement in the Defect Detection Rate; yet, Defect Removal Rate was significantly improved (8.66 defects removed/hr). With these results, the pilot proven its successfulness and the deployment of the PSP within the organization becomes the next step.

2 Background

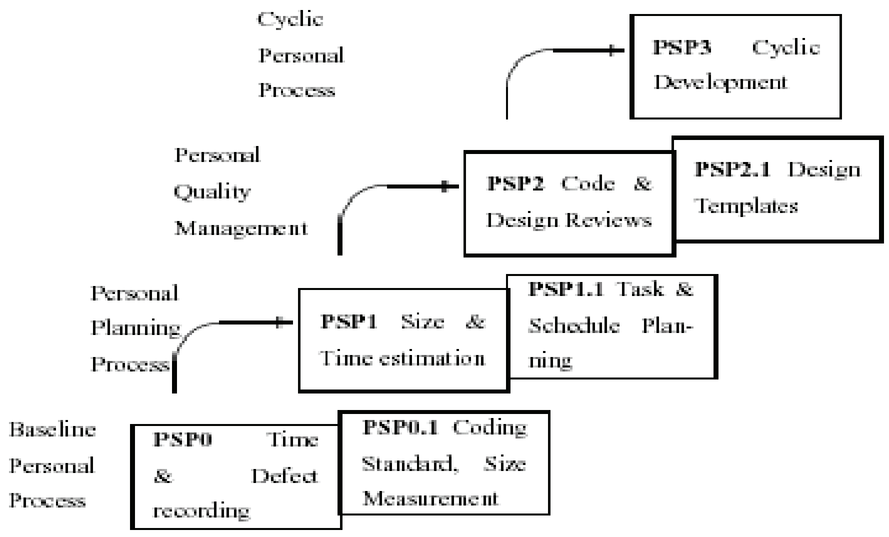
2.1 PSP Overview

PSP was developed at Software Engineering Institute (SEI). It was designed to bring discipline to the practices of individual software engineers, by providing a framework for measuring and analyzing their development work so that they produce programs of high quality and in more predictable manner. It shows them how to plan and track their work, use a defined and measured process, establish measurable goals, and track performance against these goals. It also enables engineers to manage quality from the beginning of the job, how to analyze the results of each job, and how to use the results to improve the process for the next project. The PSP can be applied to many parts of the software development process, including small-program development, requirement definition, document writing, systems tests, and maintenance and enhancement of large software systems. [Humphrey 95]

2.2 PSP Levels

PSP is divided into seven processes, they are labeled PSP0 through PSP3 (see figure '2.1), and each process has a similar set of logs, forms, scripts, and standards. The process scripts define the steps for each part of the process, the logs and forms provide templates for recording and storing data, and the standards guide the engineers as they do the work. [Humphrey 95]

Figure 2.1: PSP Levels


Cyclic Personal Process
PSP3 Cyclic Development
Personal Quality Management
PSP2 Code & Design Reviews
PSP2.1 Design Templates
Personal Planning Process
PSP1 Size & Time estimation
PSP1.1 Task & Schedule Planning
Baseline Personal Process
PSP0 Time & Defect recording
PSP0.1 Coding Standard, Size Measurement

2.3 PSP Quality Management

The PSP provides a series of practices and measures to help engineers assess the quality of the programs they produce and to guide them in finding and fixing all program defects as quickly as possible. The principal PSP quality measures are: defect density, review rate, development time ratios, defect ratios, and defects per hour. [Humphrey 95], [Humphrey 97]

In addition to quality measurement and tracking, the PSP quality management methods are early *defect removal* and *defect prevention.*

2.3.1 Early Defect Removal

The primary PSP quality objective is to find and fix defects before the first compile or unit test. This is done through the design and code review steps in which engineers personally review their work products before they are inspected, compiled, or tested. The principle behind the PSP review process is that people tend to make the same mistakes repeatedly. Therefore, by analyzing data on the defects they have made and constructing a checklist of the actions needed to find those mistakes, engineers can find and fix defects most efficiently.

2.3.2 Defect Prevention

The most effective way to manage defects is to prevent their initial introduction. In the PSP, there are three different but mutually supportive ways to prevent defects.

The *first* is to have the engineers record data on each defect they find and fix. Then they review these data to determine what caused the defects and to make process changes to eliminate these causes. By measuring their defects, engineers are more aware of their mistakes, they are more sensitive to their consequences, and they have the data needed to avoid making the same mistakes in the future.

The *second* prevention approach is to use an effective design method and notation to produce complete designs. To completely record a design, engineers must thoroughly understand it. This not only produces better designs; it results in fewer design mistakes.

The *third* defect prevention method is a direct consequence of the second: with a more thorough design, coding time is reduced, thus reducing defect injection. It was noticed that, engineers inject an average of 1.76 defects per hour; while during coding they inject 4.20 defects per hour. Since it takes less time to code a completely documented design, by producing a thorough design, engineers will correspondingly reduce their coding time. Therefore inject fewer coding defects. [Humphrey 95]

3 Pilot Planning

The pilot planning phase included creating a generic guideline on how to incorporate the PSP within the existing processes, identifying the set of measures to be used as evaluation criteria, selecting a pilot project, and identifying the training needs for the pilot project team.

3.1 Tailoring

In order to use the PSP practices within the pilot project, we needed to define an engagement model or a tailoring plan that would not jeopardize the implementation of the organization's defined process and tools. We found that we needed a generic guideline that directs us on (1) how to tailor the process of organizations to fit the PSP within and (2) how to incorporate PSP tools with the organizations' tool set. Having no such guideline available in industry or published in the research field, we decided to develop a guideline for this specific purpose, then to use it to derive our specific tailoring for the pilot project. The second step is to develop our pilot project tailoring plan using this guideline.

3.1.1 Generic Tailoring Guideline

The guideline specifies (1) generic procedures/processes that are usually used in high maturity organizations adopting the CMMI framework, (2) the specific items within each process/ procedures, which will be affected by PSP practices, (3) the tailoring options identify - for each specific item - the possible tailoring paths from which organizations can choose to implement, and (4) some guideline that might help in implementation. A sample of the guideline is shown in table '3.1' below.

Procedure	Tailoring Options		Guidelines
	Option	Tailoring Description	
Tasks Scheduling & Planning	No	Use organization tools and techniques in planning and task scheduling	
	Yes	Use PSP Planning and Task Scheduling techniques.	Using Process Dashboard
	Yes	Integrate PSP planning & task scheduling with the existing planning and scheduling tool.	Modify the existing tool to include the needed parameters suggested by the PSP Enhance the Process Dashboard to include all other parameters used by the existing tool.
Estimation	No	Use existing estimation method	
	Yes	Use PSP PROBE Estimation technique to estimate development size and effort.	Using Process Dashboard.
Defect Management	No	Existing Defect Management System will be used	
	Yes	PSP Defect management framework will be used.	Self review defects will be recorded using Process Dashboard Peer review defects will be recorded in existing tool.
	Yes	Integrating existing defect management system with the PSP Defect Management System	Modify the existing tool to include the needed parameters suggested by the PSP Enhance the Process Dashboard to include all other parameters used by the existing tool.

Table 3.1: Generic Tailoring Guideline

3.1.2 Pilot Project Tailoring Plan

Using the generic tailoring guideline, we have developed the pilot project tailoring plan. Table '3.2' describes the tailoring. We have chosen the practices that will directly impact the quality of the work products and final product.

Procedures /process	Status	Tailoring Description
Coding Procedure / Peer review	Yes	*Self-code review* will be incorporated in the project lifecycle. Organization code review checklist will be used in self-review. Organization's defined Coding Standards will be used.
Procedure for Defect Management	Yes	Design, coding, and compile defects-detected by the work product owner - will be recorded as per the *PSP Defect management (Defect types standards, defect recording)* framework using *PSP tool* (Process Dashboard). Other organization's mandated parameters will be appended to the defect report generated from the Process Dashboard. Testing Defects will be logged in the testing defects management system.
Project Tracking and Reporting	No	The organization's tracking system will be used to track the effort
Software design procedure /Peer review	Yes	Merging *PSP Design Template* contents with the organization's design template. *Self-design review* will be incorporated in the project lifecycle. Design review checklist will be updated to reflect changes applied to the design template.
Measurement and analysis	Yes	The project will use the organization's identified measures. Process Dashboard will be used in self-reviews defect recording. ** Refer to section 3.2 for details*
Other Procedures	No	

Table 3.2: Case Study-Pilot Project Tailoring Plan

3.1.3 Measurements Framework

The next step was to define the measurements framework that we will use to monitor and measure the quality of the work products and product. We identified–from one side–all the organization's metrics used in measuring the quality of the product and–from the other side–the PSP recommended metrics (see figure '3.1' below).

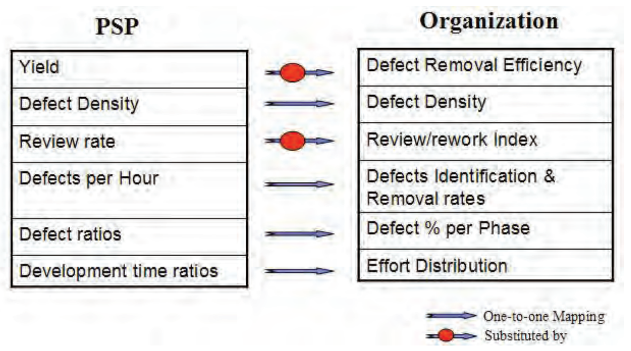


Figure 3.1: PSP vs. Organization Metrics

We found that the two sets of metrics are almost the same, so we decided - for the different metrics - to use the organization's defined ones so as to enable us to compare the project's performance with the organization's baselines. Following are the specifications of these metrics.

- *Defect Removal Efficiency (DRE):* this metric is an indicator of the efficiency of removing defects for the product. Calculated by, Defects found in phase / Total Defects attributed to this phase
- *Review Index:* Review effort / technical effort for creating the reviewed work product.
- *Rework Index:* Rework Effort per review or test / Technical Effort for creating the reviewed or tested work.

3.3 Pilot Project

We selected a seven-month development project using a waterfall lifecycle. The development team had one PSP Certified developer, while other team members went through a customized lightweight PSP training that focused on the selected practices. The training also included the usage the selected tool.

The focused training was designed as follows:

- *First,* we gave the team high-level orientation on the PSP which included the purpose and the objectives of the PSP, the PSP levels, and some published industry results [Humphrey 95], [Humphrey 00].
- *Second,* we took the team through a deeper training on the selected practices as shown in table '3.3' below.

Selected Practices	Customized Training
Time and Defect recording	Summary for Chapter 2, *A Discipline for Software Engineering.* [Humphrey 95] Orientation on time and defect tracking modules in Process Dashboard Tool.
Design & Code self reviews	Summary for Chapter 8, *A Discipline for Software Engineering.* [Humphrey 95]

Table 3.3: PSP Customized Training

4 Pilot Execution

4.1 PSP in Design Phase

In this section, will explore the measures collected during the execution of the design phase.

Table '4.1' shows a summary of the measures collected during the design phase. We have separated the values of self-reviews and the peer reviews.

Metric Name	PSP Design Self Review	Design Peer Review
Defect Identification Rate (Defects Found/hr)	0.63	0.33
Defect removal Rate (Defect Removed/hr)	10.6	1.75
Review Index	14.60%	10.72%
Rework Index	0.87%	2.04%

Table 4.1: Results from Design Phase

We have also performed further analysis on the defects which covered (1) defects by severity, and (2) defects per type. Defect severity determines the impact of the defect on the product, and it is defined in four categories from sev1 to sev4, with sev1 as the highest and sev4 as the lowest. Regarding the defect types, we used the PSP Defect Types Standard.

Defects by Severity: self-review has discovered all high severity defects (sev2) found in design, and triple the number of medium severity defects (sev3) of that discovered in peer review. This shows the high efficiency of self-reviews. Figure '4.1' represents the defects found in both types of reviews with respect to their severities.

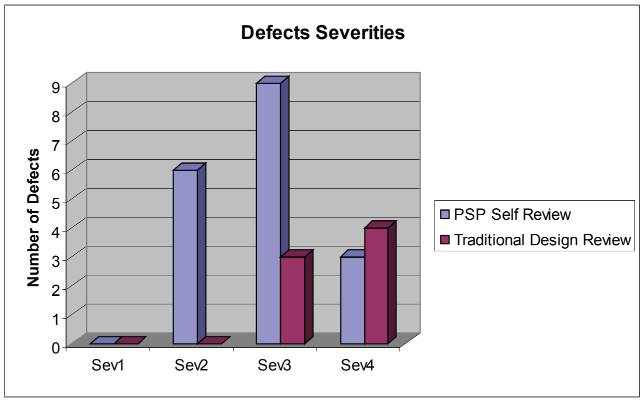


Figure 4.1: Design Defects Types per Severity

Defects by Type: self-review has discovered defects from all types, thus, reflect the effectiveness of the self-review. Figure '4.2' represents the defects found in both types of reviews with respect to their types.

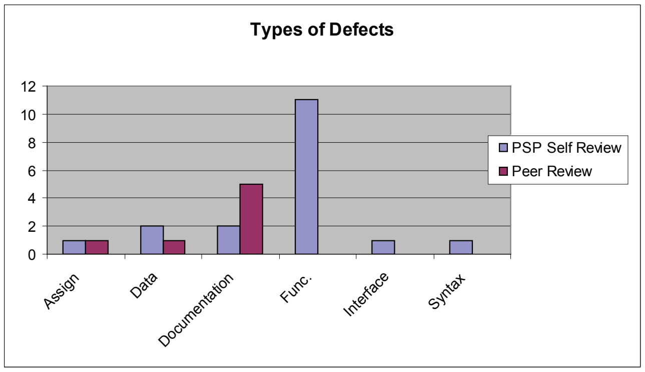


Figure 4.2: Design Defects Types Per Review Type

4.2 PSP in Coding Phase

In this section, will explore the measures collected during the execution of the coding phase. Table '4.2' shows a summary of the measures collected during the coding phase.

Metric Name	PSP Code Self Review	Code Peer Review
Defect Identification Rate (Defects Found/hr)	1.04	2
Defect removal Rate (Defect Removed/hr)	36.45	5.22
Review Index	10.08%	10.08%
Rework Index	0.3%	2.32%
Defect Density (defects/ KLOC)	0.007749	0.014903

Table 4.2: Results from Coding Phase

We have also performed the defect analysis as we did in the design phase.

Defects by Severity: Both review types found defects with severity 3 and 4 with no significant differences in the number of defects in each category.

Defects by Type: No significant differences between both types of reviews except in one defect type, which is "Data". After further analysis, we noticed that the majority of defects of type "Data" were found in peer review, and after performing a deeper analysis we discovered that (a) All "Data" defects in self review were identified by developer 1 in his self review, (b) All "Data" defects in peer review were slipped from developer 2 and were identified by developer 1 in his peer review on developer's 2 code.

5 Pilot Evaluation

Summary

The pilot results showed that 97.4% of defects were detected before delivery, with 75% of known defects were removed before testing phase. Both Design DRE and Code DRE has improved by more than 25% compared to the organization baselines, achieving the organization improvement objective set for both metrics. There was a minor improvement in overall Defect Density (0.016692 Defect / KLOC) and this could be attributed to the minor changes in the design process. There was no improvement in the Defect Detection Rate; yet, Defect Removal Rate was significantly improved (8.66 defects removed/hr).

Details

- Defects percentages per phase were re-distributed, transferring the defects discovery bottleneck from the testing phase to the coding phase –53% of the total defects- and having 75% of the defects discovered before entering the testing phase.
- Overall DRE has been improved drastically, which was a result of adding additional verification checkpoints in the development lifecycle.
- Design DRE – 89.2% - indicates that most of the design defects were discovered in the design phase, identifying and fixing 100% of high severity design defects before entering the coding phase.
- Coding DRE improved to reach – 66%- with a remarkable improvement in preventing high severity defects from slippage - only 8% of the slipped defects were of high severity. This reflects the positive impact of conducting the coding self-review within the coding phase.
- Rework effort was reduced in a way that made the rework and the review indices of the two reviews together fell within the statistical boundaries of the organization's baselines.
- Review effort - combining both review types' effort – has exceeded the organization's statistical boundaries. This is expected as we introduced an additional review type within the development cycle.
- Defect identification rate is slower than the organization's average rate.
- Defect removal rate was improved as a result of the high removal rate for the defects identified during the self-review. This shows that it is much faster to fix defects found by oneself, in his/her own product, than to fix those found by a peer.

6 Deployment

6.1 Recommendations and Considerations

The pilot indicated that incorporating PSP practices within the process of the organization have improved the quality of work products and final delivery drastically in comparison with the organization's baselines. It also showed that organizations start to harvest PSP benefits independent from TSP launching. With these results, the pilot proven its successfulness and the deployment of the PSP within the organization becomes the next step. We have identified our recommendations and some aspects to be considered for the deployment.

- The generic tailoring guideline that we have developed have facilitated the tailoring of the organization's processes to indoor the PSP practices and guided us in identifying the usage model of the Process Dashboard.
- Applying self-reviews on design and code have contributed to achieve major improvement in the quality of deliverables, detecting most of the design defects before entering the coding phase, and improved the percentages of coding defects detected before going to testing.
- The PSP design template did not have any impact on reducing the defects injected during the design, as the template was almost the same as the organization's design template.
- PSP defect management was very useful allowing us to conduct our analysis on the defect level rather than conducting it on the phase level.
- Using the Process Dashboard in defects recording has improved defects data entry, collection, and analysis. However, integrating the tool with the organization's toolset is not an easy task due to the various activities involved in its deployment; starting with the investigations that should be done on the PSP tools, passing through the integration with the existing toolset, and ending with the deployment.
- Measures required by the PSP usually do not fully match with the organizations' set of used measures, adding complexity to adopt the full set of the PSP measures.

References

[Humphrey 95]
Humphrey, W. A Discipline for Software Engineering. Reading, MA: Addison-Wesley, 1995.

[Humphrey 97]
Humphrey, W. Introduction to the Personal Software Process. Reading, MA: Addison-Wesley, 1997.

[Ferguson 97]
P. Ferguson, Watts S. Humphrey, Soheil Khajenoori, Susan Macke, and Annette

Matvya. "Results of Applying the Personal Software Process," Computer, Vol. 30, No. 5, May 1997, pp. 24–31

[Wall 05]
Daniel S. Wall, James McHale, Marsha Pomeroy-Huff. Case Study: Accelerating Process Improvement by Integrating the TSP and CMMI. December 2005. CMU/SEI-2005-SR-012

[Noopur 05]
Noopur Davis, Brice Erickson. Applying the TSP in Intuit. SEPG 2005, Seattle, WS

[Karina 05]
Karina Cedillo, Accelerating CMMI Implementation with PSP and TSP in a small organization, SEPG 2005.

[Humphrey 00]
Humphrey, W. The Personal Software Process (PSP). TECHNICAL REPORT CMU/SEI-2000-TR-022.

[SEI CMMI 00]
CMMI SE/SW/IPPD v1.02. Staged representation.CMU/SEI-2000-TR-030.

[Humphrey 98]
Humphrey, W. "The Software Quality Index," Software Quality Professional, 1, 1 (December 1998):. 8-18.

[Humphrey 96]
Humphrey, W.S., "Using a Defined and Measured Personal Software Process," IEEE Software, May 1996.

[Humphrey 98]
Humphrey, W.S.," Three Dimensions of Process Improvement, Part III: The Team Process", SEI, CROSSTALK The Journal of Defense Software Engineering, April 98

Authors

Adham Shaher Abbass
Innovation Deployment Lead
IBM Egypt, Cairo Technology Development Center (C-TDC)

Adham Shaher is a senior member in the process improvement team in IBM C-TDC where he leads the implementation of process improvement and innovation programs.

Before joining IBM, Adham was working as a software engineer where he led and practiced the execution of the various development lifecycle phases starting from requirements gathering till testing phase.

Adham holds a BS in Information from Faculty of Computers and Information, Cairo University, Egypt, and now preparing a master in "quality improvement". His research interests include software engineering, innovations management, and software quality management. He has contributed as a speaker in SEPG Europe 2008. He is also a Certified Software Quality Analyst from QAI Florida, and he has supported IBM C-TDC to obtain CMM Level 3 in 2004 and CMMI L5 in 2005.

Amr Kamel
SEI Authorized PSP Instructor and TSP Coach

Dr. Kamel has a strong background in the theoretical and practical aspects of software engineering. His theoretical background was developed and sharpened by acquiring two postgraduate degrees in the field and attending a series of professional training courses offered by prestigious training providers in the world. An extensive experience in the software industry has complemented this theoretical background to augment his credentials as a software engineering professional.

For over 20 years in the field, Dr. Kamel worked and provided consultations, locally, regionally and internationally, in software development, software architecture & design, software process improvement, quality assurance and testing. He participated as a consultant and ATM in a number of CMMI and ISO 15504 based process improvement efforts. He also led more process improvement programs based on TSP/PSP concepts.

Dr. Kamel has developed a number of programs and courses in software engineering and software quality assurance for academic degrees and for industry patrons; authored and co-authored a number of research papers in the field; and participated in, as an invited speaker, participated in and/or chaired many local, regional and international academic and professional gatherings.

Through his positions as a software process consultant at SECC, assistant professor at Cairo University as well as his internationally accredited professional certifications, Dr. Kamel has the unique chance of leading the industry and academia in the field while continuously prospering his professional knowledge and interpersonal skills.

TOWA's TSP Initiative: The Ambition to Succeed

Gerardo López, Yuri Ontibón, and Miguel Serrano

Introduction

In the last years, there has been an important national initiative in Mexico to promote the adoption of the Team Software Process (TSP) [3,5], in order to position the Mexican software industry as an international player. Such an initiative has been led by the "Instituto Tecnológico y de Estudios Superiores de Monterrey" (ITESM), in collaboration with the SEI, with the participation of leading companies, as well as the funding and support from Federal and State Governments through the ProSoft initiative [2,6]. The TSP initiative includes participants from Government, Academia, and Industry. The goal of the Mexican government is to foster growth and improvement of the software development and TI related services industry, from a revenue of USD $4 billion in 2007 [7] to USD $15 billion by the year 2013 [1]. The strategy takes into account several strengths of the Mexican industry, such as [7]:

- Geographical proximity to the USA, the main consumer of IT services in the world.
- A stable macroeconomic environment.
- World class infrastructure with competitive cost.
- Abundant human talent.
- Trade agreements which facilitate the exchange of services with many countries with preferential access to goods and services generated in Mexico.

The approach for this initiative is based on developing a world class industry that produces high quality software and services, and one of the main components of the strategy is to foster the adoption of the TSP in organizations.

TOWA is one of the organizations which are participating in the initiative. It is a small software developing company, with offices in Monterrey and Mexico City. Currently, TOWA has 180 employees, with the goal of achieving 3,000 employees over the next 6 years; as well as the purpose of delivering services in Mexico and overseas. In order to achieve this objective the company is relying on two fundamental factors: 1) a software process improvement strategy focused on creating high quality products, and 2) a first-class and experienced executive team that is focused on quality (process improvement) that leads the company to success.

TOWA has decided to adopt PSP and TSP as its basic process methodology in order to achieve the best standards of quality, and to help TOWA's growing pace with well established processes. In the last year, TOWA has signed a collaboration agreement with ITESM and entered the ProSoft program to support its TSP adoption. As of July 2008, TOWA has five PSP instructors and eight trained TSP coaches; and it expects to have sixteen TSP certified coaches as well as nine PSP instructors by the end of 2008. Also, as of July 2008, TOWA has twenty certified PSP developers, and the organization is training all developers, expecting to certify most of them.

The Need for Quality

The organization's executives strongly believe that quality is the key that will enable them to achieve their goals. With the same belief, in the past, its CEO created a software company that reached 6,000 employees in Mexico. They are aware of the importance in keeping best practices, commitment, and support while the company is growing. TOWA's executives consider themselves experienced "veterans" in the industry of software development and IT services. They have more than 30 years experience in developing applications on many platforms and technologies, (e.g., IBM, COBOL, CICS, DB2, IMS, etc.). At one time, they achieved high quality levels in software development by following best practices such as structured programming and sound software engineering principles.

However, over the years, quality levels have become an issue. Gerardo López (TOWA's CEO and founder) mentions: "As the time passed and we 'veterans' got far from the day-to-day practice, and when the technology 'improved' and it was possible to code in front of a monitor, when the technology became more powerful making graphic and web applications possible, when so many technologies came into play in a quick pace, and new young personnel joined the workforce, the good practices got lost. There is no doubt that we 'veterans' weren't able to transmit these good practices to the new generations, and we weren't able to adapt to the changing times and new technologies, and we didn't have the elements to convince the new developer generations about continuing and adapting the best practices to modern environments and technologies. Many of us 'veterans' evolved from the technical field to 'better management and executive positions', and we didn't find the way on how to adapt, evolve, and transfer the good practices to the new generations of engineers. We were 'busy'; the young newcomers were without experience, the technologies were changing very fast, and all of us together arrived to a situation that's just out of control".

The founders of TOWA created the organization with their focus on quality and capability to grow fast without losing the quality focal point. The challenge in TOWA relies on how to get the new generations (which believe they're working reasonably well) to understand how to apply the old "good practices" while developing with the new technologies. Some of these old "good practices" in TOWA were:

- A strong in-house requirements analysis methodology supported by a CASE tool.
- Quality focus based on product inspections performed by an experimented coach, as well as careful personal code and products reviews.
- Design and Coding standards (structured code and design) that facilitate the understanding of the implementers as well as better inspection of the code, and also supports a better use of the programming language.
- Implementation of "reusable models" that encapsulate the main expected functionality for the most common tasks. These "models" are reusable code that save time to programmers, make all the programs of the same kind (i.e., reports, master-detail screens, etc.) have the same structure, and allow the programmer to focus on the specific aspects of a program, instead of focusing on all the details.
- Provide expert coaching to the team members, especially on the production of Requirement Specification, Design, and Code.

TOWA believes the CMMI is a good model to help guide the improvement initiative, but does not solve the problem completely. As a reference model, it guides on "what" should be done, but it does not describe "how" things should be done. There are many questions, such as: Who is in charge of the correct design and building of software components? What measurements should be used? How to support the continuous process improvement? How to grow without losing the good practices? The answer to many of these questions was found in the TSP/PSP developed by Watts Humphrey at the SEI.

How do the new Humphrey ideas help with these preoccupations? For the organization PSP, they include three features that fortify themselves:

- To do a task (e.g., to develop a program), a well defined process is needed. The steps to create the product have to be organized as a mental process of how the product is being created, i.e., to develop a program these steps could be: 1) Understand completely and correctly what is required, 2) Design and build the program using artifacts adequate to the technology and the specific applicative area, and 3) Finish with a phase that resumes the learning acquired during the execution of the process (the Postmortem). To be considered a smart process it has to take into account the human nature of making mistakes (insert defects), and include review activities performed by the creator of the product, as well as inspections performed by at least one person different than the producer. It has been done in the past, before meeting the PSP, with excellent results by TOWA's 'veterans'.
- Collect data and measurements during the execution of the task. It allows for generating metrics of size, effort, and defects. The effort measure has been used by many organizations. Measuring size consistently is more complicated, and it requires precise operational definitions and execution. Also, it is different according to the task and the technology being used. Few people collect this information in a way that generates useful data. And lastly, count and classify defects, and identify when they have been inserted and/or eliminated, and how much it cost to eliminate them. It is relatively simple, but requires a lot of discipline (when somebody finds a defect, especially if it was inserted by them, human nature dictates they just want to fix it and forget about it). All these metrics can be analyzed individually or as a group; using some quantitative methods and statistics, it is possible to achieve conclusions and make assertive predictions about the results that will be obtained. The value of the metrics is that they aid us in making good estimations, planning, and managing projects adequately, as well as predicting future performance. In the past, we measured effort, but missed measuring size and defects consistently; we think it was one of the ingredients that were missing to perpetuate the work with "good practices". We see a great value on correctly executing those measuring practices.
- Besides using the metrics for estimation, planning, and prediction, they must be used, first of all, to ensure the correct execution of the established process. By analyzing the data, it is possible to intuit if the established process is being used correctly and – if necessary – to take the required corrective actions. On the other hand, by analyzing aggregated information, it is possible to identify areas of opportunity for improving the process (e.g., changing or simplifying the process, improving some of its artifacts, describing it better, etc.). Those practices are extensively supported in the information collected (e.g., metrics, lessons learned, etc.). If we had these practices in the past, we should've been able to perpetuate and improve them to adapt to changing times; especially when the technology evolved, we should have timely detected when and how these good practices should have evolved and tailored at the right time.

When we met Humphrey's ideas on PSP and TSP, it was immediately clear to us that they had great potential on getting the good practices to come back and continually improve them, as well as its implementation. It was also clear that it meant a titanic job, especially if the organization planned to grow from 180 to 3,000 professionals during the next 6 years. However, if we didn't find a solution, it would make growing impossible.

We found the TSP to be a natural evolution of PSP in team work, using a good management that includes an ordered process, considering and taking care of human resources as the most important component of the process, taking advantage of all of their skills, such as their capacity to be enthused, to be compromised, to learn and to act smartly, and the capacities that the group synergy can produce.

TSP Introduction Strategy

How did we begin to implement TSP? The first step was to commit ourselves with this process, firmly believing that it was the right path no matter how challenging it could be. We thought that if the solution to producing quality software at a reasonable cost and in a predictable way wasn't PSP/TSP, at least for sure the solution would have to be something really close to it.

To cultivate this commitment in ourselves, we take the following actions:

- Continuously talk about what we are going to do and what we are going to achieve.
- Train our teams constantly.
- Buy as many books as possible about the topic at hand, and motivate our people to read them.
- Discuss with our customers what we are going to do and what we are going to achieve.
- Commit to ourselves and to our customers about our goals for quality and PSP/TSP.
- Disseminate within other organizations and clients and even within the competition, the potential of these ideas. If we don't convince them, at least we will have convinced ourselves more.

Soon enough, it was clear to us that the resistance to change was huge; but instead of underestimating that difficulty, we decided to work harder and more committed.

We also think that once we have created a significant mass of "believers", that things are going to evolve more naturally and become easier. Nevertheless, the growing of the organization will make the task challenging. We know that growing erodes any set of good practices.

In our TSP/PSP implementation strategy, the first step was to take a great project that was about to start, "Orbita", a 55,000 hour project for Multipack, one of the largest Mexican courier/shipping companies, with over 4,000 employees. The project was the biggest we had at the time, and we wanted to take a big step hoping and aiming for an excellent result, in applying our former ideas and methodologies mixed with the TSP/PSP concepts. To do that:

- We assigned the project with a management team of four "believers", each one with a big responsibility according to their experience, history, skills, and motivation; and someone who would be willing to 'get their hands dirty' in order to achieve success. They got the following responsibilities:
  a. One manager was in charge of TSP implementation in the project; we had to start from scratch, we had no tool, and we created our own process, taking into account the strengths of past experiences and the PSP practices tailored to our needs.
  b. A second manager was in charge of the Software Requirements Specifications. In this area we had a very robust methodology that we have used for many years, and that is supported by a CASE tool built in-house. The tool in itself impressed everyone who has gotten to know it, but the really important thing was the underlying methodology. We created a process and tailored PSP for adapting to the requirements specification task. This process includes phases to conceptualize the requirement before gathering and writing the detail requirement (similar to design before code). We also included some features such as personal reviews, product inspections performed by an expert coach, definition of a product standard, definition of a size unit, and the tailoring of PSP metrics for the process, i.e., calculating the Yield, PQI, COQ, etc. specific for the analysis process.
  c. A third manager was in charge of software design, one of the most complicated tasks. We hadn't established a methodology that satisfied us completely, which left us with a lot of separate ideas. Fortunately, we had very clear ideas on what should be the end result of the component specification that integrates a software system. This is a task that many teams skip and they go from functional specifications to code. We also elaborated a specific PSP process to specify "construction tasks", using the same concepts we used to create the analysis process; also, the construction phase includes a detailed design process. This manager also got the responsibility of adapting the process that the company already had (defined to cover the Process Areas of CMMI Level 3), to work together with the PSP/TSP process.

d. Lastly, a fourth manager was in charge of the Code. We tailored our own PSP process, taking the Humphrey principles and complementing them with the good ideas we had used in the past.

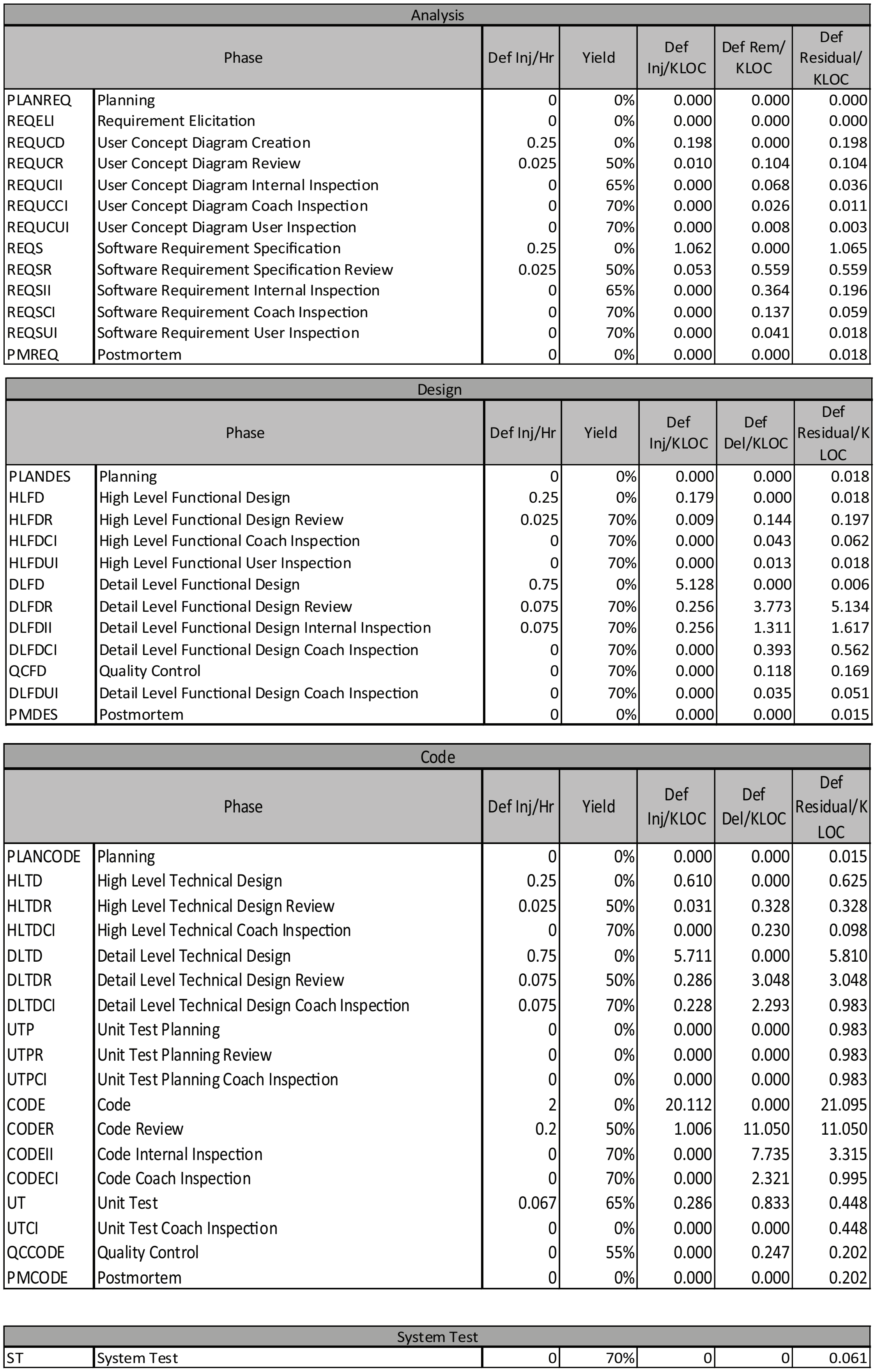
- The management team for the project is extraordinary, in our concept of the “Dream Team”. In fact, we integrated this team as part of our large project proposal, and it allowed us to win the project, competing with some of the most prestigious software development companies in Mexico, including two which are CMMI Level 5 companies. We knew that a project of these characteristics required such a team. This project is expected to be an important advance in TOWA’s history, and this team would be responsible of implementing these practices in the rest of the organization.
- As each team was integrated, the members were trained in the practices required, including the PSP aspects adequate to the project activity that they performed. There are a total of four teams: one for analysis, another for design, and two others for code. With these teams in parallel, the project is using a superimposed waterfall lifecycle.
- Since we did not have an organizational tool, we used the PSP tool (from the SEI) to allow people to capture their individual data and we configured the tool with our process. We created a program that consolidates the individual information in a central database; on that database, we generate status and quality information.

Our approach was strongly supported with PSP and TSP, but we needed to complement it with our “good coding practices” of the past. For that reason, we created another team to develop “models” with the same concepts that we had already used before. Accomplishing a good understanding between the “past good practices” and the unplanned practices of most of today’s Java programmers, was a big challenge. As we advanced, we achieved results that we consider of great value. For the “veterans”, the Java capability to build reusable code was amazing; for the young java expert programmers, too. It is a fact that the “experts” knew Java, but they haven’t actually “designed” with Java; they just thought that was what they were doing when they were coding. This task was developed by four people; the work they have done is exceptional, and it is the foundation to create much better things in the future.

Current Status

The project is running now; all 35 team members are very enthusiastic and really committed. For all of them, what they’re doing is new, and the learning is intensive every day. We have an ambitious quality goal to deliver the code to system test with less than 0.2 Def/KLOC, if we achieve it, considering a 70% system test yield, we will achieve 0.06 Def/KLOC when we deliver the product to the client. It would mean a 5 sigma process (6 Sigma = 3.4 Def/million = 0.034 Def/KLOC, 5 Sigma = 233 Def/million = 0.233 Def/KLOC). To define the goal, we developed a quality plan for the whole project (Figure 1).

Currently, we are working on the Analysis and Design phases, and we are close to starting on the Coding phase. Since the project is still running, we don’t have any conclusive information yet. We will now describe some of the measurements and information we have analyzed.



Analysis						
Phase		Def Inj/Hr	Yield	Def Inj/KLOC	Def Rem/ KLOC	Def Residual/ KLOC
PLANREQ	Planning	0	0%	0.000	0.000	0.000
REQELI	Requirement Elicitation	0	0%	0.000	0.000	0.000
REQUCD	User Concept Diagram Creation	0.25	0%	0.198	0.000	0.198
REQUCR	User Concept Diagram Review	0.025	50%	0.010	0.104	0.104
REQUCII	User Concept Diagram Internal Inspection	0	65%	0.000	0.068	0.036
REQUCCI	User Concept Diagram Coach Inspection	0	70%	0.000	0.026	0.011
REQUCUI	User Concept Diagram User Inspection	0	70%	0.000	0.008	0.003
REQS	Software Requirement Specification	0.25	0%	1.062	0.000	1.065
REQSR	Software Requirement Specification Review	0.025	50%	0.053	0.559	0.559
REQSII	Software Requirement Internal Inspection	0	65%	0.000	0.364	0.196
REQSCI	Software Requirement Coach Inspection	0	70%	0.000	0.137	0.059
REQSUI	Software Requirement User Inspection	0	70%	0.000	0.041	0.018
PMREQ	Postmortem	0	0%	0.000	0.000	0.018

Design						
Phase		Def Inj/Hr	Yield	Def Inj/KLOC	Def Del/KLOC	Def Residual/K LOC
PLANDES	Planning	0	0%	0.000	0.000	0.018
HLFD	High Level Functional Design	0.25	0%	0.179	0.000	0.018
HLFDR	High Level Functional Design Review	0.025	70%	0.009	0.144	0.197
HLFDCI	High Level Functional Coach Inspection	0	70%	0.000	0.043	0.062
HLFDUI	High Level Functional User Inspection	0	70%	0.000	0.013	0.018
DLFD	Detail Level Functional Design	0.75	0%	5.128	0.000	0.006
DLFDR	Detail Level Functional Design Review	0.075	70%	0.256	3.773	5.134
DLFDII	Detail Level Functional Design Internal Inspection	0.075	70%	0.256	1.311	1.617
DLFDCI	Detail Level Functional Design Coach Inspection	0	70%	0.000	0.393	0.562
QCFD	Quality Control	0	70%	0.000	0.118	0.169
DLFDUI	Detail Level Functional Design Coach Inspection	0	70%	0.000	0.035	0.051
PMDES	Postmortem	0	0%	0.000	0.000	0.015

Code						
Phase		Def Inj/Hr	Yield	Def Inj/KLOC	Def Del/KLOC	Def Residual/K LOC
PLANCODE	Planning	0	0%	0.000	0.000	0.015
HLTD	High Level Technical Design	0.25	0%	0.610	0.000	0.625
HLTDR	High Level Technical Design Review	0.025	50%	0.031	0.328	0.328
HLTDCI	High Level Technical Coach Inspection	0	70%	0.000	0.230	0.098
DLTD	Detail Level Technical Design	0.75	0%	5.711	0.000	5.810
DLTDR	Detail Level Technical Design Review	0.075	50%	0.286	3.048	3.048
DLTDCI	Detail Level Technical Design Coach Inspection	0.075	70%	0.228	2.293	0.983
UTP	Unit Test Planning	0	0%	0.000	0.000	0.983
UTPR	Unit Test Planning Review	0	0%	0.000	0.000	0.983
UTPCI	Unit Test Planning Coach Inspection	0	0%	0.000	0.000	0.983
CODE	Code	2	0%	20.112	0.000	21.095
CODER	Code Review	0.2	50%	1.006	11.050	11.050
CODEII	Code Internal Inspection	0	70%	0.000	7.735	3.315
CODECI	Code Coach Inspection	0	70%	0.000	2.321	0.995
UT	Unit Test	0.067	65%	0.286	0.833	0.448
UTCI	Unit Test Coach Inspection	0	0%	0.000	0.000	0.448
QCCODE	Quality Control	0	55%	0.000	0.247	0.202
PMCODE	Postmortem	0	0%	0.000	0.000	0.202

System Test						
ST	System Test	0	70%	0	0	0.061

Figure 1. Quality Plan

Defect Information

We have analyzed defect information derived from analysis and design. The standard for "Defect type" that we are using for analysis and design documents is shown in Figure 2 and explained in [8].

Type	Name	Description
10	Ambiguous Statement	The statement can be interpreted to mean several different things. Two or more statements or descriptions conflict or contradict each other.
20	Incomplete Item	The statement or description does not seem to address the aspects of the situation it attempts to describe. The statement or description that must be included in the document is missing.
30	Incorrect Item	The statement or description is incorrect.
60	Confusing Items	The statement or description confuses the reader. The statement is not clear.
70	Redundant Items	The statement repeats another statement and detracts from clarity rather than adding to it.
80	Illogical Item	The statement does not make sense in reference to other statements within the same document or other documents to which it refers.
90	Non-verifiable Item	The statement (usually a requirement or functional description) cannot be verified by any reasonable testing method.
100	Unachievable Item	The statement (usually a requirement or functional description) cannot be true in the reasonable lifetime of the product.
110	Applicable Stdrds Not met	Internal or industry standards for the document in question are not met in accordance with organization policy.
120	Not Traceable	Items cannot be traced to the appropriate previous or subsequent documents.
130	User Definition Change	User change of the requirement or definition given for a product and its implied rework.

Figure 2. Standard for Defect Types

Some of the information that we have analyzed is shown in Figures 3 and 4.

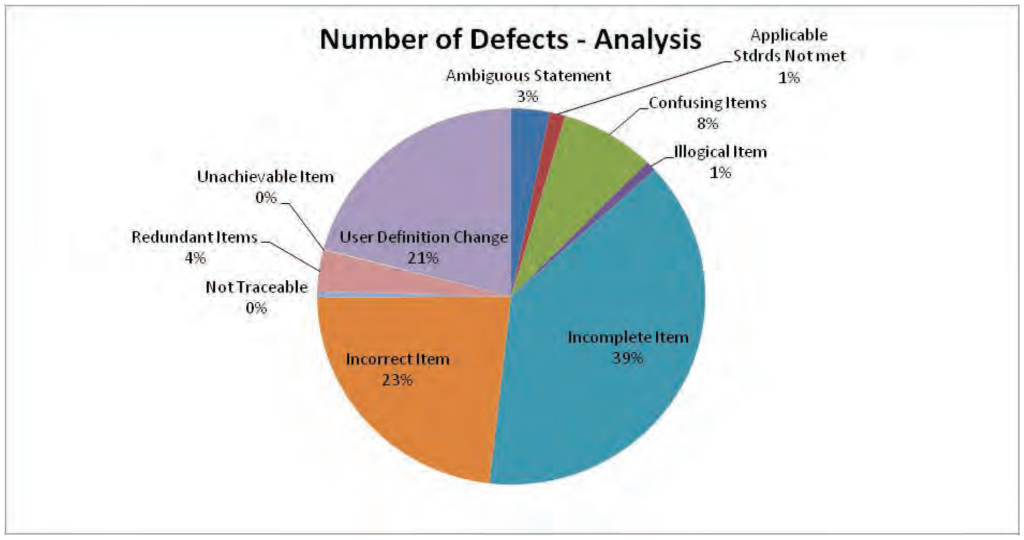


Figure 3. Number of Defects - Analysis

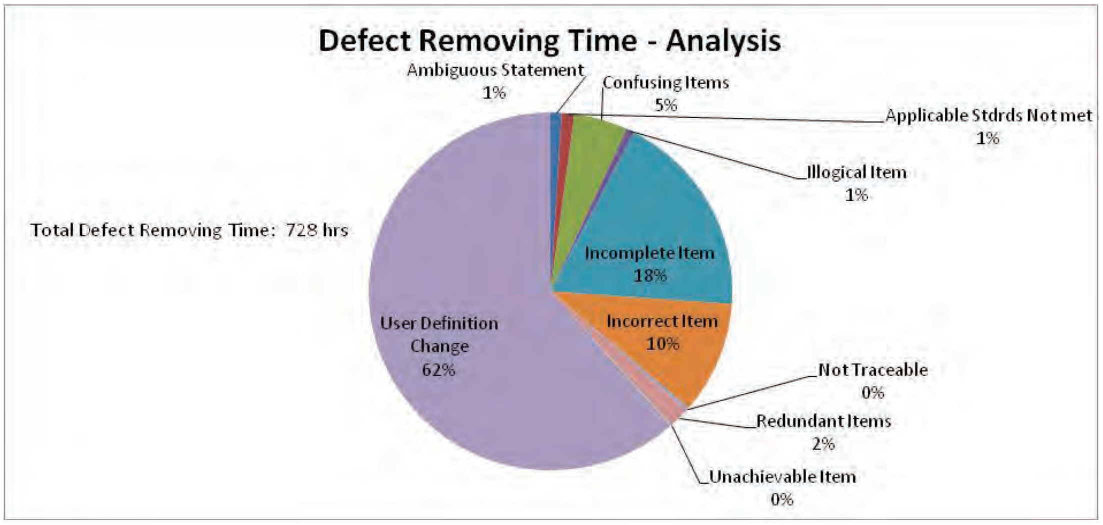


Figure 4. Defect Removing Time - Analysis

In the Analysis phase, we focus on the "Incomplete Item" and "Incorrect Item" defect types, adding specific items to the checklist; at the beginning of the project, we had only basic checklists for the products. Also, we show the information to the analysis team and decided together on implementing a format for recording the information collected during the requirements elicitation meetings.

For the "User Definition Change" type of defect, we have an agreement with our client that they can change the requirements during the analysis process without implying a change of scope. Only when the requirements document is finally signed (at the end of the analysis process), any change of requirement generates a change of scope. The client's first reaction to this information was to justify it with the contract; after we explained that our main interest was to identify risks that could delay the project and taking into consideration that the customer has the same interest we do of not delaying the project, they reacted positively looking for ways to reduce the changing of definitions. Showing the project's information to the customers has helped them realize the impact the project has on cost and schedule for any requirement change, and made them take actions to better organize their ideas before meeting with the analysis team.

We now show a similar analysis for the design phase in Figures 5 and 6.

The information for Design was similar to that of Analysis. However, in this case – in addition to updating the checklist – we decided to include a new phase to our Analysis process. The initial process didn't have a phase where the Analyst formally delivers and explains the product to the designer; in the initial process, the designer took the product from the CASE tool. The new phase to the analysis process was called Requirements Delivery". We even found the CASE tool to be a really good tool; the context is important, as well as an explanation from the analyst to the designer.

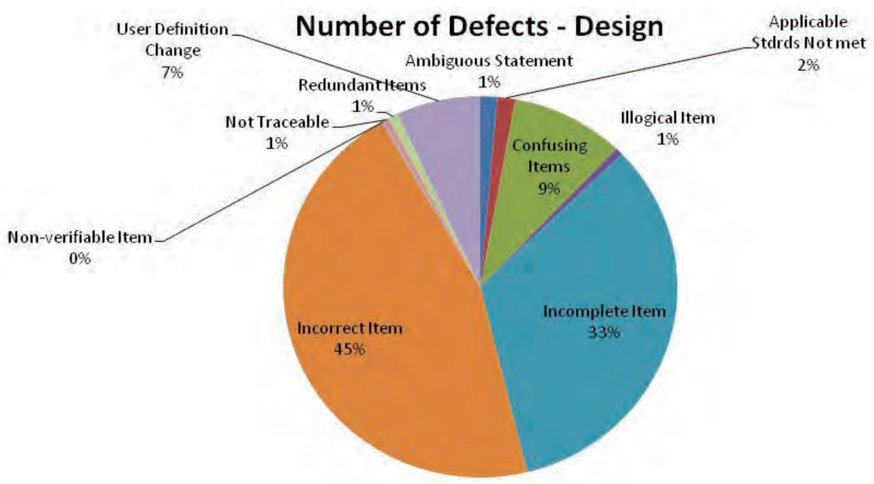
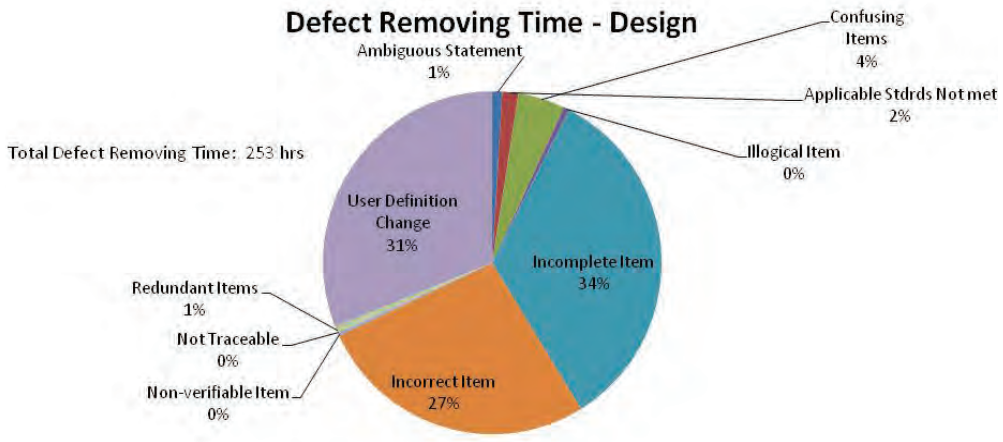
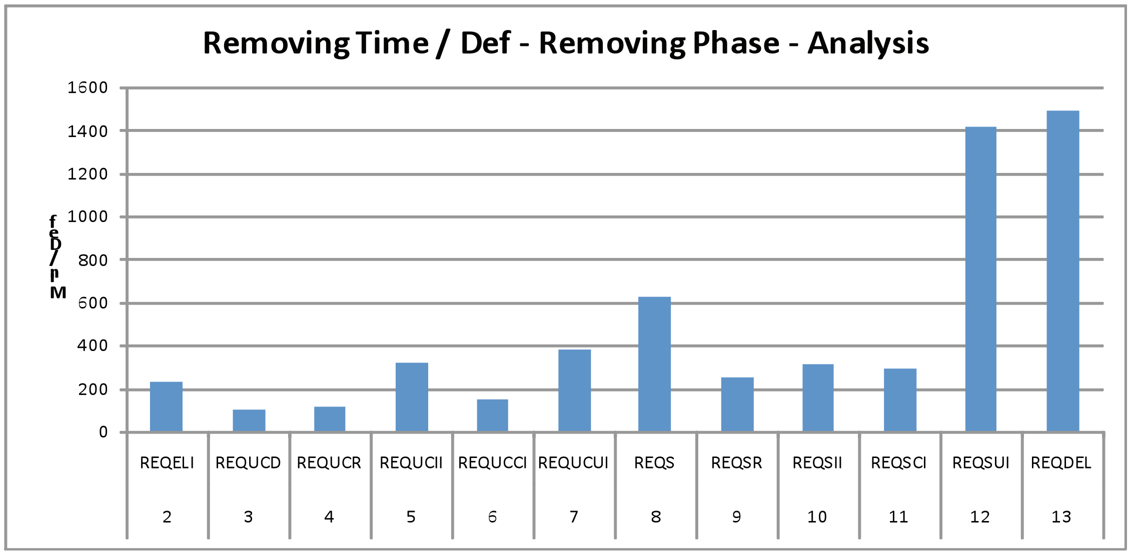


Figure 5. Number of Defects - Design

Figure 6. Defect Removing Time - Design



On Figures 7 and 8, we analyze the time invested on removing defects according to the phase the defect has been removed from (Minutes/Def).



Phase	
PLANREQ	Planning
REQELI	Requirement Elicitation
REQUCD	User Concept Diagram Creation
REQUCR	User Concept Diagram Review
REQUCII	User Concept Diagram Internal Inspection
REQUCCI	User Concept Diagram Coach Inspection
REQUCUI	User Concept Diagram User Inspection
REQS	Software Requirement Specification
REQSR	Software Requirement Specification Review
REQSII	Software Requirement Internal Inspection
REQSCI	Software Requirement Coach Inspection
REQSUI	Software Requirement User Inspection
REQDEL	Requirements Delivery
PMREQ	Postmortem

Figure 7. Defect Removing by phase - Analysis

Phase	
PLANDES	Planning
HLFD	High Level Functional Design Creation
HLFDR	High Level Functional Design Review
HLFDCI	High Level Functional Design Coach Inspection
HLFDUI	High Level Functional Design User Inspection
DLFD	Detail Level Functional Design Creation
DLFDR	Detail Level Functional Design Review
DLFDII	Detail Level Functional Design Internal Inspection
DLFDCI	Detail Level Functional Design Coach Inspection
QCFD	Quality Control
DLFDUI	Detail Level Functional Design User Inspection
DESDEL	Design Delivery
PMDES	Postmortem

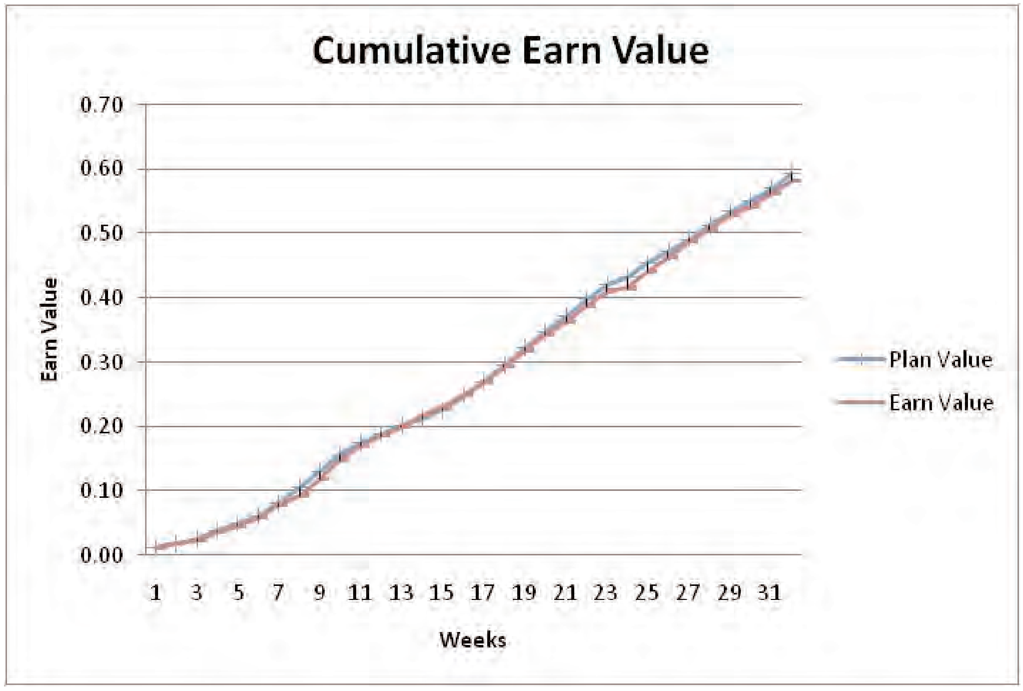
Figure 8. Defect Removing by phase - Design

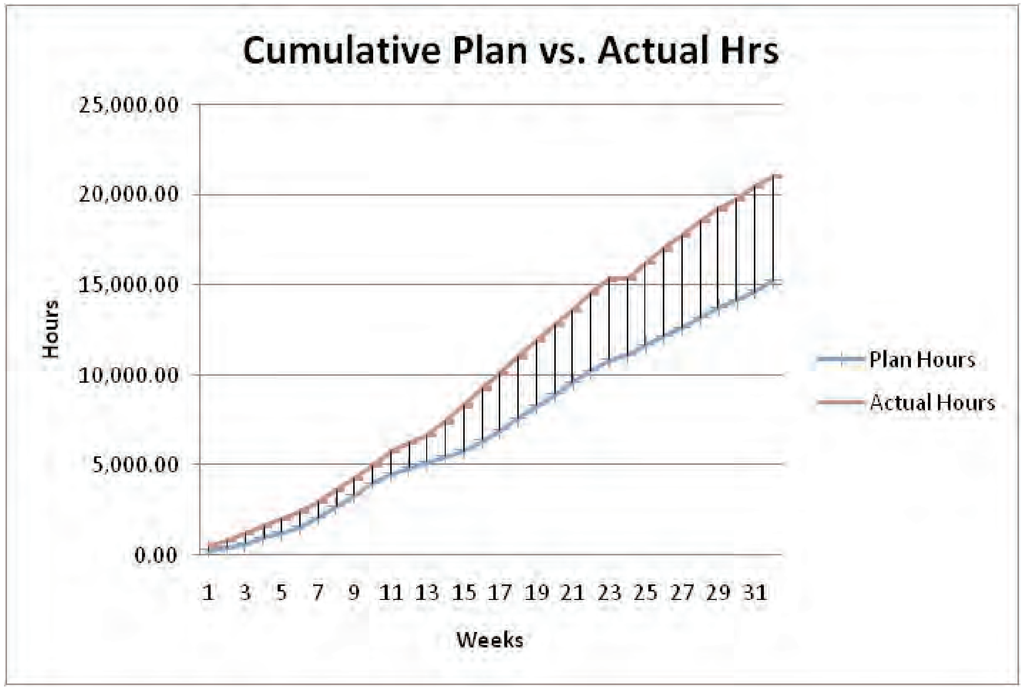
The results are consistent with the theory: removing defects costs more the later they are removed regarding the product's life cycle. In this case, it is considerably expensive to eliminate defects later in the process. This information is useful for getting the team to pay special attention to removing defects early in the process. This is their information; it's neither theoretic concepts nor someone else's information. That's why they believe in it.

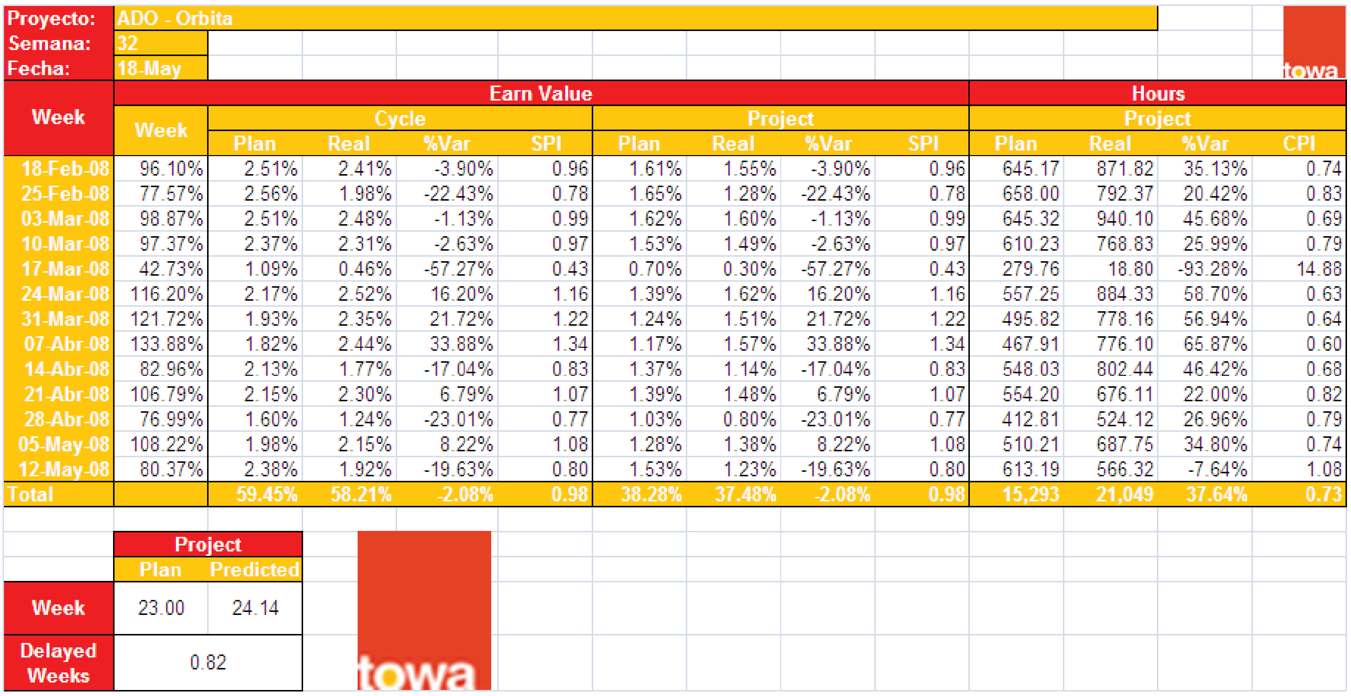
Status Information

We are also generating status information based on earn value. Some of the reports are shown in Figure 9.

Figure 9. Earn-value







Proyecto:	ADO - Orbita												
Semana:	32												
Fecha:	18-May												
Week	Earn Value									Hours			
	Week	Cycle				Project				Project			
		Plan	Real	%Var	SPI	Plan	Real	%Var	SPI	Plan	Real	%Var	CPI
18-Feb-08	96.10%	2.51%	2.41%	-3.90%	0.96	1.61%	1.55%	-3.90%	0.96	645.17	871.82	35.13%	0.74
25-Feb-08	77.57%	2.56%	1.98%	-22.43%	0.78	1.65%	1.28%	-22.43%	0.78	658.00	792.37	20.42%	0.83
03-Mar-08	98.87%	2.51%	2.48%	-1.13%	0.99	1.62%	1.60%	-1.13%	0.99	645.32	940.10	45.68%	0.69
10-Mar-08	97.37%	2.37%	2.31%	-2.63%	0.97	1.53%	1.49%	-2.63%	0.97	610.23	768.83	25.99%	0.79
17-Mar-08	42.73%	1.09%	0.46%	-57.27%	0.43	0.70%	0.30%	-57.27%	0.43	279.76	18.80	-93.28%	14.88
24-Mar-08	116.20%	2.17%	2.52%	16.20%	1.16	1.39%	1.62%	16.20%	1.16	557.25	884.33	58.70%	0.63
31-Mar-08	121.72%	1.93%	2.35%	21.72%	1.22	1.24%	1.51%	21.72%	1.22	495.82	778.16	56.94%	0.64
07-Abr-08	133.88%	1.82%	2.44%	33.88%	1.34	1.17%	1.57%	33.88%	1.34	467.91	776.10	65.87%	0.60
14-Abr-08	82.96%	2.13%	1.77%	-17.04%	0.83	1.37%	1.14%	-17.04%	0.83	548.03	802.44	46.42%	0.68
21-Abr-08	106.79%	2.15%	2.30%	6.79%	1.07	1.39%	1.48%	6.79%	1.07	554.20	676.11	22.00%	0.82
28-Abr-08	76.99%	1.60%	1.24%	-23.01%	0.77	1.03%	0.80%	-23.01%	0.77	412.81	524.12	26.96%	0.79
05-May-08	108.22%	1.98%	2.15%	8.22%	1.08	1.28%	1.38%	8.22%	1.08	510.21	687.75	34.80%	0.74
12-May-08	80.37%	2.38%	1.92%	-19.63%	0.80	1.53%	1.23%	-19.63%	0.80	613.19	566.32	-7.64%	1.08
Total		59.45%	58.21%	-2.08%	0.98	38.28%	37.48%	-2.08%	0.98	15,293	21,049	37.64%	0.73

	Project	
	Plan	Predicted
Week	23.00	24.14
Delayed Weeks	0.82	

The information shows that the project has been kept on schedule; it has been necessary to invest additional time, but the information analyzed on time has allowed us to take corrective actions in an opportune fashion.

Next Steps

We have very challenging goals. On the one hand, we have the goal to grow from 180 to 3,000 employees over the next 6 years (half of the time it took our CEO to achieve this goal in his former company). We also have the commitment to achieve the highest quality levels possible (deliver our clients with products on time, under cost, and free of defects). Generating products with quality will be the fuel that allows our organization to grow continuously during the next years. When we get to the middle goal of 500 members, we will focus on the USA market; at that moment, our process should be mature and capable and our products high quality. Our quality should be an outstanding offering to the USA market; somehow, our capability will be 'certified' by the Software Engineering Institute (as the SEI is working to create a TSP Certification process for companies).

We have to train enough PSP instructors and TSP coaches. We don't want PSP instructors and TSP coaches dedicated exclusively to train and coach; we want them actively participating in the projects and complementing their activities with teaching PSP and coaching teams. PSP/TSP should become a strategic tool for supporting our growth and making us able to compete.

We also have to develop TSP tools integrated to our administrative, operative, and financial systems. We think it is important that these tools help us evolve stably and sustainably. We know that everything will be changing continuously as our ideas mature, applying them in different scenarios and growing rapidly. Our systems should be a support and never a restriction or obstacle.

Working together with ITESM and the Mexican government, we are committed to promoting PSP and TSP and making that Mexican industry earn a distinction for its quality levels. We are looking for cooperating schemes that allow us to learn fast together. Many of the main universities in the country are taking the PSP/TSP initiative seriously, and training teachers that would later train software engineering students in their respective universities.

Our goal is to give Mexico worldwide recognition for achieving the best software quality levels; the task is vast, and there is a lot of work to do, but the path will be filled with satisfaction.

References

[1] Alba, S., "PROSOFT Como Política Pública", Secretaria De Economia Mexico - La Marcha Del ProSoft, 2006.Available at: http://www.politicadigital.com.mx/IMG/pdf/SuplementoProsoft.pdf.

[2] Dicon, C., "Mexican Advocates See TSP As a Way to Establish Quality Reputation", SEI News, 2007. Available at: http://www.sei.cmu.edu/news-at-sei/features/2007/06/01-feature-2007-06.html..

[3] Humphrey, W., PSP A Self Improvement Process for Software Engineers, Addison-Wesley, 2005.

[4] Humphrey, W., TSP Coaching Development Teams, Addison Wesley, 2006.

[5] Humphrey, W., TSP Leading a Development Team, Addison Wesley, 2006.

[6] Miller, P., "The Mexican TSP Initiative: Positioning the Mexican Software Industry Through TSP/PSP", SEI News, 2007.Available at: http://www.sei.cmu.edu/news-at-sei /features/2007/01/01-feature-2007-01.html

[7] Prosoft Secretaria de Economia Mexico, "PROSOFT 2.0 - Programa De Desarrollo Del Sector De - Servicios De Tecnologías De Información", Secretaria De Economia Mexico - La Marcha Del ProSoft, 2008. Available at: http://www.cysp.com.mx/Ima/Amiti /Documentos%20Descargables /08_03_agenda_prosoft_2.pdf

[8] IEEE Std 1044-1993. IEEE Standard Classification for Software Anomalies, 1993.

Authors

Yuri Ontibón, Corporate VP of TSP-PSP Strategy (Authorized PSP Instructor, PSP Developer, PMP)
TOWA
Yuri Ontibón is Corporate VP of the TSP-PSP Strategy at TOWA. Before joining TOWA, Ontibón was IT Director and consultant for several companies in Latin America. Ontibón received a Bachelor's degree in Systems Engineer from University Piloto in Colombia, and he also received a graduate degree as Specialist in Business from El Rosario University in Colombia as well. He is PSP Developer, PSP Authorized Instructor and TSP Coach Candidate by the SEI. He is also PMP by the PMI.

Mr. Ontibón has over 18 years of experience in Software Development and Project management.

Email: yuri.ontibon@towasoftware.com

Miguel Serrano, Director (Certified SCAMPI High Maturity Lead Appraiser, Trained PSP Instructor and Trained TSP Coach, PSM Instructor)
MS SPI Solutions
Miguel Serrano is Director and Principal Consultant at "MS SPI Solutions". He is also Adjunct Researcher in Software Engineering at the Center for Research in Mathematics (CIMAT) Mexico. Miguel Serrano is Certified SCAMPI High Maturity Lead Appraiser (CMMI Levels 2-5, Classes A, B, and C) by the Software Engineering Institute (SEI). In addition, he is Trained PSP Instructor and Trained TSP Coach, and authorized PSM Instructor. Miguel Serrano received a Masters Degree in Business - ISDS (USA), a Master in System Science (USA), and a Ph.D. in Computer Science (USA).

Email : Miguel.Serrano@msspisolutions.com

Gerardo López, President & CEO
TOWA
Gerardo López founded TOWA in 2004 with the goal of becoming the most successful and largest Latin-American Software Developer and AMS provider for the NAFTA market. Before founding TOWA, López founded Softtek in 1982 and lead its development towards being the largest Latin American company in the Software Development sector. López received a Bachelor's degree in Electrical Engineer from ITESM in Mexico and spent two years working for a Masters Degree in Computer Sciences at the University of Texas at Austin. He also received a graduate degree from IPADE (Business School) in Mexico.

Mr. López has over 34 years of experience in Software Development as an Engineer, Entrepreneur and Businessman.

Email: Gerardo.Lopez@TowaSoftware.com

Uses of Monte Carlo Simulation for TSP^{SM} Teams

Robert Stoddard, Software Engineering Institute
David Tuma, Tuma Solutions, Inc.
James Van Buren, The Charles Stark Draper Laboratory
David R. Webb, 309th Software Maintenance Group, Hill Air Force Base

1.1 Abstract

As noted in previous reports[1,2], the Team Software Process^{SM} (TSP) as it is currently practiced lacks the application of the statistical process control techniques required by the Capability Maturity Model Integration® (CMMI) Levels 4 and 5. Monte Carlo Simulation holds great promise in filling these gaps for TSP. It could be used to perform such analyses as (1) the uncertainty of a team finishing a particular project plan by a specified date, or (2) the forecast date a team may commit to within a specified range of certainty (such as 75% or 90%), or (3) the range of expected defect escapes from a team's development activities. This would more directly support the requirements of the CMMI's high maturity process areas by extending TSP to provide ranges in a statistically recognized way to TSP's planning, plan monitoring, and quality management activities, which currently use only provide point estimates. Within the TSP planning and plan monitoring framework this technique can be applied during both the launch process and during project execution. Since TSP provides the process models, all that is needed is tool support, historical data, and an understanding of the distributions inherent in that historical data.

This paper provides a brief theoretical overview of the Monte Carlo analysis technique, including its statistical limitations, reviews the key TSP process performance models (project planning and quality management) from a Monte Carlo perspective, and discusses the historical data available to a TSP project that can be used to support Monte Carlo analysis of those TSP process performance models. It then explains how Monte Carlo is being used by our TSP teams at Hill AFB, the insights and benefits the teams have received from the use of Monte Carlo, and why we believe this TSP implementation of Monte Carlo fulfills some of our organization's CMMI maturity level 4 and 5 process requirements. Finally, it discusses the next steps in applying Monte Carlo to TSP in general.

1.2 Overview

In July 2006, the 309th Software Maintenance Group at Hill Air Force Base, Utah was appraised at a CMMI Maturity Level 5. One focus project had been using the Team Software Process (TSP) since 2001. TSP is generally considered a "Level 5 process;" however, during the preparation for the assessment, it became obvious to the team that even the stringent process and data analysis requirements of the TSP did not completely address CMMI requirements for several process areas. The team was able to work within the TSP structure to successfully address these issues[2]. Since the time of the assessment, the new CMMI version 1.2 has changed the interpretations for several of the high maturity process areas. Somewhat paradoxically, the chief problem for TSP teams is their implementation of the Quantitative Project Management (QPM) Process Area (PA). The reason for this has to do with the understanding and application of process and subprocess variability and process performance models which, while addressed in the Person Software Process (PSP)^{SM} course, are neither addressed nor utilized in the current TSP tool suite. The purpose of this paper is to describe our approach to fulfilling the QPM PA requirements while adhering to basic TSP principals.

There are many possible approaches to meeting the CMMI high maturity requirements. We chose to use the TSP principals we follow to derive the desired quality attributes of our solution. These attributes can then be used to measure if our high maturity implementation remains true to TSP ideals. The TSP principals[3] are listed in Table 1. The quality attributes are listed in Table 2. The gist of these attributes is (1) that the team should remain true to its TSP principals, (2) that any process changes should be made with the goal of helping the team achieve its business goals, and (3) that management overhead related to implementing these high maturity processes should be minimal.

Superior work meets business needs
Superior products meet customer needs and desires on time with quality
It is faster and cheaper to build quality products
Those that do the work own the process and plans
Motivated teams do the best work
Quality is managed (therefore it must be measured)
Improvement in work practices is an ongoing responsibility
The capability of processes and technologies is quantitatively understood and used to guide planning and plan execution
Teams use coaching and mentoring support to help them execute their projects and grow over time

Table 1: TSP Principals

Be useful for teams during project execution (more so than during project postmortem)
Not require a great deal of overhead
Not require a great deal of new training
Focus on a typical TSP team's needs. This probably means helping teams achieve their quality and schedule goals
Stress quality, because as PSP teaches: from quality comes schedule and cost
Be consistent in theory with PSP and TSP training and ideally should match the theories stressed during PSP and TSP training

Table 2: Quality Attributes of Desired Solution

The basic strategy examined in this paper is to leverage the Monte Carlo technique to add an understanding of the inherent variance to a team's TSP schedule, cost, and quality plans. The team then manages the *likelihood* of meeting its project goals rather than managing to the *point solution* as currently practiced. By defining the TSP scope as schedule, cost, and, quality plans, we also define and limit which subprocesses are of interest for statistical management.

1.3 Monte Carlo Background

The Monte Carlo Method is any technique using random numbers and probability distributions to solve problems[4,5]. This method belongs to a class of techniques that compute confidence (or prediction) intervals around estimates. The calculation of prediction intervals in PSP's PROBE methods A and B also belong to this class. Monte Carlo uses the brute force of computational power to overcome situations where solving a problem analytically would be difficult. Monte Carlo Simulation iteratively uses the Monte Carlo Method many hundreds or even thousands of times to determine an expected solution. Monte Carlo was first extensively studied during the Manhattan project, where it was used to model neutron behavior, where the probabilistic properties of individual neutrons were well known, but the collective properties of many neutrons constrained by differently shaped containers could not be analytically determined[6].

The basic approach of Monte Carlo is captured in Table 3 and is demonstrated in the section 1.3.1

Step #	Description
1	Create a parametric model
2	Generate random inputs
3	Evaluate the model and store the results
4	Repeat steps 2 and 3 (x-1) more times
5	Analyze the results of the x runs

Table 3: Monte Carlo Steps

1.3.1 PSP Example

As an example of applying this approach we will use the TSP quality plan model to assess and compare the defects removed during test for a PSP2.1 process execution for an individual who has completed PSP student training versus and an individual who is an experienced programmer but who is not using PSP quality practices. Assume the plan is that described in Table 4.

Process Phase	Plan Time (min)
Plan	15
Design	60
Design Review	30
Code	60
Code Review	30
Compile	5
Test	30
Postmortem	10

Table 4: Hypothetical PSP 2.1 Plan

The first step of Monte Carlo is to create the parametric model. Table 5 and Equation 1 capture the example parametric model. In this example, we use a common PSP and TSP quality model that describes certain process phases as injecting defects as a function of time in phase and other process phases as removing defects as a function of process yield[7]. We will assume that the Design and Code injection rates are log normally distributed and that all other distributions are normal. We will use the TSP historical data as the basis for the quality plan[8]. Note that the historical data typically describes only the mean of the distribution; however, for purposes of this example we will assume the standard deviations of each of the distributions as show in Table 5. Furthermore, the historical data is only for trained engineers. It is known that design defect injection rate declines 8.3% and the code defect injection rate 34.6% during the PSP course[9]. We will assume that experienced engineers untrained in PSP thus have a slightly higher defect injection rates and have essentially 0% yields during personal reviews.

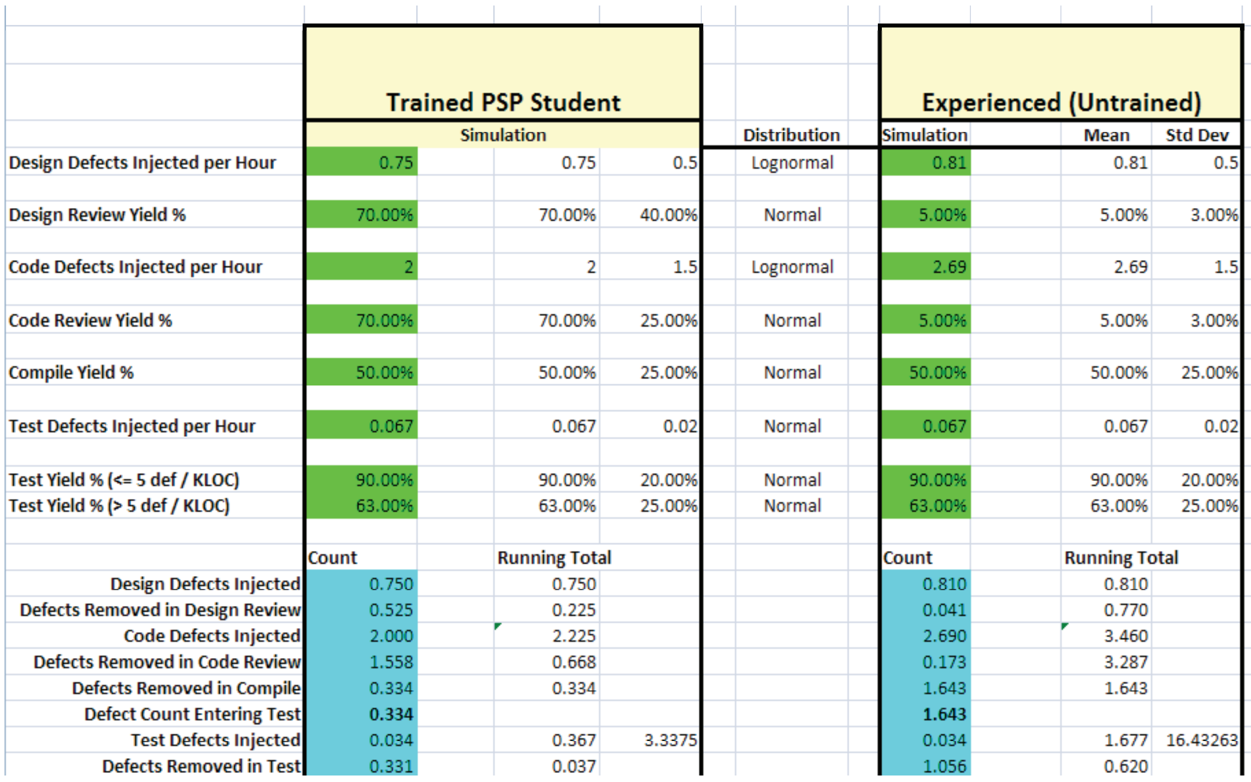
		PSP Trained Student		Experienced Developer	
Defect Injection Rate (defects / hour) per Phase	Distribution	Mean	Standard Dev	Mean	Standard Dev
Design (D)	log normal	0.750	0.50	0.810	0.50
Code (C)	log normal	2.000	1.50	2.690	1.50
Test (T)	normal	0.067	0.02	0.067	0.02
Phase Yield (defects removed / defects present)	Distribution	Mean	Standard Dev	Mean	Standard Dev
Design Review (DR)	normal	70%	40%	5%	3%
Code Review (CR)	normal	70%	25%	5%	3%
Compile (CMP)	normal	50%	25%	50%	25%
Test (<= 5 defects / KLOC)	normal	90%	20%	90%	20%
Test (>= 5 defects / KLOC)	normal	63%	25%	63%	25%

Table 5: Hypothetical Historical Data

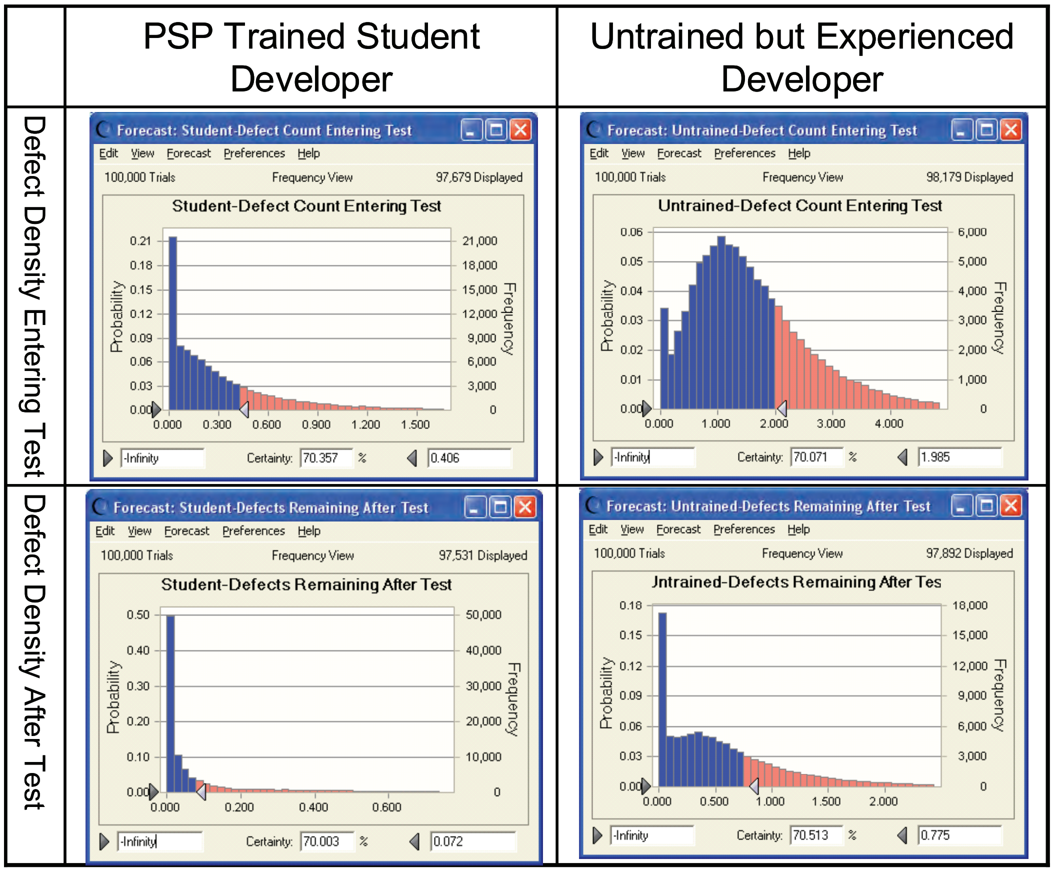
$$Defects_T = Yield_T \times (1-Yield_{CMP}) \times (1-Yield_{CR}) \times (Defects_C + (1-Yield_{DR}) * Defects_D)$$

Equation 1: Calculation for Defects Present when Test Begins

The next steps are to run the simulation, store the results of the random values of the uncertain factors, such as defect injection and removal, and then to analyze the results identifying the 70% baseline values. We implemented the parametric model in Excel using an add-on software package called Crystal Ball (see Figure 1). Figure 2 and Figure 3 are the outputs of 100,000 simulation runs for the two hypothetical students, of the defect count entering test and the defects remaining after test.



	Trained PSP Student				Experienced (Untrained)		
	Simulation			Distribution	Simulation	Mean	Std Dev
Design Defects Injected per Hour	0.75	0.75	0.5	Lognormal	0.81	0.81	0.5
Design Review Yield %	70.00%	70.00%	40.00%	Normal	5.00%	5.00%	3.00%
Code Defects Injected per Hour	2	2	1.5	Lognormal	2.69	2.69	1.5
Code Review Yield %	70.00%	70.00%	25.00%	Normal	5.00%	5.00%	3.00%
Compile Yield %	50.00%	50.00%	25.00%	Normal	50.00%	50.00%	25.00%
Test Defects Injected per Hour	0.067	0.067	0.02	Normal	0.067	0.067	0.02
Test Yield % (<= 5 def / KLOC)	90.00%	90.00%	20.00%	Normal	90.00%	90.00%	20.00%
Test Yield % (> 5 def / KLOC)	63.00%	63.00%	25.00%	Normal	63.00%	63.00%	25.00%
	Count	Running Total			Count	Running Total	
Design Defects Injected	0.750	0.750			0.810	0.810	
Defects Removed in Design Review	0.525	0.225			0.041	0.770	
Code Defects Injected	2.000	2.225			2.690	3.460	
Defects Removed in Code Review	1.558	0.668			0.173	3.287	
Defects Removed in Compile	0.334	0.334			1.643	1.643	
Defect Count Entering Test	**0.334**				**1.643**		
Test Defects Injected	0.034	0.367	3.3375		0.034	1.677	16.43263
Defects Removed in Test	0.331	0.037			1.056	0.620	



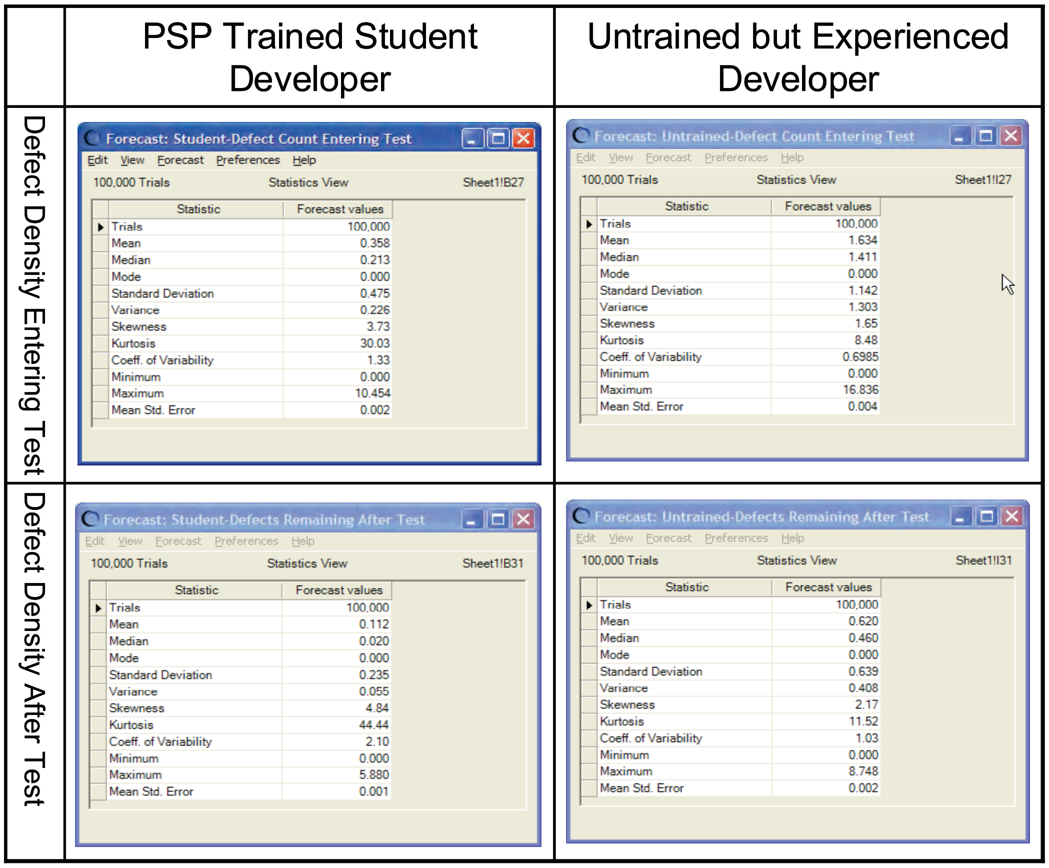
Student-Defect Count Entering Test: 100,000 Trials, Frequency View, 97,679 Displayed; -Infinity, Certainty: 70.357 %, 0.406

Untrained-Defect Count Entering Test: 100,000 Trials, Frequency View, 98,179 Displayed; -Infinity, Certainty: 70.071 %, 1.985

Student-Defects Remaining After Test: 100,000 Trials, Frequency View, 97,531 Displayed; -Infinity, Certainty: 70.003 %, 0.072

Untrained-Defects Remaining After Test: 100,000 Trials, Frequency View, 97,892 Displayed; -Infinity, Certainty: 70.513 %, 0.775

Figure 2: 100,000 Runs of the Monte Carlo Simulation for Quality


PSP Trained Student Developer
Untrained but Experienced Developer
Defect Density Entering Test
Defect Density After Test
Edit View Forecast Preferences Help

Forecast: Student-Defect Count Entering Test — 100,000 Trials, Statistics View, Sheet1!B27

Statistic	Forecast values
Trials	100,000
Mean	0.358
Median	0.213
Mode	0.000
Standard Deviation	0.475
Variance	0.226
Skewness	3.73
Kurtosis	30.03
Coeff. of Variability	1.33
Minimum	0.000
Maximum	10.454
Mean Std. Error	0.002

Forecast: Untrained-Defect Count Entering Test — 100,000 Trials, Statistics View, Sheet1!I27

Statistic	Forecast values
Trials	100,000
Mean	1.634
Median	1.411
Mode	0.000
Standard Deviation	1.142
Variance	1.303
Skewness	1.65
Kurtosis	8.48
Coeff. of Variability	0.6985
Minimum	0.000
Maximum	16.836
Mean Std. Error	0.004

Forecast: Student-Defects Remaining After Test — 100,000 Trials, Statistics View, Sheet1!B31

Statistic	Forecast values
Trials	100,000
Mean	0.112
Median	0.020
Mode	0.000
Standard Deviation	0.235
Variance	0.055
Skewness	4.84
Kurtosis	44.44
Coeff. of Variability	2.10
Minimum	0.000
Maximum	5.880
Mean Std. Error	0.001

Forecast: Untrained-Defects Remaining After Test — 100,000 Trials, Statistics View, Sheet1!I31

Statistic	Forecast values
Trials	100,000
Mean	0.620
Median	0.460
Mode	0.000
Standard Deviation	0.639
Variance	0.408
Skewness	2.17
Kurtosis	11.52
Coeff. of Variability	1.03
Minimum	0.000
Maximum	8.748
Mean Std. Error	0.002

Figure 3: 70% Prediction Interval for Quality Model

Figure 1: Excel/Crystal Ball Quality Model Simulation

If we use the classic PSP quality management techniques the point solutions are 0.04 defects/KLOC remaining after test for the "Trained" student and 0.62 for the "untrained" but experienced developer. However, this approach does not account for normal variability built into each of the subprocesses. On occasion, for example, the experienced programmer will deliver near 0 defects, while the PSP trained coder will release some defects. But, what is the likelihood of these occurrences? With Monte Carlo simulation, we can determine with a 70% confidence that the "Trained" student would have no more than 0.40 defects/KLOC entering test with no more than 0.07 defects/KLOC escaping test, while the "Untrained" student would have no more than 1.985 defects/KLOC entering test with no more than 0.775 defects/KLOC escaping test (see Figure 3). This analysis would give us fairly high confidence that employing the PSP techniques would provide an order of magnitude difference in the quality of the released product, regardless of the experience of the programmer.

Another output of Monte Carlo simulation is sensitivity charts. These capture which process steps are the most influential in producing the result. Figure 4 contains the sensitivity charts for both the "Trained" and "Untrained" developers' defect counts entering test. With real data and distributions these charts are particularly useful to quantitatively identify which subprocesses to focus on for both oversight and improvement actions.

1.3.2 Monte Carlo Limitations

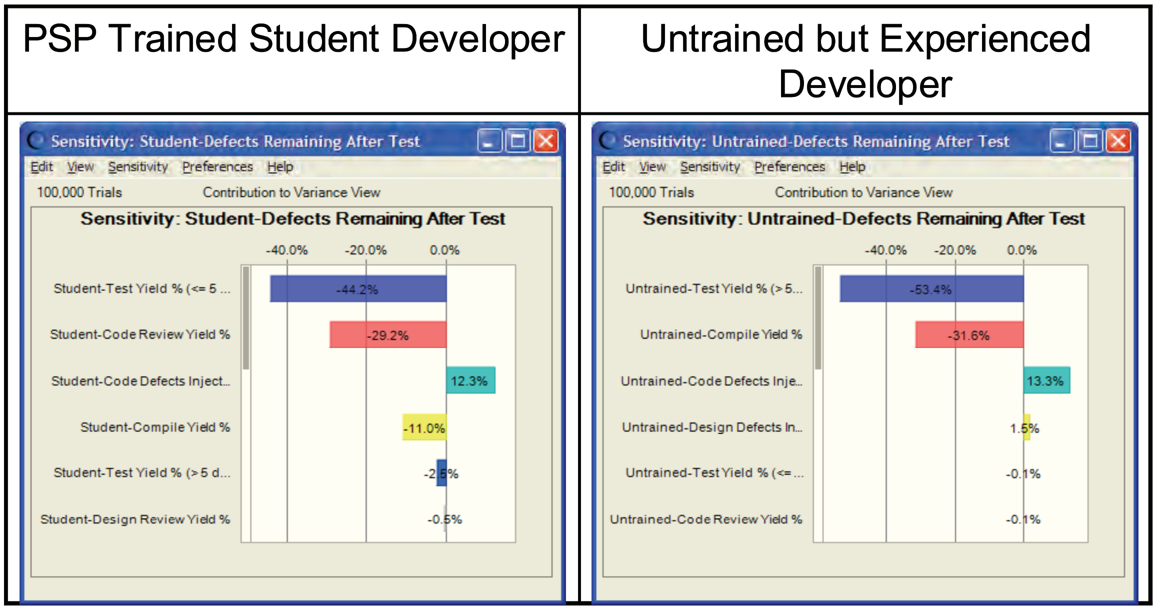
Monte Carlo can only approximate real world situations. The parametric model must be tuned to observed events. It is the accuracy of the parametric model with respect to the real world that determines the accuracy of the Monte Carlo simulation. We did not observe any of the other theoretical issues with Monte Carlo that have been identified in the literature (i.e. random number generation, computational power of computer implementing Monte Carlo, computation time of parametric model) during our application of Monte Carlo to TSP.

1.4 Process Models

1.4.1 Theory

The CMMI describes process performance models within the Organizational Process Performance (OPP) Process Area, specifically practice 1.5 "Establish Process-Performance Models". Additionally, within the Quantitative Project Management (QPM) Process Area, SP 1.4 "Manage Project Performance", sub-practice 4, the CMMI describes one of the key uses of process performance models – to statistically manage project execution. To reduce confusion with the various planning models within the PSP and TSP, the aspects of a CMMI process performance model are summarized in Table 6:

Figure 4: Monte Carlo Simulation – Sensitivity Charts



1	Process performance models predict either interim or final project performance outcomes,
2	They model uncertainty within predictive factors resulting in a prediction and/or confidence interval of the performance outcome,
3	They possess predictive factors which are clearly tied to processes and subprocesses,
4	At least one or more of the predictive factors are controllable by the project so that mid-course corrections may be made to secure successful project outcomes,
5	They enable "what-if" analysis by project staff to enrich the planning and replanning effort,
6	The models tend to be either statistical, probabilistic or simulation in nature to properly account for the inherent variation of factors and outcomes,
7	A collection of these process performance models serve to address needs across the lifecycle, as needed, and to link upstream and downstream processes.
8	These models may also serve to boost the activities of corrective action by helping to anticipate project and process issues, diagnose problems, evaluate alternative solutions and predict outcomes related to deployment and adoption. These aspects directly map to both the Corrective Action and Resolution (CAR) and Organizational Innovation and Deployment (OID) Process Areas.

Table 6: Process Performance Model Properties

Within this paper, Monte Carlo analysis serves as the tool that enables simulation (Property 6) that can be used to predict final project outcomes (Property 1) and enables what-if analysis (Property 5) in a TSP context. There are two process models that should be understood by all TSP practitioners: the planning model and the quality management model. These two models address Property 7. The planning model is a statistical analysis of the accuracy of the TSP planning (i.e., launch and relaunch) and replanning processes (e.g., weekly meetings). The quality management model is measured across the entire development process and takes into account all development subprocesses. The key aspects of these two models are reviewed below.

1.4.2 TSP Planning Model

During a TSP launch the individual engineers determine their schedule and task plans. The task plan identifies each task the engineer has been assigned and includes an estimate of the task hours needed to complete the task. Tasks are broken down into items of about 10 hours, and are sorted in order of expected task completion. The schedule plan is simply an estimate of how many hours the engineer will work on these tasks during each week of the project. The schedule plan only includes time to be spent on the items in the task plan – it does not include time spent on meetings or non-project work. Table 7 captures the Task Plan and Schedule Plan for a hypothetical individual.

Task Plan

Task Name	Planned Cost
Task 1	5 hours
Task 2	11 hours
Task 3	7 hours
Task 4	8 hours
Task 5	13 hours
Task 6	6 hours
Etc…	

Schedule Plan

Week	Planned Time
1	12 hours
2	12 hours
3	0 hours
4	8 hours
5	12 hours
Etc…	

Table 7: Hypothetical Task and Schedule Plan

From this information, TSP calculates an expected completion date for each task and for the entire project. These planned dates are calculated using a simple, direct-time-driven approach. For example, an individual might estimate that their list of tasks will require 300 total hours of work. This work could be completed on the date when the schedule plan reaches the "300 total hours" mark. Critical path analysis is typically unnecessary because TSP teams plan and manage their work with an unprecedented level of detail.

Each engineer has his or her own personal task plan and schedule plan. These personal plans are rolled up to the team level, where project management can determine the planned end date for the project, and for various components or milestones within the project. If some team members are over-tasked, their personal plan makes this apparent immediately, and the team can rebalance workload to optimize resource utilization.

During the execution of the project, individual team members measure the actual time spent on each task and the actual time spent each week, and record task completion dates. Standard earned value techniques are used to measure progress and to quantify cost and schedule variances. By comparing their actual progress to their planned progress, teams can calculate a projected project completion date and make mid-course corrections as necessary. Figure 5 is a graphical example of a TSP Earned Value chart at about the 20% point of a project.

1.4.3 TSP Quality Management Model

The TSP Quality Management Model is another important process performance model or series of models within TSP practice. This model is a simple prediction of defects injected and removed during the software process. Injected defects are estimated using average defect injection rates (defects per hour per process phase) for product development phases such as design and code. Defect removal is estimated for quality process phases, such as reviews, inspections and test, using average process phase yields (the percent of remaining defects removed by the activity). (See Figure 6.)[10] Using this model, projects can predict, during a launch, the final delivered product quality and can determine if the team's process is capable of producing the desired quality. If the quality level does not meet the desired quality level of the product then process changes or even other development processes can be considered. Historical team data (or if none is present, community benchmark data) describing defect injection rates and defect removal yields are used.

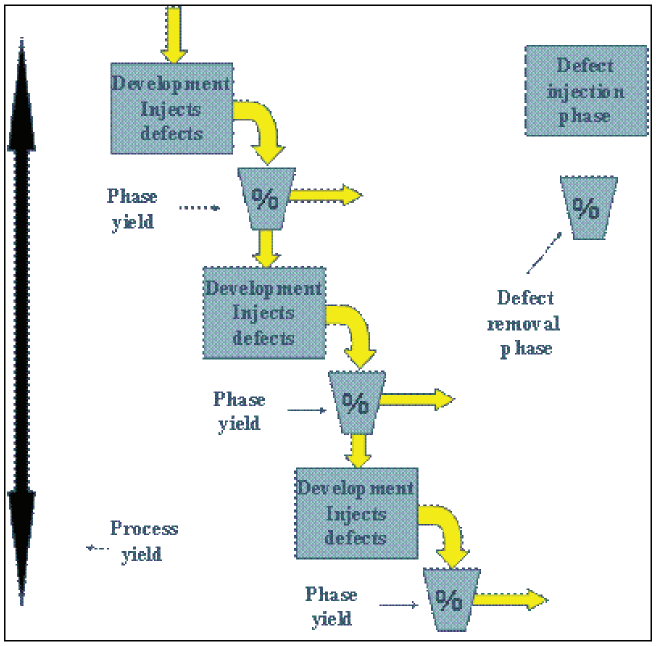


Figure 6: TSP Defect Injection and Removal Model

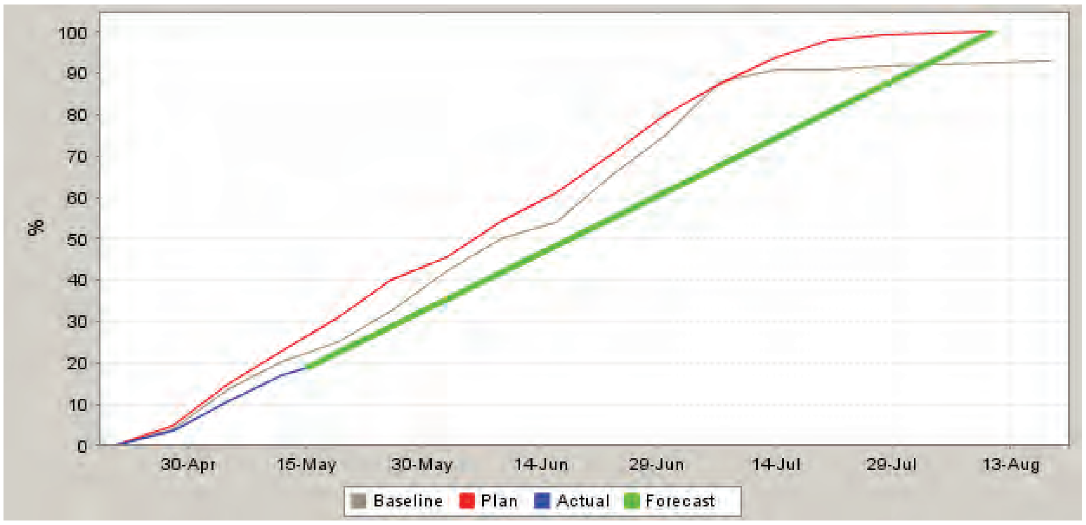


Figure 5: Hypothetical Earned Value Chart

1.4.4 Opportunities for Analysis

Since both these models use averages and do not address variability, they do not completely meet the needs of a high maturity process performance model. However, it is evident that there are data available to create such a high maturity model. In choosing which model to initially analyze, it has become evident to the authors, that the variability of the defect recording process (i.e., how each individual logs defect data) is quite large, leading to undesirable statistical effects. Task, schedule and effort data, though, appears to be recorded with much greater accuracy and precision by nearly all TSP teams. After some data analysis, we feel that, while a more robust quality model is needed and could be easily created using a procedure similar to the example in section 1.3.1, nearly all TSP teams could immediately benefit from process performance models using task, schedule, and effort data. We did note that a model measuring the consistency of the recording of quality data may also be immediately useful to teams, but it was not pursued for this paper.

1.5 Hill AFB TSP Teams

Hill Air Force Base has been using the TSP since 1998. In fact, TaskView, a Hill project, was an early TSP pilot and the birthplace of the Process Dashboard PSP/TSP tool.[11]
In 2001, the software group at Hill decided to use TSP on the Ground Theater Air Control System (GTACS) project, to help reduce defects, improve productivity, and increase customer satisfaction. This approach was very successful, resulting in a greater than 400% increase in productivity and a near zero defect rate which continues today. In fact, the performance of this team has been so extraordinary that the customer recently moved primary responsibility of the project from the prime contractor to this government organization. As a result, we have TSP schedule, effort, task and quality data going back to 2001on the GTACS team, all recorded using the Process Dashboard tool. Additionally, 309 SMXG has data since 2006 from two TSP software coding teams supporting the Marines' Expeditionary Fighting Vehicle (EFV); these teams have also been using the Process Dashboard since their inception.

1.5.1 Planning Data

Since the GTACS team has not had issues with product quality, the team's focus has been on how to meet its customer's aggressive schedule and productivity demands. Because of these issues the team has determined it needs to statistically understand its planning accuracy. Due to this team's focus, the author's earlier findings that quality data accuracy may be questionable, and the earlier findings that task, schedule and effort models could be used by many other TSP teams, we have devoted the remainder of this paper to our findings using 309 SMXG data for both GTACS and EFV with respect to the TSP planning model.

Since TSP teams collect data at both the team and individual levels, one of the first questions we asked ourselves is "what is the distribution of the individual and team planning data?" We were interested in learning the answer to two questions: 1) how accurately was available task time estimated; and 2) how accurately was the effort required to complete the tasks estimated?

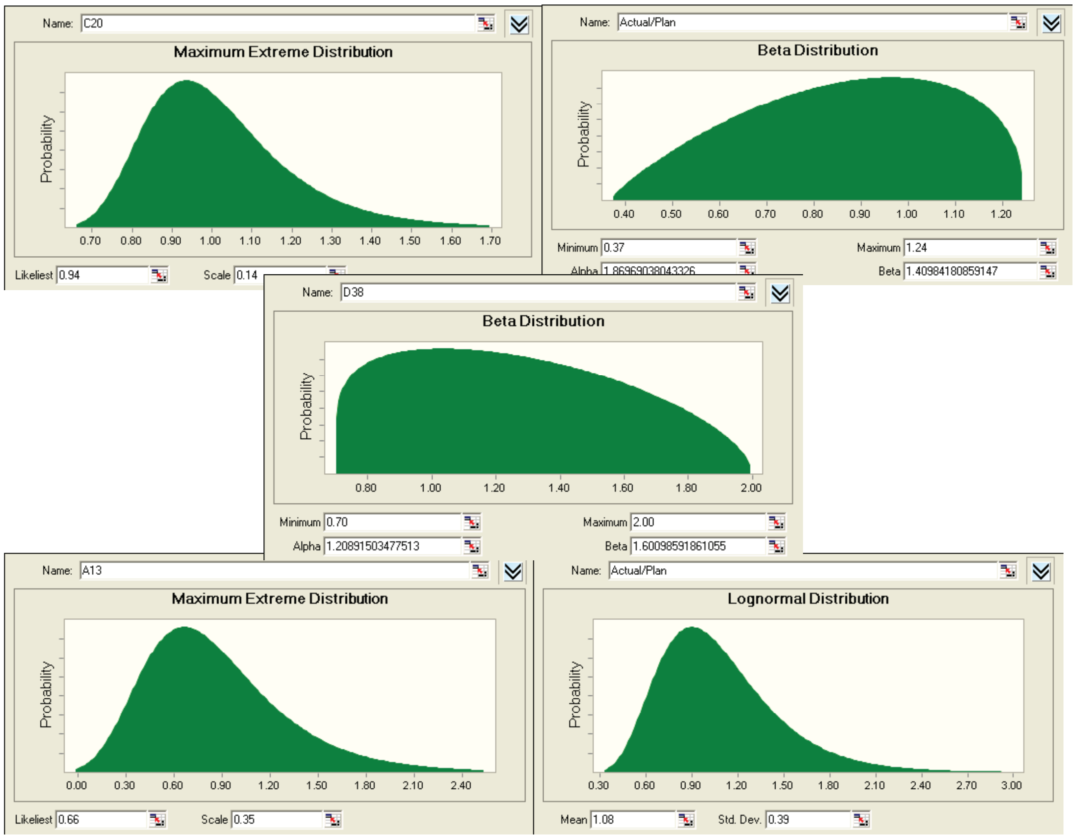
1.5.2 Planning for On Task Time

Focusing on the first question, we analyzed the team's schedule planning templates, where, during the launch, the available task hours are estimated for each week of the project. We determined an available task time estimating accuracy metric for each week as a ratio of actual to planned task time (Actual task time / Planned task time), where time was tracked in minutes. We then extracted the data from the Dashboard for a recent six month GTACS project. If the metric was near 1, then the team did a great job of estimating available task hours; if closer to zero, a poor job. When zero time was planned for a week and zero hours were worked then the metric was determined to be a one (1.0), or a perfect estimate. If zero time was planned and time was *worked* then the planned time was changed to 1 minute to eliminate infinity for calculation purposes. Using this data set, we ran the Crystal Ball "Fit" tool[12] on this data on both a team and individual basis. The best fit to a distribution was determined by selecting the distribution with the lowest Anderson-Darling rating.[13] At the individual level, there were multiple distributions in the data (see Figure 7).

We then modified the plan accuracy metric to remove the 0% (time planned but no time worked) and infinity (time worked but no time planned) data points by adding a second distribution to the model. For each individual we calculated the percentage of data points where the schedule plan either had time planned and no time worked or no time planned and time was worked. This value was 7%.

We then ran the Crystal Ball "Fit" tool on the remaining data (Table 8). Crystal Ball uses the Anderson-Darling (A-D) test to determine goodness of fit against 21 different statistical distributions. Crystal Ball defines a "good" fit as having an A-D score of less than 1.5. We expected the distribution to be log-normal because the metric was calculated by dividing a plan by an actual. We determined that log-normal fit all team members for which we could determine a distribution. As expected the parameters (mean, standard deviation) for each individual varied quite a bit, but the overall team parameters reflected good team performance. Note that the Beta distribution did not fit all team members and while the MaxExtreme distribution did fit all team members its applicability (determining the maximum value of a distribution) does not match well with our intended usage. The lognormal distribution has the additional benefit that it is theoretically covered during PSP training. For these reasons we were convinced that the lognormal distribution was the best distribution for modeling schedule planning accuracy.

Figure 7: Individual Distributions of Schedule Plan Estimating Accuracy



Name: C20
Likeliest 0.94 Scale 0.14

Name: Actual/Plan
Minimum 0.37 Maximum 1.24
Alpha 1.86969038043326 Beta 1.40984180859147

Name: D38
Minimum 0.70 Maximum 2.00
Alpha 1.20891503477513 Beta 1.60098591861055

Name: A13
Likeliest 0.66 Scale 0.35

Name: Actual/Plan
Mean 1.08 Std. Dev. 0.39

Team Member	Best Fit	A-D Score	Log-Normal A-D Score	Log Normal Mean	Log Normal Std Dev
E1	LogNormal	0.3355	0.3355	1.08	.39
E2	Beta	0.1551	0.3704	.87	.24
E3	MaxExtreme	0.5311	0.7841	.91	.64
E4	Insufficient Data (medical)				
E5	Beta	0.2603	0.3418	1.26	.36
E6	Max Extreme	0.3491	0.4892	1.02	.19
Team	Max Extreme	0.3798	0.5569	.98	.2

Table 8: Schedule Planning Best Distribution Fit

1.5.3 Task Planning

As we moved to our second question, how accurately was the effort required to complete the tasks estimated, we again needed to determine the distribution of the data. Once more, we extracted estimated and actual task time per completed task from the Dashboard for a 55 week period. This represents almost two full project phases covering the project's launch and first relaunch. We determined the estimating accuracy metric for each task once more as a ratio of actual to planned time (Actual task time / Planned task time). We ran Crystal Ball on this data on an individual basis. The data was exponentially distributed with a maximum at 0.0.

Upon further analysis, we interpreted the data as having at least two distributions: one dealing with the tasks that were tailored out of the project and the other with tasks that were actually performed. 18% of all tasks were "completed" with no time expended. This is most likely an artifact of the team's TSP planning process where the team's full process is mapped onto each requirement. During execution, process steps are tailored out if they do not add value or are no longer needed (e.g., previous postmortem's have indicated that one particular inspection can be safely combined with a later inspection if the item under development meets certain parameters). Also if an item passes an inspection with no anomalies then the developer records no time in the inspection phase, since no developer time was required to find and fix defects.

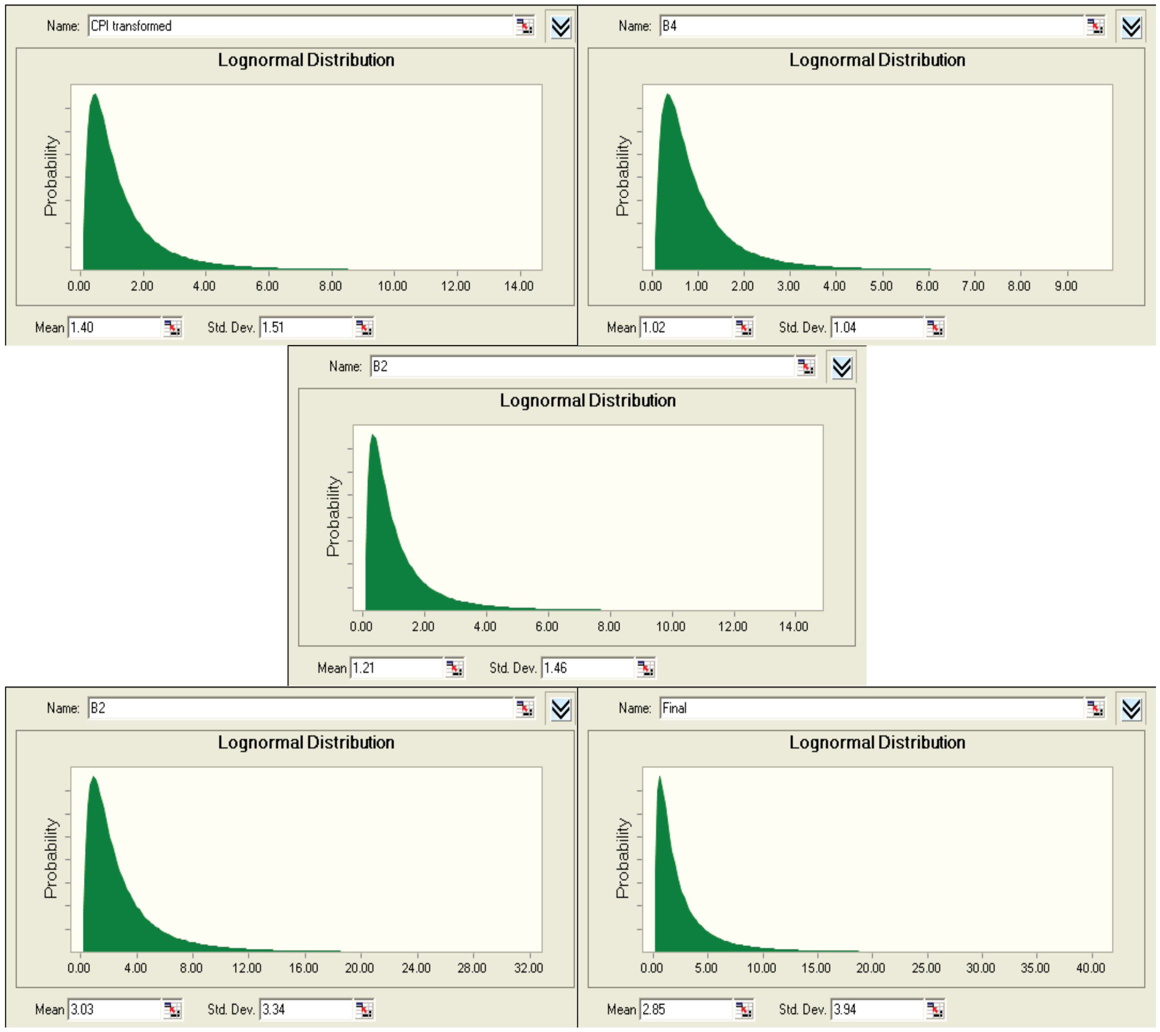
We modified the task estimating accuracy parametric model to remove these values (time planned but no time worked) by adding a second distribution to the model. Each week, for each task that has not started there is a chance that it will finish with no time allocated. For the team the value of this metric is 1.25%. For individuals the range was .53% to 4.52%.[14]

For tasks that were actually performed Table 9 captures the distribution.[15] We used Crystal Ball to determine which distribution was most appropriate. Lognormal was determined to be the best fit, both because it was adequate and it was known to the team members (Figure 7Figure 8 and Figure 9).

Member	Mean	Median	Standard Deviation
Team	1.66	1.00	2.39
E1	1.21	.98	1.54
E2	1.09	.68	1.69
E3	1.36	1.00	1.47
E4	2.11	1.34	1.85
E5	3.19	1.71	4.82
E6	2.73	1.49	2.92

Table 9: Task Planning Accuracy

Figure 8: Distribution of Adjusted Task Plan Estimating Accuracy per Individual



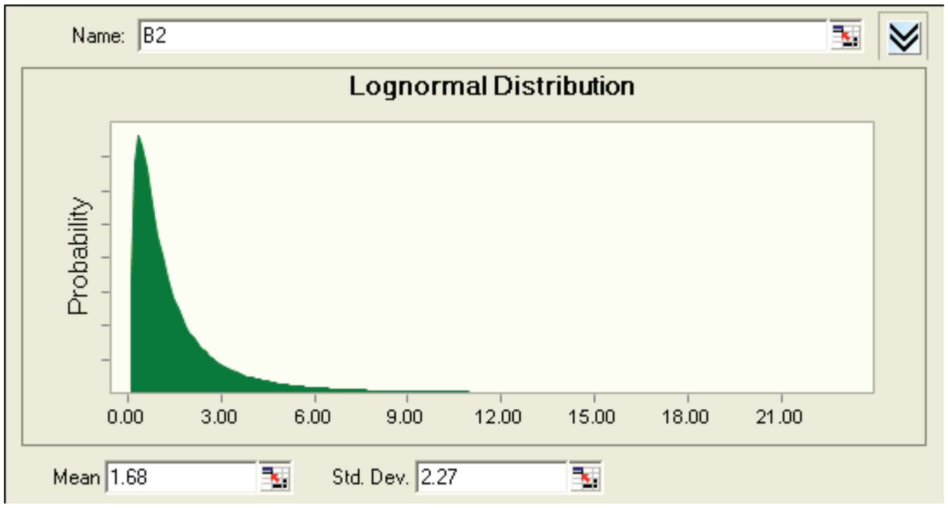
Name: CPI transformed
Mean 1.40 Std. Dev. 1.51

Name: B4
Mean 1.02 Std. Dev. 1.04

Name: B2
Mean 1.21 Std. Dev. 1.46

Name: B2
Mean 3.03 Std. Dev. 3.34

Name: Final
Mean 2.85 Std. Dev. 3.94



Mean 1.68

Std. Dev. 2.27

Figure 9: Distribution of Team Data for Task Plan Estimating Accuracy

1.6 TSP Planning Parametric Model

1.6.1 Earned Value - Projected Completion Dates

TSP teams have an unprecedented amount of fine-grained earned value data that they can use to analyze their progress. Because on-time delivery is often a high priority, most TSP teams use their earned value data to calculate a projected completion date, and monitor that date closely.

Within the EV literature, many techniques are described for calculating a projected completion date from earned value data. In general, these techniques work from an assumption that future progress on remaining work will follow the same patterns as historical progress on past work. (As every project manager knows, this can be a dangerous assumption, which must be continually challenged. But it nevertheless serves teams quite well.)

For example, the simplest approach draws a line from the project start to the most recent data point to gauge the rate of historical progress; then, it extrapolates that line to 100% to generate a projected completion date. We call this the Simple Forecasting Method.

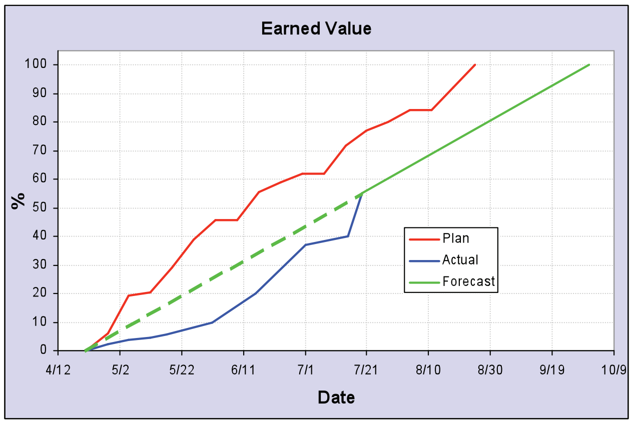


Figure 10: Typical TSP Earned Value Graph with Forecast Completion

That simple extrapolation technique, however, would only hold true if staffing levels are constant through the life of the project. This would produce an inaccurate projection if team members were joining or leaving the project on various dates, or if vacation and training causes available time to fluctuate. Fortunately, TSP teams can use the data in their schedule plan to improve this forecast. A common technique is to measure the total amount of progress made so far (earned value), and divide by the total amount of time in the schedule plan so far. The resulting ratio roughly approximates the rate at which value is being earned relative to the plan. (For example, you might find that each hour of planned time has earned 1% of project value.) This ratio can be applied to future weeks in the schedule plan to determine when the project might reach 100% complete. We call this the Rate Forecasting Method.

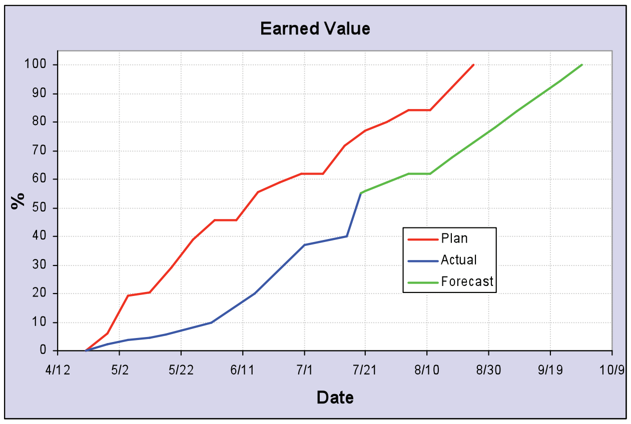


Figure 11: Earned Value with Forecast Task Completion Data

That technique is very simple, and can easily be calculated. But it is not without problems. Its most significant failing is that it is highly skewed by "in progress" work (tasks that have been started but not yet completed). Since in-progress work earns zero credit, the calculated rate is identical to the rate that would be produced if team members had "sat on their thumbs" for all of those in-progress hours. Furthermore, when the rate is applied to future weeks in the schedule, it begins with an assumption that all of those in-progress hours still need to be expended all over again. Thus, this technique counts in-progress effort against the team twice, and produces a very pessimistic forecast projection.[16] Furthermore, it is not amenable to parametric analysis, because one single number – the rate – combines data from several unrelated sources of variation.

1.6.2 Earned Value – Parametric Model

To perform a successful analysis of an earned value plan, we must break apart the independent sources of variation. At a minimum, two sources of variation are easily identifiable: cost estimating error and schedule estimating error. That is, the estimated times in the task plan have a measurable error, and the estimated times in the schedule plan have a measurable error. If we can characterize these errors mathematically, we can use them as the basis for a parametric analysis.

Of course, cost estimation error has received a great deal of analysis over the years, especially from PSP. At the end of a project, the overall estimating bias can be described with the cost performance index (CPI):

CPI = Cost Performance Index = Total Planned Cost / Total Actual Cost

We could apply a similar calculation to the rows in our schedule plan, to produce a number we will call the direct time performance index:

DTPI = Direct Time Performance Index = Planned Schedule Time / Actual Schedule Time

When the project is done and we have these two ratios describing estimating biases, we could hypothetically produce a "revisionist project plan," as follows:

- Make a copy of the original project plan.
- Divide the estimated cost for each task by the CPI. By definition, the resulting numbers would sum to the actual project cost.
- Divide the estimated time for each week by the DTPI. By definition, the resulting numbers would sum to the actual time in the schedule.

By definition, this planned completion date in this revisionist plan would be equal to the actual completion date of the project.

During the project we can apply a similar analysis, based upon the CPI and DTPI calculations for the work that has been performed so far. The resulting projection will be based on an assumption that the estimating biases observed so far will hold true for remaining tasks. For example, if tasks are taking twice as long as expected, then we probably underestimated the work by 50%. And if we're getting two-thirds of the time we planned per week, that trend is likely to continue in the future. The resulting projection will be roughly equivalent to the extrapolations calculated by the Rate Forecasting Method.[17]

1.6.3 Earned Value – Parametric Analysis of Variation

Of course, this extrapolated completion date is based only upon the end-to-end average CPI and DTPI ratios. In PSP parlance, we have used "PROBE Method C" – a linear averaging method. As we know, we have another source of information we can analyze: the variability of the individual data points. For example, even if the project has a CPI of 1.0, we might see that individual tasks have estimating errors of ±50%. A statistical analysis might tell us, "your CPI is currently 1.0 – but that might be an accident due to the set of tasks you happened to complete first. If the future tasks have estimating errors as large as the tasks so far, the final CPI of the project might be as large as 1.3 or as low as 0.65."

In order to perform such an analysis, we must have a way to characterize the variability of the individual data points we have collected. As paragraphs 1.5.2 and 0 show, it seems quite possible to identify statistical distributions that model these sources of variability – but only if we calculate those distributions one team member at a time. This, then, forms the basis of our parametric model:

- For a single individual, identify probability distributions that characterize their cost and schedule estimating errors.
- At the team level, collect task and schedule plans from each individual, each with their own cost and schedule distributions.
- Run a Monte-Carlo simulation with tens of thousands of trials:
  - In each trial, select a random value from the cost distribution and a random value from the schedule distribution for each individual.
  - Use those numbers to construct "revisionist plans" for each individual.
  - Roll up the data from those revisionist plans to create a projected team completion date.
  - Record that date as the result of a single Monte-Carlo trial.
- After a sufficient number of trials have been run, the trial values can be analyzed to produce a probability distribution for the team completion date.

- The 50% probability value from that distribution can be taken to be the "most likely value." Alternatively, the team can pull 70% or 90% values to view prediction intervals.

Of course, the standard disclaimers for prediction intervals still apply. The resulting range does not tell the team that they have a "70% chance of completing by date X." Instead, it tells them that "there is a lot of variability in your data. I've extrapolated a completion date of Y, but the based on the variability in your data, that extrapolation could have just as easily fallen anywhere between X and Z, assuming project performance matches historical data." Figure 12 is a graphical depiction of applying this technique to an Earned Value estimate. The green "cloud" represents the 70% prediction interval as calculated using Monte Carlo techniques; while the calculations and data used are far different, this prediction interval concept is the same as taught to students during the PSP course.

1.6.4 Real-World Use and Application

Stepping back from the math, there is a commonsense way to interpret the resulting values. During the first few weeks of a project, the projected completion date can jump around wildly as team members mark tasks complete. Inexperienced teams can be alarmed to see a projected completion date several years away – but coaches know that these swings are a normal byproduct of limited historical data. Through parametric analysis, we can show the team a projection accompanied by an uncertainty range. When the uncertainty range is large, the team should take the projection with a grain of salt or ignore it entirely.

These uncertainty ranges give teams valuable new insights, not only into their current project progress, but also into the quality of their planning process. As TSP coaches, we often caution teams that earned value projections are only as good as the plan from which they are derived (garbage-in, garbage-out). When uncertainty ranges are wide – and especially when they grow wider as the project progresses – this serves as a red flag that the team is working against a poorly constructed plan. In such a case, the projected completion date shouldn't be trusted; the team may be overdue for a relaunch, or they may need to analyze their launch planning process to identify opportunities for improvement.

Of course, the calculations described above are nontrivial. To use these techniques successfully, a team must have tools that can perform the parametric analysis and calculate the uncertainty ranges. Although various commercial applications provide Monte-Carlo support, the earned value parametric model can be extremely complex to simulate. Fortunately, teams using the open-source Process Dashboard application can benefit from these uncertainty ranges for free. Since 2003, the Process Dashboard has been performing the parametric analysis described above, and calculating completion date ranges at both the individual and the team level.[18]

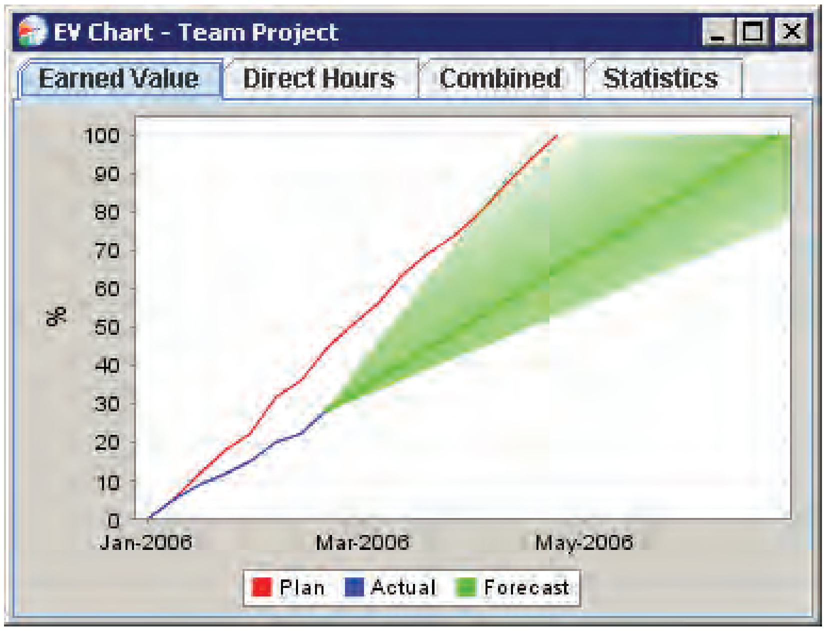


Figure 12: Earned Value Forecast using Monte Carlo Simulation and 70% Prediction Interval

Finally, although the discussion above has described the use of current project data to construct distributions for cost and schedule variability, this is obviously not a hard requirement. Data from completed projects (or from across the industry) could be used to calculate baseline distributions. With such baselines in hand, parametric analysis can be performed during a team launch to generate uncertainty intervals for the planned completion date.

1.6.5 Corrective Actions and Process Improvement

Of course, there are multiple ways to apply these distributions in real-life projects. One simple way is by the use of control charts or control "like" charts. For example, if an individual's variability of task hour estimating data is lognormal with known mean and standard deviation and was planned with this distribution in mind, then each week the actual to planned ratio can be plotted on a chart with a logarithmic scale for mean and standard deviation (Figure 13). If the individual stays within the control limits, then their plan should account for this kind of variability. If the individual has a point outside of these limits, then the situation should be examined for possible replan or process improvement. The team can also use these charts to look for opportunities to narrow the control limits and/or move the mean closer to zero. While these are not control charts, strictly speaking, they follow the control chart patterns and point out true corrective action and process improvement opportunities. In Figure 13 investigation of the outlier point indicated that it was a unit test task that had been planned at about nine and a half hours and took about half hour to complete. For this task the test equipment worked the first time (about 50% chance), the test passed during its first execution (not that unusual), and no test procedure refinement was needed (unusual).

1.7 Usage

We are using the Monte Carlo technique described in section 1.7 for planning and not for quality management. This is primarily a data quality and quantity issue. As shown previously we have significant amounts of data that we believe to be accurate related to schedule and task plans. We have much less data and data of a perceived much lower quality for quality management purposes. A specific quality management subprocess is only executed a few times per cycle by individual engineers. We have also noted a continual coaching issue with the recording of all defects found, especially those found in unit test and personal reviews.

Not surprisingly, we have noted variation at the individual level in the accuracy of schedule and task planning. The model we propose uses this variation to both predict project completion schedule and cost and to quantitatively focus management and improvement on schedule and task planning accuracy. Instead of using the current CPI and SPI methods to estimate completion the new model uses the natural variation of individual team members planning accuracies as input to Monte Carlo. The team statistically manages the individual task and schedule planning accuracy metrics. By statistically managing we mean that the variation in performance is understood, assignable causes are managed and over the long term improvements are made in both the bias and variation of the estimates.

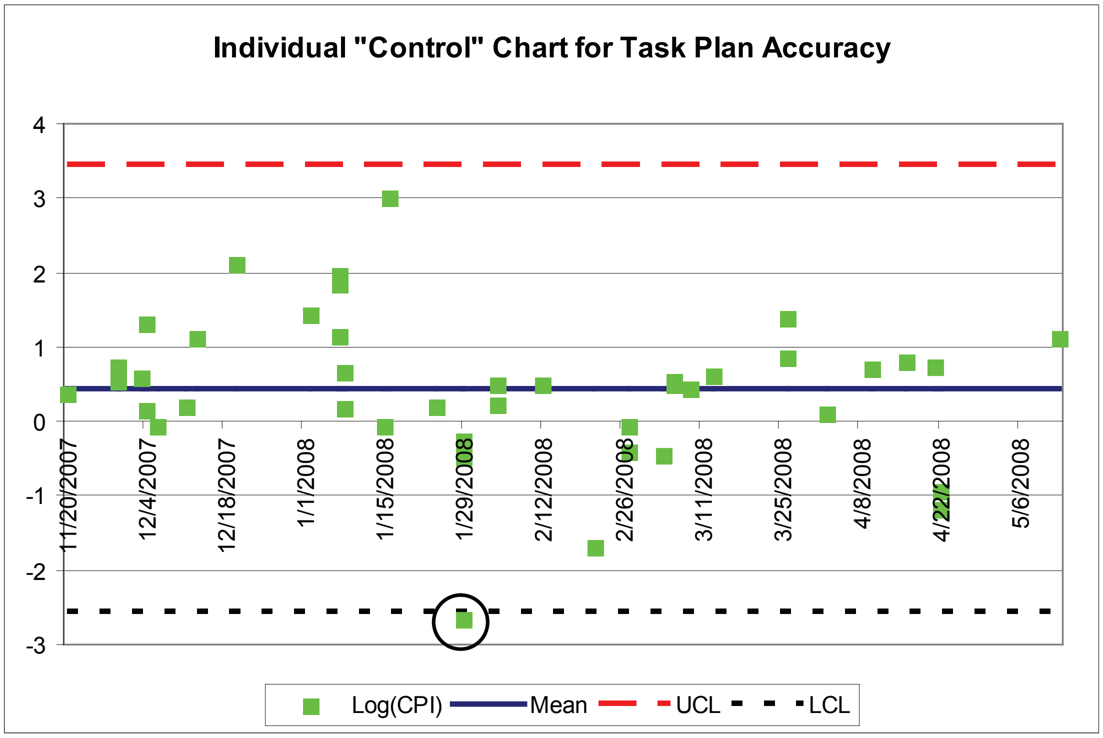


Figure 13: Control "Like" Chart for Task Hour Estimating Accuracy

1.8 CMMI Statistical Management and TSP

In section 2 we noted a list of quality attributes that the implementation of these high maturity enabling process changes should meet. The gist was that the changes help the team reach its business related goals while not adding overhead. The below table discusses each attribute. The conclusion is that the Monte Carlo Analysis described above helps the team meet its goals and while not adding significant overhead to team's management practices.

Attribute	Comment
Should be useful for teams during project execution (more so than during project postmortem)	Teams are able to use Monte Carlo simulation output weekly during team meetings to manage commitments at a level of confidence. Envisioned extensions (see paragraph 1.9) will allow them to monitor process performance of key sub processes during project execution using XmR like analysis.
Should not require a great deal of overhead	Tool support enables calculations and modeling to be performed in the background. TSP theory allows consistent definition of parametric model.
Should not require a great deal of new training	The only new concepts described and in use are the use of Monte Carlo to determine prediction interval. Enhancements (see paragraph 1.9) may require knowledge of sensitivity charts and XmR theory.
Should focus on a typical TSP team's needs. This probably means helping teams achieve their quality and schedule goals	The Monte Carlo enhancement described is intended to help the team understand the variability in their cost and schedule planning estimate and actual performance. An enhancement requiring tool support would enable support for the quality management process model.
Should stress quality, because as PSP teaches: from quality comes schedule and cost	We determined that the tool we have does not support Monte Carlo analysis of the quality management process model. Also we have concerns about the quality of the data related to the quality management process model and the team we were studying has historically (at least since TSP introduction) not had an issue with product quality, thus it is not an issue of interest to them.
Its theory should be consistent with PSP and TSP training and ideally should match the theories stressed during PSP and TSP training	The parametric model used by Monte Carlo uses the PSP and TSP planning and quality management process models as its basis.

In light of the above criteria, there are simple approaches to process performance modeling and statistical management that may be implemented within a TSP environment. The following is an abbreviated discussion of the CMMI Quantitative Project Management (QPM) Process Area and clarification of its applicability to the TSP environment.

Specific Goal 1: Quantitatively Manage the Project

Specific Practice 1.1: Establish the Project's Objectives

Project objectives may be defined by a decomposition of business goals and objectives that are first delineated during the TSP launch process. As opposed to business goals and objectives which are quite high level, project objectives become detailed and worded in terms of how project factors, including factors related to people, process, tools, methods, and the environment, will be controlled and managed. The earlier discussion of historical baselines of factors in terms of distributions possessing a mean and standard deviation provide a rich mechanism to identify and evaluate the reasonableness of project objectives that will meet the business goals and objectives. Monte Carlo simulation may be used to evaluate the uncertainty of performance in the project factors and it's resulting impact on the expectation of meeting the business goals and objectives.

Specific Practice 1.2: Compose the Defined Process

TSP teams employing Monte Carlo simulation, as described in this paper, as well as, other statistical regression techniques, may also leverage these models to make more informed decisions about how to establish the project's defined process. Within TSP, the implication is that the numerous controllable project factors that are known to govern individual and team performance, may be used to guide up front decisions on specific changes to staffing, methods, tools, and environment to increase the likelihood of success. TSP teams most likely would be able to fine-tune various project factors to optimize performance, This could be as simple as deciding on individual task assignments based on individual strengths and weaknesses, to as sophisticated as, deciding how to govern internal and external team member handoffs and interactions. Other decisions such as approach to the software architecture, choice of programming language, approach to software testing and interaction with the customer may be enhanced through similar use of the process performance models.

Specific Practice 1.3: Select the Subprocesses that will be Statistically Managed

In keeping with the spirit of the criteria for TSP team usage of process performance models and statistical management, this practice becomes vital to ensure a lean, efficient approach is adopted. Within an individual's process and within the processes of the TSP team operation, this practice calls for a reasonable approach to identifying where and when statistical management is warranted. Ideally, the TSP team should possess increasing amounts of historical data and baselines on project and individual factors. This data may be analyzed in terms of historical distributions. Factors possessing the greatest uncertainty or variation will represent the targets for use of statistical management. The primary focus of statistical management is to enable the reduction of uncertainty, variation and surprise in the project. Additionally, the statistical management of the various project factors that dominate the resulting performance outcomes should be considered. Instead of statistically managing outcomes, the statistical management of these project factors will provide the greatest leading indicator of undesirable individual and team performance.

Specific Practice 1.4: Manage Project Performance

Both individuals and teams may use the frequent updating of, and predictions from, process performance models to continually assess if execution is on track to successful project outcomes. Additionally, the wise use of a small set of statistical process control charts on the significant project factors will enable early feedback on both process stability and capability. All of these activities serve to arm the team with sufficient ability to take proactive actions during project execution to ensure successful project outcomes.

Specific Goal 2: Statistically Manage Subprocess Performance

Specific Practice 2.1: Select Measures and Analytic Techniques

Individuals and teams are well served to apply basic statistical and simulation techniques such as Monte Carlo simulation and regression analysis to analyze and predict subprocess performance. An advantage that TSP teams have is that their training and culture already supports the focus on quantitative analysis and management by data. TSP teams have consistently shown a lower adoption curve with regards to most of the CMMI High Maturity measurement and analysis.

Specific Practice 2.2: Apply Statistical Methods to Understand Variation

As discussed already, the use of Monte Carlo simulation and statistical regression represents a very mature approach to quantifying and understanding variation in project factors and the impact of that variation on the performance outcomes of projects. Historically, teams have conducted regression analysis of controllable project and process factors to predict outcomes. Once the regression is completed, the regression equations are modeled in Excel with a Monte Carlo simulation add-on. Essentially, the TSP team members can then indicate the degree of uncertainty and the degree of control that they will exert on the factors of the regression equation, which then is fed into the simulation so that the resulting expectation of project outcomes may be realistically evaluated.

Specific Practice 2.3: Monitor Performance of the Selected Subprocesses
Monitoring performance of the key work subprocesses within the TSP team may be quite simple. Simple periodic observation of the critical statistical process control charts and comparison of process performance model predictions of interim outcomes with actual performance of the interim outcomes fully implements the intent of this specific practice. This activity may be easily integrated into the daily and weekly management actions of the TSP team.

Specific Practice 2.4: Record Statistical Management Data
Recording statistical management data is already an inherent activity within the TSP environment. Maintaining baselines (distributions, central tendency, variation) of key project and process factors, as well as, TSP interim and final outcomes, is already an inherent activity of TSP and thus does not represent a new activity. The primary new activity here may be that TSP teams may find it useful to collect and store data on an extended set of factors above and beyond the set of factors currently taught in the TSP method. TSP teams, however, should proceed with the business and customer goals as the primary focus, and ensure that the statistical management supports those goals.

In summary, both CMMI process performance modeling and statistical management of critical work activities can play a significant role in TSP team operations. The Monte Carlo simulation example in this paper demonstrates a model that meets many of the desirable attributes in a CMMI process performance model.These attributes include the modeling of variation of factors to predict interim and final outcomes with an expected range or distribution of behavior.As will be seen in section 1.9, other desirable attributes still need to be considered and added to this Monte Carlo simulation example to fully represent the intent of CMMI process performance models, such as controllable factors tied to critical TSP work processes used to make simulation or statistical predictions of outcomes.

1.9 Next Steps

Use of Monte Carlo to enhance PSP and TSP is a field rich in opportunities. Although out of scope of this paper, the modeling described could and should be extended to statistical modeling of attributes of the PSP and TSP environment, attributes of the exhibited team competencies, and of individuals themselves, to enable better predictions of plan accuracy and product quality. This would enrich the current TSP process models, providing further competitive leverage and links to achieving CMMI High Maturity. In paragraph 1.3.1 the examples of sensitivity charts were included. These can be used for process analysis to identify key process areas for performance monitoring during project and execution and high value improvement areas during project post mortems.

Community data is needed. In paragraphs 1.5.2 and 0we derived the planning accuracy distributions for a small set of TSP practitioners. Baseline data including distribution derived from the TSP community would be useful when individual historical data is not available. This data probably should be sorted into (TSP) experienced and inexperienced bins since experienced practitioners will most likely have their personal historical data to use. This historical data should be enough to fully support both the TSP planning and quality management process models.

Tool support can also be improved. Some TSP tools do not support Monte Carlo. They could be extended to support it. The dashboard tool that we have used could also be extended. The rest of this paragraph identifies potential extensions to the dashboard. Two options for bootstrapping Monte Carlo are needed. The historical data set to use should be user specified, controllable, and optional. The user should also optionally be allowed to specify a default distribution type and parameters. To enable process performance understanding and long term improvement, the best fit distribution type and parameters should be reported to the user. Also an XmR like capability could be added to allow the user to review their ongoing performance with respect to their expected performance. Monte Carlo support should be extended to the TSP quality management process model.

Finally, stepping back from the CPI and DTPI based parametric model described in paragraph 1.5, one should note that it implements what the PSP/TSP community might call PROBE “Enhanced Method C”. This enhancement adds a variability dimension to the standard method C and allows us to compute prediction intervals when using Method C.

1.10 Closing

Last year we discussed our TSP team’s CMMI maturity level 5 assessment experiences and noted several areas where our expectation that TSP came out of the box as a level 5 process was not met. As reported last year most of these areas where relatively easy to address without violating any TSP principals. The one area where our implementation was clearly not optimal from the CMMI perspective was in the area of process performance models. Our hope that this paper has shown that the Monte Carlo method, some extensions to common universally known TSP process models, and focused tool support allows TSP to meet this ideal of coming out of the box as a level 5 process.

1.11 Acronyms

A-D	Anderson-Darling
CAR	Causal Analysis and Resolution
CMMI	Capability Maturity Model Integration
CPI	Cost Performance Index
DTPI	Direct Time Performance Index
EFV	Expeditionary Fighting Vehicle
EV	Earned Value
GTACS	Ground Theater Air Control System
KLOC	Thousand Lines of Code
OID	Organizational Innovation and Deployment
OPP	Organizational Process Performance
PROBE	PROxy Based Estimating
PSP	Personal Software Process
QPM	Quantitative Project Management
SPI	Schedule Performance Index
TSP	Team Software Process
XmR	Individuals and Moving Range (Control Chart)

1.12 References/Bibliography

SM Team Software Process, TSP, Personal Software Process, and PSP are service marks of Carnegie Mellon University.

1 **McHale, James, and Daniel S. Wall.** "Mapping TSP to CMMI." CMU/ SEI-2004-TR-014. CMU, 2004.

® Capability Maturity Model Integration and CMMI are registered in the U.S. Patent and Trademark Office by Carnegie Mellon University.

2 **Webb, David, Miluk, Gene, and Van Buren, James,** CMMI Level 5 and the Team Software Process, A Case Study of the CMMI Level 5 Assessment of a TSP Team, 2007 TSP Sympo-sium.

3 Original list - **Willett, Alan and Fagan, Eileen,** The Teacher, The Bathtub, and the Bureau-crat, 2007 TSP Smposium (presentation). List has been modified locally.

4 **Weisstein, Eric W.** "Monte Carlo Method." From MathWorld—A Wolfram Web Resource. http://mathworld.wolfram.com/MonteCarloMethod.html

5 **Wittwer, J.W.,** "Monte Carlo Simulation Basics" From Vertex42.com, June 1, 2004, http://vertex42.com /ExcelArticles/mc/MonteCarloSimulation.html

6 The Beginning of the Monte Carlo Method, N Metropolis, Los Alamos Science, Special Issue, 1987. http://library.lanl.gov/cgi-bin/getfile?00326866.pdf

7 **Humphrey, Watts,** PSP: A Self Improvement Process for Software Engineers, Addison-Wesley, 2005. Page 244

8 **Humphrey, Watts,** TSP – Coaching Development Teams, Addison-Wesley, 2006. page 149.

9 **Humphrey, Watts,** TSP – Coaching Development Teams, Addison-Wesley, 2006. page 146.

10 **Humphrey, Watts,** PSP: A Self Improvement Process for Software Engineers, Addison-Wesley, 2005. Page 144

11 **Ken Raisor, Rick Reynolds, and David Tuma,** Process Dashboard version 1.9, copy-right © 1998-2007, http://processdash.sourceforge.net/

12 **Oracle and/or its affiliates,** Crystal Ball User Manual, Version 7.3.1, Copyright © 1988, 2007, pp 29 - 35 (http://www.crystalball.com/)

13 **Oracle and/or its affiliates,** Crystal Ball User Manual, Version 7.3.1, Copyright © 1988, 2007, pp 31, 127 (http://www.crystalball.com/)

14 The dashboard task and schedule report does not report the start date of a task. Thus for pur-poses of this calcu-lation a we assume that tasks are worked without inter-ruption until completed, that all tasks started at the phase launch, and that each engineer has 10 task hours per week. We then can use the actual task hours, actual date completed, and the phase launch date to determine the number of weeks that each task had 0 task hours applied.

15 The team members are ordered roughly by when they completed their PSP training.

16 One could argue that pessimistic forecast dates motivate team members, but that is not a sound mathematical basis for choosing a statistical method. For informed de-cision-making, we seek to use mathematical techniques that are as accurate (balanced between over- and under-estimates) as possible. Sometimes a team may want a pessimistic forecast value to ensure they do not over commit. In that case, it would be far better to choose the upper range of a confidence interval (as described below) than it would be to add an arbitrary amount of calendar time accidentally pro-duced by unaccounted-for effort.

17 The two techniques will produce exactly the same pro-jection when no "in progress" tasks are present. As the total effort expended on "in progress" tasks grows, this technique will continue to produce estimates that aim to be as accurate as possible, while the rate technique will produce increasingly pessimistic forecasts.

18 **Tuma, David A.** "A Statistical Model for Personal Earned Value, and Implications for Project Planning." 31 Dec. 2004. http://processdash.sourceforge.net/ev.html.

Authors

Robert W. Stoddard II
Senior Technical Staff
Software Engineering Institute
Robert Stoddard is a Senior Member of the Technical Staff within the Software Engineering Institute currently responsible for the development and delivery of measurement training associated with process improvement and CMMI High Maturity. In concert with this work, Robert recently co-authored a book entitled “CMMI and Six Sigma:Partners in Process Improvement” via Addison-Wesley and has delivered a number of tutorials on CMMI High Maturity measurement practice. Prior to joining the SEI in 2005, Robert served for 7 years, as both a Quality Director and Distinguished Member of Technical Staff, within Motorola - including the leadership of the world-wide deployment of Six Sigma training for Motorola University. Prior to Motorola, Robert served for 14 years with Texas Instruments in the field of quality and reliability. Robert is an active member of both the American Society for Quality and the IEEE Reliability Society and holds 5 ASQ certifications. Additionally, Robert is a Motorola-certified Six Sigma Master Black Belt.

David Tuma
Software Consultant
Tuma Solutions, LLC
David Tuma is an independent consultant, supporting high-maturity project teams through the development of open source support tools. David received a BS in Computer Science and Engineering from the Massachusetts Institute of Technology. He is an SEI authorized PSP Instructor.

James K. Van Buren
Software Program Manager
Charles Stark Draper Laboratory
Jim Van Buren is a Program Manager for the Charles Stark Draper Laboratory currently supporting the Ground Theater Air Control System (GTACS) project team at Hill Air Force Base. He is a SEI authorized PSP instructor and TSP coach. He holds a BS in Computer Science from Cornell University and has over 25 years software development and management experience.

David R. Webb
Senior Technical Program Manager
520th Software Maintenance Squadron, Hill AFB
David R. Webb is a Senior Technical Program Manager for the 520th Software Maintenance Squadron of the 309th Software Maintenance Group at Hill Air Force Base in Utah, a CMMI Level 5 software organization. Webb is a project management and process improvement specialist with over twenty years of technical, program management, and process improvement experience on Air Force software. Webb is a SEI authorized instructor of the Personal Software Process, a Team Software Process launch coach, and he has worked as an Air Force manager, SEPG member, systems software engineer and test engineer. He is a frequent contributor to technical journals and symposiums, and he holds a Bachelor’s Degree in Electrical and Computer Engineering from Brigham Young University.