AgileEVM – Earned Value Management in Scrum Projects

Tamara Sulaiman, PMP, CSM
SolutionsIQ
tsulaiman@solutionsiq.com

Brent Barton, CSM, CST
SolutionsIQ
bbarton@solutionsiq.com

Thomas Blackburn, PMP, CSM
InfoTech, Inc.
Thomas.blackburn@infotechfl.com

Abstract

Scrum is an Agile Project Management Framework. This framework specifically focuses on maximizing Return on Investment (ROI). Scrum, however, does not define how to manage and track costs to evaluate actual ROI against the vision. A reasonable cost measurement that integrates with Scrum would help provide an additional feedback loop.

We adapted Earned Value Management (EVM), using values defined in Scrum. The result is called AgileEVM (Agile Earned Value Management) and is a simplified set of earned value calculations. From the values in Scrum, we derived a release date estimate using mean velocity and from this equation, generated an equivalent equation using traditional EVM techniques, thus establishing the validity of using EVM with the Scrum framework. Finally, we used this technique on two projects to further test our hypothesis. This investigation also helped us determine the utility of AgileEVM.

1. Introduction

Scrum is an Agile Project Management Framework. Many Agile project methods, especially Scrum, emphasize incremental, multi-level planning and discourage fully planning software projects down to the finest level of decomposition in the early stages. The basis for discouraging big up-front plans is the amount of inherent ambiguity present in complex software systems that makes fully defining the requirements up front nearly impossible. Further, Scrum focuses on frequently prioritizing requirements to maximize Return on Investment (ROI) [1]. In order to evaluate actual ROI, project costs must evaluated and used to determine whether release plans have sufficient business value or additional re-planning is necessary. One challenge of evaluating ROI is the Scrum framework does not integrate costs and leaves this up to the implementers of Scrum.

"Earned Value Management (EVM) [is] a method for integrating scope, schedule and resources, and measuring project performance." [2] This technique is used in many industries. If we can validate a way to use this accepted practice on Scrum, then we have the tools we need to help provide balanced what-if scenarios for release planning for scope, schedule and budget. EVM techniques, however, assume complete planning of a project to discrete work package levels, then assigning cost and duration to these packages so EVM's utility has been questioned for Agile projects.

To address this, we define Agile Earned Value Management (AgileEVM), a simplified set of earned value calculations adapted from traditional EVM using Scrum metrics. By demonstrating the mathematical relationship between EVM calculations and Scrum metrics we intend to prove that Earned Value Management (EVM) is valid for use on Scrum projects. It should be noted that while we believe that these statements hold true for other Agile methods, we focus only on Scrum projects because that is the framework used in our research.

To demonstrate the validity and utility of AgileEVM, we present three main areas: research, AgileEVM development and implementation on two projects. First, we review the available literature on earned value methods to establish the validity of EVM techniques on software development projects and to document previous studies applying Earned Value techniques on Agile projects. Second, we define terms and a process for using AgileEVM on Scrum projects and correlated EVM with standard Scrum burndown methods. Finally, we introduce AgileEVM in two very different Scrum projects so we could validate our assumptions.

2. Research

The literature review revealed an abundance of publications validating the use of EVM for traditional

project management, particularly on defense and construction projects. In use since the 1960's, EVM is a well recognized project management technique; included in the Project Management Institute's PMBOK [2]. We concentrated on publications relating earned value to software development projects in general, and relating earned value specifically to Agile software projects.

In the Spring, 1995 issue of Acquisition Quarterly, Major David S. Christensen and Daniel V. Ferens published a paper, "Using Earned Value for Performance Measurement on Software Development Projects"[3]. The processes described for obtaining and using EVM data are clearly based on a heavyweight, high-ceremony waterfall software development life cycle.

An early piece by Quentin W. Fleming and Joel M. Koppelman, "Earned Value Project Management – A Powerful Tool for Software Projects" [4], validated the appropriateness of using EVM on software development projects. In the aforementioned article and in *Earned Value Project Management* (2nd edition), they reiterated their view on the appropriateness of EVM for software engineering projects [5]. The EVM methodology described is based on exhaustive initial planning, formal work breakdown structures, detailed work packages associated with dates and dollars, and detailed tracking and analysis.

The earliest mention of EVM together with agile practices was a paper presented by Steven H. Lett of Lockheed-Martin, at the European SEPG in 1998. Mr. Lett's paper, "An Earned Value Tracking System for Self-Directed Software Teams" [6] describes an earned value tracking system (EVTS) for self-directed work teams (SDWT) that involved traditional project management planning tasks. Detailed reports from individual team members were required on a regular basis and involved a metrics engineer to collate and analyze the data obtained.

In the February, 2003 issue of PMForum [7], Glen Alleman described a method for an Agile project manager to use earned value techniques on iterative projects. The method advocated the use of 'testable requirements' and 100% unit test coverage.

At the June 2003 Agile Conference in Salt Lake City, Glenn Alleman, Micheal Henderson, and Ray Seggleke presented a paper entitled "Making Agile Development Work in a Government Contracting Environment - Using Earned Value to Measure Velocity" [8]. This large government-contracted development project mandated earned value management (EVM). In this paper they describe introducing agility by using certain Agile methods, including XP-like engineering practices.

In his book *Crystal Clear – A Human Powered Methodology for Small Teams,* Alistair Cockburn describes the similarity between a "burn up" chart and an EVM chart [9]. He warns that EVM credits tasks completed, not necessarily integrated code. He concludes that the Agile burn up chart provides more accurate information as to the state of the project than the earned value chart.

We reviewed the threads from two very active discussion groups: http://groups.yahoo.com/group/scrumdevelopment and http://groups.yahoo.com/groups/agilemanagement. Opinions generally centered around two themes. There is much debate about the meaning of the term "value" and confusion between EVM, business value and economic value. The second theme considers that EVM techniques can be used on Scrum projects, but asserts that there is no tangible utility to using them.

We found the results of the review encouraging, validating the use of EVM on traditional software development projects. We found articles describing the use of Agile practices on software projects using EVM. What we did not find was conclusive evidence demonstrating the applicability and value of using EVM on Agile projects.

3. Development of AgileEVM

Our implementation of AgileEVM concentrates on measuring progress at the release level, rather than at the sprint level or at the product level. We feel this to be the most appropriate way to use earned value management formulas on agile projects. It is true that these formulas can easily be used to measure progress of a project with multiple releases, but this would require that the backlog of multiple releases be identified and estimated. An example of where this may be a good approach would be when each Sprint has a production release and we want to use AgileEVM for helping us with the entire product life cycle.

In comparison with the requirements of traditional EVM, AgileEVM leverages work items that are integral to the Scrum process. We measure progress at the end of each sprint when actual sprint velocity and actual costs are known. In five key terms below (see Table 1) Traditional EVM applications are compared to AgileEVM applications:

Table 1: Comparison of EVM terms

Performance Measurement Baseline (PMB)	
Traditional EVM	The sum of all work package schedule estimates (duration and effort).
AgileEVM	Total number of story points planned for a release (*PRP*)
Schedule Baseline - often integrated in PMB	
Traditional EVM	The sum of all work packages for each time period calculated for the total duration.
AgileEVM	The total number of planned sprints (*PS*) multiplied by sprint length.
Budget at Complete (*BAC*)	
Traditional EVM	The planned budget for the release or project.
AgileEVM	The planned budget for the release.
Planned Percent Complete (*PPC*)	
Traditional EVM	What % complete did we expect to be at this point in the project? Can be a subjective estimate, or a calculation of the dollar value of the cumulative tasks planned to be complete by this point in time divided by the performance baseline
AgileEVM	The number of the current sprint (*n*) divided by the total number of planned sprints (*PS*).
Actual Percent Complete (*APC*)	
Traditional EVM	The dollar value of work packages actually completed divided by total dollar value of the budget at complete.
AgileEVM	The total number of story points completed (potentially shippable increments) divided by the total number of story points planned.

Initial values needed for AgileEVM are straightforward. The values in Table 2 provide the ability to create an initial release baseline to measure progress against.

Table 2: Initial release parameters

Name	Definition
BAC	The amount budgeted to be spent on the release
L	The length of time for each sprint
PS	Total number of sprints planned for the release
SD	The start date for the release.
PRP	Total points planned for the release

At the completion of each Sprint, we capture four data points. This is sufficient to calculate Scrum metrics and AgileEVM.

Table 3: Sprint data points

Name	Definition
n	Sprint number - starts at 1
PC	Points Completed - The points of work completed from the Release Backlog during the Sprint
PA	Points Added - The points added (or subtracted) to the Release Backlog during the Sprint
SC	Sprint Cost - What was spent

Change control in Scrum is managed and reviewed each Sprint cycle. Upon acceptance of the work completed in the Sprint and the adjustments to the release backlog, this can be considered a new project baseline. No extra weight or ceremony is imposed on the Scrum process except we explicitly review the release plan in the context of the budget.

To calculate earned or planned value we must have an accurate representation of actual and planned percent complete. Story points, as described in Mike Cohn's *User Stories Applied* [10], represent an estimate of effort and are therefore a valid measure of percent effort complete. This is especially true if the release includes testable requirements (Alleman [9]) in the form of Stories that must pass and be accepted by the Product Owner. We define actual percent complete, *APC*, to be the ratio of Story points completed to Story points planned. We further define planned percent complete, *PPC*, to be the ratio of Sprints completed to Sprints Planned. These are summarized in Table 4.

Table 4: AgileEVM definitions

Name	Definition
PRP	Planned Release Points. See (18) for further discussion
RPC	Release Points Completed. See (19) for further discussion
APC	Actual Percent Complete of Release. This is the ratio of Points completed to Points planned
PPC	Planned percent complete.

We use the following EVM definitions and equations from the PMBOK: [2]

Table 5: Standard EVM definitions and equations

Equation	Definition	
BAC	Budgeted Cost at Completion: This is the initial budget for the release	
AC	Actual Cost: This corresponds to budgeted costs in PV for release	
*PV = PPC * BAC*	Planned Value:	(1)
*EV = APC * BAC*	Earned Value	(2)
CV = EV - AC	Cost Variance	(3)
SV = EV - PV	Schedule Variance	(4)
CPI = EV / AC	Cost Performance Index	(5)
SPI = EV/PV	Schedule Performance Index	(6)
*ETC = 1/CPI * (BAC – EV)*	Estimate To Complete	(7)
EAC = AC + ETC	Estimate At Complete	(8)

4. Demonstrating a Mathematical Correlation of Scrum Metrics and EVM Metrics

There is a correlation of the schedule forecasts using Scrum compared to Estimate at Completion (*EAC*) forecasts using EVM. The logical argument starts with a presentation of the hypothesis and then proceeds through a derivation of a Release Date (*RD*) function. Following this, we derive the Release Date using mean velocity of Sprints. Finally, we calculate a release date using EVM from the mean velocity. Inspecting this result, we validate a reasonable measure for Actual Percent Complete using *EAC*, thus establishing the correlation.

4.1 Hypothesis

Release date estimates using *EAC* calculations provided by EVM correlate to Mean velocity predictions provided by Scrum.

More specifically, let RD_m stand for the calculated Release Date using method *m*. If the projected release date based on the mean velocity (RD_v) equals the projected release date based on EVM (RD_{EV}), then EVM techniques can be used on Scrum projects to calculate *EAC* with equal precision to traditional plan-driven projects described in the PMBOK (ANSI/PMI 99-001-2004 standard) [2]. Restated, if

$$RD_v = RD_{EV} \tag{9}$$

then EVM techniques related to *EAC* can be used on Scrum projects.

4.2 General Equation of a Release Date

Deriving a release date RD_m using any method *m*, needs an initial position, let it be named start date (*SD*). Then a release date is simply an offset from the start date. This offset can be determined by the number of Sprints and the Sprint length. From this we can define the offset to be the number of sprints multiplied by the duration of the Sprints. Thus,

$$RD_m = SD + N \cdot L \tag{10}$$

where *N* equals the number of Sprints and *L* equals the length of each Sprint in calendar days. Note that (10) assumes equal Sprint lengths. It is mathematically simpler to limit this discussion to equal Sprint lengths. Although unusual, Sprint lengths are occasionally adjusted to accommodate environmental situations. For these anomalies, one can, in most cases, redefine the Sprint Length to be equal to the Greatest Common Factor of the different Sprint lengths without invalidating this method.

Upon inspection of (10), *N* is the only variable, where *SD* and *L* are initial parameters that do not change for a release. Thus, *N* is the single independent variable. See Fig. 1: Graph of a release date.

$$RD_m = f(N) \tag{11}$$

So RD_m is a linear function dependent on *N*.

$$RD_m = SD + L \cdot N \tag{12}$$

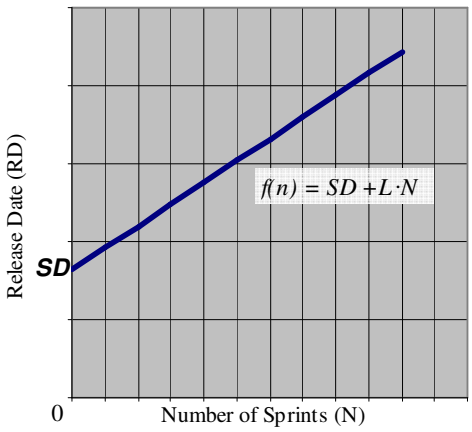

Fig. 1: Graph of release date

4.3 Using Mean Velocity to Define the Equation of a Release Date

Considering (12), we need to define N in terms of an estimate of how much work can be done per Sprint. Let v equal the average work completed per Sprint, which is known as the mean velocity, or often just velocity. The total number of Sprints N decreases as velocity increases, so it is inversely proportional to N. Let W = the total work planned to be done in the release. Then the total work factored by the mean velocity equals the total iterations N for $v>0$.

$$N = \frac{W}{v} \tag{13}$$

For Sprint n, where $n \geq 1$, let

$$N = n + n' \tag{14}$$

where n' is the estimated Sprints after n. Also, let W' be the work to be done in sprints n'. Then (13) can be rewritten

$$n' = \frac{W'}{v} \tag{15}$$

Combining (14) and (15), then substituting into (12), we have

$$RD_v = SD + L \cdot \left(n + \frac{W'}{v} \right) \tag{16}$$

4.4 Using Mean Velocity to Determine Sprints Left

Clearly, if there have been no completed Sprints, an estimate of an initial velocity must be made. Once underway, measures of actual velocity can be used to estimate future velocities. Empirical measures, such as "yesterday's weather," are effective for evaluating what can be accomplished in Sprint Planning. For release planning, however, applying the arithmetic mean to determine the average velocity (work done per Sprint) is a common practice.

To provide measures for work, we need to introduce a valuation for sizing Release Backlog Items (RBI). This allows us to measure how much work is completed for velocity calculations. The method for sizing these can be any numerical value. A few examples are: Story Points, T-Shirt Sizes with Ideal Team Days applied, Function Points and Working Days[1].

For convenience, let us define the units for these numerical values to be points, abbreviated p. Thus, for the items in the Release Backlog, there exists a same-sized set P of point values, $P = \{p_1, p_2, p_3, \ldots; p_i \geq 0\}$ where the total point value, named Planned Release Points (PRP) can be calculated as

$$PRP = \sum_{P} p_i \tag{17}$$

Before the inception of any Sprint, when $n = 0$, denote the initial Planned Release Points: PRP_0. For each Sprint n, define any new RBI points added PA_n and the points completed PC_n. Thus, the total release points from inception through Sprint n can be calculated as

$$PRP_n = PRP_0 + \sum_{k=1}^{n} PA_k \tag{18}$$

Further, define RPC_n as the total release points completed through Sprint n. This can be calculated as

$$RPC_n = \sum_{k=1}^{n} PC_k \tag{19}$$

Finally, let the number of points remaining at Sprint n be denoted PR_n. This can be found by subtracting the Completed Release Points from the Planned Release Points. Thus,

$$PR_n = PRP_n - RPC_n \tag{20}$$

We can now redefine $W' \equiv PR_n$ from (15) , so

$$n' = \frac{PR_n}{v} \tag{21}$$

To calculate the number of Sprints left we need to calculate N. Substituting (21) in (14), we have

$$N = n + \frac{PR_n}{v} \tag{22}$$

1 One needs to be cautious when using values with units such as time or cost. These can lead to confusion and potential misinterpretation or miscalculation.

Next, we need a more explicit way to express v in terms of data we capture. Let n define the n^{th} Sprint where $n>0$ and let v_n be the velocity of Sprint n. Then the velocity for Sprint n is the number of points completed in Sprint n. *Thus,*

$$v_n = PC_n \tag{23}$$

The arithmetic mean of the velocity v can then be calculated using the standard equation:

$$v = \frac{1}{n} \cdot \sum_{k=1}^{n} v_k \tag{24}$$

where $n>0$.

Substituting (23) into (24) yields:

$$v = \frac{1}{n} \cdot \sum_{k=1}^{n} PC_k \tag{25}$$

Substituting (20) and (25) into (22) generates the following equation.

$$N = n + \frac{PRP_n - RPC_n}{\frac{1}{n} \cdot \sum_{k=1}^{n} PC_k} \tag{26}$$

We can simplify this equation by factoring n from each operand and substituting RPC_n from (19).

$$N = n \cdot \left(1 + \frac{PRP_n}{RPC_n} - \frac{RPC_n}{RPC_n}\right) \tag{27}$$

This can be immediately simplified to:

$$N = n \cdot \left(\frac{PRP_n}{RPC_n}\right) \tag{28}$$

From the description of Actual Percent Complete (*APC*) in Table 4, we can construct APC_n as the ratio of release points completed (RPC_n) and total points (PRP_n) for sprint n. Thus, let

$$APC_n = \frac{RPC_n}{PRP_n} \tag{29}$$

or

$$\frac{1}{APC_n} = \frac{PRP_n}{RPC_n} \tag{30}$$

Substituting (30) into (28) causes the equation to simplify to:

$$N = \frac{n}{APC_n} \tag{31}$$

Finally, we can calculate the Release Date from (12) using this simplified formula:

$$RD_v = SD + L \cdot \left(\frac{n}{APC_n}\right) \tag{32}$$

4.5 Deriving a Release Date based on EAC from RD_v

If we can derive an equation in the form of (12) from (32) for a release date based only on EVM terms, we will have correlated it to mean velocity. Considering (31), we can manipulate this equation as follows.

$$\frac{n}{APC_n} = n \cdot \left(1 + \frac{1}{APC_n} - \frac{APC_n}{APC_n}\right) \tag{33}$$

and

$$\frac{n}{APC_n} = n \cdot \left(1 + \frac{BAC}{BAC} \cdot \left(\frac{1 - APC_n}{APC_n}\right)\right) \tag{34}$$

But from (2), $EV = APC \cdot BAC$, so

$$\frac{n}{APC_n} = n \cdot \left(1 + \frac{BAC - EV}{EV}\right) \tag{35}$$

From (5), $CPI = EV / AC_n$, *or* $EV = CPI \cdot AC_n$ so

$$\frac{n}{APC_n} = n \cdot \left(1 + \frac{BAC - EV}{CPI \cdot AC_n}\right) \tag{36}$$

From (7), $ETC = 1/CPI * (BAC - EV)$, so

$$\frac{n}{APC_n} = n \cdot \left(1 + \frac{1}{AC_n} \cdot ETC\right) \tag{37}$$

or

$$\frac{n}{APC_n} = \frac{n}{AC_n} \cdot \left(AC_n + ETC\right) \tag{38}$$

But, from (8), $EAC = AC + ETC$, so

$$\frac{n}{APC_n} = n \cdot \frac{EAC}{AC_n} \tag{39}$$

That is both (n / APC_n) and $(n * EAC / AC_n)$ will calculate total Sprints. Also, for Sprint n we have equivalent values for determining APC_n.

$$APC_n = \frac{RPC_n}{PRP_n} = \frac{AC_n}{EAC} \tag{40}$$

Substituting (39) into (32),

$$RD_v = SD + L \cdot \left(n \cdot \frac{EAC}{AC_n}\right) \tag{41}$$

Thus we have derived RD_v solely in EVM terms so, let us define RD_{EV} such that $RD_v = RD_{EV}$. Finally,

$$RD_{EV} = SD + L \cdot \left(n \cdot \frac{EAC}{AC_n}\right) \tag{42}$$

5. AgileEVM Test Implementation

With AgileEVM defined, and equipped with a spreadsheet for tracking purposes, we were ready to try this out on some test projects. We chose two very different projects for the initial test set. The goals for the test were:

1. Validating the correspondence of the projected release dates by comparing mean velocity and EVM analysis.
2. Determine if the EVM metrics provided additional value beyond the established Scrum metrics for making project decisions.
3. Analyze whether the impact of cost analysis adds value to technical teams for making good decisions.

Two projects using the Scrum methodology were tracked using both the AgileEVM and the burndown method. These projects have very different adaptations to the Scrum framework, one utilizing very short Sprints with a small team, and the other being a single large team with monthly sprints. Both projects used story points for estimation purposes. It is important to note the larger team project is still active at the time of this writing, however, we feel the results of tracking the projects for nine months is sufficiently demonstrative to report. The smaller team, Project A, has recently been released to production.

For both projects, a data gathering spreadsheet was developed to collect the data and perform both the velocity trend analysis and the AgileEVM calculations on a Sprint-by-Sprint basis. This method provided for lightweight data collection along with easy comparison of the results between the burndown metrics and the AgileEVM metrics.

Project A is a one-half million dollar web application project. The team is developing a product enhancement and therefore is closely monitored by company management. The collocated team has five to seven members and is experienced with the Scrum process. Due to dynamic business requirements, sprints are one week long. Both the AgileEVM and Scrum metrics are communicated to the entire team at the end of each sprint. Key management stakeholders are presented with the Burndown charts and AgileEVM metrics on a monthly basis.

Project B is a very large, externally funded project. The project size is about three million dollars for the first eighteen month phase. The Project B team is much larger than usual with 15-18 members, and continues to add team members every couple of sprints. This team is new to Scrum and Agile development methods. They are also collocated for the majority of the day. Sprints are 30 calendar days long. Unlike Project A, the primary project guide on Project B is the Scrum Burndown chart, not the AgileEVM metrics. We were still able to validate the correspondence of the data but circumstances limited our ability to interpret decision support value.

5.1 AgileEVM Test Results

Our analysis of the data from both projects demonstrated indistinguishable results between the predicted release dates using the mean velocity method and the earned value method (see Fig. 2 and Fig. 3). This result was as expected; following the mathematical analysis proving that these two formulas are equivalent. We also tracked release date projections based upon atypical calculations for *EAC* [2]. We noticed that in the early Sprints, the atypical calculations provided more realistic predictions. This is consistent with the nature of discovery inherent in Agile projects.

The velocity of both projects fluctuated each Sprint. Project A had cyclic velocity with several highs and lows (see Fig 4: Project A velocity). Project B started stabilizing about Sprint 5 (see Fig. 5: Project B velocity). Because of the fluctuations in velocity, the mean varies considerably from the actual velocity for some Sprints; though comparing the predicted release dates to the burndown charts demonstrates a close relationship.

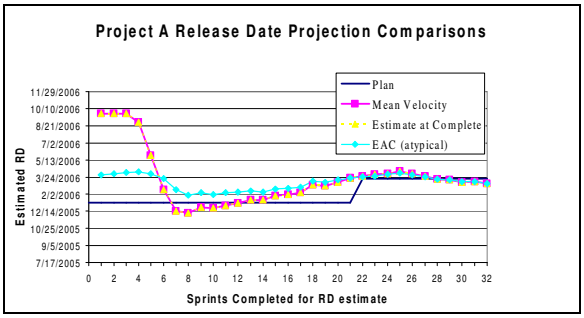

Fig. 2: Project A release date projections

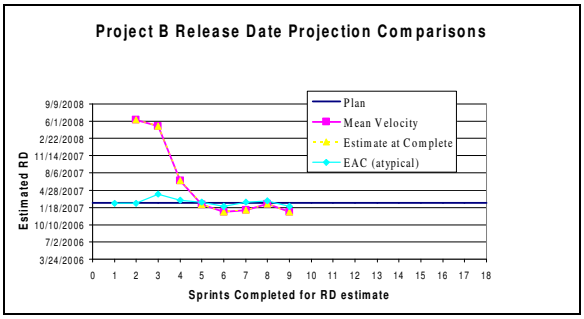

Fig. 3: Project B release date projections

The combined burndown and performance index charts provide evidence of the additional value provided by AgileEVM. Consider Project A (see Fig. 6: Project A Burndown, CPI, and SPI). This

project had a 30% budget buffer and the CPI index clearly shows the erosion of this buffer that the burndown doesn't recognize. Adjusting the planned release date in Project A at the end of Sprint 21 caused the SPI to move from an index value of 0.7 to 0.97 (see Fig. 2 and Fig. 6). This aligned the planned and projected release dates along with the AgileEVM indicators.

Project B has been adjusting scope down. The SPI and CPI is registering the potential ROI of reducing scope in Sprint nine (see Fig. 7: Project B Burndown, CPI, and SPI). This scope reduction is demonstrated by comparing the flattening velocity of Sprints six through nine to the indices (see Fig. 5: Project B velocity). Note that the same result can also be interpreted by evaluating the burndown graph and the velocity. This action of reducing scope has kept the cost and schedule constraints within current expectations.

The data also showed strong correlation in the cost information provided by AgileEVM and manual planning calculations. Given the expected behavior of the indices in traditional EVM, it is clear that the CPI, SPI and EAC behaved as expected given the budget, schedule, and scope.

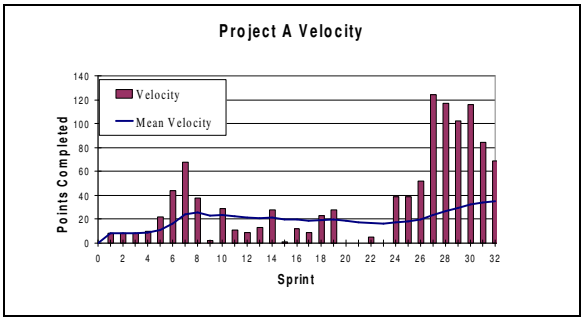

Fig. 4: Project A velocity

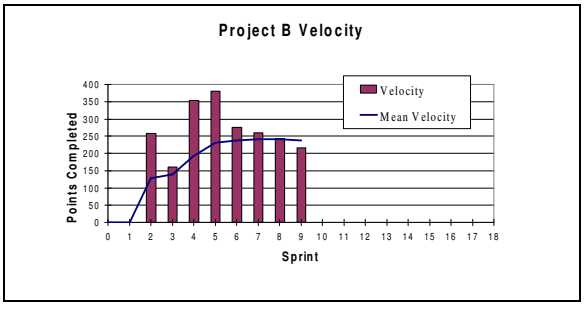

Fig. 5: Project B velocity

An interesting aspect is the way the information was used on Project A. The metrics providing the most value to team members and management were the CPI, SPI, and EAC. These metrics were requested by both executive management and team members alike. Given that significant change is expected on an Agile project, it was necessary to remind team members and management that the forecast release date, percent complete, Effort To Complete and Estimate At Complete are based on what is actual “right now”. This is exactly the same as with the burndown method.

Executive management was comfortable with the AgileEVM metrics and found them useful in making decisions concerning the direction of the product. Reviewing this data helped them validate decisions and the need for process change.

5.2 The Agility of AgileEVM

A key question of the usefulness of AgileEVM is the level of agility. Would the implementation of the method add "drag" to a Scrum team’s velocity? If so, would the impact to velocity outweigh the added value of the information?

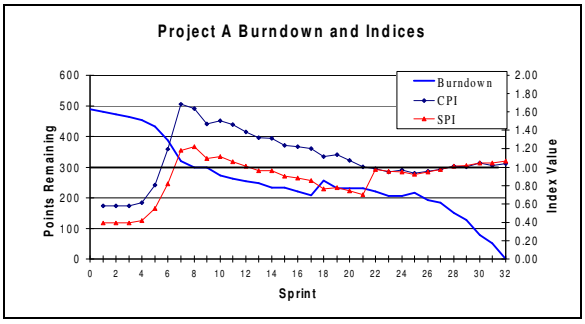

Fig. 6: Project A Burndown, CPI, SPI

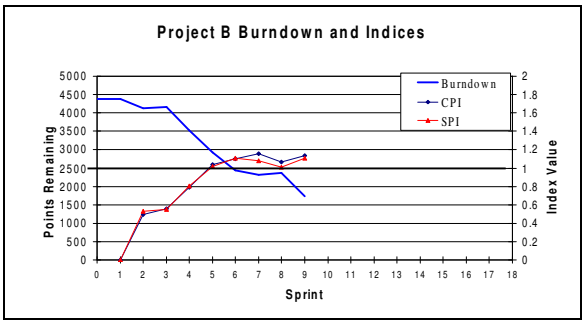

Fig. 7: Project B Burndown, CPI, SPI

The question was posed to the team on Project A: "Did the process of collecting or calculating the AgileEVM metrics add weight to the project?" The answer was a resounding, "No! The data used in AgileEVM is data that is already available. The calculation and tracking of the metrics was automated

in the spreadsheet that the ScrumMaster maintained. Communication of the metrics to the project team was as easy as posting the results on the wall, and having the information available during the project retrospective discussions."

Most of the developers reacted positively to the availability of the data, feeling that if an analysis was needed in order to determine a cause of a certain issue, like a sudden loss in velocity, the data is readily available to review. The team was able to quickly identify changes in key metrics, and this aided in the continuous improvement of the team.

The Product Owners for both projects felt that the AgileEVM metrics did not provide any more schedule insight than the burndown chart provided. It is interesting to note that the Product Owners for these projects do not have budget responsibility. We agree that without the need to manage cost performance, AgileEVM does not add significant value above traditional burndown methods.

The ScrumMaster/project manager for Project A, who does have budget responsibility, found the data particularly useful for validating project status, tracking and forecasting potential budget issues, identifying alternative strategies by running "what if" scenarios using AgileEVM to predict results, as well as communicating status and issues with the product owner and management. .

The ScrumMaster/project manager on Project B did not use or report the AgileEVM metrics, nor were the metrics shared with team members or management stakeholders. Data was collected, analyzed and compared to actual results without impacting the team. The very fact that the process of collecting and calculating the data did not impose on the team at all gives further credence to the view that the AgileEVM process passes the agility test.

6. Conclusions

In researching the validity and value of using EVM on scrum projects several key questions needed to be answered. First, are the metrics valid for Scrum projects? Second, is the process lightweight? Third, do the resulting metrics add value?

Assuming that the burndown trend analysis is valid for Agile projects, we have shown mathematically that the calculations for Release date using Estimate At Complete and calculations for Release date using the burndown trend analysis are the same. Empirically, we have tracked the results of two projects and have shown the indistinguishable results of the data generated from both methods. We feel that the validity of the AgileEVM techniques is established.

The implementation of the AgileEVM process has no noticeable impact on a Scrum team's velocity. Also, the value of the data was confirmed by the team who had access to the metrics, as well as the ScrumMaster and management stakeholders for the project. Thus, we are encouraged that the AgileEVM metrics do indeed add value to Scrum projects.

For the ScrumMaster, it is clear that metrics that are familiar go a long way to ease the discomfort that new, unfamiliar methodologies can induce. The analysis that AgileEVM provides, along with the burndown method, helps to substantiate intuition and provides executives with quantitative data in a consistent manner. The cost analysis, with its forecast Estimate at Complete and Estimate to Complete are valuable to Agile stakeholders calculating estimated ROI. Agile stakeholders who are responsible for making budget decisions find this information extremely valuable.

Our recommendation is that AgileEVM be used in conjunction with the Burndown chart and team velocity as supporting data. One important caveat is that change is expected on Agile projects and so the AgileEVM metrics are derived from what is true at each Sprint boundary.

Providing the team and Agile stakeholders with useful and understandable data is vital to the "rudder" with which the Scrum team steers toward better processes and continuous improvement. By providing the burndown and AgileEVM metrics together, the team is better equipped to succeed.

References

[1] Schwaber Ken, *Agile Project Management with Scrum*, Microsoft Press, Redmond, WA, 2004.

[2] "*Guide to the Project Management Body of Knowledge - PMBOK® Guide 2003 Edition*", Project Management Institute, Pennsylvania, 2003.

[3] David S. Christiansen, Daniel V. Ferens, "Using Earned Value for Performance Measurement on Software Development Projects, Acquisition Review Quarterly, DAU Press, Fort Belvoir, Virginia, Spring 1995, pgs 155 – 170, http://www.suu.edu/faculty/christensend/EVonSWprojects.pdf.

[4] Fleming, C.D., Koppelman, Joel M. Koppelman, *Earned Value Project Management – 2nd Edition*, Project Management Institute, Newtown Square, Pennsylvania, 2000.

[5] Quentin Fleming and Joel Koppelman, “Earned Value Project Management – A Powerful Tool for Software Projects”, Crosstalk - The Journal of Defense Software Engineering, Ogden, UT, July 1998. http://www.stsc.hill.af.mil/crosstalk/1998/07/value.asp

[6] Stephen H. Lett, “An Earned Value Tracking System for Self-Directed Software Teams", European SEPG 98, Conference Proceedings, 1998, http://www.benchmarkqa.com/PDFs/software_teams.pdf.

[7] Glen B. Alleman, “Project Management == Herding Cats, A Field Report, Agile Project Management”’ PMFORUM, PMFORUM.ORG, February 2003, http://www.pmforum.org/viewpoints/2003/0203agilepm.htm

[8] Glen B Alleman and Michael Henderson, “Making Agile Development Work in a Government Contracting Environment – Using Earned Value to Measure Velocity, Agile Development , Salt Lake City, Utah, June 25 – 27, 2003. http://www.niwotridge.com/PDFs/ADC%20Final.pdf

[9] Cockburn, Alastair, *Crystal Clear: A Human-Powered Methodology for Small Teams*, Pearson Education Inc. Upper Saddle River, NJ, 2005

[10] Cohn Mike, *User Stories Applied: For Agile Software Development*, Pearson Education, Inc. Boston, MA, 2004.

[11] Cohn Mike, *Agile Estimating and Planning*, Pearson Education, Inc. Boston, MA, November 2006.

[12] Glen B Alleman, Nine Best Practices, The Software Management Framework, 3 June 1999.

[13] David Christensen, “The Estimate At Completion Problem: A Review of Three Studies,” Project Management Journal, Project Management Institute, March 1993 pp24:37-42.

[14] David S Christensen, Ph.D., “Determining An Accurate Estimate At Completion”, National Contract Management Journal, Arlington MA, 1993, pp 25:17-25.