EARNED SCHEDULE FOR AGILE PROJECTS

By Robert Van De Velde, PhD, PMP

ABSTRACT

Earned Schedule for Agile projects (AgileES) combines the speed and responsiveness of Agile with the accuracy and control of Earned Schedule. AgileES adds value to common Agile tools that are used for assessing schedule performance. It also removes an Agile concern with Earned Value's schedule performance metrics. Using Earned Schedule on Agile projects is not without controversy, but there is strong, objective proof that it is valid for Agile projects. In short, AgileES deserves a place in the Agile tool kit.

Since its introduction a decade ago, Earned Schedule (Lipke, 2003) has improved the ability of projects to measure and track schedule performance. Traditional Earned Value Management (EVM) measures schedule performance in units of currency. Earned Schedule measures it in units of time—a richer and more intuitive metric.[1] While the use of Earned Schedule on plan-driven projects is growing, its adoption by Agile projects is stalled. There are several reasons for the gap.

Today, Agile projects use EVM (AgileEVM) to manage cost performance. Their use of EVM for schedule management is limited—there is doubt that it adds value beyond what is provided by common Agile tools. Consider this observation concerning the use of EVM on two Agile test projects:

> *The Product Owners for both projects felt that the AgileEVM metrics did not provide any more schedule insight than the burndown chart provided. … We agree that without the need to manage cost performance, AgileEVM does not add significant value above traditional burndown methods. (Sulaiman, T., Barton, B., & Blackburn, T., 2006)*

There is also a concern with EVM's traditional schedule performance metrics. As a project approaches completion, EVM's metrics invariably show schedule performance improving. At the end of a project, the performance appears to be perfect, even if the project is delivered late. That is counter-intuitive, if not simply misleading.

Finally, there are concerns within the Agile community about using EVM in any form. Based on negative experience, some claim that EVM is merely bureaucratic overhead. Others, who have had positive experience with EVM, dispute the claim. The debate persists.

AgileES applies Earned Schedule to Agile projects and addresses all of these concerns. AgileES practice demonstrates value-add for traditional burndown methods and improved accuracy versus traditional EVM schedule performance metrics. AgileES *theory* objectively proves the validity of AgileES for Agile projects.

AgileES Practice

AgileES quantifies schedule performance efficiency. Earned Schedule measures the amount of time earned on a project. The Schedule Performance Index for time (SPI_t) compares the amount of time earned to the actual time, thus indicating how well or poorly time is being used on the project. The SPI_t yields a profile of schedule performance that goes beyond burn charts and gives Agile teams additional information to use in identifying and correcting deviations.

A recent project that used AgileES illustrates the point. The project, call it *AgileEStest*, was a classic Agile project: it was a software development initiative done by a team with several years of Agile experience. The team prepared a product backlog, a relative sizing, and a release plan.

The planned staffing level on the project was ten team members, all of whom were co-located. The sprints were two weeks long, and the original end date was four months after the start. The budget was a half-million dollars.

Each week, the team gathered EVM data for a spreadsheet that was used to do Agile and AgileES calculations. No additional data collection was required for AgileES. Metrics generated from the spreadsheet were used in regularly scheduled team meetings. Grooming of the release plan was done at the end of each sprint.

Consider a snapshot taken part way through the project (Figure 1). The snapshot includes Burndown and AgileES metrics that will be explained presently.

The project had a bad start. Actual staffing levels were significantly lower than planned. As a result, schedule performance efficiency was low. At the end of Sprint 3, the staffing level was brought to the full complement of ten, but due to the delay already incurred and the addition of a large number of new release points, a major plan revision was required. The number of sprints and budget were increased, the target date was delayed, and the contents of the sprints were re-organized. With the new baseline, the project was essentially restarted.

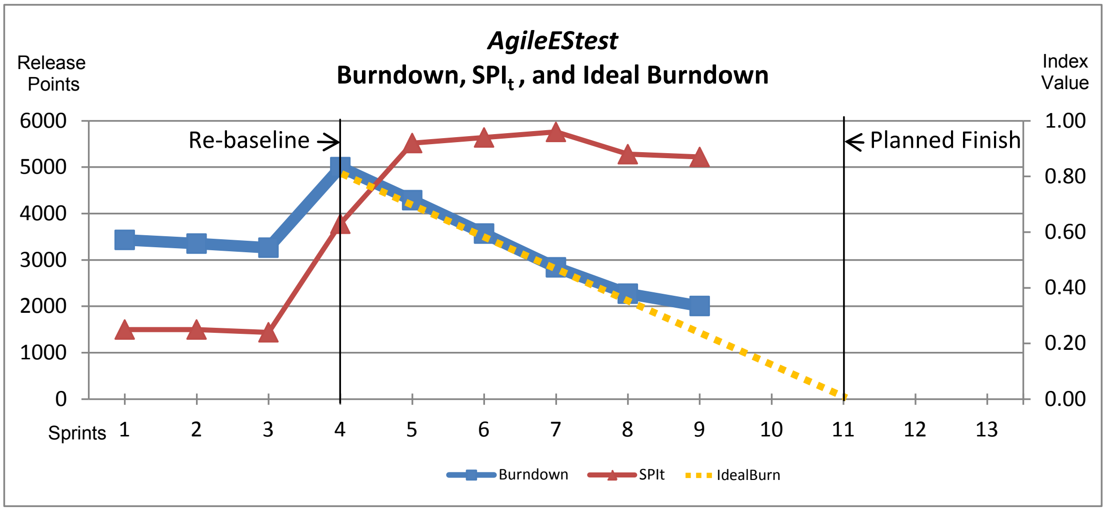

Figure 1: *Project AgileEStest* Burndown, SPI$_t$, and Ideal Burndown

The chart focuses on the restarted project. It depicts the burndown (the blue squares) beginning with the number of release points actually completed at the end of Sprint 4 and running as far as Sprint 9. For those unfamiliar with burndown charts, they are commonly used on Agile projects for assessing schedule performance. The charts show how much work has been completed and, by inference, how much remains on the project.

The chart in Figure 1 also contains an Ideal Burndown line (the yellow dotted line). It runs from the first sprint of the re-baseline to the last planned sprint. It starts with the number of release points that should have been completed at the end of Sprint 4 and finishes at Sprint 11 with no remaining release points. If the actual Burndown runs below the line, the project is ahead of schedule, and if it runs above the line, the project is behind schedule.

After the re-baseline, the Burndown line shows a steady decline in the number of remaining release points. Through Sprint 8, the results appear to run on or slightly above the Ideal Burndown line, indicating that the project is on or slightly behind schedule. At Sprint 9, the results jump above the Ideal Burndown line, indicating that the project is definitely behind schedule.

AgileES clarifies what is happening. As shown by the SPI_t line (the red triangles), schedule performance efficiency improves over the first four sprints of the new baseline, but after the fourth, Sprint 7, performance steadily declines. While the Burndown line highlights the deviation at Sprint 9, the shortfall began at Sprint 7. Root cause analysis of the delay should start there, rather than with the later sprint.

Burndown charts show the outcome of schedule performance efficiency: greater efficiency means more release points are consumed, less efficiency means fewer release points are consumed. The outcome is that the project is either on, ahead of, or behind schedule. By contrast, AgileES assesses schedule performance efficiency itself and not just the outcome of that performance efficiency. It shows explicitly how time is being used and thereby adds insight to what is provided by common Agile tools such as burn charts.

The discussion thus far has been framed exclusively in terms of *SPI*$_t$, intentionally omitting EVM's Schedule Performance Index (SPI). As alluded to earlier, *SPI* is a problematic metric. The *SPI* is the ratio between the Earned Value and the Planned Value. At the end of a project, the Earned Value equals the Planned Value, by definition. As a project approaches its finish, the *SPI* begins to rise, regardless of the actual performance. Even late projects end with a perfect *SPI*.

Unlike the *SPI*, Earned Schedule's *SPI*$_t$ accurately reflects time performance throughout a project's life cycle. Again, the test project illustrates the point.

By the time it completed, *AgileES* test had exceeded its planned finish by two sprints. Nonetheless, the *SPI* rose steadily from Sprint 10 onwards and ended at 1.00. In contrast, the *SPI*$_t$ trended downwards from Sprint 11, ultimately finishing at .80. Given the project's finish date, the *SPI*$_t$ presents a more accurate picture of time performance than does the *SPI*.

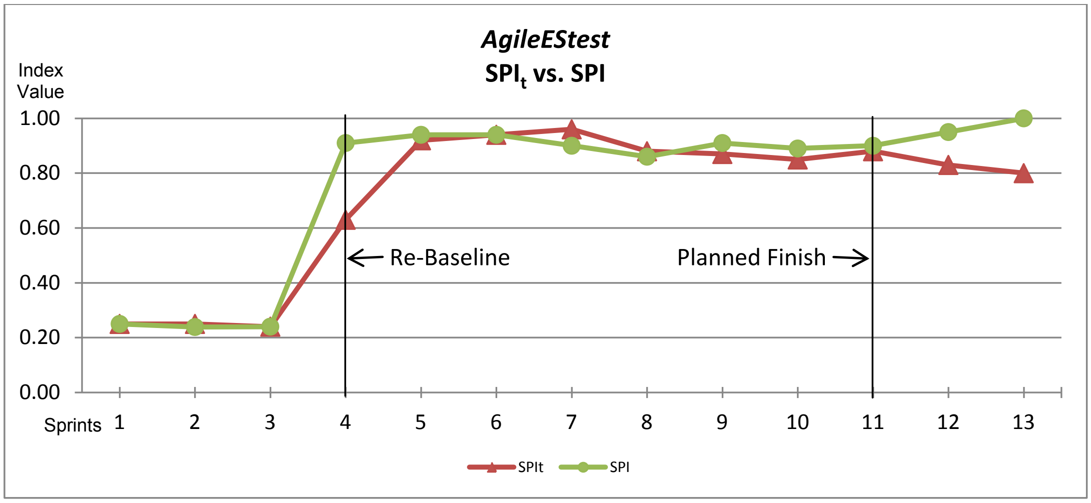

Figure 2: *Project AgileEStest* SPI$_t$ vs. SPI

In summary, there are practical reasons for applying Earned Schedule to Agile projects. As shown by the test project, AgileES offers new insight into burn charts, and it is more accurate than EVM's *SPI*. The utility of AgileES is a good reason to use it, but practicality does not prove validity.

AgileES Validity

Mathematical deduction proves validity. But, why is it necessary to prove validity? Why is the utility of AgileES practice not enough to justify its use on Agile projects?

Practice is a matter of experience, and within the Agile community, there is experience that runs counter to AgileES practice. Consider the following statement:

> *Artificial measures such as EVM typically prove to be overhead at best, whose only value is to cater to the dysfunctional bureaucrats infesting many organizations. (S. Ambler, 2011)*

On the other hand, there is experience consistent with AgileES practice. Consider the following response to the last statement:

> *Ignore the arguments against EV from those [who have] not … deployed it successfully. … You wouldn't take agile advice from someone who has not successfully deployed agile in a domain and context similar to yours. Don't do the same for anything else. (G. B. Alleman, 2011)*

Experience, although valuable, is not conclusive proof. Rather than countering one subjective claim with another, we need an objective approach. To prove the validity of AgileES for Agile projects, therefore, Earned Schedule metrics were deduced from Agile metrics. Test projects then provided empirical verification of the math.

The proof incorporates both the approach and specific theorems from previous work on AgileEVM (Sulaiman, Barton, Blackburn, 2006). As the full proof is lengthy, only an outline is included here, describing key steps. A link to the full proof is included in the References (Van De Velde, 2013).

The proof starts with AgileEVM's release date equation (Sulaiman, T., et al., 2006). The Release Date at mean velocity *v* (RD_v) equals the Start Date (*SD*) plus an offset. The mean velocity is the average work per sprint. The offset is the sprint Length (*L*) times a performance factor. The performance factor is the current sprint number (*n*) multiplied by 1 over the Actual Percent Complete at Sprint n (APC_n).

$$RD_v = SD + L \times \left(n \times \frac{1}{APC_n}\right) \tag{1}$$

In AgileEVM (Sulaiman, T., et al., 2006), the Actual Percent Complete is defined as the ratio between the total number of Release Points Completed at Sprint *n* (RPC_n) and the total number of Planned Release Points at Sprint *n* (PRP_n). That is,

$$APC_n = \frac{RPC_n}{PRP_n} \tag{2}$$

That ratio is equivalent to the ratio between the Planned Duration at a particular velocity *v* (PD_v) and the Earned Duration at that velocity (ED_v). The following steps establish the equivalence.

The Planned Duration at velocity *v* (PD_v) is the Length of time (*L*) that the overall project is expected to take in order to complete all planned release points at the planned velocity (PV_n). The planned velocity is the number of release points to be completed in each sprint. The PD_v sets a baseline expectation on the total duration of a project. In formal terms,

$$PD_v = L \times \left(\frac{PRP_n}{PV_n}\right) \tag{3}$$

The Earned Duration at velocity *v* (ED_v) is the time that is actually spent completing the release points that are done up to that point in time, given the planned velocity. In formal terms,

$$ED_v = L \times \left(\frac{RPC_n}{PV_n}\right) \tag{4}$$

Given the definition of APC_n, PD_v, and ED_v, common terms cancel out with the following result:

$$\frac{1}{APC_n} = \frac{PRP_n}{RPC_n} = \frac{PD_v}{ED_v} \tag{5}$$

Substituting for the performance term in equation (1), we have:

$$RD_v = SD + L \times \left(n \times \frac{PD_v}{ED_v}\right) \tag{6}$$

Next, Earned Schedule terms replace the duration velocity terms (PD_v and ED_v). The equivalence between the Planned Duration for velocity (PD_v) and the Planned Duration for Earned Schedule (PD_{ES}) is easy to demonstrate. For both PD_v and PD_{ES}, the planned duration is the difference between the Finish Date and the Start Date. So, $PD_v = PD_{ES}$.

The Earned Duration for velocity (ED_v) was just defined. There needs to be an equivalent earned duration term for Earned Schedule. The Earned Duration for *ES* (ED_{ES}) equals the length of time during which the total number of release points actually done (RPC_{AT}) is greater than or equal to the cumulative number planned at the end of each sprint from the first to the last ($CPRP_i$). In formal terms,

$$ED_{ES} = L \times \left(\sum_{i=1}^{j} (RPC_{AT} \geq CPRP_i)\right) \tag{7}$$

For simplicity, we assume here that the total number of release points completed is exactly equal to the total number planned as of the end of a sprint. That is, there is no fractional amount. (Accounting for fractional amounts adds considerably to the complexity of the full proof.)

Now, we prove that ED_v and ED_{ES} are equivalent. The equivalence rests on the fact that ED_v and ED_{ES} are simply different ways to count the number of units of planned velocity in the release points completed.

An example illustrates the point (Table 1). At the end of Sprint 5, 50 release points have been completed. At a planned velocity of 10 release points per sprint, 5 sprints have been earned. Similarly, if we compare the 50 points completed with the running total of planned points at the end of each sprint, we see, again, that 5 sprints have been earned. Generalizing, we say that $ED_v = ED_{ES}$.

Sprint	1	2	3	4	5	6	ED
RPC_n	0	15	35	40	50	-	5
PV_n	10	10	10	10	10	10	
$CPRP_n$	10	20	30	40	50	60	5
RPC_{AT}	50	50	50	50	50	50	

Table 1: $ED_v = ED_{ES}$

Substituting the Earned Schedule terms for the velocity terms in (6), we have:

$$RD_v = SD + L \times \left(n \times \frac{PD_{ES}}{ED_{ES}}\right) \tag{8}$$

Then, we move from the Agile context to the Earned Schedule domain. To do so, we show the equivalence between ED_{ES} and the calculated amount of schedule earned (*ES*). (For simplicity, the outline varies here from the full proof.)

The key to the equivalence is the fact that the method for sizing release points can be any numerical value (Sulaiman, T., et al., 2006). So, we can substitute weighted release points for release points throughout the equations. Starting with (7), we first substitute ED_v for ED_{ES}. Second, we multiply all release points in the equation by Rate (*R*), giving us:

$$L \times \left(\frac{R \times RPC_n}{R \times PV_n}\right) = L \times \left(\sum\nolimits_{i=1}^{j} (R \times RPC_{AT} \geq R \times CPRP_i)\right) \tag{9}$$

We cancel common terms and use the equivalence between ED_v and ED_{ES} to simplify the first term. Given that the Rate times the Release Points Completed yields the Earned Value and the Rate times the Cumulative Planned Release Points yields the Planned Value, we substitute in the second term and get the following result:

$$ED_{ES} = L \times \left(\sum\nolimits_{i=1}^{j} (EV_{AT} \geq PV_i)\right) \tag{10}$$

As the second term is the calculated amount of schedule earned (not including the fractional amount), ED_{ES} equals *ES*.

Having shown the equivalence between ED_{ES} and *ES*, we substitute into (8), yielding:

$$RD_v = SD + L \times \left(n \times \frac{PD_{ES}}{ES}\right) \tag{11}$$

Finally, factoring in standard Earned Schedule definitions for SPI_t and for the Estimate at Completion for time (EAC_t), the AgileES release date equation follows.[2] That is, the Release Date for AgileES (RD_{ES}) equals the Start Date (*SD*) plus an offset. The offset consists of sprint Length (*L*) times a performance factor. The performance factor is the current sprint (*n*) times the Estimate at Completion for time (EACt) divided by the Actual Time (*AT*).

$$RD_{ES} = SD + L \times \left(n \times \frac{EAC_t}{AT}\right) \tag{12}$$

Thus, the AgileES release date equation is deduced from Agile's velocity release date equation, proving the validity of Earned Schedule for Agile projects.

The math was tested on two projects. Both the velocity and the AgileES release dates were calculated throughout the projects. Given the math, the results were not surprising. They showed that the Agile velocity and Earned Schedule release date estimates were indistinguishable (see Figure 3 for the results from *AgileEStest*). The correlation between the mean velocity release date equation and the AgileES release date equation was thereby empirically confirmed, re-enforcing the validity of Earned Schedule for Agile projects.

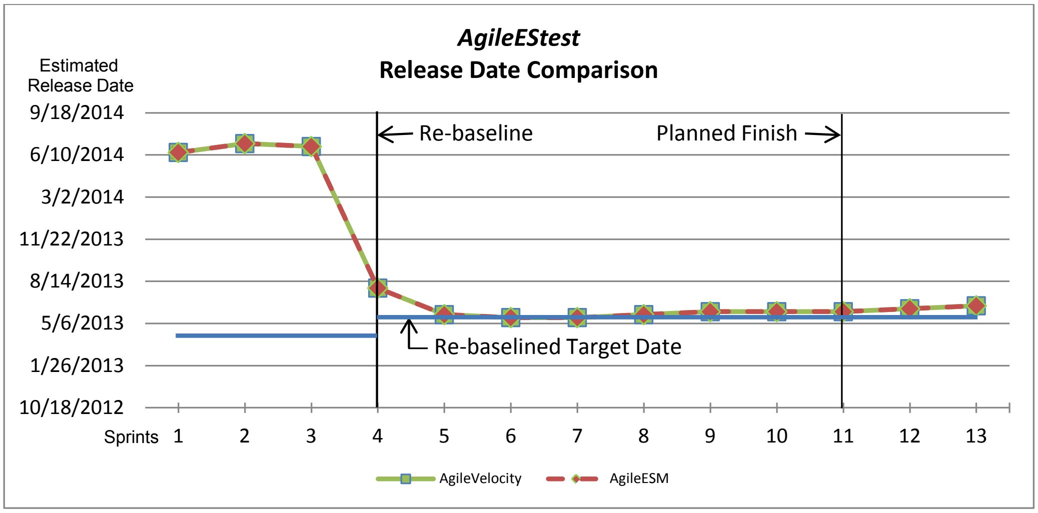

Figure 3: *Project AgileEStest* Release Date Comparison

CONCLUSION

AgileES Added Value: AgileES release date estimates do not by themselves add value to Agile velocity release date estimates. As stated by the math and verified by experimental data, AgileES release date estimates and velocity release date estimates are indistinguishable. Similarly, AgileES and burn charts both indicate whether or not a project is on schedule. The value-add comes from the fact that AgileES assesses schedule performance efficiency, indicating how well or poorly time is being used on the project. That provides insight into schedule performance that goes beyond burn charts and traditional EVM measures.

Validity of AgileES for Agile Projects: The added value of AgileES is a good practical reason for its application to Agile projects, but there is also a theoretical reason for its validity. The outlined proof suggests how the AgileES release date estimate follows from Agile metrics. The detailed proof definitively establishes the correlation between the AgileES release date equation and the velocity release date equation. Test projects provide empirical verification of the math.

Lightweight Process: Although it was mentioned only in passing above, AgileES uses the same data that is ordinarily collected on Agile projects—no additional data collection is required, minimizing the overhead for Agile teams. There is a small increase in time for analysis of the additional data provided by AgileES. Finally, either a purchased tool is required, or there is a one-time effort to incorporate Earned Schedule calculations and results into existing tools.

AgileEVM+AgileES: AgileES is a natural complement to AgileEVM. AgileEVM's strength in cost management is supplemented by AgileES' strength in schedule management. The combination gives Agile teams better tools for managing their projects.

FOOTNOTES

1. The concept behind Earned Schedule is that the amount of time earned on a project is the time at which the value currently earned should have been earned (Lipke, 2003, 2009).

   The calculation of Earned Schedule is in two parts:

   a) the number of complete periods in which the current Earned Value (*EV*) equals or exceeds the cumulative Planned Value (*PV*) for the period;

   b) the fractional amount for the first period in which the Earned Value (*EV*) does not equal or exceed the cumulative Planned Value for the period (*PV*).

The expression of Earned Schedule duration in periods adds flexibility to the calculations: the periods can represent any unit of time. Earned Schedule periods can be rendered in specific time units by multiplying the number of periods by the Length (*L*) of a period. Thus, we can state the duration of *ES* in specific time units (ES) as follows:

$$ES = L \times \left(\sum_{i=1}^{j} (EV_{AT} \geq PV_i) + \frac{EV_{AT} - PV_i}{PV_{i+1} - PV_i} \right) \qquad \text{(i)}$$

2. From the standard Earned Schedule equation, other metrics have been derived (Henderson, 2004), including the Schedule Performance Index for time (SPI_t) and the Estimate at Completion for time (EAC_t).

$$SPI_t = \frac{ES}{AT} \tag{ii}$$

Dividing the Planned Duration based on ES (PD_{ES}) by SPI_t yields the EAC_t. That is,

$$EAC_t = \frac{PD_{ES}}{SPI_t} \tag{iii}$$

REFERENCES

Alleman, G.B. (2011, May 4). More EV and Agile Confusion [Blog post]. Retrieved from http://herdingcats.typepad.com/my_weblog/2011/05/alert-earned-value-is-not-the-same-as-business-value.html

Ambler. S. (2011, May 3). Agile and EVM Strategies. [Blog post]. Retrieved from http://www.drdobbs.com/architecture-and-design/agile-and-evm-strategies/229402730

Henderson, K. (2004, Spring). Further Developments in Earned Schedule. The Measurable News.

Lipke, W. (2003, Summer). Schedule is Different. The Measurable News.

Lipke, W. (2009). Earned Schedule. Lulu Publishing.

Sulaiman, T., Barton, B., & Blackburn, T. (2006). AgileEVM—Earned Value Management in Scrum Projects. Agile '06: Proceedings of the Conference on AGILE 2006 (pp. 7-16). IEEE Computer Society.

Van De Velde, R. (2013). AgileESM: Earned Schedule for Agile Projects. http://www.projectflightdeck.com/media/AgileES_Full_Length_Proof.pdf

ABOUT THE AUTHOR

Robert Van De Velde owns and operates ProjectFlightDeck.com, a company focused on Earned Schedule products and services. He has recently extended Earned Schedule to Agile projects, packaging the practice, tools, and theory as AgileESM© (Agile Earned Schedule Management). For more information, click on the following link http://www.projectflightdeck.com/AgileESM.php.

As a project manager, Rob has a long track record of delivering IT programs and projects in a variety of domains, including financial services, natural resources, telecommunications, and health care. His project management accomplishments have been recognized by several awards, including two CIPAs (Canadian Information Productivity Awards). He has successfully used both plan-driven and Agile processes on his projects.

Rob holds a PhD, a PMP, and a Black Belt in MS Project. He has spoken at Project World, EVM World, and Ryerson University. His articles have appeared in Projects@Work, PMWorldToday, and Journal of Systems Management.

Contact Information
Robert Van De Velde
Owner/Operator
ProjectFlightDeck.com
Robert.vandevelde@projectflightdeck.com
3416 Sawmill Valley Drive
Mississauga, Ontario
Canada L5L 3A4
(905) 828-0508