Earning Value the Agile Way: Using Story Points to Generate EV Metrics

By Eric Christoph, PMP, EVP

May 16, 2011

Project managers who want or need to provide Earned Value (EV) metric data on their Agile development project must determine how to generate earned value without incurring the planning and execution overhead typical to Earned Value Management (EVM) and which Agile was created to avoid. In order to do this, managers need to find common ground between the two methodologies and understand the strengths and weaknesses of each. This paper discusses the difference between the EVM and Agile approaches, and describes a method for generating EV metrics on an Agile project without sacrificing the advantages of Agile.

Since the publication of the Agile Manifesto in 2001 Agile has become an accepted and in some cases required approach within the IT industry, inspiring many "Agilistas" to question existing project management best practice as defined by organizations such as the Project Management Institute (PMI) and the US Federal Government. In response, PMI has recently developed the PMI Agile Certified Practitioner certification and the US Congress passed a law[1] that directs the Secretary of Defense to develop a new acquisition process for IT systems. Some project managers in the federal space are even seeing contract requirements to use both approaches on the same project.

Earned Value Management vs. Agile

In the IT arena EVM and Agile are competing management approaches. The majority of EVM and Agile practitioners are familiar with only one or the other of these approaches, as the two are not often seen as natural complements. In addition, there are widespread misconceptions about both approaches and plenty of examples where each has been abused, misused, or otherwise made a scapegoat for poor management or engineering practice. Before discussing how to generate EV metrics for an Agile project it will be useful for many readers to review and compare the two in terms of how they support the management of the project.

Earned Value Management 101

Earned Value Management (EVM) is an approach that compares the cost and time that was planned for completing a project with the cost and time actually spent. The key assumption in the approach is that the project manager is able to define what it means to be "complete". Defining project completion starts with the creation of a work breakdown structure (WBS) as shown in Figure 1:

[1] US Public Law 111-84, The National Defense Authorization Act for Fiscal Year 2010, Section 804

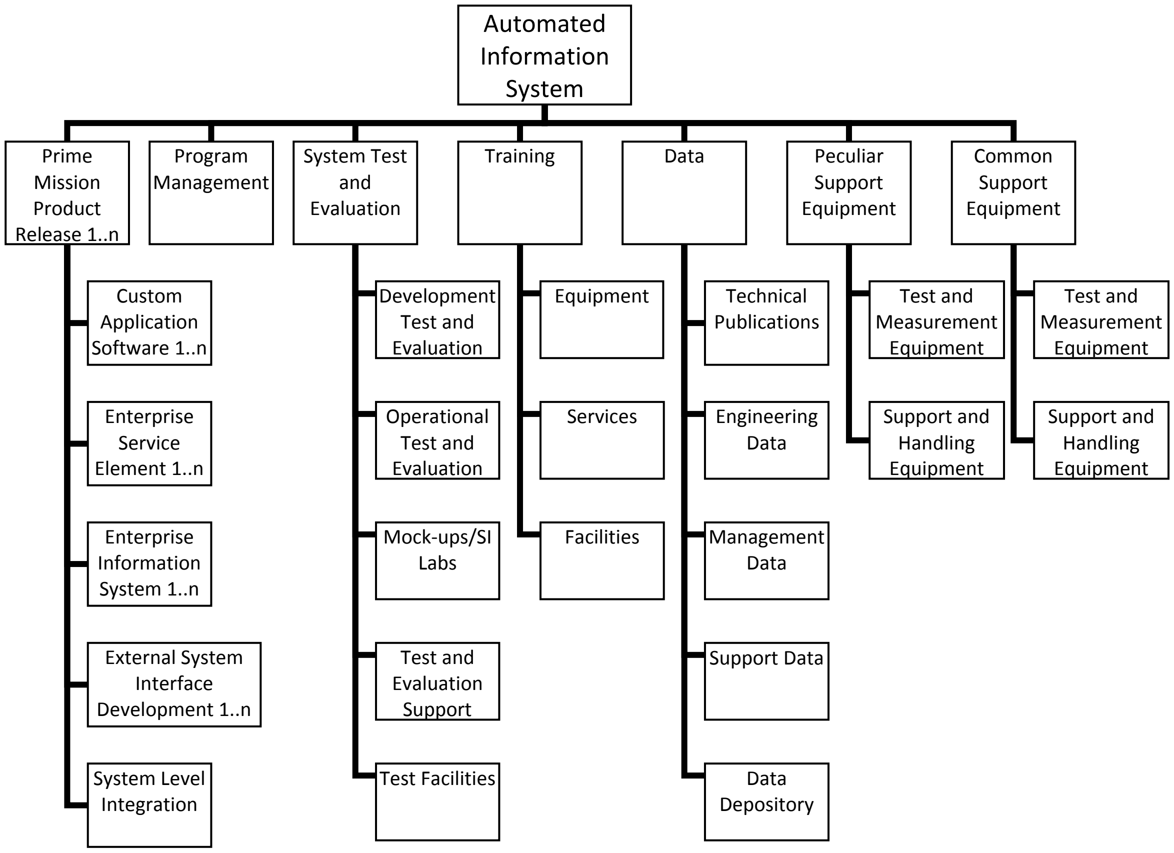
Automated Information System
Prime Mission Product Release 1..n
Program Management
System Test and Evaluation
Training
Data
Peculiar Support Equipment
Common Support Equipment
Custom Application Software 1..n
Enterprise Service Element 1..n
Enterprise Information System 1..n
External System Interface Development 1..n
System Level Integration
Development Test and Evaluation
Operational Test and Evaluation
Mock-ups/SI Labs
Test and Evaluation Support
Test Facilities
Equipment
Services
Facilities
Technical Publications
Engineering Data
Management Data
Support Data
Data Depository
Test and Measurement Equipment
Support and Handling Equipment
Test and Measurement Equipment
Support and Handling Equipment

Figure 1: Partial Work Breakdown Structure for a Military IT System

The WBS is used to organize the scope of the work. Once the work is organized, each element is analyzed and a cost and schedule estimate is developed to support a time-phased plan called the performance measurement baseline (PMB). Value is “earned” during project execution when scope is completed. At any given point in time the variances between the costs incurred, the amount that had been planned to be spent, and the value of scope completed (earned value) are calculated and used to forecast the project completion cost and time. Figure 2 below shows an EVM performance chart with plan, actual costs, earned value, and various forecasts:

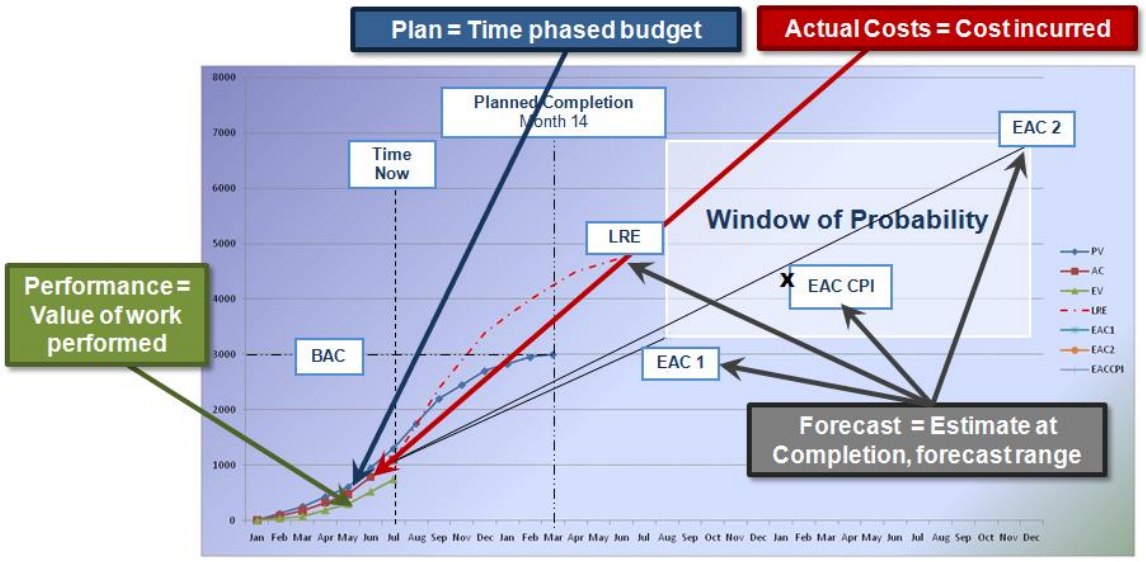

Figure 2: EVM Performance Chart

In EVM, all value is determined by the plan, which makes scope definition and control absolutely critical to realizing the benefits of the approach. If you can adequately define and manage scope then you have the ability to identify performance variances early, take corrective action, and accurately forecast project completion. Used correctly, EVM supports extremely efficient and effective completion of all types of projects.

The Agile Alternative

Agile development is in many ways a response to poor management, whether that be project, engineering, or corporate. A review of the history of the establishment of Agile finds references to "Dilbertesque" organizations and the need for "freedom from the inanities of corporate life[2]." The Manifesto for Agile Software Development reads, in its entirety, as follows:

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

- *Individuals and interactions over processes and tools*
- *Working software over comprehensive documentation*
- *Customer collaboration over contract negotiation*
- *Responding to change over following a plan*

That is, while there is value in the items on the right, we value the items on the left more.

Agile represents a set of principals which are implemented through methodologies such as Scrum, XP, Test Driven Development, or Feature Driven Development. These approaches feature the use of integrated teams of developers and product owners, a

[2] http://www.agilemanifesto.org/history.html

product backlog, and iterations or sprints of varying lengths. Agile methods with extremely short or highly variable length sprints are generally less suitable for adapting earned value techniques. Fixed phase Agile approaches, such as Scrum and Feature-Driven Development, are easier because the sprint can be tuned to coordinate with required reporting cycles.

Agile methods break tasks into small increments focused on the development of specific features with the minimum necessary planning. Using Scrum terminology, the Product Backlog is the full set of requirements or user stories that could be potentially selected in future sprints. The Sprint backlog represents the defined set of requirements and/or deliverables that will be met for a specific iteration, which should be those items in the Product Backlog that are highest in business value as determined by the product owner. It is important to note that the items in the Product Backlog are allowed and even encouraged to change, but that the Sprint backlog should be fixed once a Sprint starts. Sprints last for short, fixed time frames that typically last from two to four weeks. Each sprint involves a team working through a full software development cycle including planning, requirements analysis, design, coding, unit testing, and acceptance testing when a working product is demonstrated to stakeholders. This minimizes overall risk and allows the project to adapt to changes quickly.

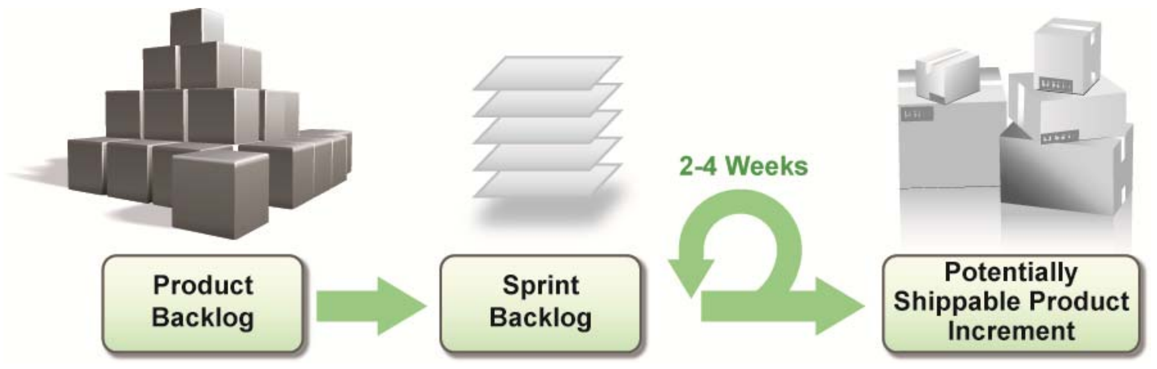

Figure 3: Agile Scrum Process

Velocity is the Agile measure of productivity. Velocity is the sum of the estimates of delivered and accepted features per iteration, and is measured in the same units as feature estimates, whether this is story points, requirements, or any other suitable unit.

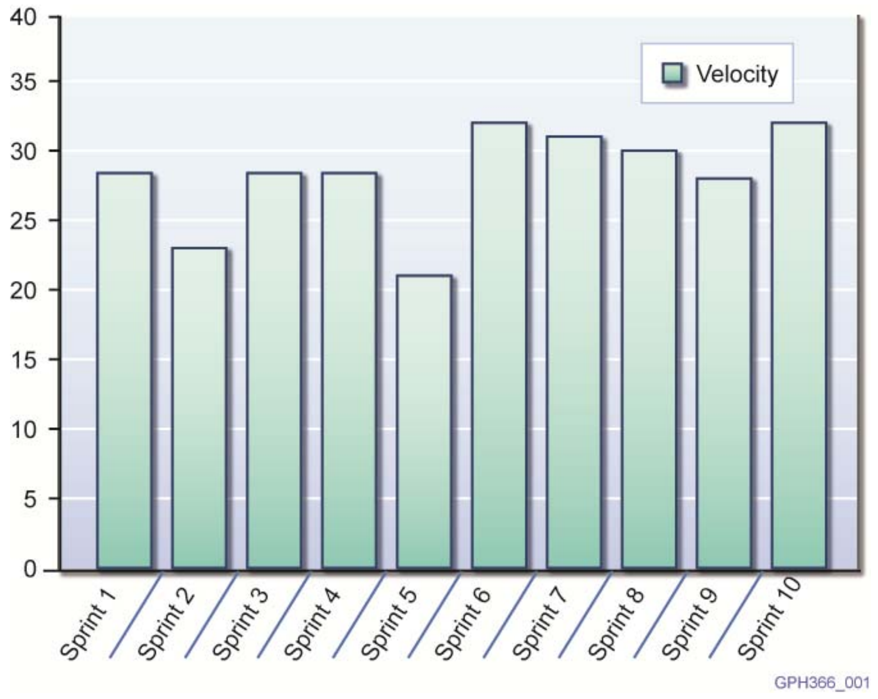

Figure 4: Velocity Chart

Since Agile typically assumes a fixed time period for each iteration and a fixed team performing the work, the Velocity measure provides a very powerful and simple to use forecasting capability.

Comparison of EVM and Agile

In terms of the iron triangle EVM tries to keep scope constant (or at least controlled) in order to manage cost and schedule while Agile works exactly opposite:

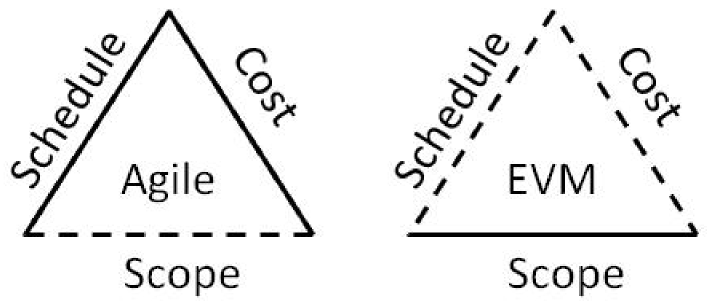

Figure 5: The Iron Triangle in EVM and Agile

The usefulness of the performance metrics that EVM and Agile generate reflects this difference in approach. As long as scope is managed, EVM metrics provide measures of actual productivity relative to plan that allow the project manager to make informed decisions about resources and the sequence of activities to be performed to bring a project in on time and on schedule. Cost and schedule performance indices are only predictive of future results if the scope of the work is well understood; having a “rubber baseline” negatively affects the usefulness of these metrics and makes forecasting less accurate.

For an Agile project, as long as the team makeup is relatively constant and the sprint durations are similar then the Velocity measure provides a valuable indicator of how long it might take to “burn down” a given product backlog. On the other hand, if the duration of sprints and the team makeup are constantly changing during an Agile project then the Velocity measure provides no predictive value[3].

In terms of which approach is preferable, Agile has tremendous advantages over EVM whenever scope is unknown or unpredictable. Examples where this situation exists include when stakeholder requirements are not defined or in operations and maintenance, for instance when prioritizing defect fixes or feature requests. In these cases Agile provides a very lightweight approach to demonstrating value to stakeholders in a frequent, iterative manner. Agile may also be preferable on smaller projects with skilled teams that understand the goals of the project and the technology very well, and where the overhead and expertise required to do detailed planning is neither desired nor required.

The weakness of Agile is that it sacrifices overall completion efficiency in order to provide immediate capability. Of particular concern on some jobs is that Agile masks this tradeoff because the approach never requires that completion be defined. Since all re-work, refactoring, retrofitting, etc. is just another product backlog item to be included in a future sprint there is no measure of how much work it took to get any component of a system developed. The development of large, complicated systems with significant interfaces between components will nearly always benefit from the planning discipline and performance visibility at the component level provided by EVM systems.

Why Would I Want EV Data on an Agile Project?

There are two common situations where a project manager might want to generate EVM metrics for an Agile project:

- When the team makeup and, to a lesser extent, the sprint durations cannot be held constant, such as in a mixed development and operations environment
- When EVM is required on a contract

EV metrics provide a measure of cost productivity which does not exist in Agile. Stated another way, CPI measures how much was developed for the money spent while Velocity only measures how much was developed. Cost productivity measures allow the project manager to show that he or she is managing stakeholder resources effectively even when problems with resource availability are impacting project feature development. In the case where EVM is required by the stakeholder, generating EV metrics allows the project manager to meet contract reporting requirements.

[3] Measures other than Velocity are used on Agile projects, including measures of customer satisfaction, feature delivery, etc. However, Velocity is the closest Agile comes to a productivity measure and is the most comparable with the EV CPI metric.

Any approach for generating EV metrics on an Agile project should maintain the simplicity and low process overhead of Agile methods. This means concentrating on approaches for getting EV metrics while being willing to forgo typical earned value management processes, such as scheduling. In order to generate EV performance metrics you need three key pieces of data:

1. A budget to accomplish a defined portion of the project scope (Planned Value, or PV),
2. A dollarized measure of accomplishment (Earned Value, or EV), and
3. The actual costs incurred by the project while completing the work (Actual Costs, or AC).

This data is used to develop EV performance metrics such as CPI and SPI:

Metric	Use
Cost Performance Index (CPI) = EV/AC	Shows accomplishment relative to actual costs incurred
Schedule Performance Index (SPI) = EV/PV	Shows accomplishment relative to planned costs budgeted

Generating Earned Value Using Story Points

Generating EV data requires that there be some level of the project where work is planned and then accomplishment is measured against that plan. In Agile, this is the sprint. At the beginning of each sprint, features or user stories in the product backlog are collaboratively evaluated and prioritized by the Agile team. Those features that are determined to provide the highest business value are moved to the sprint backlog, where they are evaluated using a set of criteria based upon complexity, experience with a similar requirement, planned difficulty, required expertise, or other appropriate factor(s). The result of this evaluation is a list of features to be completed during the sprint, each with a number of story points:

Date Submitted	Sprint Planned	Req #	Requirement	Story Points	Completed	Earned
1-Jan-11	1	25	Streamline the Current e-mail process	1	No	0
16-Dec-10	1	16	Allow the user to choose the e-mail format	1.15	No	0
30-Nov-10	1	9	Create a new front page for the application	1.05	No	0
14-Nov-10	1	12	Allow for Selection of sensitivity level for e-mail	1	No	0
29-Oct-10	1	3	Auto populate all relevent data in the entry form	1.1	No	0
13-Oct-10	1	5	Allow the document to be hard copy printed	1.15	No	0
27-Sep-10	1	7	A record of the e-mail must be kept	1.05	No	0
27-Sep-10	1	8	A full audit of emails being printed	1.15	No	0
27-Sep-10	1	9	Ability to conduct content searches	1.05	No	0

27-Sep-10	1	10	Error message to read "Invalid Entry"	1.05	No	0
29-Oct-10	1	11	Auto populate user signature	1.1	No	0
27-Sep-10	1	11	Enable page numbering on e-mails	1.15	No	0
			Total	**13**		**0**

Figure 6: Example Sprint Backlog Feature List with Story Points

So far, this is just basic Agile sprint planning (for a more detailed discussion of estimating with story points read Agile Estimating and Planning by Mike Cohn). All that is required in order to generate EV metrics is that the total costs be planned for the sprint, and that the total costs for the sprint be divided by the total planned story points to generate a story point value:

Sprints	Story Points	Hours	Hours/SP	$	$/SP
Sprint 1	13	580.00	44.62	$35,090	$2,699
Sprint 2	16	720.00	45.00	$43,560	$2,723
Sprint 3	12	580.00	48.33	$35,090	$2,924
		1,880.00		$113,740	

Figure 7: Calculating Story Point Value ($/SP)

The example in Figure 7 shows hours as well as dollars. Dollars tend to be a better indicator of value, but if they are not available hours may be used instead. As each feature or story point is completed and accepted, note that and keep a log showing when that occurred and what the story point value of the item was. All completed items should be logged in this manner regardless of whether or not they were in the original sprint backlog.

Date Submitted	Sprint	Req #	Requirement	Story Points	Completed	Earned Value
1-Jan-11	1	25	Streamline the Current e-mail process	1	Yes	$ 2,699
16-Dec-10	1	16	Allow the user to choose the e-mail format	1.15	Yes	$ 3,104
30-Nov-10	1	9	Create a new front page for the application	1.05	Yes	$ 2,834
14-Nov-10	1	12	Allow for Selection of sensitivity level for e-mail	1	No	$ -
29-Oct-10	1	3	Auto populate all relevent data in the entry form	1.1	Yes	$ 2,969
13-Oct-10	1	5	Allow the document to be hard copy printed	1.15	Yes	$ 3,104
27-Sep-10	1	7	A record of the e-mail must be kept	1.05	No	$ -
27-Sep-10	1	8	A full audit of emails being printed	1.15	Yes	$ 3,104
27-Sep-10	1	9	Ability to conduct content searches	1.05	Yes	$ 2,834
27-Sep-10	1	10	Error message to read "Invalid Entry"	1.05	Yes	$ 2,834

29-Oct-10	1	11	Auto populate user signature	1.1	Yes	$ 2,969
27-Sep-10	1	11	Enable page numbering on e-mails	1.15	No	$ -
5-Feb-11		1	Interface to existing server	1.15	No	$ -
7-Feb-11		2	User logins from O/s	1.15	No	$ -
15-Feb-11		6	Replace underlying O/s	1.15	Yes	$ 3,104
18-Feb-11		13	Enable workflow	1.05	Yes	$ 2,834
			Total	**17.5**		**$ 32,388**

Figure 8: Capturing Earned Value During a Sprint

As each requirement is completed, it "earns" value according to the story point value determined for the sprint, in this case $2,699 per story point completed.

Generating Earned Value Metrics

Project performance measurements are now available which can be utilized to track development efficiency (CPI) as well as velocity. The following example shows both metrics being tracked across several iterations. This information tells the analyst not only what got done, but how efficiently it was done and can be used to identify the impact of changing requirements on performance:

Sprints	Velocity - Plan	$ - PV	$/SP - Plan	AC	Velocity - Actual	EV	CPI (EV/AC)	SPI (EV/PV)
Sprint 1	13	$ 35,090	$ 2,699	$ 31,250	12	$ 32,388	1.04	0.92
Sprint 2	16	$ 43,560	$ 2,723	$ 45,887	17	$ 46,291	1.01	1.06
Sprint 3	12	$ 35,090	$ 2,924	$ 33,006	11	$ 32,164	0.97	0.92
		$ 113,740		$ 110,143		$ 110,843	1.01	0.97

Figure 9: Sample EV Metrics

It is important to note that EV in this example is based on a story point value that changes at each new iteration, but does not change during the iteration. In order to hold a constant story point value the entire product backlog would have to be measured against the full planned costs of the project. If the product backlog were stable enough for that to be feasible then the project should probably consider using standard EVM and not Agile.

Is This Really Earned Value Management??

In a word, no. Earned Value Management is designed to be used when scope can be held relatively constant, and it measures the actual cost of completing work relative to the planned cost to complete that work. Agile is designed to be used when scope is in flux but resources can be held relatively constant. EVM attempts to achieve cost and schedule targets through careful planning, while Agile attempts to deliver immediate business value through simplified planning and close team communication. The two approaches operate under different assumptions and their metrics are designed to work within those constraints.

However, sometimes the assumptions do not hold. When resources get re-prioritized during a project (or even during a sprint) CPI provides a measure of productivity that “floats” with the available resources. Combined with measures to track the amount of churn in the sprint backlog, the project now has a very rich data set that can be used to show the impact of changing requirements in a way visible to all stakeholders. This in turn will help drive more stable requirements definition, and that will help all projects, whether they are Agile or EVM.

Special thanks to my colleagues Linda Girdner, PMP, EVP, CPA and Brian Thomas, Ph.D., PMP, CSM for their help in developing the concepts in this paper.

About the Author

Eric Christoph, PMP, EVP is Vice President for Performance Management at L-3 Communications STRATIS. Eric has a keen interest in the design and implementation of program control systems for organizations of all types, and is a member of several related industry and government working groups, including the PMI Standards Consensus Body and the NDIA Program Management Systems Committee, where he serves as L-3 Communications representative and at-large board member. Eric's current efforts are focused on developing management information systems for large IT providers that are capable of supporting both project and service management functions. He can be reached at eric.christoph@L-3com.com.