Software Project Planning, Statistics, and Earned Value

Walt Lipke and Mike Jennings
Software Division, Directorate of Aircraft Management, Oklahoma City Air Logistics Center

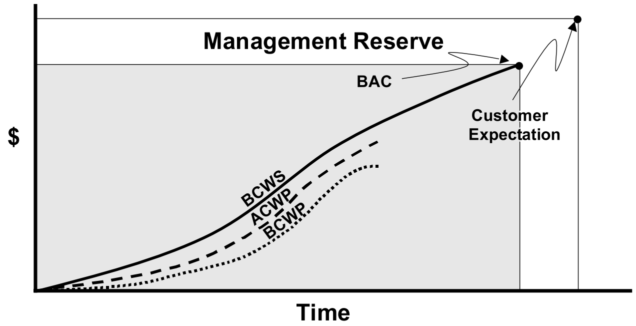
Increasingly, statistical methods are being applied to earned value (EV) data [1, 2]. In a previous publication [3] the author has discussed the use of statistical process control (SPC) with the earned value indicators and cost and schedule performance indexes. This application provides strategies and methods for gauging the performance of software projects and achieving project commitments. As an extension of managing performance, this article branches to using the statistical representation of the EV information to prepare project plans. Interestingly, the statistical approach yields not only the expected cost and completion date, but the management reserve required for an acceptable level of risk.

Nearly 15 years ago the Test Software and Industrial Automation branches of the Directorate of Aircraft Management began using EV methods to manage software development. During this time several refinements in the application of EV were made. The work breakdown structures have evolved and are much more sophisticated today than they were for the first few projects where EV was applied. The planning and scheduling practices have improved tremendously from simple paper and pencil tabulations to the use of commercially available automated tools. With these tools *what if* scenarios can now be performed during planning to account for possible risk areas. Along with these improvements, the ability to predict project outcomes and to strategize needed recovery actions was significantly enhanced three years ago by applying EV cost and schedule performance indexes [4]. Project managers now have a tool to help them choose the appropriate recovery strategy along with necessary actions.

Although these are significant improvements that have aided greatly in managing software projects to achieve the required performance at the negotiated cost and completion date, software organizations today are feeling pressure to apply the control chart method of SPC. Control charts began in the 1930s and were applied to manufacturing processes to maintain quality control of assembly line products. Control chart concepts and methods fell out of favor in the United States by the 1950s, but were revived in the 1980s. Dr. Edward Deming became an international celebrity from the impact control charts had on the success of the Japanese production of automobiles.

The application of SPC to software management is not very straight-forward; the automobile assembly-line application does not translate directly to software. The rate of software development is low, and none of the products are prepared to identical specifications. Yet a belief persists that SPC control charts must be used in order for software management to know that the development process is in control. Just as in manufacturing, if anomalous behavior occurs in the software process it should be recognized and corrected in order to minimize the impact on the delivered product.

Along with the rest of the software industry, we have struggled to develop a meaningful application of SPC control charts. EV indicators and cost and schedule performance indexes is proving very useful. In the publication [3] cited previously, the indicators were shown to be a management tool for statistically representing project performance. That paper also provided project managers with methods to overcome poor performance, and it alluded to further application in the areas of software project planning and process improvement. This paper addresses project planning, including the quantification of risk in both cost and schedule. Mitigating the risk with management reserve is included in the discussion.



BCWS = Budgeted Cost of Work Scheduled
ACWP = Actual Cost of Work Performed
BCWP = Budgeted Cost of Work Performed
BAC = Budget at Completion
CPI = Cost Performance Index [CPI = BCWP / ACWP]
SPI = Schedule Performance Index [SPI = BCWP / BCWS]

Figure 1. *Earned Value*

Earned Value

For this subject the book [5], *Cost/Schedule Control Systems Criteria, The Management Guide to C/SCSC*, by Quentin Fleming, is highly recommended for a more complete discussion of EV and its application to project management. For our specific application, an understanding of the EV indicators, cost performance index (CPI), and schedule performance index (SPI) is needed. To begin this discussion, refer to the point on Figure 1 labeled budget at completion (BAC). BAC is the performance expectation of the project; it identifies the cost and completion date for the project manager. Similarly, the point labeled customer expectation is the price and product delivery completion date promised to the customer. (The customer expectation is different from the planned project performance to allow for anticipated risk.)

EV management tools are based upon establishing a project baseline to achieve BAC. The project's performance is tracked against that baseline. The baseline performance is illustrated by the S-curve in Figure 1 marked BCWS, i.e., budgeted cost of work scheduled. The in-process performance tracking is facilitated by the two remaining curves shown: actual cost of work performed (ACWP) and budgeted cost of work performed (BCWP). BCWP is the earned value to date; it is a representation of the completion of project tasks and is traceable to the values allocated to project tasks during the planning phase.

During project execution, CPI and SPI provide information of performance efficiency. The CPI describes the rate of achieving earned value with respect to the funding outlay. SPI is the rate of achieving earned value with respect to the schedule baseline, BCWS. These two indicators, taken together, have been shown to be a very useful software management tool [3, 4].

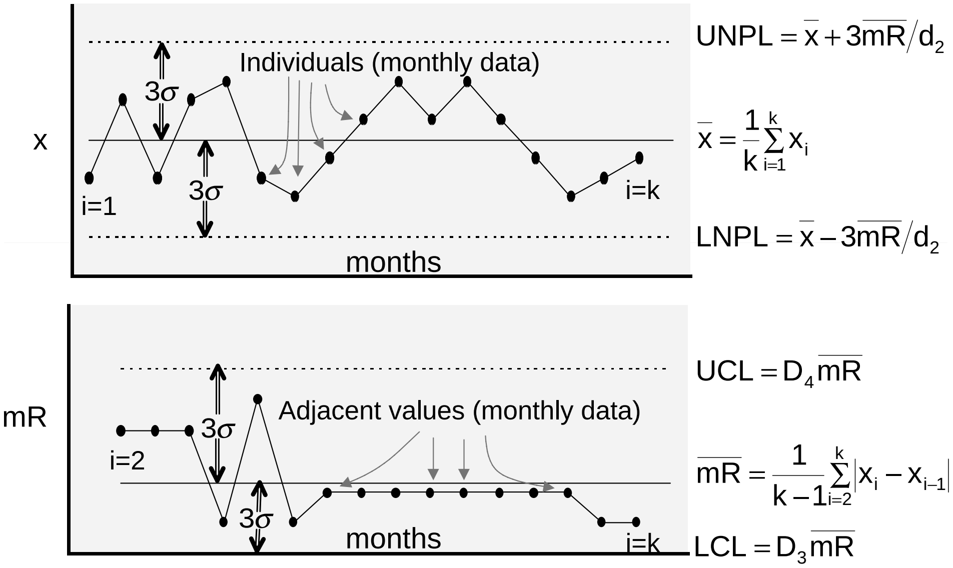
Statistical Process Control

There are several methods of performing SPC: scatter diagrams, run charts, cause and effect diagrams, histograms, bar charts, Pareto charts, and control charts [6, 7]. Although the other methods are useful, the application of control charts will be the only SPC application discussed in this paper.

As mentioned earlier, the question the software industry desires to answer is, "How do I know if my software development process is in control?" The answer is in the use of control charts. The inherent statistical variation in the process gives definition to anomalous behavior. In the statistical sense, anomalous behavior has an extremely low numerical value for its probability of occurrence. Consequently, if anomalous behavior is not observed, then the process is in control.

There are several applications for control charts with a division between two basic types, i.e., those with attribute data and those with variable data. An attribute is a characteristic that is either present, or is not. Conversely, a variable characteristic is measurable on a continuous scale. The control chart method chosen for our application is termed *Individuals* and *Moving Range*. Symbolically, it is shown as XmR, where X represents individuals, and mR is the moving range. This method can be used for both attribute and variable data. It is the appropriate control chart choice when only a single data point is available per sampling. The Individuals and Moving Range method fits our application because CPI and SPI are variables, and there is just one data point per month for each.

Figure 2 illustrates the XmR control chart. The symbol X used in the figure represents CPI^{-1} or SPI^{-1}. The symbol mR represents the difference in X's value between i = n and i = n+1. The equations, along with the value of the correction factors necessary for creating the chart are also included in the figure. The correction factors (d_2, D_3, and D_4) are derived from statistical theory, and are used to calculate the control limits of the process (i.e thresholds beyond which a measurement has an extremely low probability of occurrence).



For subgroup n=2 (i.e., adjacent data points): d2=1.128, D3=0, d4=3.268

Figure 2. *Control Chart, Individuals and Moving Range*

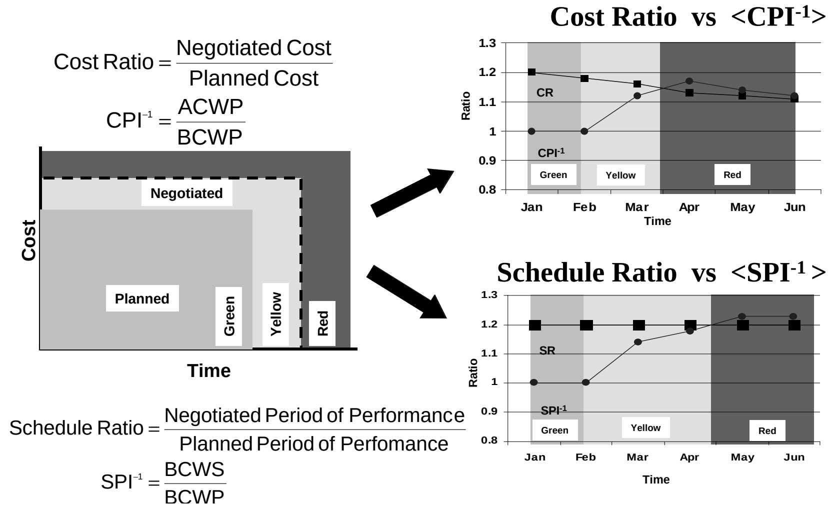
Control limits (upper and lower natural process limits [UNPL, LNPL] and upper and lower control limits [UCL, LCL]) are established at six sigma (6 σ), where sigma is a standard statistical measure of the variation in the process being observed. Measured values outside of the six sigma limits have a probability of occurrence of only 0.27 percent—virtually zero. any measured value occurring outside of these limits is an anomaly requiring management attention.

One of the applications of control limits is testing the capability of the process. For a product to be satisfactory for a customer, it will normally have a specification that establishes its acceptability in terms of upper and lower limits. By comparing the product limits to the process limits, we can predict the percentage of product not expected to meet customer requirements. The calculated value is the probability or risk of manufacturing unacceptable product.

Performance Analysis

As mentioned earlier, we have merged EV and SPC to create another software management tool [3] depicted in Figure 3. By establishing performance limits in the form of cost and schedule ratios, the average of the monthly values for CPI^{-1} and SPI^{-1} can be compared, respectively. The ratios are simply the quotients of the customer requirement to the expected project performance. The cost and schedule ratios establish the poorest perform-

Figure 3. *Performance Analysis*



ance efficiency allowed for the project to achieve the customer requirement.

It is a simple matter to color code the performance. If performance indicates the project will be completed within the planned cost and schedule, the status is green. In this case, if the average value of CPI^{-1} and SPI^{-1} (subsequently shown as $<\text{CPI}^{-1}>$ and $<\text{SPI}^{-1}>$) is reported to be 1.0 or less, then we can expect the plan to be achieved. If performance indicates the project plan will be exceeded, yet the customer requirement will be met, then the status is yellow. For this case, the values of $<\text{CPI}^{-1}>$ and $<\text{SPI}^{-1}>$ will exceed 1.0, but be less than the comparable ratio. Of course if the indicators show efficiencies in excess of the ratios (i.e., the limit above which performance is unacceptably poor), then the status is red.

Likewise from the article referenced earlier in this section [3], the color coding of the performance status leads to a recommended management action, i.e., adjustment of overtime or staffing, realignment of personnel, or customer negotiation. The calculation methods for adjusting overtime and staffing are also discussed in the article.

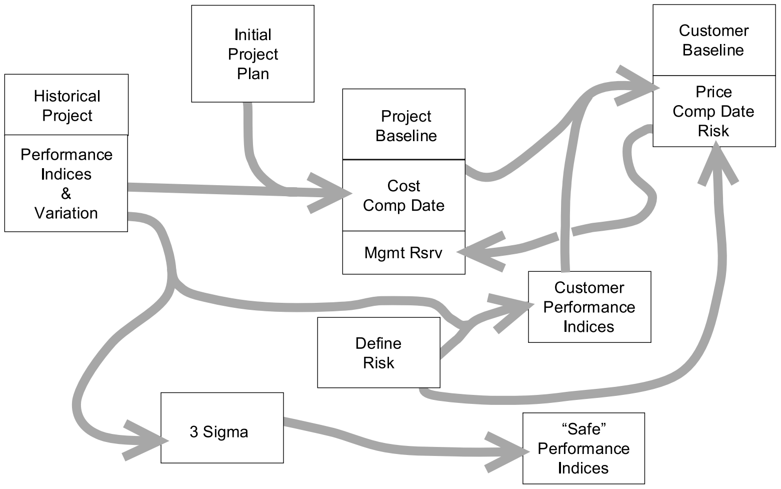
Project Planning

We have used the EV-SPC method for approximately one year as a tool for managing software developments. It occurred to us that the method could also assist with project planning. The basic idea is that data ($<\text{CPI}^{-1}>$, $<\text{mR}_c>$, $<\text{SPI}^{-1}>$, and $<\text{mR}_s>$) from past projects could be used to amend the draft project plan to establish a project baseline (cost and completion date). Using the project baseline and risk level that the company or project manager is willing to accept, computations can be made of the customer price (without profit) and delivery completion. The differences between customer values and project baseline then determine the values for management reserve. The planning process is schematically shown in Figure 4.

In the figure the computation of *safe* performance indexes is shown, i.e., the highest average values of CPI^{-1} or SPI^{-1} that will not exceed the 3 sigma variation from the historical average value. There are times when to be safe is not much more expensive in cost and schedule than for the risk defined. This condition occurs when the process is highly refined and is indicated by extremely small values of $<\text{mR}>$. Certainly, if being safe does not make the project's bid noncompetitive, it is the ideal project plan. However, in general all project managers will need to accept some risk of failure to win the contract. This planning method quantifies the risk and computes the commensurate management reserve.

Before elaborating on the computations, the expected result of this planning method needs to be discussed. Our hypothesis is that, at minimum, we could expect the new project to execute closer to the plan. After all, the adjustment to the project baseline is, in essence, the use of a lesson learned. Also, if little change is made to the planning methods, then we could expect the variation of the performance indexes from the new project to be approximately the same as those from the historical project. On the other hand, if improvements are made in the planning method or the software development process, it becomes more difficult to predict the result. Nevertheless, what we hope for is performance indexes closer to the planned values and decreased variation.

Figure 4. *Planning Method*



Calculation Procedure

This procedure follows the flow depicted in Figure 4. It may be helpful to consult the figure as the calculations are described.

Step 1. Select the historical project that has the greatest amount of similarity to the project being planned. The variables of programming language used, staff experience, software engineering environment etc., variation in the work breakdown structure (WBS), and values allocated to the tasks are to be considered in selecting the historical project. Also, the planning team's biases should be considered for historical and new projects, along with the customer's behavior attributes. Once the selection is made, obtain the historical data: $<\text{CPI}^{-1}>_h$, $<\text{mR}_c>_h$, $<\text{SPI}^{-1}>_h$, $<\text{mR}_s>_h$. Note that the performance of tasks, which are out of scope with respect to the contract, may be imbedded in these numbers. If the value of these tasks is known, it is appropriate to remove their effects and adjust the historical data accordingly.

Step 2. Develop the draft plan to establish the initial budget at completion (BACi) and period of performance (POPi).

Step 3. Create the baseline project plan by using the performance efficiencies from historical project data to adjust the initial plan.

$\text{POP}_i \times <\text{SPI}^{-1}>_h = \text{POP}_p$

[Start Date + POP_p => Expected Completion Date (CD_p)]

$\text{BAC}_i \times <\text{CPI}^{-1}>_h = \text{BAC}_p$

At this point, it is probably worthwhile for the planning team to reflect again on the differences between the historical and new projects. To finalize the baseline plan, Steps 2 and 3 may need to be iterated.

Step 4. Next, calculate the safe performance indexes for future reference. Recall from the earlier discussion that it may not cost much to be safe. Later on a comparison will be made between safe vs. risk performance.

$<CPI^{-1}>_{3[\sigma]} = 1.0 + 3 <mR_c>/1.128$

$<SPI^{-1}>_{3[\sigma]} = 1.0 + 3 <mR_s>/1.128$

The formulas used above are shown in Figure 2.

Step 5. Define the acceptable risk level. Defining risk is a tricky business. The project manager desires sufficient management reserve to cover all anticipated risk. But, the company wants to win the contract, which means accepting more risk. Therefore, the planning team feels pressure to lower the price and shorten the schedule. The level of risk selected will be a compromise between these two extremes. Generally, the risk associated with achieving the project cost is lower than the risk for delivering the product on time. The planning method allows for establishing risk levels for both cost and schedule. For the remainder of the discussion, the risk acceptable is equated to the probability of failure (P_f).

Step 6. Establish the boundary for red performance. The upper specification limits (USL_c and USL_s) for $<CPI^{-1}>$ and $<SPI^{-1}>$, respectively, are calculated from statistical data. The procedure is general to both cost and schedule; the calculations for each are performed identically. To illustrate this, an example of the calculations for 30 percent risk is shown below.

The probability of failure can be determined from the mathematical table of the cumulative normal distribution [8]:

$$P_f(z) = 1 - F(z), \text{ where } F(z) = \int_{-\infty}^{z} \frac{1}{\sqrt{2ð}} e^{-z^2/2} dz$$

(The representation of F(z) shown is one of the forms available in mathematical tables; there are other forms that can be used equally as well.)

For risk = 30 percent, $P_f(z) = 0.30 = 1 - F(z)$
Thus, F (z) = 0.70

Using the mathematical table of the cumulative normal distribution, identify F(z) values adjacent to F(z) = 0.70.

F (.52) = 0.6985, F (.53) = 0.7019

Now, estimate the value of z for 30 percent risk (z (@30 percent)) by interpolation.

$$z\ (@30\%) = 0.52 + \left(\frac{0.7000 - 0.6985}{0.7019 - 0.6985}\right)(0.53 - 0.52)$$

Thus, z (@30%) = 0.5244

Using the z equation [6], calculate the value of the USL.

z = (x – u) / [σ] , where x is the value of the point (in this example, USL), u is the average value (in this example, 1.0), and [σ] is the standard deviation (in this example, <mR>/1.128)

Therefore, z (@30%) = (USL – 1.0) / (<mR>/1.128)

With some algebraic manipulation, the equation can be solved for USL.

To illustrate the calculation, we will use a value from an actual project, <mR> = 0.2652.

USL (@30%) = (0.5244) (0.2652)/1.128) + 1.0

Performing the math, the performance index representing the red boundary for 30 percent risk is computed to be 1.1233 for our example.

Step 7. Establish the customer baseline. Knowing the value of USL, the customer baseline can be computed.

$POP_p \times USL_s = POP_c$

Start Date + POP_c = Customer Completion Date (CD_c)

$BAC_p \times USL_c$ = Price (without profit)

The above calculation should be repeated using the appropriate safe index from Step 4 in place of its respective USL multiplier. As stated earlier, to be safe may not raise the price or increase the schedule significantly if the development process is very refined (the average value of <mR> is small).

Step 8. Calculate the management reserve. The management reserve is simply the differences between the customer and project baselines.

For Schedule Reserve, $MR_s = POP_c - POP_p$ (workdays)
or
$MR_s = CD_c - CD_p$ (calendar days)
For Cost Reserve, MR_c = Price – BAC_p (dollars)

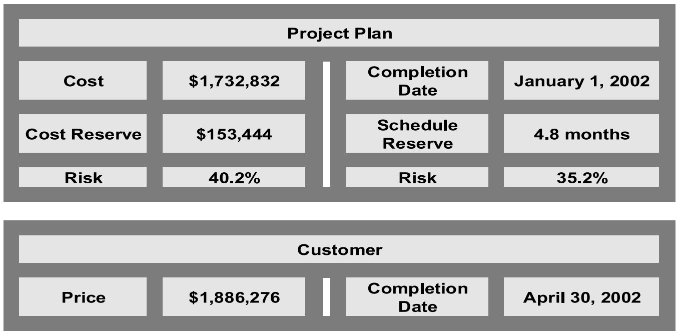
Prototype Application

The planning method and calculation procedures were tested against both a historical and current project. The in-work project was planned using knowledge gained from the performance of the historical project. For reference, the products from the historical project were 52 test program sets (TPS). TPSs are a combination of the software and hardware needed to test the performance and diagnose the failures of electronic circuit cards. The project ran for five years and had a peak staffing of 12 engineers. The in-work project was to develop nine TPSs. It has eight engineers presently assigned, and is 29 percent complete.

As we discussed in the calculation procedure, better results are expected when data is used from historical projects having similar attributes to the project to be planned. In this case, the projects are highly similar. The circuit cards requiring the TPSs are similar in structure, application, and component technology. The automated test system, which uses the TPSs for performing the circuit card testing, is the same for both projects. The programming language used for developing the software is also identical for both. The specification defining the TPSs came from the same customer; there are only minor differences. The planning team, project leader, and three engineers are common to both projects. The planning team used the WBS from the historical project with only minor changes. Schedules for the two projects overlapped for approximately nine months.

The historical project has the following performance values: $<CPI^{-1}> = 1.12$, $<mR_c> = 0.40$, $<SPI^{-1}> = 1.18$, $<mR_s> = 0.46$. The numbers were derived from 61 monthly data points. The $<CPI^{-1}>$ and $<SPI^{-1}>$ values indicate poorer than expected performance. While the project had enough cost reserve to avoid an overrun, it did not plan for schedule reserve and did experience problems completing the product deliveries on time. The amount of variation is somewhat larger than the values of <mR> from our other software development areas. Thus, it was thought that better planning was needed.

The results of using the EV-SPC planning method are shown in Table 1. Here you see that the planning team covered a greater amount of schedule risk (35 percent probability of failure) than was accommodated for cost (40 percent probability of failure). To mitigate the anticipated risks, the cost reserve is 8.8 percent of the expected project cost, and the schedule reserve is 15.6 percent of the planned period of performance. As a point of emphasis, the methods described provide a significantly more



Project Plan			
Cost	$1,732,832	Completion Date	January 1, 2002
Cost Reserve	$153,444	Schedule Reserve	4.8 months
Risk	40.2%	Risk	35.2%

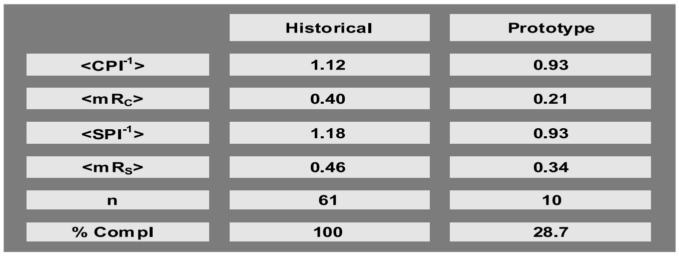
Customer			
Price	$1,886,276	Completion Date	April 30, 2002

Table 1. *Method – Results*

complete understanding of the probability of project failure. By having the risk strategy quantified, we can logically expect improved business practice and software project management.

The performance results taken from the new project bear out the hypothesis made earlier. These results are presented in Table 2. As predicted the inverse performance index values are closer to the expected values of 1.0 than for the historical project; cost is closer by 41.7 percent while schedule is closer by 61.1 percent. However, the statistical variation seen in the new project is considerably smaller, something we did not expect. Cost variation is reduced by 47.5 percent, and schedule variation is down by 26.1 percent.

Table 2. *Performance*



	Historical	Prototype
<CPI^{-1}>	1.12	0.93
<mR_C>	0.40	0.21
<SPI^{-1}>	1.18	0.93
<mR_S>	0.46	0.34
n	61	10
% Compl	100	28.7

Reflecting on the two projects, adjustments were made to the earned values that were assigned the various WBS tasks during the new project planning. The expectation of improving the project planning is the removal of statistical variation from some of the common cause [6, 7] entities. But overall, the reduction is enhanced from other lessons learned by the planning team, project leader, and the three experienced engineers. They used those lessons from the historical project to guide themselves, and especially the new members of the project team. Thus far many pitfalls experienced during the historical project have been avoided. Significant process improvement is evident.

Summary

The software project planning method proposed in this article incorporates the use of past project performance data, earned value management, and statistical process control. The method provides several outputs:

- Project cost and customer price.
- Expected and customer completion dates.
- Management reserve for both cost and schedule.
- Quantified risk for cost and schedule.
- Statistical quantification of process improvement.

We have shown in the example application that the method may have merit. In general, the results predicted were observed. The new project is performing closer to the plan with less variation. Certainly this is improved software project performance. And with improved performance, it is expected that customers will be increasingly satisfied. Of course the bottom line to achieving customer satisfaction is gaining additional business. We believe the software management tool presented in this article can help to achieve these goals.◆

References

1. Gordon, Creaghe, Risk Analysis and Cost Management, 16th Annual College of Performance Management Conference, May 2000.
2. Sen, Surhita, EVM Concepts in SPC, 16th Annual College of Performance Management Conference, May 2000.
3. Lipke, Walt and Vaughn, Jeff, Statistical Process Control Meets Earned Value, CrossTalk, June 2000.
4. Lipke, Walter H., Applying Management Reserve to Software Project Management, CrossTalk, March 1999.
5. Fleming, Quentin W., *Cost/Schedule Control Systems Criteria, The Management Guide to C/SCSC*, Probus, Chicago, 1988.
6. Pitt, Hy, *SPC for the Rest of Us*, Addison-Wesley, Reading, Mass., 1995.
7. Florac, William A. and Carleton, Anita D., *Measuring the Software Process*, Addison-Wesley, Reading, Mass., 1999.
8. Crow, Edwin L., Davis, Francis A., and Maxfield, Margaret W., *Statistics Manual*, Dover Publications, New York, N.Y., 1960.

About the Authors

Walt Lipke is deputy chief of the Software Division at the Oklahoma City Air Logistics Center. The division employs approximately 600 people, primarily, electronics engineers. He has 30 years of experience in the development, maintenance, and management of software for automated testing of avionics. In 1993 with his guidance, the Test Program Set and Industrial Automation (TPS and IA) functions of the division became the first Air Force activity to achieve Level 2 of the Software Engineering Institute Capability Maturity Model® (SEI CMM). In 1996, these functions became the first software activity in federal service to achieve SEI CMM Level 4 distinction. The TPS and IA functions, under his direction, became ISO 9001/TickIT registered in 1998. These same functions were honored in 1999 with the IEEE Computer Society Award for Software Process Achievement. Lipke is a professional engineer with a master's degree in physics.

Mike Jennings is chief of the Avionics Test Software Section, an organization within the Software Division at the Oklahoma City Air Logistics Center. His section develops and maintains software for the automated testing of avionics systems from several weapon systems. He has more than 10 years of automated testing experience in various software development and leadership positions. Jennings earned a bachelor's degree in electrical engineering from Oklahoma State University.

OC-ALC/LAS
Tinker AFB, Okla. 73145-9144
Voice: 405-736-3341
Fax: 405-736-3345
E-mail: Walter.Lipke@tinker.af.mil
Michael.Jennings@tinker.af.mil