Software Project Status and Completion Estimation with Earned Value Management

Frank Tsui
Software Engineering, Southern Polytechnic State University

Abstract:

Earned Value Management (EVM) is a project tracking, control and management technique that has been used mostly by large, government projects [1, 4, 6, 7]. In recent years, EVM is gaining popularity among some software development managers [2, 3, 5, 9, 10]. One of the most difficult tasks in managing software projects is to accurately estimate status and to forecast the future. For example, estimating the software project size, complexity, quality and schedule has never been easy. EVM includes techniques which utilize in-project data to be used to estimate the current project's schedule and cost status along with estimated project completion schedule and cost. In this paper, we examine these techniques, discuss its potential problems as applied to software development, and propose an improved model for forecasting within the EVM paradigm.

Introduction:

With the recent high-profiled problems encountered in the US government's Affordable Health Care system, HeakthCare.gov site, software project management is once again a front-and-center topic these days [8]. Earned Value Management (EVM) technique has been a staple with many government sponsored projects since its inception as a financial management tool approximately fifty years ago [1, 7]. It is basically a cost and schedule monitoring and control technique. This technique requires the project manager to initially carry out the following steps:

- Perform a Work Breakdown Structure of the project tasks
- Assign an estimated effort, called Budgeted Cost (BC), for each task.
- For each of the tasks also assign and estimate the start and end date.
- The sum of all the BC's, called Budget at Completion (BAC), is the estimated total project effort.

The heart of the technique is the project tracking and control portion which requires the project managers to perform the following after all the initial BCs are estimated.

- For each of the Budgeted Cost tasks, track its actual effort or Actual Cost (AC) expended.
- At each project status reporting day, compare the planned versus the actual project information.
- Estimate the current project Schedule Performance Index (SPI) and Cost Performance Index (CPI).
- Compute the Estimate at Completion (EAC) for cost and schedule, utilizing CPI and SPI.

The main focus of this paper is to analyze the Schedule Performance Index (SPI), Cost Performance Index (CPI) and Estimate at Completion (EAC) for cost and schedule, discuss its applicability to software development projects, and propose an improved and modified model for software development environment.

In the next Section, we will review the basic terminology and concepts of EVM. Then, in the Problems with Project Status and Estimation at Completion Section, we will explore some of the weaknesses of SPI, CPI and EAC. Lastly, we will propose some alternatives and an improved EVM model for project status and estimation computation for a software development project.

Basic EVM Concept and Definitions

In this section we review the basic definition and concepts of Earned Value Management model. After performing the Work Breakdown Structure of a project, the effort required of each task of the project is

estimated. The effort estimated for each task is called Budgeted Cost (BC). The summation of all the BCs of the project is the total project cost, called the Budget at Completion (BAC). Thus BAC = $\sum$ BC. The Work Breakdown Structure of a typical software development project at the macro-task level includes: requirements processing, designing, implementation, testing, integration and releasing. And for each of these macro tasks, there is an associated BC. For example, there is an estimated effort for testing, and thus there is a BC for testing. BAC would then be the total estimated software development project effort for requirements processing, designing, implementation, testing, integration and releasing tasks.

For any specific project status meeting at time, t, the total estimated effort of all the tasks that are scheduled to be completed is the Budgeted Cost of Work Scheduled (BCWS). However, it is possible that those scheduled tasks in BCWS are not necessarily the same as those that are actually completed at time, t. The total estimated effort, or the budgeted cost, of those scheduled tasks to be completed that are actually completed, or performed, is called Budgeted Cost of Work Performed (BCWP). Thus the scheduled BCWS and the actually completed of those scheduled tasks, BCWP, may or may not be the same at any specific time, t. The third metric is the actual cost (AC), not the budgeted cost, of those tasks that are completed. The sum of these AC's is called the Actual Cost of Work Performed (ACWP).

An example will clarify these definitions and concepts. Figure 1 shows the project status of a software development project on March 20. The project status discussion will all be based on the March 20th date. Note that the first column lists all the major software development tasks. The second column shows the estimated effort cost of each task in person hours. This second column represents the BC of each of the tasks. The sum of all the entries in the second column is the estimated total project effort cost, or Budget at Completion (BAC). Thus BAC for this project is 80+70+200+95+5, or 450, person hours. In the third column, the actual effort expended for each of the major tasks is recorded. One can see what these actual effort costs are on March 20th. The fourth, fifth, sixth, and seventh columns respectively list the estimated task start dates, actual task start dates, estimated task completion dates, and actual task completion dates. It is important to note that in the definitions of BAC, BCWS, BCWP and ACWP only completion dates are utilized. But we have included task starting dates here in Figure 1 even though EVM uses only task completion dates.

Figure 1: Project Status on March 20

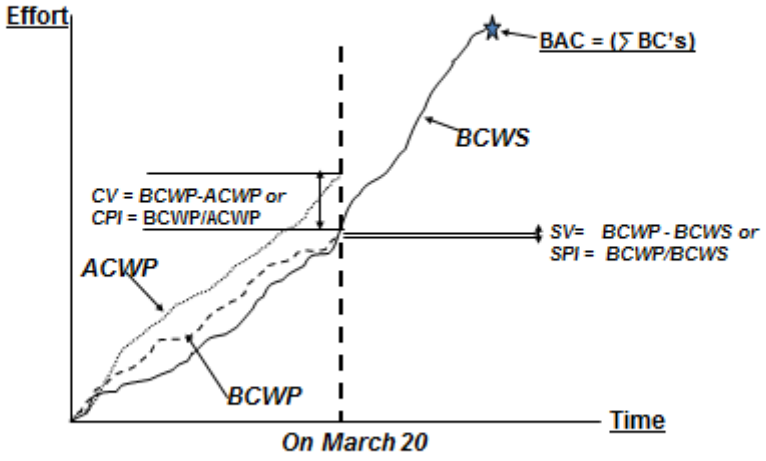
Major Task	Estimated Effort (BC) In person hrs.	Actual Effort Expended (AC) In person hrs.	Estimated Start Date **	Actual Start Date **	Estimated Completion date	Actual Completion date
Requirement	80	80	Jan 15	Jan 15	Feb 20	Feb 20
Design	70	78	Feb 15	Feb 15	March 15	March 10
Implement	200	55	March 1	March 1	April 25	-----
Test	95	40	Feb 15	Feb 20	May 15	-----
Integrate	5	0	May 10	-----	May 20	-----

*** Start dates are usually not part of EVM*

Although efforts are spent on many tasks, note that only the Requirement task and Design task were planned to be completed by project status meeting date of March 20th. Their estimated completion dates are February 20 and March 15, respectively. Thus on March 20th, the Budgeted Cost of Work Scheduled or BCWS is 80 + 70, or 150 person hours. Since both of the planned work tasks, Requirement and Design, are also completed (performed), Budgeted Cost of Work Performed or BCWP is also 80 + 70, or 150 person hours. The Actual Cost of Work or ACWP performed is 80 + 78, or 158 person hours.

We can track project status with a pair of measurements that compare planned effort versus expended actual effort, Schedule Variance (SV) and Cost Variance (CV), where SV = BCWP – BCWS and CV = BCWP – ACWP. Alternatively, one may use a pair of indexes, Schedule Performance Index (SPI) and Cost Performance index (CPI), to track project status. SPI is defined as BCWP/BCWS, and CPI is defined as BCWP/ACWP. When SPI > 1, the project is ahead of the planned schedule. When CPI > 1, the project is under-running the cost. When both SPI and CPI are equal to 1, the project is on target for both the schedule and cost. When SPI is <1, the project is behind schedule, and when CPI <1, the project has a cost over-run situation. An interesting thing to note in EVM is that both the schedule status and cost status of the project is measured by the one metric, amount of effort. Figure 2 shows the same project status in a graphical form.

Figure 2: Graphical Representation of Status on March 20



The y-axis in figure 2 represents the "effort" in person hours, and the x-axis represents the "time" in days. The solid line represents the accumulative BCs, which ends as the BAC; this is the BCWS line. The heavy dotted line represents the BCWP, and, in this example, it meets the BCWS line on March 20th. ACWP is the light dotted line, which on March 20th is above the BCWP line. The gap or delta between BCWP and ACWP is shown as CV. This cost variance status may also be represented as CPI. On March 20th the CPI = BCWP/ACWP = (150/158) =.95 < 1, and we have a small cost over-run. The gap between BCWP and BCWS is shown as SV, which on March 20th is really zero. The schedule status represented

by the alternative index SPI = BCWP/BCWS = (150/150) = 1, and the project is tracking exactly on schedule.

Lastly, EVM is not only used as a project status tracking and monitoring mechanism, but it may also be utilized as a predictive tool. EVM model provides a computational formula for Estimate at Completion (EAC) for project schedule and cost. The estimation process is heavily based on "in-project" information. The two basic formulae for EAC are:

1. (for schedule) EAC_S = ACWP + (BAC-BCWP)/SPI and
2. (for cost) EAC_C = ACWP + (BAC-BCWP)/CPI

The estimation at completion is based on the actual efforts that are already expended, ACWP, and what is remaining to be completed, (BAC-BCWP). In the case of schedule, the "what is remaining to be completed" portion is influenced by the hitherto "in-project" data of SPI. Thus (BAC-BCWP) is divided by SPI. In the case of cost, the "what is remaining to be completed" portion uses the hitherto "in-project" data of CPI and divides (BAC-BCWP) by CPI. Thus on March 20th, EACs = 158 + (450 – 150)/1 = 458, and EACc = 158 + (450-150)/.95 = 158 + 316 = 474. We have rounded up the EAC_c number to an integer. So, for this example, the estimation for schedule at project completion is 458 person hours, and the estimation for cost at project completion will be 474 person hours. These projections say that, on March 20th, the Estimate at Completion for both schedule and cost will exceed the initial BAC estimate of 450.

In the next section we will further analyze this prognostication part of EVM for software development projects and then offer some improvements.

Problems with EVM Project Status and Estimation at Completion

In this section we will explore the estimation capabilities of EVM as applied to software development projects. In the above example, the project status meeting on March 20th indicates that SPI = 1 and CPI <1. The Estimate at Completion for schedule and cost both projected a schedule delay and cost over-run. The two metrics of EVM, EAC_c and CPI, on March 20 do agree. The project status shows cost over-run and also prognosticates a cost over-run at project completion.

At the same March 20th status meeting, SPI = 1 shows that the project is tracking to schedule, but EACs = 458 says that the project is estimated to exceed the estimated schedule of 450 and that there will be a delay. These seemingly contradicting statements from SPI and EACs metrics point out that there is a potential weakness in EVM measuring system, which uses the same metric of "effort" to measure both schedule and cost. Recall, that we have also purposely included task starting dates in Figure 1 even though EVM does not utilize them. When we examine the task start dates in Figure 1, we notice that the Test task did not start on time. For seasoned managers, the delay in an actual task start date is a much more significant concern. Thus, examining SPI metric alone is not a strong enough tracking measurement for project schedule status. One more item stands out in Figure 1 that affects the accuracy of project status as represented by SPI and CPI. Both Implement and Testing tasks have started and have expended some effort. Depending on how we handle the inclusion of effort of these partially completed tasks, SPI, CPI and the resulting EAC would all be very different.

A more subtle problem exists in the EAC computations for software development projects. Although incorporating in-project data, SPI and CPI, in the projection of the future makes sense, it is important to check the applicability of the in-project data. In the example above the CPI and SPI indices are based on the "in-project" data of Requirement and Design efforts. The remaining tasks are Implement, Test and Integrate. In large software development projects, these different macro tasks are performed by different people, utilizing different techniques and tools. Thus, the in-project performance data of Requirements and Design tasks are most likely not very applicable in the projection of the remaining, different tasks.

Moreover, in software development, the effect of a defect often propagates across the different phases or tasks of the project. Some of these defects are found and eliminated early, but some escapes inspections and reviews, creating unexpected major problems in down-stream activities. The "quality" affect in software projects can create major havoc to schedule and cost. This may be relatively unique to software projects, but it is not captured by the EVM model when computing EAC.

In the next section we will offer suggested improvements to the EVM model to alleviate some of the problems discussed in this section.

Suggested Improvements to EVM for Software Projects

In this section we offer some suggestions to be included in EVM when managing software development projects. First, let's look at the partial completion of tasks which are not computed into the BCWP. Note that partial efforts of Implementation and Testing tasks in Figure 1 are not included into the schedule status, SPI, and the cost status of CPI. To be more accurate, we need to capture these partial efforts. We suggest the following newly altered scheme. Compute the proportional effort, using the scheduled time-duration, for those tasks that are partially completed on the project status date. We propose the following definition and modified measurements. If a task j, is only partially completed, then let yj represent the partial-BC of the original estimated BC for task j. For any task j, let estimated elapsed time (EET) = (scheduled task j end date – scheduled task j start date). Then for any incomplete task j, the percentage of schedule (PS) = (project status date – scheduled task j start date)/(task j's EET) . Then, if xj is the original BC, yj = (xj * PS). The newly modified terms would then be:

- NewBCWS = ∑(completed BC's) + ∑(yj's)
- NewBCWP = ∑(completed BC's) + ∑(AC's of partially completed tasks)
- NewACWP = ∑(completed AC's) + ∑(AC's of partially completed tasks)
- NewSPI = (NewBCWP/NewBCWS) and NewCPI = (NewBCWP/NewACWP)

In using the originally scheduled start and end date for duration, note that even if the project is started late the proportional effort will result in the originally planned percentage. Thus the late start date problem mentioned earlier is partially handled by the new scheme. Include those proportional efforts into the NewBCWS for project status. Then compute the NewBCWP utilizing the actually expended effort of those partially completed tasks as the planned effort. Different from before, NewBCWP is now a mixture of completed planned efforts and partially completed actual efforts. NewACWP is just the sum of all those actually expended efforts. There is one caution that must be folded in. In the event that a task k is started before the planned start date and the project-status date also occurs before that task k's planned start date, then task k's PS will be negative. In such a situation, we would not include, for that project status meeting, the partially completed task k in the ∑(yj's) of NewBCWS.

For our above example, on March 20th, we are 20 days into a planned elapsed time of 55 days (March 1 to April 25) for Implementation task. This is approximately 20/55 = .36 of the estimated 200 person hours for Implementation Task. Thus we have 200 x .36 = 72 person hours as the estimated effort for Implementation. Using the similar approach, we arrive at 95 x .31 = 30 person hours as the estimated effort for Testing. The NewBCWS is now 80+70+72+30 = 252 person hours. The NewBCWP would now be 80+70+55+40= 245 person hours. The NewACWP = 80+78+55+44 = 257 person hours. Then the NewSPI = 245/252 <1, and the NewCPI = 245/257 <1. With the NewSPI and NewCPI, the project status is that the project is slightly behind schedule and slightly over-cost. This modified measurement scheme incorporates partially completed projects and alleviates the delayed actual start date problem. We still strongly advise project managers to not only look at the EVM metrics, which involve mostly efforts, but also study the actual dates during a status meeting with charts such as Figure 1.

The second major area of potential problem with the current EVM model is the non-discriminative usage of the in-project SPI and CPI for EAC. Note that in Figure 2, the y-axis represents Effort and the x-axis represents the Time. This means that with EVM, aside from the initial forecasting of BC, there is little concern with the different types of activities. In software development, we move from one type of task such as requirements analysis to another type of task such as design. Thus the x-axis in Figure 2 would be more meaningful if it represented tasks, rather than just time. EAC, in forecasting the future, need both in-project NewSPI and NewCPI and some data about the yet to work on different tasks. If the NewSPI and NewCPI include many of the partially completed remaining tasks, then they probably include enough in-project characteristics of the remaining tasks that we can just compute the EAC by replacing SPI and CPI with these NewSPI and NewCPI. The "remaining effort" would then be represented as follows: (i) (BAC-BCWP)/NewSPI and (ii) (BAC-BCW)/NewCPI.

In the case where most of the remaining tasks have not yet started and thus there really is very little in-project data to help, we can't just use NewSPI and NewCPI as the modifier of (BAC-BCWP). We will need to look outside of the present project and recast the (BAC-BCWP) modifier with some past tasks' history. This newer modifier is currently under research and is a discussion for a future paper.

Conclusion:

In this paper we examined the traditional EVM approach and its applicability to software projects. Some of the problems of accurately representing software project status and projecting the completion are examined. A modified approach to include start date and partially completed tasks in EVM is proposed as an improved formulation of the model.

References:

[1] W.F. Abba, "Earned Value Management - Reconciling Government and Commercial Practices," Project Management, special Issue, January/February 1997, pp58-67.
[2] P.G. Armour, "The Business of Software - How We Build Things," Communications of ACM, January 2013, vol.56, No1, pp 32-33.
[3] I. Attarzadeh and O.S. Hock, "Implementation and Evaluation of Earned Value Index to Achieve an Accurate Project Time and Cost Estimation and Improve Earned Value Management System," International Conference on Information Management and Engineering, Kuala Lumpur, Malaysia, April 2009.
[4] D. Christensen, "The Estimate At Completion Problem: A Review of Three Studies," Project Management Journal 24, March 1993, pp 37-42.
[5] H. Erdogums, "Tracking Progress through Earned Value," IEEE Software, September/October 2010, pp 2-7.
[6] Office of Secretary of Defense, Earned Value Management, www.acq.osd.mil/evm, accessed December, 2012.
[7] Q.W. Fleming and J.M. Koppelman, Earned Value Project Management, 4th Edition, Project Management Institute, Inc., 2010.
[8] J. Markon and A. Crities, "Health-care Web site's lead contractor employs executives from troubled IT company," The Washington Post, November 15, 2013.
[9] T. Sulaiman, B. Barton, T. Blackburn, "AgileEVM – Earned Value Management in Scrum Projects," Proceedings of the AGILE 2006 Conference, Minneapolis, USA, July 2006, pp 7-16.
[10] F. Tsui, "Analysis and Application of Earned Value Management in Software Development," International Conference on Software Engineering and Practices, Las Vegas, USA, July 2013.