Avoid Self-Inflicted Wounds in Applying CMM to ATP and Support

By David B. Putman
Hill Air Force Base

If you have ever found yourself thinking that the Capability Maturity Model (CMM®) does not apply to you, you are not alone. Unfortunately, you may not be aware that the source of the problem may not be the CMM. The cause generally goes back to the method the organization chose for implementing the CMM concept. Size and critical computer resources are classic examples of areas in which the organization may need to step out of the box in order to look at the underlying concept related to requirements implementation.

Exploring the Size Metric

Years ago someone in our organization defined the size of a project as source lines of code (SLOC). This became a size metric. The waivers for tracking size quickly followed; the rationale being, SLOC does not make sense when providing a maintenance level of support or designing hardware. This quickly led to more fuel for the fire as to why the CMM did not apply in numerous areas. Many failed to consider an alternative size metric. Those that developed alternative metrics, however, often failed to recognize that the alternative did indeed meet the concept of the size metric.

We eventually broadened the definition of the size metric so it could be applied to all projects. I recently explored the size concept with Will Hayes, senior member of the technical staff of the Software Engineering Institute (SEI). This discussion helped confirm that the present concept of the size metric should have been implemented when the Project Planning Key Process Area was originally addressed. The only problem was that parties were so busy arguing SLOC that they failed to see what was right in front of them.

One of the concepts in the CMM is to document the process used when preparing an estimate (i.e. capture the thought process, data, etc.). Documenting the estimating process reduces the dependency on expert opinions and improves the repeatability of the estimates. Comparing the actuals to the estimates helps improve the accuracy of the next estimates.

The CMM refers to size in relation to estimating the cost and schedule required to develop a product. With that concept in mind, a simple definition of cost and schedule can be defined as:

Cost = Size * Productivity in Dollars
Schedule = Size * Productivity in Days

Where,

Size = a measure or indicator of the amount of work to be performed in terms other than dollars or hours;
Productivity = a cost or schedule metric that indicates the rate at which the measurement of work can be performed.

As the product is decomposed into smaller elements and the organization better understands its capabilities, the equations may be expanded as shown below:

$$\text{Cost} = (\Sigma S_X{*}P_{X \text{ in Dollars}}) * (1 + \text{Percent\_Risk})$$
$$\text{Schedule} = (\Sigma S_X{*}P_{X \text{ in Days}}) * (1 + \text{Percent\_Risk})$$

Where,

S_X = Size of a particular task or part of the product;
P_X = The productivity to perform the task or develop that part of the product;
Percent Risk = a optional percentage that addresses such areas as:

— A range (e.g. from 0 to 0.25) that takes into account the team's learning curve, training, experience and motivation. For example, if the project is assigned to the top performers in the organization, the product may be completed as originally estimated; if the project is assigned to less experienced team members the project may take 25 percent longer to complete.
— A correction factor for the estimator's bias. In this example the estimate is dependent upon an expert's opinion. This correction factor recognizes that the time it would take the expert to complete the task may vary from the time it would take the typical employee to complete the task.
— Potential impacts resulting on dependencies on subcontractors, procurement, or other activities outside the organization's control. For example, many estimates will include a schedule buffer that takes into account the average time that it takes for the procurement of piece parts. Even though the average time has been taken into consideration, there is a risk associated with the fact that the parts may not be received in the average time frame.

The sections below explore using the size metric when preparing estimates for efforts related to developing and maintaining automatic test equipment (ATE) product. The sections are broken out in the following areas:

- Conceptual approach for estimating the cost for ATE Test Program Set (TPS) maintenance.
- Conceptual approach for estimating TPS development.
- Conceptual approach for estimating test station replacement and sustainment activities.

Estimating ATE TPS Maintenance Cost Conceptually

Applying the concept of size to maintenance activities is fairly easy, but the managers of ATE TPS software maintenance activities often look at the size metric from the wrong perspective. Maintenance estimates can be calculated using the following definitions:

Size = The number of maintenance tasks (analysis/updates) that can be anticipated over a specified time frame (e.g. a quarter or a year). A review of historical data and trends can quickly result in a size estimate.

$\text{Productivity}_{\text{in Dollars}}$ = The average cost per maintenance task.

$\text{Productivity}_{\text{in Days}}$ = The average cycle time per maintenance task.

The necessary manpower to support the anticipated workload can be easily calculated once cost and schedule information has been estimated. Using the definition of size identified above, the size metric can easily be tracked. Examples of items related to size that could be tracked include:

- The number of maintenance tasks received each month.
- The number of maintenance tasks closed each month.
- The number of maintenance tasks open at the time of the

monthly snapshot.

- The number of maintenance tasks in a work stoppage condition (i.e. the work stoppage is out of the control of the organization) at the time of the monthly snapshot.
- The average number of maintenance tasks per employee at the time of the monthly snapshot.

The workload level may have an impact on average cost and schedule. Many ATE customers will fund for a guaranteed level of maintenance support to cover a specific time frame. In this situation the average cost or schedule (cycle time) may be highly dependent upon the level of the workload in comparison to the guaranteed level of support. Figure 1 uses the concept of the economic supply and demand curves to represent the maintenance workload.

The curves shown in Figure 1 demonstrate that if the customer sends the team one task after funding for a guaranteed support level of $1 million then the cost per task is $1 million. The average cost per task decreases as more tasks are received until the average cost per task stabilizes when the team is fully loaded with work.

On the left side of the chart the average cycle time per task may start out higher than necessary due to the employees' concern about their future. On this side of the chart the employees may feel that they are faced with the dilemma of working themselves out of a job vs. *nursing* the project. This dilemma may lead to morale issues even though the team may be fully funded for the current time frame.

The average cycle time per task in Figure 1 will start to increase as resource limitations (manpower, equipment availability, etc.) start impacting the workload.

The optimum point for both cost and schedule occurs in the chart where the two lines cross on the graph (point B). Changes in the data must be well understood to determine whether the process is getting better or worse, or if the level of the workload is causing a shift along the curve. Process changes will raise or lower the curve. Workload changes (e.g. the number of tasks received) may cause a shift along the curve to the right or the left. Other workload changes, such as changes to the average complexity of the workload, may raise or lower the curve.

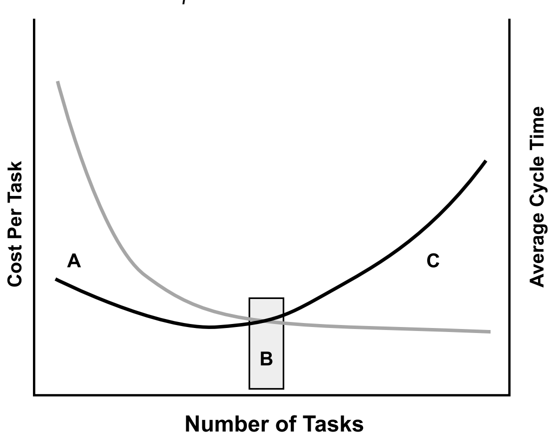
Estimating TPS Development Conceptually

In the early 1980s, I was given a cookbook formula for estimating TPS development efforts. This formula was developed in the '70s and worked well for circuit boards that contained small-, medium-, and some large-scale integration circuits. The formula was based upon the total number of integrated circuit (IC) pins in reference to the total number of input and output pins on the circuit card. The original formula did not work well as the integrated circuits became more complex. However, a variation of this original concept may work very well for estimating the size of the work to be performed for developing TPS.

Assigning a weighting factor to the various IC families can enable the project lead to calculate a number representing the complexity of the circuit card. It may also be possible to estimate the testability of the circuit card by comparing the number of input and output pins to other known parameters on the unit under test (UUT). Table 1 gives an example of how the complexity of the circuit card might be estimated in determining the amount of effort necessary to develop the TPS software. This example does not include developing component models for automatic test program generator simulators, interface test adapter (ITA) fabrication and other TPS development tasks that also need to be included in the final estimate.

Taking advantage of historical data, an organization can explore the weighing concepts discussed above in an effort to develop a reasonable correlation between the UUT complexity and the hours expended during the TPS development. The scatter

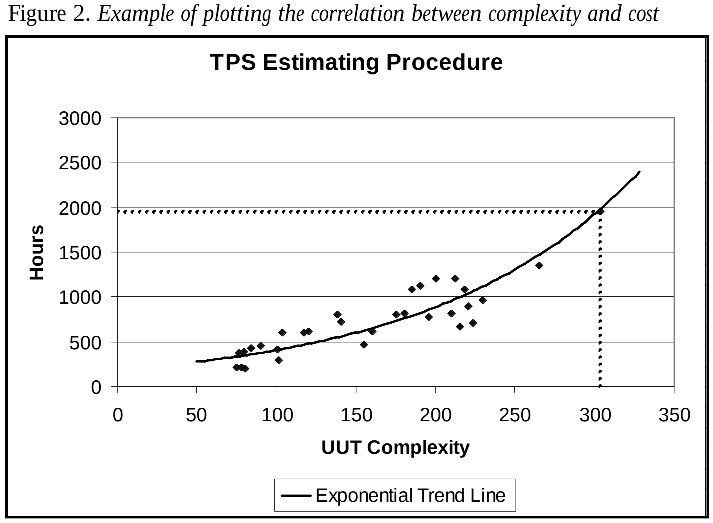
Figure 1. *Cost vs. schedule for maintenance tasks*



Component	Quantity	Weight	Rough Complexity
Small-, medium-scale integration	14 *	1 =	14
Counters, shift registers, etc.	8 *	2 =	16
Memory devices (programmable (array logic, read-only, random access …).	16 *	4 =	64
Communication devices (universal asynchronous receiver transmitter, RS-232, IEEE-488, serial, parallel…)	4 *	15 =	60
16 bit microprocessors, micro-controllers, …	1 *	75 =	75
32 bit microprocessors	0 *	100 =	0
Testability = [(quantity of ICs) * (20 pins/IC avg.) / (Total number of input/ output pins)] **2	= [(43*20)/100]**2		74
		TOTAL Complexity =	303

Table 1. *Example of a method to calculate a numerical complexity value*

Figure 2. *Example of plotting the correlation between complexity and cost*



Software Description	No. of Functions	Time per Function	Total Time (in hours)
Digital multi-meter software drivers ((type of measurement, scale, filter, front/rear …)	5 *	30 =	150
Timer/counter (type of measurement, scale, impedance …)	5 *	30 =	150
Power supply drivers: five identical supplies providing +/- 20 VDC, 10 ADC power supplies (voltage, current)	2 *	20 =	200
Station self test (enter No. of tests) *[requirement is to … such as test each stimulus and measurement at high-scale, low-scale, and mid-range]*	57 *	15 =	855
		TOTAL Labor Hours =	1355

Table 2. *Way of documenting process to estimate labor for the test station software*

diagram in Figure 2 shows an example of correlating the complexity to the effort. Spreadsheets, such as Excel, can calculate a variety of trend lines so that the user can find the best fit to the data. Using the graph shown in Figure 2, the cost to develop the TPS code for the example in Table 1 is $\text{Cost}_{\text{in Hours}} = 2.07\ e^{(0.0074*303)} \sim 1950$ hours of labor.

The cost of the ITA design, parts, and fabrication can be calculated using a table similar in nature to Table 1 but designed to meet the needs of the ITA estimates (see also section 2.3).

Estimating Test Station Replacement and Sustainment Activities Conceptually

Refurbishing or replacing the test stations involves similar types of work as developing a TPS from scratch. The activities include the design, the purchase of equipment and piece parts, the fabrication, and the development of software drivers, station self tests, and other software applications (e.g. test executives, post processors, program debuggers, TPS analysis applications, etc.).

Table 2 shows an example of how the software costs may be estimated. An example of estimating the hardware costs is shown in Table 3. Most ATE leads are very familiar with preparing a cost breakdown as shown in Table 3 for the hardware costs, but the similar practice for the software costs as shown in Table 2 does not seem to be as common. A similar table could also be developed to document the estimated fabrication costs of the items such as the cables, installing the instruments into the station, installing the cooling fans, etc.

Exploring Concepts Behind Critical Computer Resources

A discussion on risks is warranted before exploring the CMM concept for managing critical computer resources. From a simplified viewpoint risks can be grouped into two areas:

- Risks that may impact the team's ability to develop the product.
- Risks that may impact the product's ability to meet the performance requirements.

From a pure software viewpoint the critical computer resources (CCR) are the risks that may impact the product's ability to meet its performance specification. TPS developers and maintainers have been quick to point out that CCR is not appli-

Hardware Description	Quantity	Estimated Cost	Total Cost
Digital multi-meter	1 *	$2,500 =	$2,500
Timer/counter	1 *	$2,500 =	$2,500
DC power supplies	5 *	$1,000 =	$5,000
Oscilloscope	1 *	$15,000 =	$15,000
Wave form analyzer	1 *	$8,500 =	$8,500
IBM-compatible PC system (computer, monitor, keyboard…)	1 *	$5,000 =	$5,000
Piece parts (mating connectors, pre-fabricated cables, power strips, fans … *this should be done at a reasonable level in an itemized format)*		$8,250 =	$8,250
			TOTAL Equipment Cost = $46,750

Table 3. *Way of documenting process to estimate cost of the test station hardware*

cable (or rarely applicable) in the ATE environment. Removing the focus on the word *computer* reveals that the concept of managing critical resources is applicable in the ATE environment.

Tables 4 and 5 show two of the formats that an organization may choose for assigning a risk factor (R_F) to each potential risk. These tables assign a probability of occurrence and a severity to each of the risks identified. The tables also provide a method for determining a R_F that relates to the action required for each risk. The R_Fs used in Table 6 are based upon the R_Fs identified in Table 5 and assume that the organization has defined the actions as:

R_F = 1: No follow on action is required.
R_F = 2: The risks will be monitored and the probability and severity updated when necessary.
R_F = 3: A risk mitigation strategy will be developed.

By categorizing of the risks as development and performance risks, simple check sheets can be developed that will help in identifying and tracking them. For example, the left column of Table 6 could be used as a boilerplate or check sheet for TPS development risk management activities. It is highly probable that the

Table 4. $R_F = P * S$

<=100%	5	5	10	15	20	25
<80%	4	4	8	12	16	20
<60%	3	3	6	9	12	15
<40%	2	2	4	6	8	10
<20%	1	1	2	3	4	5
Probability		1	2	3	4	5
		Low		Med		High
		Severity				

Table 5. R_F = *the value identified in the cell*

<=100%	1	2	**3**	**3**	**3**
<80%	1	1	2	**3**	**3**
<60%	1	1	2	**3**	**3**
<40%	1	1	2	2	**3**
<20%	1	1	1	2	2
Probability	Low		Med		High
	Severity				

Table 6. *TPS Performance Risks*

Risk	P	S	R_F	Action
Product Development Risks				
Critical Personnel: Key personnel, critical to the successful completion of the product, may leave the organization.	1	5	2	This concern will be monitored
Support Environment: The organization may be unable to provide the necessary support environment necessary for the development of the product (e.g. computer access, software application tools, testing and integration environment.).	4	5	**3**	**Mitigation Plan:** The success of the project is highly dependent upon the availability of the ATE for integrating and testing the TPS. The owners of the ATE (production shop) have signed the SOW showing their intent to support the development to the maximum extent possible. However, production items take precedence over developmental TPSs. This risk has been identified in the proposal and any cost and schedule impacts will be negotiated with the customer if sufficient ATE is not available.
Procurement of Piece Parts: The organization may be unable to get the piece part hardware in a timely manner.	3	5	**3**	**Mitigation Plan:** The development schedules for the TPSs were expanded to allow xx days for the procurement of the parts necessary for the ITAs. The xx day schedule extension for each TPS was based upon the historic average of the number of days we have waited for the delivery of parts.
Product Performance Risks = Resources critical to the Performance of the product				
Available RAM: The amount of RAM available in the CPU may impact the successful operation of the product.	1	1	1	Automated segmentation utilities are used to assure the program segments do not exceed 80 percent of the available RAM.
Throughput: The CPU throughput (speed, run-time, etc.) may impact the successful operation of the product.	1	1	1	N/A:
Available Disk Space: The amount of disk space may impact the successful operation of the product.	1	1	1	N/A: The ATE has sufficient disk space to host approximately xx TPSs. Unused TPSs are deleted by the ATE operator (when necessary) to free up disk space; these TPSs can be quickly reloaded should they be needed in the future.
Other Test Station Resources (Power)				
Unit Under Test (UUT Power: The ATE may be unable to meet the power requirements for the UUTs (e.g. No. of DC/AC power supplies, voltage levels, current requirements, ripple, etc.).	5	4	**3**	**Mitigation Plan:** Full load testing of the gun controller circuit card requires providing 28 VDC at 50 ADC to the gun firing . circuitry. The power supplies in the ATE cannot provide this requirement. Two options have been identified in the proposal (1) use an external power supply to provide the power or (2) do not test the circuit under full load. The first option raises the development costs and increases the shops support costs (calibration, repair and spares), the second option has a risk of not catching a small percentage of the darlington transistor failures. We will implement the solution that is negotiated with the customer.
Cooling: The ATE may be unable to provide the cooling necessary for testing the UUT.	2	1	1	N/A: There is a small risk that certain UUTs may need cooling that has not been identified in the test specifications. If necessary, small fans can be installed in the ITA.
Other Test Station Resources (Input Signals)				
Waveforms: The ATE may be unable to provide the necessary waveforms to meet the requirements of the UUTs (e.g. number of signals, frequency, amplitude, shape, etc.).	2	3	2	Monitor: Ancillary equipment of additional hardware design may be necessary.
DC reference: The ATE may be unable to provide the necessary DC references to meet the requirements of the UUTs (e.g. number of signals, voltage level, precision).	2	3	2	Monitor: Ancillary equipment of additional hardware design may be necessary.
Pneumatic Inputs : The ATE may be unable to provide other necessary input signals.	5	5	3	The Sample Data Assembly requires a special timing signal in order to function properly. This timing signal is generated from an on-board clock signal. The proposal includes the cost and schedule necessary to design and instal this function into the interface test adapter.
Other Test Station Resources (Measurement System)				
AC Voltage Measurements (range, resolution, accuracy, etc.)				N/A: Calculations can be made during the development of the TPS to convert Peak-to-Peak values stated in the specifications to true RMS measurement readings.

risks shown in Table 3 were considered during the development of the TPS proposals, but often the results of this thought process were not always documented.

In looking at TPS maintenance activities, the problem analysis may reveal that one of the TPS Performance Risks identified in Table 6 is the cause of the identified problem. However, with the emphasis switched from TPS development to TPS maintenance, a performance problem is no longer a risk but an issue that must be addressed. In this case the organization may chose to condense all individual ATE resources identified in Table 6 to a single entry such as:

Risk: The engineering analysis may reveal that the equipment in the ATE and ITA may not meet the performance requirements of the UUT.

Mitigation: None. The engineering analysis and recommendation report sent to the customer will identify the performance issue and when possible make recommendations as to how the performance problem can be corrected.

Conclusion

The original intent of this paper was to show ways that the CMM could be applied in the area of supporting automatic test programs. TIS has gone a step farther by removing the software emphasis in TIS policy and guiding documentation; this enables us to apply the CMM concept to hardware engineering as well as software engineering. Hopefully this paper will help others who are struggling with CMM implementation issues to step out of self-perceived boundaries and to further explore the project management concepts behind the CMM.◆

About the Author

David B. Putman is the Chief of the Operational Flight Program branch in the Technology and Industrial Support Directorate, Software Engineering Division at Hill Air Force Base, Utah. He has more than 21 years experience in ATE, two years with Hughes Aircraft, and more than 19 years with the Air Force. During his involvement with ATE, he was the senior engineer within the Avionics Software Support Branch for nine years, and he supervised ATE engineering teams for two years. He supervised the F-16 OFP system design and integration test teams for one year, and he was the SEPG lead when OO-ALC was assessed at a CMM Level 5. He has a bachelor's degree in electrical engineering from the University of Utah and a master's degree in business administration from Utah State University.

OO-ALC/TISF
7278 Fourth Street
Hill AFB, Utah 84056
Voice: 801-777-4726
Fax: 801-777-8069
E-mail: david.putman@hill.af.mil

New CMMI℠ Product Integrates Processes

PITTSBURGH—Now organizations currently using different models for separately improving systems and software engineering can use one newly released model to improve, train and assess process more commonly and consistently.

CMMI-SE/SW Version 1.0, an integrated model for systems engineering and software engineering improvement, was released in August 2000. The integrated model is designed for product-development organizations to improve their engineering and project-management processes. It incorporates the best features of its source models: Capability Maturity Model for Software (SW-CMM®) V2.0 draft C and EIA/IS-731 Systems Engineering Capability Model (SECM).

This new model will enable organizations to build on previous investments in improvement based on the SW-CMM or the SECM, and at the same time to benefit from the standardization and commonality of the integrated model.

It was developed by the Capability Maturity Model Integration (CMMI℠) Project, a collaborative effort sponsored by the Office of the Under Secretary of Defense for Acquisition, Technology, and Logistics and the National Defense Industrial Association with participation by government, industry, and the Software Engineering Institute.

Use of Electronic access to CMMI-SE/SW V1.0, and more information about the CMMI product suite, are available at www.sei.cmu.edu/cmmi or by calling SEI customer relations at 412-268-5800.

Bill Pollak, public relations coordinator
Software Engineering Institute
Office: 412-268-5656 Fax: 412-268-5758
CERT/CC media relations: 412-268-4793

The updated CrossTalk Theme Announcement is now available on the World Wide Web at www.stsc.hill.af.mil/CrossTalk/themes.doc **Newly-added themes include Software Engineering Careers, Web-Based Applications, and Software Odyssey: Cost, Schedule, Quality. Call Heather at 801-586-0095 for more information.**