September 2009 Vol. 12 No. 3

SoftwareTech NEWS

www.softwaretechnews.com

Managing For Success

The Data & Analysis Center for Software
https://www.thedacs.com/

http://iac.dtic.mil/dacs
Unclassified and Unlimited Distribution

Agile Earned Value and the Technical Baseline

AGILE METHODS ARE PRIMARILY USED FOR COMMERCIALLY-DEVELOPED SOFTWARE WHEN THERE ARE FREQUENT DELIVERIES OF USABLE SOFTWARE THAT MEET THE CUSTOMER'S HIGHEST AND MOST CURRENT PRIORITIES.

by Paul J. Solomon (Performance-Based Earned Value)

In the April 2009 *STN,* John Rusk published "Earned Value for Agile Development." This article augments Rusk's points by providing guidance for using Agile methods with earned value management (EVM) when the Performance Measurement Baseline (PMB) is *not* linear. Other topics are measuring progress towards defining and implementing the technical baselines and accounting for deferred functionality.

Agile methods are primarily used for commercially-developed software when there are frequent deliveries of usable software that meet the customer's highest and most current priorities. This article addresses the usefulness and implementation of Agile methods when using EVM to manage major defense programs during the Engineering and Manufacturing Development (EMD) phase.

Applicable Agile EVM points

First, a recap of John Rusk's points that apply to major programs:

- We need overall targets for scope, time and cost so that we can track our progress towards them.
- The EV for a task accrues when the task is completed. You get no points for partial completion of a task.
- The burndown chart works best when it covers a period of time that is big enough to feel like the big picture. Cover roughly 3 to 6 months…the whole set of iterations that make up one release
- You will find unforeseen nuances of the previously-identified requirements. These are "derived requirements"– essential to the implementation but not explicit in the original user-facing requirements. ..Simply implement these as they arise, without reflecting them *at all* in the EVM system (PMB).

Regarding the last point, during software design and development, features are often defined that support a configuration-managed requirement but do not change the contract scope. For example, if there is a requirement to inform the pilot if an enemy threat has been detected, the customer and contractor may later agree that the threat shall be communicated to a pilot by a flashing light on the display console and/or by an audio signal. The subsequent customer agreement to that feature is normally not a change to the contractual statement of work (SOW) or reason to change the budget.

Points not applicable to major programs

Next, John Rusk's assumptions and points that are not applicable to major defense programs follow:

- Agile projects are designed to have a linear output over time producing roughly the same amount of output in each iteration; therefore, we don't need to compute an s-curve to draw our PMB line.
- Only working software features earn value. There is no earned value associated simply with designing something – you only score points when it is designed, built and tested.

During EMD of Major Defense Acquisition Programs (MDAPs) or Major Automated Information System (MAIS) programs, neither the time-phased budget nor the planned physical outputs are linear. The buildup of resources and the mix and timing of measurable output always result in an s-curve.

Although the development of the product baseline does not produce working software, Agile methods should be considered for work products such as validated requirements or features in the technical baseline. The tasks and work products that support the completion of the technical baseline require continual customer approval, long before there are software builds and tests. The phases and work products of an EMD contract for a major program follow.

EMD Phases, Milestones and Baselines

EMD has two phases; Integrated System Design (Design) and System Capability and Manufacturing Process Demonstration

(Demonstration). During Design, the technical baseline evolves from the Capability Development Document (CDD) to the Functional Baseline to the Allocated Baseline, and finally, to the Product Baseline.

Table 1 shows the technical baselines, technical reviews, and completion criteria per *Operation of the Defense Acquisition System* (DODI 5000.02) and *Defense Acquisition Program Support Methodology* (DAPS).

A list of typical work products that support the product baseline, listed by CMMI process area, is shown in Table 2 shown on following page.

During Design, there is frequent customer interaction until the product baseline is approved. Agile methods may be useful during completion of work products that lead to the approved baseline. For example, the backlog may include the completion of trade studies, test cases, operational scenarios, and validated requirements that require customer involvement and approval. The Agile product backlog may be organized and prioritized by configuration item (CI). The output of the monthly iteration will be the scheduled subset of the CI's, system functionality, interfaces, or complete hardware and software detailed design.

During Demonstration, the "working software" is produced. However, it is not delivered to the customer for operational use as is done commercially. That will happen after EMD is concluded.

The Design and Demonstration work products that are dependent on customer approval for completion of the work package must be scheduled in the Integrated Master Schedule (IMS), with monthly milestones, and budgeted in the PMB. Then, they may be included in the Agile product backlog as Product Backlog Items (PBI). However, to use Agile methods with EVM, the following EV requirements and discipline must be applied.

Document/ Baseline	DoD Source	Description	Review Milestone Criteria
Capabilities Development Document (CDD)	DODI 5000.02 (a) performance parameters necessary to complete design of the system	Detailed operational	**Technology Development, Milestone B**
Functional Baseline	DAPS Methodology V2.0 (DAPS) (b), 4.3.1.C35	System functional requirements as captured in system specifications (functional baseline). All required system performance is fully decomposed and defined in the functional baseline.	**System Functional Review (SFR):** System's lower-level performance requirements are fully defined and consistent with the mature system concept and lower-level systems requirements trace to top-level system performance and the CDD.
Allocated Baseline	DODI 5000.02	Allocated baseline (to hardware, software, human/support systems).	**Preliminary Design Review (PDR)**
Allocated Baseline	DAPS, 4.3.1.C38, 4.3.1.Q64		**PDR** Assesses the system preliminary design as captured in performance specifications for each configuration item (CI) in the system (allocated baseline), and ensures that each function in the functional baseline has been allocated to one or more system configuration items. The software functionality in the approved allocated baseline is consistent with the updated software metrics and resource-loaded program schedule.
Product Baseline	DODI 5000.02	Defines system functionality and interfaces and complete hardware and software detailed design for each CI.	**Critical Design Review (CDR): Completion of Integrated System Design Phase:** 1. System functionality and interfaces, complete hardware and software detailed design are defined. 2. Product baseline for all CIs is established.
Product Baseline	DAPS, 4.3.1.C41, 3.2.2.Q78		**CDR:** Assesses the system final **design** as captured in product specifications for each CI in the system (product baseline), and ensures that each product in the product baseline has been captured in the detailed design documentation. The software functionality in the approved product baseline is consistent with the updated software metrics and resource-loaded schedule.
Product Baseline	DAPS, 4.2.2.C18, 4.3.1.C49		**System Verification Review (SVR):** Assess the system final **product**, as evidenced in its production configuration, and to determine if it meets the functional requirements (derived from the CDD and draft Capability Production Document) documented in the Functional, Allocated, and Product Baselines. All system performance specification qualification test requirements have been successfully completed.

(a) *Operation of the Defense Acquisition System* (DODI 5000.02)
(b) *Defense Acquisition Program Support Methodology* (DAPS)

Table 1: Technical baselines, technical reviews, and completion criteria

Maintain the EVM Baseline

There are some fundamental problems in planning and measurement when using Agile methods within EVM's process constraints. Most problems stem from Agile's focus on

EMD Phase	Review Milestone	Typical Work Products by CMMI Process Area (PA)
Integrated	CDR	• Decision Analysis and Resolution PA • Trade studies • Requirements Development PA • Requirements for verification/validation • Test cases and expected results • Derived requirements • Product and product-component requirements • Interface requirements • Interface requirements • Product component operational concepts, scenarios and environments • Technical Performance measures (TPM) • Validated requirements • Technical Solution PA • Product component operational concepts and scenarios • Technical data package • Product-component descriptions • Key product characteristics • Required physical characteristics and constraints • Interface requirements • Material requirements • Verification criteria to achieve requirements • Environments and operating/usage scenarios • Comprehensive interface • Interface specs. • Interface control documents • Interface design and documents
System Capability & Manufacturing Process Demonstration	SVR	• Implemented design • Software code • Fabricated parts

Table 2: Typical Work Products per CMMI Process Area

near term planning and rewards. Some of the characteristics of Agile methods and EVM and a discussion of their differences follow.

Agile methods have the following characteristics, per Rusk's article:

- Next iteration of work is detail planned in work package
- Product burndown is a planning package for remaining features
- Features often deferred from the current iteration to the product burndown
- Features and priorities frequently revised

The EVMS Standard includes four principles regarding planning, performance measurement, and variance analysis:

- Plan all work scope for the program to completion
- Integrate program work scope, schedule, and cost objectives into a performance measurement baseline plan against which accomplishments may be measured
- Objectively assess accomplishments at the work performance level
- Analyze significant variances from the plan, forecast impacts, and prepare an estimate at completion based on performance to date and work to be performed

One EVMS Guideline requires maintaining the PMB, the time-phased scope, schedule, and associated budget through the end of the contract. Agile's focus on meeting near-term customer priorities may lead to a loss of focus on progress towards the next major technical review or software build. During development of the functional, allocated and product baselines, the team may fail to track progress towards meeting the success criteria for the SFR, PDR, and CDR. During Demonstration, the continual reprioritzation and revision of the backlog may blur vision of progress towards meeting all the requirements in the baselined blocks and builds. By placing the remaining PBIs in a planning package, the team may fail to establish sufficient, interim milestones and fail to perform variance analysis of the impact of schedule and cost variances on downstream tasks and block releases.

In the next sections, methods and examples will describe how to fit EV to Agile methods while developing working software. The same concepts and techniques may also be applied to the Design phase.

Working Software

Working software is produced during Demonstration. Also, during Demonstration, the product baseline may also be changed if new customer requirements emerge or if there are tradeoffs to balance cost, schedule, and technical objectives. The following example illustrates how to apply Agile methods with EVM when there is a plan to build a block of software with incremental builds.

Define Baselines for each Build (Technical, Schedule, Cost)

Once the Product Baseline is approved, establish the schedule in the IMS and the cost baseline in the EVM database. Finally, revise the backlog and burndown or burnup curves to be consistent with the IMS and EVM plans.

When planning incremental builds, allocate the functional requirements in the Product Baseline to each build in each block. Document each build's technical baseline. Apply budget to each block and build relative to its estimated software development effort. Then allocate budget to each requirement in the backlog. The budget may be equally distributed to the requirements or it may be allocated relative to the estimated software development effort to meet the requirement. The requirement's business value may also be the allocation basis.

The burndown curve, including all work and planning packages, is a basis of the PMB. Before locking down the PMB, ensure that resources will be available when needed including software engineers with the functional expertise needed to develop and test the scheduled requirements.

Now the first monthly iteration can be planned and executed. Although all requirements in the product baseline must be achieved, one of the early team decisions will be to prioritize the builds in each block, the requirements within the builds, and finally, the content of the next iteration.

Agile EV Example

Note: The following discussions of Agile methods and deferred functionality are extracted from the book, Performance-Based Earned Value®,[i] by Ralph Young and myself. The examples of deferred functionality are from the Proceedings of the 2009 Systems and Software Technology Conference (SSTC).

EV is often reported as a percent complete of its underlying tasks and work products (base measures). Per A. Cockburn, when selecting a measure upon which to base earned value, the best results are achieved when the measure is directly related to indicating that the desired functionality has been implemented. The product requirements are an excellent measure for use in determining earned value measures since they are directly related to evaluating progress in implementing the functionality required by the system. Each of the functional requirements is decomposed to a set of lower level, derived requirements. The set of higher and lower level requirements facilitate software design. The coded software implements the software design.[ii]

Cockburn also states that agile project teams measure progress not according to how many requirements have been gathered but by how much running functionality has been designed, programmed, and implemented (features that run). He also recommends that other units of accomplishment, besides features that run, be used to measure progress. These include use cases, individual steps in use cases, user interface widgets (frames, pull-down lists, buttons), interface calls used by applications, and user documentation.[iii]

In order to utilize requirements as the basis for taking earned value, the developer must have a requirements traceability system that provides the capability to track requirements from the level of the system requirements through software requirements, builds, Computer Software Configuration Items (CSCI), design, code and unit test, and to test procedures for all test phases.

Deferred Functionality

For valid reporting of project status, earned value should reflect the results of deferring functionality from its baselined iteration, build, or block.

When functionality is deferred from the current iteration to the backlog, even if it has customer concurrence, the deferral has the following major impacts:

1. If all the requirements planned for iteration are not completed, then the earned value for the deferred requirements cannot be earned as part of the iteration. It is behind schedule.
2. The work package which receives the deferred requirements will require additional resources to complete. The unearned budget is transferred to the new work package.
3. Although requirements are deferred to a new work package, the cumulative earned value must continue to show a behind schedule condition. To maintain the schedule variance, schedule the deferred effort in the first month of the new work package.

To illustrate how deferred functionality should be quantified at the work package level, assume that the plan is to develop multiple builds with incremental functionality. The functional requirements are allocated to each build and documented in the requirements traceability matrix. Each build has a separate work package for implementation of code. The completion criteria for each work package include:

- All baseline requirements have been coded, unit tested, and integrated into the build.
- The build has passed peer and customer review
- Documentation for the build has been completed
- The build has been recorded as complete in the configuration management process

- The build is released for higher level integration and test

SOW: Software Requirements in 2 Builds:			
Build	Allocated TRs	Budget/TR	Bac
A	100	5	500
B	60	5	300

Figure 1: Allocated Testable Requirements

In this example, assume that there are two builds. The allocated testable requirements (TR) and Budget at Completion (BAC) are shown in Figure 1. The control account for Builds A and B is shown in Table 3.

	Jan	Feb	Mar	Apr	May	Jun	Jul	Total
Build A								
Planned Reqs. met	25	25	25	25				100
Budget/Req.: 5 hours								
BCWS current (cur)	125	125	125	125				500
BCWS cumulative (cum)	125	250	375	500				500
Build B								
Planned Reqs. Met					20	20	20	60
BCWS cur					100	100	100	300

Table 3: Control Account for Builds A and B

Assume that Build A is behind schedule at the end of April with only 90 requirements being met (coded, unit tested, and integrated). At this point, earned value would be 450 hours. The schedule variance (SV) is –50 hours. The April status of Build A is shown in Table 4.

There is a decision to release Build A short of its targeted functionality and baselined requirements. There will be no additional work on Build A subsequent to its release. The requirements that have not been met are deferred into Build B. To report earned value status, close the Build A work package. Open a new work package for the next iteration, Build B, and transfer the deferred requirements (10) and budget (50) to the Build B work package.

	Jan	Feb	Mar	Apr	Total
Build A					
Planned Reqs. Met cur	25	25	25	25	100
Actual Reqs. Met cur	20	20	25	25	90
BCWS cur	125	125	125	125	500
EV cur	100	100	125	125	450
BCWS cum	125	250	375	500	
EV cum	100	200	325	450	
Schedule variance (SV):					
Reqs. Met	-5	-10	-10	-10	
SV	-25	-50	-50	-50	

Table 4: Control Account Status - April

Place the budget in the first month of the Build B work package to preserve the schedule variance. Table 5 illustrates the results of the transfer at the beginning of May.

	Apr	May	Jun	Jul	Total
Close Build A work package:					
Schedule variance (cum):					
Req. Not Met	-10				-10
BCWP remaining	-50				-50
Build B					
Before Replan					
Planned Req Met		20	20	20	60
BCWS cur		100	100	100	100
Plus transfer budget form Build A:					
Req. Not Met		+10			
BCWP remaining		+50			
After Replan:					
Planned Req. Met		30	20	20	70
BCWS cur		150	100	100	350

Table 5: Control Account Status – Beginning of May

The earned value status at May month end is shown in Table 6. Only 20 requirements were completed in May. So there is still a schedule variance (SV) of – 50 (10 requirements).

	May	Jun	Jul	Total	
Build B After Replan:					
Planned Reqs. Met	30	20	20	70	
BCWS cur	150	100	100	350	
Actual Reqs. Met cur	20			20	
EV Cur	100			100	
Schedule vairiance cum:	-10				
Reqs. Met	-50				
SV	-50				

Table 6: Control Account Status – End of May

Please note that the aggregate PMB (Build A plus Build B) was maintained despite the deferred functionality. During variance analysis, the team should discuss the impact of the behind schedule condition on revised backlog, including the estimated slip, if any, to subsequent releases (of builds and blocks).

Negative EV

Sometimes earned value has been taken for completing a PBI but that item is later returned to the backlog for rework. It is no longer acceptable by the customer. This may result from subsequent tests or other analysis. In order to show true progress against the cumulative plan and towards meeting final objectives, negative earned value should be taken.

Negative earned value is appropriate for accurate status reporting. EVMS Guideline 30 states "Control retroactive changes to …work performed…Adjustments should only be made..to *improve the accuracy* of performance measurement data." Clearly, failure to make such adjustments will overstate true technical progress, reported earned value, and the Cost Performance Index.

What About Revised Scope?

In the previous example, there was no change to the Product Baseline or scope, even if the PBIs were reprioritized. However, at the month end review meeting, such as a Sprint review, the customer may decide to revise the backlog by adding or subtracting requirements or features.

If the Product Baseline is revised, there may be cause to revise the contract SOW and program budget. In this case, there will be a contract change that will flow down to a revised PMB.

More frequently, there is only a revision to the derived requirements and features in order to meet cost and schedule objectives. Many features and PBIs are derived requirements. They are derived from higher level functional requirements that are unchanged. After customer approval, the inclusion of additional or fewer features in the backlog may change the estimated completion dates of the remaining tasks, builds, and blocks and the Estimate at Completion. However, there is normally no justification to change the PMB unless the contract SOW and approved Product Baseline are changed.

Summary

The use of Agile methods on a major defense acquisition program may result in earlier completion of the technical baselines and implementation of the Product Baseline. However, the Agile user's focus on the results of monthly iterations may cloud knowledge of performance towards meeting the program's technical, schedule, and cost objectives. Earned value discipline and continual focus on the technical baseline must be maintained.

Agile methods enable a quick response to performance deviations and changing priorities. However, the PMB should be maintained, earned value should reflect true technical performance (after accounting for deferred functionality and rework), and the program should continuously monitor the impact of variances on final cost and schedule objectives.

Note: The author realizes that this topic is relatively new and that programs and organizations may develop best practices for integrating Agile methods with EVM. Please send your practices to me for inclusion in a possible future article.

Endnotes

[i] Solomon, Paul and Young, Ralph. *Performance-Based Earned Value.*®©IEEE Computer Society/Wiley (November 2007): Chapter 13.

[ii] Cockburn, Alistair. *Crystal Clear.* Addison Wesley (November 2004):99-102

[iii] Cockburn, A. *A Governance Model for Incremental, Concurrent, or Agile Projects. CrossTalk* (February 2006):13-17

About the Author

Paul Solomon, PMP is the co-author of the book, Performance-Based Earned Value®. He is internationally recognized as a leader, teacher, and consultant on Earned Value Management (EVM). He published many articles on EVM, systems engineering, software engineering, and risk management. Most are available on his website. He retired from Northrop Grumman Corporation where he led the use of EVM on programs including the B-2 Stealth Bomber, Global Hawk, and F-35 Joint Strike Fighter. He has taught thousands of professionals and led EVMS implementation, compliance reviews, Integrated Baseline Reviews, independent assessment reviews, and process improvement teams. He is qualified to lead EVMS certification reviews.

Author Contact Information

Email: Paul Solomon [paul.solomon@pb-ev.com]