# Earned Value and Agile Reporting

Anthony Cabri, Mike Griffiths
*Quadrus Development Inc.*

## Abstract

*This paper reviews the concepts of Earned Value Management established in traditional project management, and determines whether and how they can be applied to software development projects following an Agile methodology.*

*First the origins and concepts of Earned Value are reviewed, followed by its application in traditional projects. Then the application of Earned Value Management to Agile software projects is investigated.*

## 1. Introduction

This paper reviews the origins and concepts of Earned Value, followed by its application in traditional projects. It then investigates the application of Earned Value Management to Agile software projects.

## 2. History of Earned Value

Earned Value is a project management technique to measure, at a specific date, the progress and performance of a project against the plan, and to estimate future performance. Earned Value considers 3 dimensions: 1) planned expenditures, 2) actual expenditures, and 3) budgeted expenditures for actual work accomplished. This provides a superior view into the project state than only looking at the first 2 dimensions.

The concept of Earned Value began in the 1890's as the early industrial engineers measured performance in American factories. They defined a "cost variance" to relate "earned standards" against "actual expenses" to determine performance.

It was only in 1962 that Earned Value was formally introduced on projects by the US Navy, as part of the development of the PERT/Cost methodology. In 1996, a new set of criteria were produced to encourage adoption in the private industry, by making the criteria more 'user friendly'. The National Defense Industrial Association (NDIA) developed these 32 criteria and named it the Earned Value Management System (EVMS) criteria, currently embodied in ANSI/EIA 748.

Finally, the Project Management Body of Knowledge (PMBOK), developed by the Project Management Institute, recommends utilizing a similar set of Earned Value criteria, as part of Project Cost and Project Communications Management (Performance Reporting).

## 3. Earned Value Management

To measure the 3 dimensions described earlier for Earned Value Management, and apply them to a project, the following key values are required[1]:

- Planned Value (PV): The budgeted cost for the work scheduled to be completed up to a given point in time. Formerly known as BCWS (Budgeted Cost for Work Scheduled).
- Earned Value (EV): The budgeted amount for the work actually completed during a given time period. Formerly known as BCWP (Budgeted Cost for Work Performed).
- Actual Cost (AC): The total actual cost incurred in accomplishing work during a given time period. Formerly known as ACWP (Actual Cost for Work Performed).

The PV, EV, and AC values are combined in various ways to provide project performance metrics.

[1] Adapted from PMBOK, 3rd edition, Project Management Institute, 2004

These values can best be understood graphically, as shown in Figure 1.

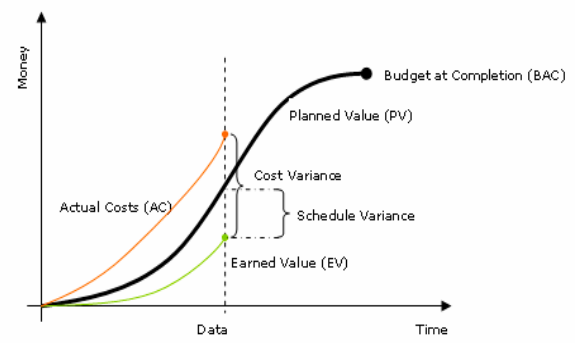


**Figure 1. Earned Value.**

The primary metrics are:

• Cost Variance (CV) = EV – AC

• Schedule Variance (SV) = EV – PV

In addition to these metrics, commonly used normalized performance indices are:

• Cost Performance Index (CPI) = EV / AC. A value < 1.0 indicates a cost overrun compared to the budget estimates; Can be interpreted as "I am getting x cents out of every $". This indicator can be used to forecast project costs at completion.

• Schedule Performance Index (SPI) = EV / PV. Can be interpreted as "I am progressing at y% of the rate originally planned". Note that this index is not a reflection of the schedule on its own, and should be reviewed in conjunction with the project plan to determine true project position.

• To forecast cost and/or schedule completion using Earned Value data, the following formulae are typically used[2]:

• Estimate at Completion (EAC) = AC + ((BAC - EV) / CPI), where BAC is the total PV at completion. This is the revised total cost estimate based on the earned value data & original scope.

• Estimate to Completion (ETC) = (BAC - EV) / CPI. This is the revised work schedule completion based on the earned value data & original scope.

The utility of Earned Value analysis in predicting future performance is subject to 3 critical success factors[3]:

*1. Quality of the project's baseline plan.* Earned Value is compared against the baseline plan, whether the plan is accurate or not. Therefore, cost 'overruns' will occur if the project costs are under-budgeted, and scope creep will occur if the initial project scope hasn't been adequately defined.

*2. Actual Performance against the Approved Baseline Plan.* i.e. whether the actual performance tracks to the baseline plan.

*3. Management's Determination to Influence the final results*. Final results for a project based on earned value projections can be modified based on management's commitment to take action as soon as deviations from the plan are observed.

## 4. Earned Value on Traditional Projects

### 4.1 Traditional Project Assumptions

Traditional projects (i.e. physical engineering projects) make the following assumptions:

1. Scope is well understood and can be fully defined at the start of a project. Therefore, while a change control process is typically in place, few scope changes are expected.

2. Project work completes in a sequential, linear nature, where current progress rates are indicators of futures rates.

### 4.2 Earned Value Application

Now that we have defined Earned Value Management, how is it applied in traditional projects?

A task-based plan is created at the beginning of the project, and this is base-lined. The plan creation steps from an earned value perspective can be described as follows[4]:

1. Determine project scope e.g. through creation of a hierarchical work breakdown structure (WBS) of project deliverables.

2. Assign responsibility for performance for each of the specific tasks.

3. Identify key project milestones.

4. Prepare master schedule & budget.

5. Prepare detail schedules and budgets.

6. Integrate detailed schedules and budgets with project masters.

Once the project is underway, actual performance data is collected and earned value analysis applied against the baseline plan at pre-defined points, as described in Section 3. Earned Value. Project managers utilize this information to monitor costs, and take action as required to correct the project course.

Scope changes are expected to be minimal, since the project scope is well thought out at the initial project planning stage. Scope changes are managed through a change control process. If there is a scope change, the plan is re-baselined.

[2] There are several ways to calculate EAC and ETC – those listed here are used when variances are typical.

[3] Fleming, Quentin, Koppelman, Joel, "Earned Value Project Management", 2nd edition, PMI, 2000, pp128-130.

[4] Fleming, Quentin, Koppelman, Joel, "Earned Value Project Management", 2nd edition, PMI, 2000

## 5. Earned value on Agile Software Projects

### 5.1 Agile Software Project Assumptions

Agile software projects make the following assumptions:

1. Project work is completed iteratively and is not sequential & linear. For example, feedback from each iteration affects the next one; complex work may be undertaken earlier in the life cycle to reduce risk of the architectural approach, etc.

2. Scope is defined at a high level at the start of a project. Only the next iteration (what is best understood) is scoped in more detail.

3. Scope changes are expected and frequent over the course of the project, as typically software users only understand what they want when they see something tangible, and agile seeks to be responsive to this.

Agile software projects make some assumptions that are fundamentally different from traditional projects, and this will affect whether and how earned value is applied.

Traditional projects determine the scope up-front; scope change is infrequent. In Agile software projects, initial scope is not assumed to be complete. Scope is fleshed out as project iterations are completed, based on user & stakeholder feedback at the end of each iteration.

Traditional projects follow a linear progression over the course of the project life cycle. Similar projects have by and large been done before, are well defined and thus inherently have less risk. Agile software projects, on the other hand, are typically unprecedented, may require R&D, a higher level of creativity, and thus work often proceeds in a non-linear fashion. As the true scope of the project becomes clear, some rework of initially completed elements may be required. Figure 2 and Figure 3 illustrate these differences. For these reasons, one cannot utilize a linear extrapolation of work completion based on work done to-date in an agile software project setting.

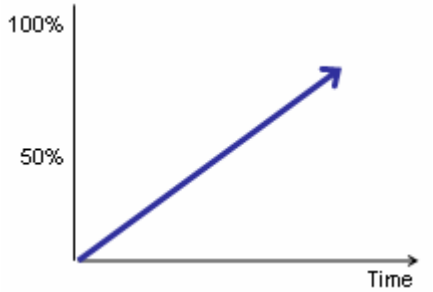


**Figure 2. Traditional project progress.**

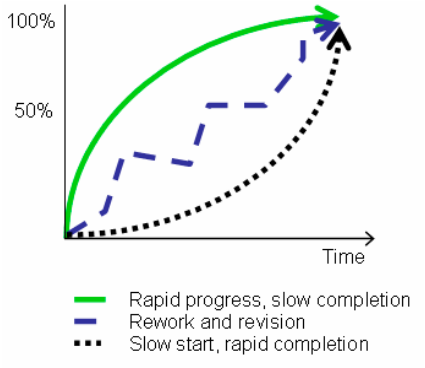


**Figure 3. Agile software project progress.**

The challenge in applying Earned Value directly to an agile software project is due to the reasons above, as they violate the first Earned Value critical success factor, namely "Quality of the project's baseline plan". The scope of an agile software project cannot be fully defined in a bottom-up fashion at the beginning, as is done in traditional projects.

Thus, directly applying Earned Value Management in agile projects will likely result in an invalid Planned Value (PV) at the start of the project, with over- or under-runs occurring during project execution; many re-baselines would be required.

So what does this mean? Is Earned Value irrelevant for Agile projects? No: The concept of Earned Value, relating actual physical progress to actual costs is just as relevant for agile projects as for any other project. Project Managers are always looking to answer questions such as "How much have we done?", "How much more is still to be done?", "What have we spent?", and "How much will this whole thing cost". The key is to look at agile project tracking mechanisms and how they can be used and/or enhanced to provide this information.

The following section reviews popular agile project tracking techniques with a discussion on how to use them to answer the above questions.

### 5.2 Agile Software Project Tracking Options

**5.2.1. Burn-Up Charts.** One of the principal agile project management tracking techniques is burn charts, and as we will see, these correspond well to the concept of Earned Value. There are 2 principal types of burn charts in agile: burn-up and burn-down.

A burn-up chart shows the increasing amount of functionality accomplished as a function of time, and is reported on a regular basis.

The benefit of a burn-up chart is an easy to understand depiction of status and rate of delivered features. This is conceptually equivalent to the Earned Value accumulated at a specific date. Figure 4 is a

sample burn-up chart, with planned and earned values depicted. The thick line shows planned value, the thin is earned value.

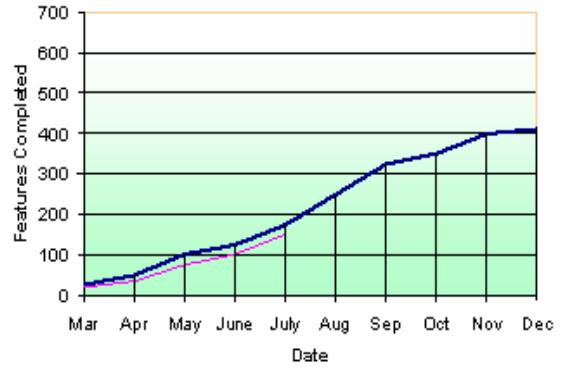


**Figure 4. Sample burn-up chart.**

Only running, tested features[5] are reported in agile software projects i.e. partially completed features are not tracked. This type of reporting provides a stronger and more believable picture of the project status, as it avoids the "feature 90% complete" syndrome.

**5.2.2. Burn-Down Charts.** A burn-down chart presents similar information, in a manner that clearly indicates how many features remain to be completed i.e. keeping the end goal in mind.

In addition to showing how many features remain, the sample burn-down chart in Figure 5 captures scope change as the project progresses. Notice how scope change increases do not occur during an iteration, but rather after the iteration has completed. This allows the project team to focus on the committed features during the time-boxed iteration. Burn-down charts succinctly capture the amount of scope change that occurs during the project, the amount and frequency of which is of interest to the project manager.

In this example chart, "features" to be developed could be XP story points, use cases or other non-functional requirements.

[5] *Running Tested Features.* (n.d.). Retrieved April 27, 2006 from http://tinyurl.com/rk5qm

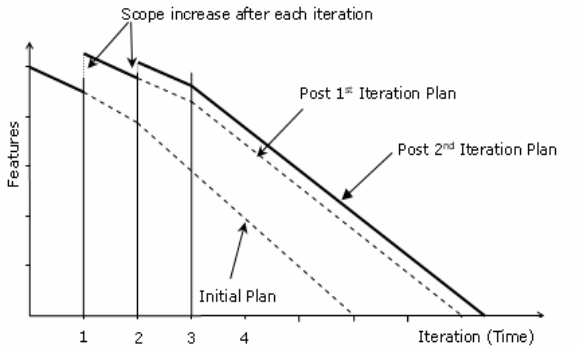


**Figure 5. Sample burn-down chart.**

It may be possible to extrapolate final completion dates from these charts, through estimation of the entire feature set. With each iteration, scope and estimates should improve so that final cost figures can be provided with better confidence; taking into account that progression may not be linear, a range estimate is recommended.

**5.2.3. Cumulative Flow Diagrams.** Cumulative Flow Diagrams (CFD) are another method for tracking progress on an agile software project, building upon the basic burn-up charts.

As with burn-up charts, CFDs clearly portray the proportional number of completed features over time. The additional benefit of CFDs derives from their depiction of work in progress. This provides further detailed understanding of the project status at any point in time, as well as allowing for early detection and correction of problems.

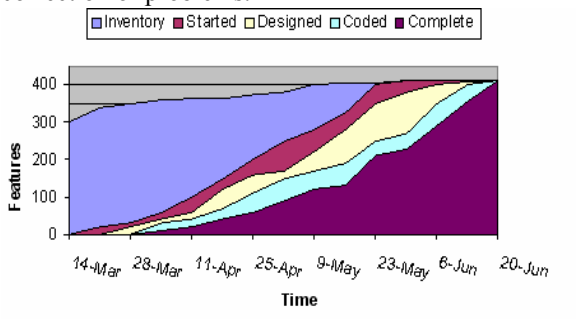


**Figure 6. Sample cumulative flow diagram.**

## 5.3. Earned Value Application

So how do these charts relate to Earned Value? While Earned Value metrics are not directly applicable, Earned Value concepts can certainly be applied.

The concept of Schedule Variance is perhaps the easiest to visualize with burn charts. In Earned Value reporting this is a cost-based measure, however with

burn charts it is measured as feature-based, as shown below. The larger the variance, the more review should be placed on the quality of the estimates being created and possibly the scope the project.

It is important to note that schedule variance is only of value on an iteration basis, as this is the level for which detailed estimates are created. If schedule variance is required on a monetary basis, this could be calculated using actual resource costs.

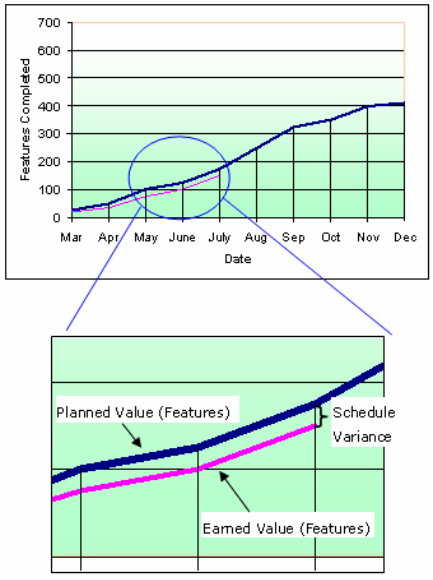


**Figure 7. Burn-up chart & schedule variance.**

As previously mentioned, scope changes that occur during the course of the project can also be reflected in the burn chart. After each iteration, as the overall scope is better understood, revised effort & cost estimates for the total project effort can be undertaken. With each iteration, these estimates should become better and converge to the final value, as can be seen in Figure 8.

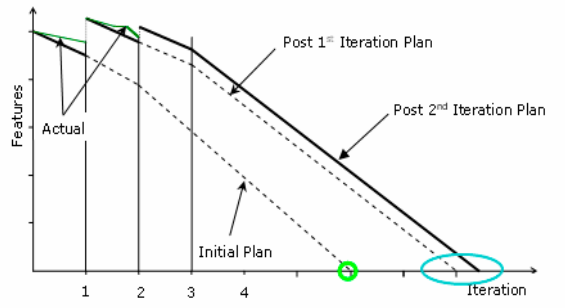


**Figure 8. Burn-down chart reflecting scope and estimate changes.**

Burn charts have typically not tracked costs. However, expenditures can be graphed on a burn chart, so trends can be related to feature completion. In the sample burn-up chart in Figure 9, 'features completed' as well as 'planned costs' and 'actual costs' are shown. From this, one can compare the cost and feature completed slopes – an increase in costs should be associated with a similar increase in features. If not, an explanation or further investigation may be required. Note, however, the limitation that costs are usually related to developer feature costs, not total project costs e.g. user training, etc.

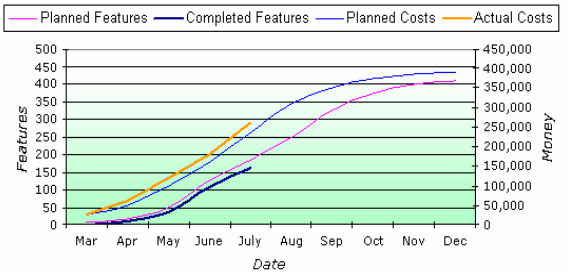


**Figure 9. Burn-up chart & costs.**

In the case of the burn-up chart in Figure 9, despite an increase in actual over planned costs, the delivered features do not meet what was planned for each iteration. Assuming a pre-approved budget, the feature list can be trimmed on a go-forward basis, as is shown in the burn-up chart in Figure 10. The example shows that the new scope is reduced by roughly 100 features from the original.

By interpreting these charts, we are able to evaluate the 3 fundamental dimensions of Earned Value, as they pertain to the development of features. Specifically, 1) planned expenditures (Planned Value) observable with the Planned Costs curve, 2) actual expenditures (Actual Cost) observable with the Actual Costs curve, and 3) budgeted expenditures for actual work accomplished (Earned Value) derivable from the Completed Features curve.

An equivalent measure to SPI can be observed as: Completed Features / Planned Features. When Completed Features are less than Planned Features, the progression rate is less than planned (and the SPI value is < 1). In this scenario, this is an indication that iteration estimates need to be further refined, and the project scope possibly re-adjusted.

Similarly, an equivalent measure to CPI can be observed as: Planned Costs / Actual Costs. An overrun occurs when the Actual Costs are over the Planned Costs, and the CPI value is < 1.

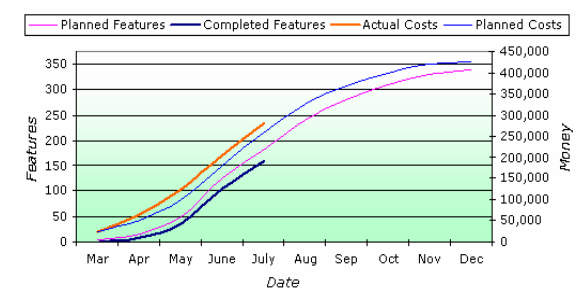


**Figure 10. Burn-up chart with trimmed scope.**

While these indicators may prove useful for snapshot status, utilizing them to predict final costs & schedules may be misleading, given the non-linear progression aspect of agile software projects. However, the above charts point out, early on, problems for which corrective action may need to be taken, a key goal for Earned Value. This is done through regular updates of the chart data (which can be accomplished via daily standups), as well as through the agile practice of short time-boxed iterations.

Finally, CFDs can also show the benefits of Earned Value concepts in the same way as burn-up charts. In addition, the work in progress provides an indication of lead time, iteration size, as well as a measure of 'knowledge work inventory' i.e. capturing the transformation state from idea to a working feature[6]. These are useful metrics in software projects, and are not available with traditional Earned Value.

## 6. Conclusion

Earned Value is a project management monitoring & reporting technique that has been developed and utilized over the course of the last 100 years in traditional engineering projects. It relies on an initial task-based baselined plan for measuring progress, and a project with well-defined scope that evolves in a sequential, linear fashion.

In agile, projects evolve in an iterative, non-linear fashion, with feedback loops that affect the initial plan. Change is expected and frequent throughout the project lifecycle, thus measuring progress relative to the initial plan will be misleading.

While there are issues with attempting to apply Earned Value to Agile projects, agile project management techniques such as burn charts (indicating the amount of functionality outstanding vs. completed over time, etc.) provide status & progress information very similar to what Earned Value attempts to measure. Costs can be added to the charts to view the information together with rate of feature completion. All of these may be more valuable to the project manager and project stakeholders in monitoring an agile project, rather than attempting to apply traditional Earned Value.

[6] Reinertsen, Donald, "Managing the Design Factory – A Product Developers Toolkit", Free Press, New York NY, 1997

## 7. References

1. Christensen, Maj. David S., USAF, "Using Performance Indices to evaluate the Estimate at Completion", *Journal of Cost Analysis* (Spring 1994).
2. Cockburn, Alistair, *Crystal Clear*, Addison-Wesley, 2004.
3. Fleming, Quentin W., Koppelman, Joel M., *Earned Value Project Management*, 2nd edition, Project Management Institute, 2000.
3. Schwaber, Ken, Beedle, Mike, *Agile Software Development with SCRUM*, Prentice Hall, 2001
4. Smith, Kenneth F, *Earned Value* position paper, August 2000.
5. Reinertsen, Donald, *Managing the Design Factory – A Product Developers Toolkit*, Free Press, New York NY, 1997.
6. Various, *A Guide to the Project Management Body of Knowledge (PMBOK),* 3rd edition, Project Management Institute, 2004.