Earned Value Management Software Projects

Robert A. Hanna, PMP, *Member, IEEE*

***Abstract*—JPL has been attempting to apply EVM techniques for the last 5 years with mixed success, especially when applied to software tasks. One of the main causes is that software presents particular challenges that were not considered during the genesis of EVM in the construction industry. A small number of tasks have been able to use EVM effectively and have derived great benefits. This paper discusses challenges specific to software management and how they can be successfully addressed.**

***Index Terms*—Cost, Schedule, Software, Management**

I. Introduction

EARNED value management is a technique for providing an objective analysis of the progress of a project in terms of program dollars. The technique integrates schedule and budget performance to provide metrics that enable good decision-making. Earned value management was first widely used in the construction industry where many of the small tasks are well understood and repeatable. Software projects exhibit more volatility than construction in general and must be addressed accordingly to gain the benefit of earned value management.

II. The Challenges

Software projects are unlike construction in several areas that make earned value management a challenge.

A. Innovation and Prototypes

In software efforts there are often tasks that are associated with exploring new structures and functionality. The designs necessary to implement the new functionality are often unknown early in the project. In order to determine the correct course of action prototypes must be created. The number and complexity of these prototypes is dependant upon the skill and experience of the design team. As a result, this can introduce volatility and rework on the project that makes using EVM difficult.

B. Defect Discovery and Resolution

All projects have defects that need to be resolved during the development process. The nature and size of the defects discovered can be hard to predict, resulting in a large amount of volatility and long delays with no positive progress (BCWP) resulting in performance indices that trends downward and without reaching an equilibrium (CPI and SPI). This reduces the ability of a software project to accurately forecast (EAC, ETC).

C. Architectural Changes

As software projects progress, more knowledge is gained concerning the implementation of functionality and high-risk features. The architectural structures needed to implement the functionality may not be apparent early in the project, so fundamental architectural changes may be needed, resulting in large amounts of unplanned rework. This unplanned rework results in negative progress (BCWP) from the EVM perspective that is not allowed by most EVM software packages.

III. Solutions

There are several approaches that can make a software project more suitable for an earned value management approach.

A. Task Definitions

On software projects a key to being able to accurately track work progress (BCWP) with earned value is the guideline used for defining task duration. On projects that are small or large, if the duration of individual tasks are too long, the tracking of that task becomes highly inaccurate (especially using the percent complete method.) If task durations are too short, the tracking becomes onerous and does not reflect the work performed. Guidelines for task duration discussed in this paper must be put into practices for successful use of EVM on software projects.

B. Cost Account Structure

Some amount of rework during development is expected in any effort. If that rework is tracked under the cost accounts used for tracking the development of functionality, the volatility of the repair effort diminishes the ability for accurate tracking of progress across the project. Through the separation of the defect resolution into a level of effort cost account, the volatility can be removed from the tracking of the development effort while allowing some flexibility to address minor architectural changes.

C. Incremental Implementation Approach

In order to reduce the amount of volatility in a software project, an incremental approach for implementing functionality is helpful. The guiding principle is that at each interim stage of development the product should be

Manuscript received October 3, 2008.

R. A. Hanna is with the Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA 91125 USA (phone: 818-354-1183; e-mail: Robert.A.Hanna@jpl.nasa.gov).

functional. This decision reduces volatility by ensuring that architectural dependencies are not so far reaching that they cannot be resolved until late in the project.

Filename:	17.doc
Directory:	D:\F Drive_new\www-smcit09\abstracts-contri_papers
Template:	C:\Documents and Settings\klittle\Application Data\Microsoft\Templates\Normal.dot
Title:	.
Subject:	IEEE Transactions on Magnetics
Author:	-
Keywords:	
Comments:	
Creation Date:	10/27/2008 5:44:00 PM
Change Number:	26
Last Saved On:	11/4/2008 5:14:00 PM
Last Saved By:	Robert Hanna
Total Editing Time:	157 Minutes
Last Printed On:	5/5/2009 2:01:00 PM
As of Last Complete Printing	
Number of Pages:	2
Number of Words:	690 (approx.)
Number of Characters:	3,802 (approx.)