



**EARNED VALUE REPORTING ON AGILE SOFTWARE DEVELOPMENT
PROGRAMS WITHIN THE DEPARTMENT OF DEFENSE**

THESIS

Elizabeth Hodson, Civilian, USAF

AFIT-ENV-MS-16-M-158

**DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY**

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

DISTRIBUTION STATEMENT A.
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the United States Air Force, Department of Defense, or the United States Government. This material is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

AFIT-ENV-MS-16-M-158

**EARNED VALUE REPORTING ON AGILE SOFTWARE DEVELOPMENT PROGRAMS
WITHIN THE DEPARTMENT OF DEFENSE**

THESIS

Presented to the Faculty

Department of Cost Analysis

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Master of Science in Cost Analysis

Elizabeth Hodson

Civilian, USAF

March 2016

DISTRIBUTION STATEMENT A.
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

AFIT-ENV-MS-16-M-158

**EARNED VALUE REPORTING ON AGILE SOFTWARE DEVELOPMENT
PROGRAMS WITHIN THE DEPARTMENT OF DEFENSE**

Elizabeth Hodson

Civilian, USAF

Committee Membership:

R. David Fass, PhD, Lt Col, USAF (Ret.)
Chair

Brandon Lucas, Lt Col, USAF, PhD
Member

Dr. Edward White
Member

Abstract

The correct measurement of earned value on agile software development programs has recently been identified as a concern by software developers implementing agile development methods. This research attempts to address the difficulties in reporting earned value on agile development programs, and the differences between the measures of completion in agile development versus earned value reporting. The major difference was identified to be in translating between the work originally planned and the work that was “recently” planned as part of the agile development process. This difference, if not transparent, leads to different measurements of earned value and inconsistent reports of a program’s progress.

Acknowledgments

Thank you to my family, friends, and colleagues who have provided countless hours supporting this thesis. Support came in many forms – pulling data, proofreading, listening, advising, and even consoling – for all of which I am very appreciative.

~ Elizabeth Hodson

Table of Contents

	Page
Abstract	iv
List of Figures	viii
List of Tables	x
I. Introduction	1
General Issue	1
Agile Background.....	3
The Agile Process.....	4
Earned Value Management Background.....	6
Earned Value Management Process	7
Problem Statement and Research Questions	10
Methodology.....	11
Assumptions/Limitations.....	12
Review	13
II. Literature Review	14
Chapter Overview.....	14
Relevant Research	14
Review	19
III. Methodology	21
Introduction	21
Data Source	21
Measures Chosen.....	22
Measures Compared	24
Data Limitations	26
Chapter Summary.....	27
IV. Analysis and Results.....	28
Chapter Overview.....	28
Part I: Applied Examples to <i>AgileEVM</i>	29
Part II: Earned Value Measurements - Data Selection.....	40
Changes to the Budget at Completion	43
Percent Completed Each Month.....	47
Schedule Variances	51
Cost Variances.....	56
Summary.....	60

V. Conclusions and Recommendations	62
Review	62
Conclusions of Research	63
Recommendations for Future Research.....	65
Summary.....	66
Appendix.....	68
JMP [®] Output for Percent Change to the BAC.....	68
JMP [®] Output for Percent Complete Each Month.....	70
JMP [®] Output for Percent Completed Each Month by Quartile Period of Time	72
JMP [®] Output for Percent Schedule Variances (SV).....	79
JMP [®] Output for Percent Schedule Variance (SV) by Quartile Period of Time.....	81
JMP [®] Output for Percent Cost Variances (CV)	90
Bibliography	92

List of Figures

	Page
Figure 1: Waterfall versus Agile Process.....	5
Figure 2: Depiction of Earned Value Management	7
Figure 3: Collaboration within CADE.....	22
Figure 4: Months under Contract by Development Method	43
Figure 5: Cumulative Growth in BAC over Time	44
Figure 6: Growth in BAC during 0-25% of Contract Period.....	45
Figure 7: Growth in BAC by 50% into Contract Period.....	46
Figure 8: Average Percent of Work Completed Each Month.....	48
Figure 9: Percent of Work Completed by Development Method Over Time.....	49
Figure 10: Mean Schedule Variance Percentage over Contract Period.....	51
Figure 11: Agile SV Percentages Reported in First Ten CPR Submissions.....	54
Figure 12: Waterfall SV Percentages Reported in First Ten CPR Submissions.....	55
Figure 13: Mean Cost Variance Percentage over Contract Period	56
Figure 14: Agile CV Percentages Reported in First Ten CPR Submissions	58
Figure 15: Waterfall CV Percentages Reported in First Ten CPR Submissions	59

Figures within Appendix

Figure A-1: Percent Change in BAC - ANOVA	68
Figure A-2: Percent Change in BAC - Test for Constant Variance.....	68
Figure A-3: Percent Change in BAC - Test for Normality	69
Figure A-4: Percent Change in BAC - Non-Parametric Test	69
Figure A-5: Percent Complete Each Month - ANOVA.....	70
Figure A-6: Percent Completed Each Month - Test for Constant Variance	70
Figure A-7: Percent Completed Each Month - Test for Normality	71
Figure A-8: Percent Completed Each Month – Kruskal-Wallis Non-Parametric Test.....	71
Figure A-9: Percent Completed Each Month - Non-Parametric Test.....	71
Figure A-10: Percent Complete During 0-25% Quartile - ANOVA	72

Figure A-11: Percent Complete During 0-25% Quartile - Test for Constant Variance.....	72
Figure A-12: Percent Complete During 0-25% Quartile - Test for Normality.....	73
Figure A-13: Percent Complete During 0-25% Quartile - Non-Parametric Test	73
Figure A-14: Percent Complete During 25-50% Quartile - ANOVA	74
Figure A-15: Percent Complete During 25-50% Quartile - Test for Constant Variance.....	74
Figure A-16: Percent Complete During 25-50% Quartile - Test for Normality.....	75
Figure A-17: Percent Complete During 25-50% Quartile - Non-Parametric Test	75
Figure A-18: Percent Complete During 50-75% Quartile – ANOVA.....	76
Figure A-19: Percent Complete During 50-75% Quartile - Test for Constant Variance.....	76
Figure A-20: Percent Complete During 50-75% Quartile - Test for Normality.....	77
Figure A-21: Percent Complete During 50-75% Quartile - Non-Parametric Test	77
Figure A-22: Percent Complete During 75-100% Quartile – ANOVA.....	77
Figure A-23: Percent Complete During 75-100% Quartile - Test for Constant Variance.....	78
Figure A-24: Percent Complete During 75-100% Quartile - Test for Normality.....	78
Figure A-25: Percent Complete During 75-100% Quartile - Non-Parametric Test	79
Figure A-26: SV Percentages – ANOVA	79
Figure A-27: SV Percentages - Test for Constant Variance	80
Figure A-28: SV Percentages - Test for Normality	80
Figure A-29: Percent Schedule Variances - Non-Parametric Test	81
Figure A-30: SV Percentages during 0-25% Quartile - ANOVA.....	81
Figure A-31: SV Percentages during 0-25% Quartile - Test for Constant Variance.....	82
Figure A-32: SV Percentages during 0-25% Quartile - Test for Normality	82
Figure A-33: SV Percentages during 0-25% Quartile – Non-Parametric Test.....	83
Figure A-34: SV Percentages during 25-50% Quartile - ANOVA.....	83
Figure A-35: SV Percentages during 25-50% Quartile - Test for Constant Variance.....	84
Figure A-36: SV Percentages during 25-50% Quartile - Test for Normality	84
Figure A-37: SV Percentages during 25-50% Quartile – Non-Parametric Test.....	85
Figure A-38: SV Percentages during 50-75% Quartile – ANOVA	85
Figure A-39: SV Percentages during 50-75% Quartile - Test for Constant Variance.....	86
Figure A-40: SV Percentages during 50-75% Quartile - Test for Normality	86
Figure A-41: SV Percentages during 50-75% Quartile – Non-Parametric Test.....	87
Figure A-42: SV Percentages during 75-100% Quartile - ANOVA.....	87
Figure A-43: SV Percentages during 75-100% Quartile - Test for Constant Variance.....	88
Figure A-44: SV Percentages during 75-100% Quartile - Test for Normality	88
Figure A-45: SV Percentages during 75-100% Quartile - Non-Parametric Test.....	89
Figure A-46: CV Percentages – ANOVA.....	90
Figure A-47: CV Percentages - Test for Constant Variance.....	90
Figure A-48: CV Percentages - Test for Normality.....	91
Figure A-49: CV Percentages - Non-Parametric Test	91

List of Tables

	Page
Table 1: The 12 Principles of the Agile Manifesto.....	4
Table 2: Common Terms used in Agile Development	6
Table 3: Earned Value Definitions	9
Table 4: Identified Methods for Recognition of Earned Value	17
Table 5: Summary of Formulas to Calculate a Release Date	31
Table 6: Results of EVM_{Sprint} and $EVM_{Program}$ Examples Summarized.....	35
Table 7: <i>AgileEVM</i> vs Traditional EVM	39
Table 8: Data Pool Extracted from CADE	40
Table 9: Final Data Selection by Service.....	41
Table 10: Final Data Selection Details	42
Table 11: Summary Statistics of Months under Contract by Development Method.....	42
Table 12: Cumulative Growth in BAC over Time.....	43
Table 13: Growth in BAC during 0-25% of Contract Period	44
Table 14: Growth in BAC during 25-50% of Contract Period	45
Table 15: Percent of CPR submissions that included a BAC Adjustment	47
Table 16: Percent Completed Each Month (BCWP/BAC) Nonparametric Comparisons All Pairs Using Steel-Dwass Method.....	49
Table 17: Summary of JMP [®] Results for Percent Completed Each Month by Quartile	51
Table 18: Percent Schedule Variance Nonparametric Comparisons All Pairs Using Steel-Dwass Method	52
Table 19: Nonparametric Comparisons by Quartile for Percent Schedule Variance	53

Table 20: Summary Statistics on SV Percentage during First Two Quartiles	54
Table 21: Summary Statistics on CV Percentage during First Two Quartiles	57

EARNED VALUE REPORTING ON AGILE SOFTWARE DEVELOPMENT PROGRAMS WITHIN THE DEPARTMENT OF DEFENSE

I. Introduction

General Issue

The Department of Defense (DoD) operates in an environment that requires agility. The ability to react quickly is a necessary skill when responding to natural disasters, conducting war operations, or when participating in international affairs. It is therefore important for programs that support these quick-tempo operations to apply this *agility mindset* on new acquisitions and development endeavors. The DoD, in support of this idea, released updates to instruction 5000.02 suggesting alternative models for structuring acquisition programs that may require an expedited schedule due to “operational urgency and risk factors”(2015:2).

Implementing an agility mindset is not an easy task, especially for an organization as large and complex as the DoD. While it is not applicable in every program or situation, Information Technology (IT) programs in particular can benefit immensely from application of this agility mindset due to “the pace of technological change and need for rapid delivery of end-user capabilities” (House Armed Services Committee, 2010:17). The Office of Management and Budget (OMB) reiterated the benefits IT programs could gain from implementing modular based development processes capable of delivering functionality into increments. This report, titled “25 Point Implementation Plan to Reform Federal Information Technology Management,” stated

that many IT programs deliver too late, and often years after work has begun. In that time “...technology will change, project sponsors will change, and, most importantly, program needs will change” (OMB, 2010:11). More recently, The Under Secretary of Defense (USECDEF) for Acquisition, Technology and Logistics (AT&L), Frank Kendall III, in his *Better Buying Power 3.0*, states that the “DoD’s military products are developed and fielded on time scales that are much longer than some commercial development timelines, particularly those associated with electronics, information technology, and related technologies” (2015:9).

In recent years, the use of a similar *modular-type* approach called *Agile Development* has gained popularity in DoD software acquisition and development programs. As it has become a more common and accepted form of development, it has also begun to raise questions related to Earned Value Management (EVM) reporting practices. Earned value management is a process that allows program managers to assess the impact of program changes on cost and schedule. Earned value is a structured approach to program management that is a required practice for most cost and incentive-type contracts greater than \$20 million and includes specific guidelines for contractor compliance (DoD, 2015:71,77). Agile development is a fluid, adaptable development process that when required to report earned value is challenging to both supporters of earned value and agile developers alike (SEI, 2011:61–62). Nevertheless, implementing an agile development approach supports the DoD’s need to respond quickly to new requirements and EVM is a required and proven tool to manage cost and schedule against those very requirements.

Agile Background

In the late 1980s, program managers began looking for a lean approach to managing programs. The competitiveness of the market required companies to respond quickly to new development if they were to succeed. During this time, a lean approach to program management began to gain traction over the traditional program management style. Hirotaka Takeuchi and Ikujiro Nonaka (1986) developed an approach they named *Scrum* because they argued that the best performing projects resided where small, cross-functional teams could adapt quickly to change just like scrum formations in the game of rugby. The authors argued that the traditional, sequential approach to development was not quick enough to respond in the current market. Instead they proposed that, similar to rugby, a team that “... tries to go the distance as a unit, passing the ball back and forth – may better serve today’s competitive requirements” (p.1). The application of these authors’ rugby analogy was put into practice specifically for software projects in 1993 by Jeff Sutherland at Easel Corporation who, with Ken Schwaber, formalized the *Scrum* development process (Schwaber, 1994). *Scrum* was very receptive to requirement changes and the desire to shorten cycle times. Hence, it became a very powerful tool in managing software development.

Scrum is not the only agile method used in software development. Other agile processes such as Extreme Programming (*XP*), Dynamic Systems Development Method (*DSDM*), and *Crystal* have joined the developmental toolbox. All of these development tools share the goal of implementing a lean process to deliver quality software in a timely manner. The creation of the “*Manifesto for Agile Software Development*” in 2001 is perceived by software developers as embracing this common purpose most effectively (SEI, 2010). The *Agile Manifesto* was an agreement, by those who vowed to uphold it, to abide by twelve principles of agile development

as shown in Table 1 (Manifesto 2001). Since published, the *Agile Manifesto* is viewed by software developers an over-arching umbrella of which the various methods (*Scrum*, *XP*, *DSDM*, *etc.*) fall under, and as such are often referenced in aggregate as “agile” development (SEI, 2010:46).

Table 1: The 12 Principles of the Agile Manifesto

1	Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
2	Welcome changing requirements, even late in development. Agile processes harness change for the customer’s competitive advantage.
3	Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
4	Business people and developers must work together daily throughout the project.
5	Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
6	The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
7	Working software is the primary measure of progress.
8	Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
9	Continuous attention to technical excellence and good design enhances agility.
10	Simplicity—the art of maximizing the amount of work not done—is essential.
11	The best architectures, requirements, and designs emerge from self-organizing teams.
12	At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

(Manifesto 2001)

The Agile Process

The agile process begins with the creation of a comprehensive backlog of user *stories* (i.e. requirements) divided into multiple periods called *sprints* or *iterations*. During each sprint, software engineers design, test, and ultimately release new features to the user at the end of the

period. An undefined number of sprints and product releases will take place before the entire system is built, making the development process very receptive to requirement changes, and ensuring that the users are getting the system they really want. Figure 1 reflects a general comparison of the agile process to a traditional waterfall development cycle in which software features are developed together and sequentially¹.

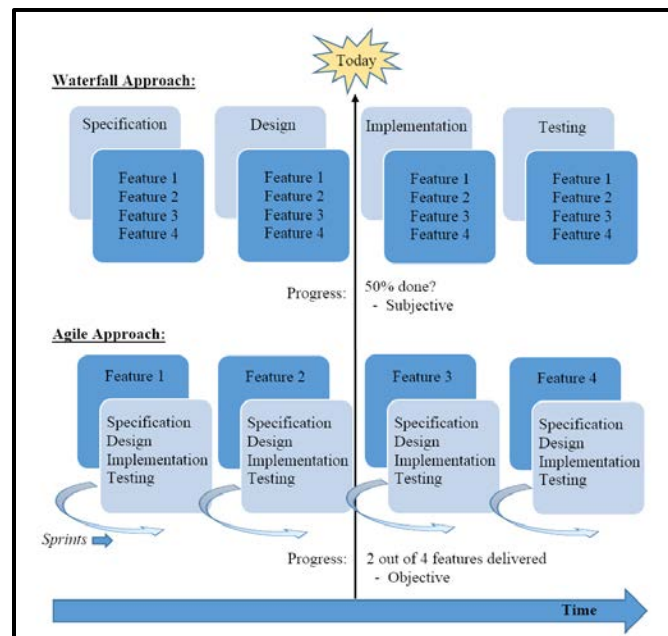


Figure 1: Waterfall versus Agile Process

In an agile software development program, progress is measured based on the number of features or user stories completed at the end of each sprint. The number of user stories that can be completed within a sprint depend upon the agile team's own assessment of the work to be completed. Over time, an agile team's average rate of completion, called *velocity*, can be

¹ Not represented in Figure 1 are incremental or spiral development processes. Similar to agile development, incremental or spiral development processes allow software to be built in phases. Agile is both an iterative and incremental process as it allows for continuous refinements to be made to pieces of software throughout development (Cohn, 2014).

measured to assist in planning for future sprints and in setting delivery dates (Sulaiman, Barton, & Blackburn, 2006). As requirements are added, they are translated into story points (or measure of effort) and added to the program’s backlog. The team’s velocity and backlog can then be used to create *burn down* charts to communicate the progression of development. These common definitions and terms used in agile development are summarized in Table 2.

Table 2: Common Terms used in Agile Development

Agile Term	Definition
User Stories	Small system function with defined success criteria that can be developed by one team within one iteration or sprint. User stories define the work that must be done to create and deliver a feature.
Story Points (SPs)	Characteristics of a user story; relative size measurement used by agile teams for estimating the amount of effort to complete the work.
Iteration	Also called a <i>Sprint</i> : Time period of fixed length during which the agile development team produces a piece completed software. Typical length is between 2-4 weeks.
Feature	Business functions or attributes of a software product or system. A single feature is typically implemented through many stories. Features provide the basis for organizing stories.
Release	The lowest level of planning in the Integrated Master Schedule (IMS) – it is a schedule for releasing software into productive use, made up of features and user stories.
Velocity	Also called <i>Cadence</i> : measures the amount of progress accomplished by the team during a sprint, often measured in story points. Used to measure how long it will take a particular team to deliver future outcomes by extrapolating on the basis of prior performance -will vary among teams.
Backlog	A prioritized list of user stories the agile team will work on at some point in the future. A <i>product</i> backlog may contain Must Haves, Should Haves, Could Haves, and even Nice to Haves. A <i>sprint</i> backlog will contain those user stories that are Must Haves for the current sprint or iteration.
Burn Up / Burn Down Charts	Representation of the number of user stories completed or left to be completed by displaying the rate of progress over a period of time.

(GAO, 2015b)

Earned Value Management Background

While the agile mindset was just beginning to take hold in the 1980s, the DoD had already spent decades trying to find the right tool to manage its large, complex development

programs. In 1967, after years of trying to devise one standard for this large intra-service organization, the DoD issued Instruction 7000.2 *Performance Measurement for Selected Acquisitions* (Abba, 2001). The procedures and guidelines within Instruction 7000.2 marked the beginning of DoD's adoption of earned value as a management process. Today the DoD has its own EVM Systems Interpretation Guide (EVMSIG) developed to facilitate the proper adoption and implementation of industry's standard "EIA 748-C *Earned Value Management System (EVMS) Standard*" (OUSD AT&L (PARCA), 2015a), and considers EVM as "one of the DoD's and industry's most powerful program planning and management tools" (DoD, 2015).

Earned Value Management Process

Earned value is a measurement of progress that communicates the amount of the budgeted work (or Budget at Completion: BAC) completed at a particular point in time. Depicted in Figure 2 (DAU, 2015), earned value is also referred to as the "budgeted cost for work performed" (BCWP).

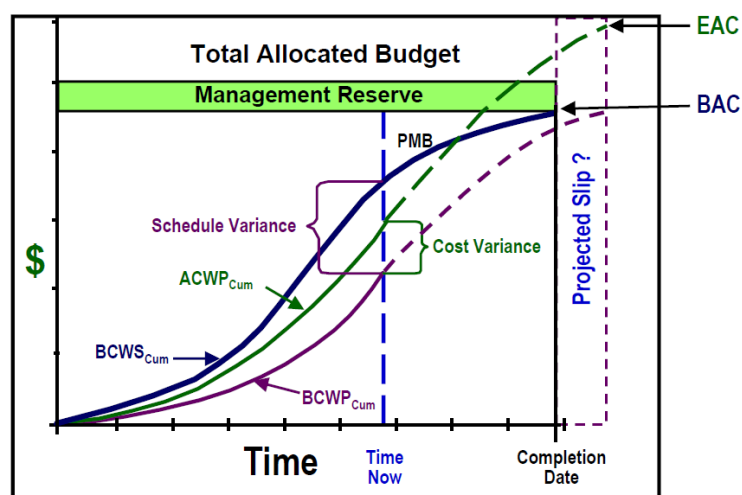


Figure 2: Depiction of Earned Value Management

When the BCWP is compared to the actual costs of work performed (ACWP), a program manager can assess the financial health of his program by relating what he thought the program would cost to what it is actually costing him. Likewise, when the BCWP is compared to the budgeted cost for work scheduled (BCWS) he can assess his program's current spending alongside his original schedule to spend it. If the value of the work completed (BCWP) is less than the work scheduled (BCWS), then his project is likely behind schedule. Using these metrics, as defined in Table 3, a program manager can ultimately ascertain her programs estimate at completion (EAC) and its relationship to her original plan (BAC).

Table 3: Earned Value Definitions

Term	Definition
Budget at Completion (BAC)	The sum of all budgets established for the contract through any given WBS/OBS level. When associated with a level it becomes control account BAC, Performance Measurement Baseline BAC, etc.
Budgeted Cost for Work Scheduled (BCWS)	Also called Planned Value (PV) – the budget for work scheduled to be accomplished within a time-period. BCWS may be expressed as a value for a specific period or cumulative to date.
Budgeted Cost for Work Performed (BCWP)	Also called Earned Value (EV) – the budget for work completed. BCWP may be expressed as a value for a specific period or cumulative to date.
Actual Cost of Work Performed (ACWP)	Also called Actual Costs (AC) – the costs actually incurred and recorded in the Earned Value Management System for accomplishing the work performed within a given accounting period. ACWP may be expressed as a value for a specific period or cumulative to date.
Cost Variance $CV = BCWP - ACWP$ $CV\% = \frac{CV}{BCWP} * 100$	A metric for showing cost performance – the difference between BCWP and ACWP. A positive value indicates a favorable condition and a negative value indicates an unfavorable condition. It may also be expressed as a percent of BCWP
Cost Performance Index $CPI = \frac{BCWP}{ACWP}$	A cost efficiency metric, similar to the CV, that depicts the amount of actual cost of work performed to the budgeted cost of work performed. A CPI greater than one indicates a favorable condition, and a value less than one indicates an unfavorable condition
Schedule Variance $SV = BCWP - BCWS$ $SV\% = \frac{SV}{BCWS} * 100$	A metric for showing schedule performance – it is the difference between BCWP and the BCWS. A positive value is a favorable condition, while a negative value is unfavorable. It may also be expressed as a percent of BCWS
Schedule Performance Index $SPI = \frac{BCWP}{BCWS}$	A schedule efficiency metric, similar to the SV, which depicts the budgeted cost of work scheduled to the budgeted cost of work performed. An SPI greater than one indicates a favorable condition and a value less than one indicates an unfavorable condition
Estimate at Completion $EAC = ACWP + ETC$	The current estimated total cost for authorized work. It equals ACWP plus the estimated costs to complete (Estimate To Complete (ETC)) the authorized work remaining. EAC does not include profit or fee.

(OUSD AT&L (PARCA), 2015a)

Problem Statement and Research Questions

Earned value management allows program managers to compare yesterday's plan with today's updates to generate tomorrow's expectations. Tomorrow's expectations include being *agile*, i.e. quick to respond to changing requirements. However, these changing requirements pose challenges for earned value metrics that rely on relatively static baselines. On the other hand, one of the premises of agile development (*as reflected in Table 1*) is that "*today's updates*" include adjustments to requirements so that functional software is delivered throughout the program. A review of relevant literature (Alleman, Henderson, & Seggelke, 2003; Ghosh, 2012; OUSD AT&L (PARCA), 2015b; Sulaiman et al., 2006) indicates these two tools can be used interchangeably - allowing earned value management to communicate progress against a plan while allowing agile to make adjustments to the plan during development. However, there is also empirical evidence (Christoph, Eric, 2015; Eagle One Solutions, 2015; SEI, 2010, 2011, 2014) to support that this interchangeability should be pursued with caution.² The objective to this research to examine the existing research and empirical data to gain insight into the difficulties of reporting earned value on DoD software programs employing agile development by addressing the following research questions.

1. Are contractors changing how they measure progress when reporting earned value on software contracts that implement agile development methods? Specifically:

² This is not suggesting the empirical evidence supports one method while dismissing the other method. In many cases those who have worked with both earned value and agile development can speak to the rewards of both approaches, but have identified in the application of both tools some lessons learned and areas in which caution is advised.

- a. Is the budgeted cost for work performed (BCWP) being incorrectly measured against an estimate at completion (EAC) rather than the budget at completion (BAC)?
 - b. If so, how would this affect the interpretation of earned value measurements reported this way?
- 2. Is there evidence from current agile development programs of inconsistent and/or atypical earned value reporting? Specifically:
 - a. Is the average budget at completion (BAC), rate of work performed (BCWP/BAC), the schedule variance percentage (SV %), and the cost variance percentage (CV %), different for contracts implementing agile development versus incremental or waterfall contracts?
 - b. Is the range and standard deviation in the budget at completion (BAC), rate of work performed (BCWP/BAC), the schedule variance percentage (SV %), and the cost variance percentage (CV %), different for contracts implementing agile development versus incremental or waterfall contracts?

Methodology

We use two methodological approaches for this research. First, we will review the calculations published by Sulaiman et al. (2006) as *AgileEVM* by applying their calculations to two hypothetical examples. Such a hypothetical framework will allow us to address the

implications of applying these authors' method to DoD programs. Second, using EVM data for DoD Electronic Systems and Software programs, we will document the differences and similarities seen between several earned value metrics on DoD contracts employing agile development methods with those that use waterfall or incremental development methods.

Assumptions/Limitations

The Office of Management and Budget (OMB) has been addressing, for the last several years, the federal government's IT spending and performance. Information Technology "includes everything from hardware and software systems to data standards to commonly agreed-upon architectural frameworks" (House Armed Services Committee, 2010:17). This relationship of hardware and software, often defined under one term "IT", results in some shared terminology and best practices identified by the OMB, the Government Accountability Office (GAO), the Project Management Institute, and the Defense Science Board to help the government effectively plan and manage its IT programs. For example, the term *modular* is identified repeatedly in reports recommending six month, incremental deliverables in support of shared service strategies (Executive Office, 2012; GAO, 2013, 2015a; OMB, 2010, 2012a, 2012b). While the meaning and usage of the words *modular and agile* as well as *incremental* and *sprints* are similar in practice, for purposes of this thesis we will focus strictly on the principles behind agile development and earned value management as they relate to only software development. Any applicability of recommendations from this thesis to that of IT programs as a whole will need further research.

Review

This chapter reviewed agile development as an approach to developing software, and EVM is a method of measuring the progress of that development. Chapter II provides a literature review on the topic of reporting earned value on agile development. Chapter III addresses the methodology used with this research, which will be analyzed in Chapter IV. Chapter V summarizes the research through discussion of the possible impacts this research has on DOD cost management and offers additional research opportunities.

II. Literature Review

Chapter Overview

The purpose of this chapter is to review literature relevant to measuring progress in an agile development setting and the implications for earned value reporting. We examine what constitutes completeness for software intensive programs and the insight provided by cost and schedule variances. Secondly, we address the volatility of requirements in an agile setting and its potential impact on earned value. Finally, we look at what we have learned from several software developers who have shared with the DoD their methods for relating earned value and agile development.

Relevant Research

While both the *Agile Manifesto* and agile development methods in general addressed the need to produce quality software quickly, neither addressed how to “forecast the future cost and schedule of the project beyond the use of ‘yesterday’s weather’ metrics” (Alleman et al., 2003). In other words, the development planned for a sprint starting this week might have transpired from yesterday’s planning meeting. While this planning momentum works for the teams developing the software, it can seem chaotic to those not familiar with the process. On the contrary, EVM’s requirements to measure against the original plan may seem futile to the agile team who has changed the plan a dozen times since it was established.

The ability to translate the progression of software development to earned value is not a new problem introduced by agile development. A DoD program is rarely one hundred percent software development. In most cases, software is only a part of a larger system and often

parceled out among multiple system components. As a result, measuring the remaining development becomes difficult. Christensen and Ferens advocated using the phases of software development as work packages to report earned value based on the criteria used to exit one phase and enter the next (Christensen and Ferens, 1995). Opponents of this method argue that a percent complete allocated at the end of each stage of development does nothing more than report a level of effort expended over time. Reporting a level of effort does not reflect the actual functionality and hence ‘real value’ invested in and delivered to the customer that perhaps agile development could (Alleman and others, 2003; Ghosh, 2012).

Consider a traditional waterfall development program in which requirements are broken down into sequential work packages and tasks. A “percent complete” at the end of the concept and requirement phases provide little insight into the complexity of work remaining (i.e. later phases of development including integration, testing, and possibly even re-work are often the most time consuming and complex). In contrast, agile development allows the prioritization of a backlog that puts the high-risk and high-value requirements at the top. The software engineers can then develop, integrate, test, and perhaps re-work the software features in the same order that the user has helped prioritize. This implies that a “percent complete” reported from an agile development program is based on actual features delivered and may be more representative of the “real value” left to deliver. It is for this reason that some argue schedule and cost variances and efficiency indices such as SPI and CPI can indicate problems more effectively and earlier on for agile programs compared to traditional waterfall development programs (Ghosh, 2012).

In summary, it appears that agile development allows for a better measurement of software completion and hence earned value. However, a potential paradox has surfaced in measuring the number of features developed as a “percent complete.” To explain, we will try

and illustrate with an example. Suppose we believe that a positive schedule variance early in a waterfall development program is less indicative of a program's completion date than a positive schedule variance in an agile development program with half of the features delivered. Then we are also confirming that we have a plan in the agile program in order to assess that we have delivered "half" of the features. Continuing with this train of thought, suppose we argue that the "half" measurement is not so much a plan as it is a *rough* estimate based upon the over-arching scope of the program and that features can still be added, removed, or moved around from one sprint to another in the true and intended spirit of agile development. If this is true, then we are supporting that having a *rough* estimate gives us better metrics than a defined plan. Hence, we have just weakened our first statement on the predictive nature of schedule variances on agile development programs. "Cost and schedule performance indices are only predictive of future results if the scope of the work is well understood..." (Christoph, Eric, 2015: 17). If we support the use of agile development and EVM on the premise that agile development will allow EVM to capture more precise measures of progress than non-agile development programs, then we also have to assume that the base to which this progress is measured is static. In other words, the plan in an agile development setting will need to be comparable to a plan in a non-agile development setting if we are to compare predictability of EVM metrics. To say that agile development supports more rigorous EVM metrics without a stable plan is a contradiction. However, to say that agile development needs a stable plan is disregarding the intention of agile development and the benefits derived from the process.

At the time of this research the Office of Performance Assessments and Root Cause Analyses (PARCA) was drafting a White Paper to document special considerations necessary when implementing earned value on agile software development programs. One of the

considerations noted was that development progress must tie to the completion of technical progress, i.e. scope (OUSD AT&L (PARCA), 2015b). In other words, reporting EVM requires a measurement against a plan to which we can measure progress. The White Paper is in part a response to several meetings between departments in the DoD³ and various software development contractors to better understand how and when the contractors are measuring “completeness” on their programs. As summarized in Table 4, there are contractors who recognize “complete” when the task or user story is 100% finished. Other contractors begin recognizing earned value the moment work commences, and yet others will wait until the product owner at demonstration accepts the work. In summary, not one particular method works for everyone. Nevertheless, regardless of the method and timing used to record earned value, a plan is required to measure it.

Table 4: Identified Methods for Recognition of Earned Value

Recognition of Earned Value	
100% Complete User Stories	• KM Systems Group - Story points are claimed “earned” at 100% completion of a user story. • allows adjustments between sprints for user stories of ‘equivalent effort’ • Lockheed Martin - Story points are claimed “earned” when the product owner accepts/approves the story at the ‘iteration demo’ • Applies a best, most likely, and worse case assumption against an average velocity to calculate the estimated effort remaining
Partially Complete User Stories	• Leidos - User Stories have interim milestones established so that partial credit can be taken before 100% completed • Raytheon - User Stories receive 50% of assigned value in story points once work is started, and the remaining 50% when completed

(KM Systems Group, 2015; Leidos, 2015; Lockheed Martin, 2015; Raytheon, 2015)

³ Hanscom’s Mission Planning Office initiated meetings with the Office of Performance Assessments and Root Cause Analyses (PARCA) after identifying several contracts using agile development needed additional guidance on reporting EVM. Since then, other offices including the Navel Center for Cost Analysis (NCCA), and the Office of the Secretary of Defense/Cost Assessment and Program Evaluation (CAPE) have had similar meetings with software contractors. Other offices may have had similar discussions to which we do not have published material or meeting minutes.

The authors Sulaiman et al. presented an approach that supported both agile development and earned value by demonstrating that the velocity calculations using story points could be interchanged with earned value measures and result in the same release dates (2006). In other words, they demonstrated that calculations for earned value and calculations in agile development can be used interchangeably. Since its publication, there have been references in literature to what Sulaiman et al. called *AgileEVM* as a way to interrelate the two tools (SEI, 2011:62, SEI 2014:20). However, the authors' recommendation came with a caveat that we must not overlook. "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" (Sulaiman et al., 2006). As we hope to illustrate with our hypothetical examples in Chapter IV, it is important when reporting earned value to define a single plan (i.e. BAC) at the beginning of the program, rather than a continually changing plan derived from the last *sprint boundary*.

One of the research questions we set out to answer was "are contractors changing how they measure progress when reporting earned value on software contracts that implement agile development methods?" Specifically we wanted to determine if the budgeted cost for work performed (BCWP) was being incorrectly measured against an estimate at completion (EAC) rather than a budget at completion (BAC). Given the inherent nature of agile development to allow for requirement changes, the "focus" for agile software developers is on the "updated" development plan or current backlog. If the work reported as "done" (i.e. BCWP) is measured against this current backlog, then the developers are measuring against an estimate to complete (ETC) rather than the BAC. This measure of work performed would then be different from the definition of BCWP and hence "changing" the how progress is measured for earned value. To some extent, we will not be able to confirm the contractors are adhering to this incorrect

application because we were unable to obtain the contractors' quantifiable backup data. This quantifiable backup data (QBD) would include the details supporting story point calculations or other measures of effort that then could be used to understand their conversion to earned value. While we do not have this supporting detail, we can discuss why such "change" to the measurement of progress is likely if the contractor reporting earned value incorrectly implements Sulaiman et al.'s (2006) *AgileEVM* calculations. The second research question we set out to answer - "is there evidence from current agile development programs of inconsistent and/or atypical earned value reporting?" The potential difference in how progress is measured could present itself in the form of inconsistent earned value metrics between agile development contracts and non-agile development contracts (or perhaps within the same agile development contract between reporting periods). Comparing means, ranges, and standard deviations of several earned value measurements between various contracts implementing different development methods, may support the inference that agile development programs are changing the way they measure progress.

Review

In conclusion, this chapter reviewed the current literature concerning the recognition of earned value on agile software development programs. Given the agile process encourages a 'faster-paced' process, and a continuous delivery of features, the rate at which programs recognize earned value is likely accelerated compared to non-agile development methods. However, this rate could be volatile and even incorrect should the measure of work performed be based upon an "updated backlog plan" rather than the BAC. Should the contractors be able to adjust the BAC to accommodate the changing requirements, we expect to see additional

volatility in the budgets of agile programs. In addition to the number of BAC changes, the timing of the changes will likely occur within the first half of the contract period when requirements are more likely to fluctuate. Finally, given any differences in the aforementioned earned value metrics, an agile program's cost and schedule variances are likely to reflect similar variation or volatility in the percentages reported. These differences will be researched using the methodology in Chapter III. The results of our data analysis are summarized in Chapter IV. Chapter V summarizes the results from Chapter IV and discussions of the impacts this research has on DOD cost management and offer additional research opportunities.

III. Methodology

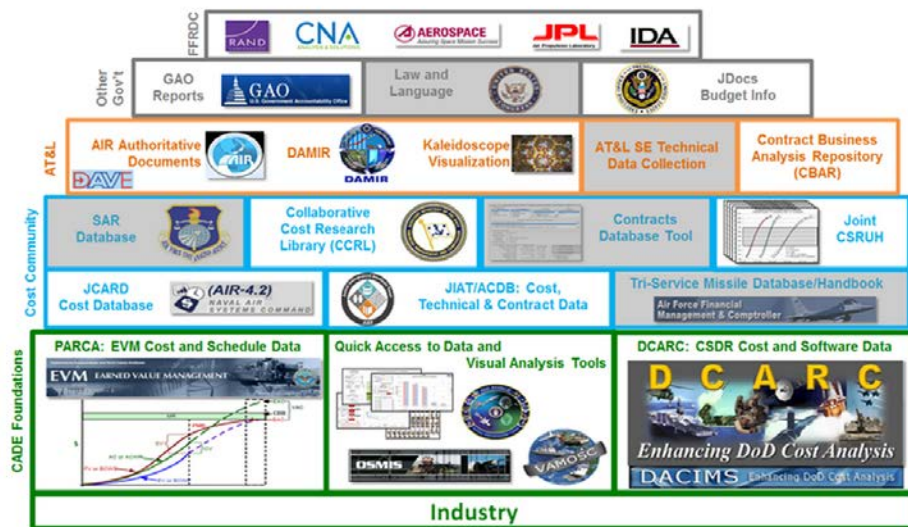
Introduction

This chapter outlines the methodology utilized for data collection and analysis. The previous chapter discussed the various approaches and expected variances when recording earned value on agile development programs. This research uses two different sets of data. The one data set will be hypothetical examples to illustrate and examine the calculations used by Sulaiman et al. (2006) and discussed in Chapter II. The second data set is from actual programs reporting into the DOD's Cost Assessment Data Enterprise (CADE). This chapter focuses on the latter data source, its selection criteria, and limitations of information. Chapter IV will cover the hypothetical data analysis as well as the results from the methodology applied to this second data set described in this chapter.

Data Source

CADE is a central repository for data collection that pulls information from various reporting systems and other data repositories across multiple agencies in an effort to ensure data is consistently reported and accessible to those who need it. Figure 3 depicts this OSD Cost Assessment and Program Evaluation (CAPE) initiative and the breadth of systems and data available. Of primary interest to this research is EVM data and software development information. The EVM central repository stores Cost Performance Reports (CPRs), Contract Funds Status Reports (CFSRs), and Integrated Master Schedules (IMSs). Likewise, the Defense Automated Cost Information Management System (DACIMS) houses Contractor Cost Data Reports (CCDRs), Software Resources Data Reports (SRDRs) and other related documents.

While each of these reports can be downloaded and analyzed individually within each repository, CADE has included several data analysis tools that pulls the information from these repositories into a file programmed in Visual Basic for Applications (VBA) through Microsoft Excel[®]. These visual analysis tools are used to download the information necessary to support this research.



(OSD CAPE, n.d.)

Figure 3: Collaboration within CADE

Measures Chosen

For this research, several data points are obtained from the CPRs within CADE. We use the contract start date, contract completion date, BAC, and the cumulative BCWP, ACWP, and BCWS to calculate the following measures:

- ***Percent of Time into Contract*** is calculated using the contract start date and the contract completion date as identified in the CPR. If there is an obvious error or omission of a contract date, the date is corrected to be the date most recently reported. For example, if a contract completion date changes in one period to six years out, and in the next several periods is reflected as the original date, the first date is assumed to be an input error and corrected before analyzing.
- ***Quartile Dummy Variables*** - Given our data set is limited in number of completed programs and they are in various stages of development, the percentage of time into the contract is blocked into 25% quarterly dummy variables. For example, the period from zero to 25% into the contract period is blocked as one quarter, 25-50% a second quarter, and so forth.
- ***Percent Change in BAC*** is defined as the change in the BAC from one reporting period to the next and is also ‘quartered’ by the period of time into the contract as discussed above.
- ***Rate at which work is performed (Monthly BCWP/BAC)*** – the monthly budgeted cost of work performed occasionally needed adjusted to account for a gap in reports. This was calculated using the difference in BCWP recorded since the last submission divided by the number of months between submissions. For example, if a program did not report for three months in a row, then the current monthly BCWP was divided by three.

Measures Compared

All analyses on the measures chosen were first reviewed in Microsoft Excel[®] for any patterns or appearance of statistical differences in the data. We used JMP[®] to validate any statistical significance in these variations and trends identified. Specifically, we used the following techniques:

Analysis of Variance (ANOVA):

Analysis of variance (ANOVA) can be used to determine whether there are any significant differences between mean values of the associated measure based upon the development method on the contract. Using an alpha of 0.05 and the hypotheses below, we look at the mean increase in the BAC, the mean percent of work completed (BCWP) each month, and the mean percent changes in the schedule and cost variances. If the p -value is greater than alpha we fail to reject the null and conclude the means are equal. If the p -value is less than or equal to alpha, then we reject the null and conclude at least one mean is different between the groups.

Means Comparisons

If we reject the null hypothesis from the ANOVA test concluding that at least one mean is different between the groups, we then need to determine the significance tests of all combinations of pairs within the group. We use the Tukey-Kramer test with an alpha of 0.05 to determine which of the groups have significant differences in their means. If the p -value is less than our alpha, we conclude that the means between the associated groups are significant.

Test for Normality

To check the validity of the ANOVA results as well as the Tukey-Kramer comparisons, we must test the assumption that each individual population is normally distributed. We utilize the Sharpio-Wilk (S-W) test to confirm whether this normality assumption holds. Like the ANOVA setting, we choose beforehand an alpha of 0.05 to conduct this hypothesis test. The null hypothesis for the S-W test is that the population is normally distributed and the alternative hypothesis is that the population is not normally distributed. If we reject the null hypothesis, we then turn to non-parametric techniques to continue our investigation of whether or not the populations are the same.

Test for Unequal Variances

In addition to testing the assumption of normality, we must also test the assumption that each population has the same variance (constant variance). We use the Levene test to check this assumption. If the p -value is greater than alpha (which we again select a priori as 0.05), we fail to reject the null and conclude the variances are equal. If the p -value is less than or equal to alpha we reject the null and conclude at least one variance is different. If we reject the hypothesis of equal variances but pass the assumption of normality then we turn to the Welch's ANOVA test which requires similar assumptions to hold as the usual ANOVA test, however, it relaxes the assumption of constant variance.

Non-Parametric Test

If we fail the test for normality with the Sharpio-Wilk, we run the Kruskal-Wallis as our non-parametric ANOVA test. The null hypothesis of this test is that the population medians are

the same between the groups. We reject the null hypothesis if the p -value is less than or equal to alpha, again set at 0.05, telling us that at least two populations are different. If we reject the null hypothesis then we run the Steel-Dwass multiple comparisons test multiple comparisons test which is the non-parametric version of the Tukey-Kramer.

Data Limitations

While CADE is extremely valuable in terms of providing a single point to retrieve information, there are still some gaps and inconsistencies in the data. For one, not all ACAT II and III programs are in the database, so many smaller programs using agile development are not captured. Secondly, only the primary method of development is identified in the SRDR visual analysis tool. Programs may contain a mix of development methods, but by choosing only the primary method to label the entire program could be mislabeling the results of this research. In fact, the primary method can be an “other” category, which is perhaps, is a mix of development methods which we decided to exclude from our data pool. Thirdly, because not all the programs chosen for this research are final, our grouping based on quartiles could change after this research is completed.

Another issue is the inconsistency in the naming conventions between the tools within CADE that make it difficult to compare programs. For example, a program name in SRDR tool may be abbreviated differently than the same program in the EVM central repository. In other cases the information includes a contract line item number (CLIN) or another identifier that is not directly traceable between tools unless uniquely familiar with the contract. Finally, as identified in the literature review, converting agile metrics to earned value metrics is dependent upon knowing the story points planned and completed. The number of story points planned and in

progress is not required as support for the earned value reported and so was not available for this research. These limitations are only negative in that they limit the power of the inferences that are possible from analyzing the data. We are still able to document several preliminary conclusions that can subsequently be enhanced, and benefit from, additional data when it becomes available.

Chapter Summary

The purpose of this chapter was to describe the methodology used for the CADE data in this research. It addressed the source of the data as well as the statistical processes applicable to evaluate the earned value measures. This chapter also addressed the limitations to the data collected. Chapter IV reviews the data itself and Chapter V discusses the implications and future considerations for agile development with earned value reporting.

IV. Analysis and Results

Chapter Overview

The purpose of Chapter IV is to document the results of the research and answers to questions identified in Chapter I.

1. Are contractors changing how they measure progress when reporting earned value on software contracts that implement agile development methods? Specifically:
 - a. Is the budgeted cost for work performed (BCWP) being incorrectly measured against an estimate at completion (EAC) rather than the budget at completion (BAC)?
 - b. If so, how would this affect the interpretation of earned value measurements reported this way?
2. Is there evidence from current agile development programs of inconsistent and/or atypical earned value reporting? Specifically:
 - a. Is the average budget at completion (BAC), rate of work performed (BCWP/BAC), the schedule variance percentage (SV %), and the cost variance percentage (CV %), different for contracts implementing agile development versus incremental or waterfall contracts?
 - b. Is the range and standard deviation in the budget at completion (BAC), rate of work performed (BCWP/BAC), the schedule variance percentage (SV %), and the

cost variance percentage (CV %), different for contracts implementing agile development versus incremental or waterfall contracts?

The first part of this chapter addresses the use of formulas presented in Sulaiman et al. (2006) and applying hypothetical program data. This framework will allow us to address the implications of applying these same formulas to DoD programs and the risk of measuring earned value based upon updated requirements and hence the estimate at completion. The second part of this chapter will use EVM data for DoD Electronic Systems and Software programs reported in the Cost Assessment Data Enterprise (CADE). This part of the analysis will allow us to determine if current agile development programs are reporting inconsistent and/or atypical earned value. Comparing the means, ranges, and standard deviations of several earned value measurements between various contracts implementing different development methods, will perhaps allow us to support the inference that agile development programs are changing the way they measure progress.

Part I: Applied Examples to *AgileEVM*

In order to illustrate the importance of defining a plan (i.e. BAC) at the beginning of the program when reporting earned value, we apply Sulaiman et al. (2006) *AgileEVM* calculations (summarized in Table 5) to two very simplified examples that we will henceforth refer to as EVM_{Sprint} and $EVM_{Program}$. Before we begin the exercise, we must explain several assumptions. First, we will refer to the “plan” and the BAC interchangeably, as both are meant to mirror what would be the program management baseline for that program. Second, we assume that the program consists solely of development labor and excludes any costs for overhead or hardware.

Third, we assume that actual labor costs are recorded in the same period in which the work is completed. Although these last two assumptions are not entirely realistic for DoD programs, to assume otherwise would unnecessarily complicate the analysis. Finally, we assume that any work added to the backlog is within the scope to deliver a functional piece of software (i.e. the added work is not for additional “nice-to-haves” or unnecessary elaborations of an existing feature). These assumptions allow us to emphasize the impact of changing the plan in reporting earned value. Therefore, the difference between the EVM_{Sprint} example and the $EVM_{Program}$ example will be the adjustment of the BAC. As we mentioned in Chapter II, the authors’ proposed calculations use what is planned or “true at each sprint boundary” (Sulaiman et al., 2006). We intend to demonstrate with our examples that adjusting EVM to what was true as of the last sprint, instead of what was true from the original program’s plan, will lead to different interpretations of earned value metrics.

Table 5: Summary of Formulas to Calculate a Release Date

Summary of Formulas to Calculate Release Date			
Name	Definition	Formula	Eq'n Ref
Initial Planned Release Points PRP_0	Sum of story points as a measure of work within the backlog at the beginning of the program, before any work has begun	$PRP_0 = \sum_p P_i$	(1)
Updated Planned Release Points PRP_n	Sum of points within the backlog from time zero plus any points added (PA)	$PRP_n = PRP_0 + \sum_{k=1}^n PA_k$	(2)
Release Points Completed RPC_n	The total points completed (PC) through sprint n	$RPC_n = \sum_{k=1}^n PC_k$	(3)
Velocity v_n	The sum of points completed through sprint n divided by sprint n	$v = \frac{RPC_n}{n}$	(4)
Actual Percent Complete APC_n	The ratio of points completed to points planned or the actual costs through sprint n divided by the estimate at completion (EAC)	$APC_n = \frac{RPC_n}{PRP_n}$ or $APC_n = \frac{AC_n}{EAC_n}$	(5)
Total Number of Sprints Required N	The ratio of the current sprint completed divided by actual percent complete or the actual cost to complete	$N = \frac{n}{APC_n}$ or $N = n * \frac{EAC_n}{AC_n}$	(6)
Earned Value (EV or BCWP)	The actual percent complete times the initial budget for the release (BAC)	$EV = APC * BAC$	(7)
Planned Percent Complete PPC	The current sprint n divided by the total number of planned sprints (PS)	$PPC = \frac{n}{PS}$	(8)
Planned Value (PV or BCWS)	The planned percent complete times the initial budget for the release (BAC)	$PV = PPC * BAC$	(9)
Schedule Variance (SV)	The difference between the earned value and planned value	$SV = EV - PV$	(10)

(Sulaiman et al., 2006)

Example: EVM_{Sprint}

Let us assume our program is at the end of its third sprint for a particular release of software. We have already spent \$30,000 for the development team to complete 45 story points. We originally planned the release to require 150 story points of effort over 10 sprints at a total cost of \$100,000. However, at the last sprint review meeting, we learned that the development team added an additional 15 story points to the backlog, which is expected to cost an additional \$10,000. Using the reference numbers in the fourth column of Table 5:

- (1) The initial planned release points (PRP_0) = 150
- (2) The new planned release points (PRP_n) = $150 + \sum_{k=1}^3(15)$, So that $PRP_3 = 165$
- (3) Release points completed at end of sprint three: $RPC_3 = 45$
- (4) Velocity is equal to 15 story points per sprint: $v = \frac{RPC_3(45)}{n(3)}$
- (5) Actual percent complete using both formulas (story points and EAC):
 - a. Story points: $APC_3 = \frac{RPC_3(45)}{PRP_3(165)}$, So that $APC_3 = 27.27\%$
 - b. Estimate at completion: $APC_{n3} = \frac{AC_3(\$30,000)}{EAC_3(\$110,000)}$, So that $APC_3 = 27.27\%$
- (6) Number of total sprints required using both formulas (story points and EAC):
 - a. Story points: $N = \frac{n(3)}{APC_3(27.27\%)}$, So that $N = 11$
 - b. Estimate at completion: $N = 3 * \frac{EAC_3(\$110,000)}{AC_3(\$30,000)}$, So that $N = 11$
- (7) The earned value is $EV = 27.27\% * \$110,000$, So that $EV = \$30,000$
- (8) The planned percent complete is $PPC = \frac{n(3)}{N(11)}$, So that $PPC = 27.27\%$
- (9) The planned value (or BCWS) is $PV = 27.27\% * \$110,000$, So that $PV = \$30,000$
- (10) The schedule variance is $SV = \$30,000 - \$30,000 = \$0$

The EVM calculations in this example assumed that the development work added could also be added to the budget (BAC). In other words, we assumed that the plan could be adjusted to reflect what was “true” as of last week’s sprint review meeting.

Example: $EVM_{Program}$

In this example, we assume the same scenario in EVM_{Sprint} except that we do not change the plan (BAC) to reflect the additional story points added. We recognize that the additional work does not simply “disappear.” Our intention with this example is to illustrate the different interpretations that result based upon our plan figure.

- (1) The initial planned release points (PRP_0) = 150
- (2) The new planned release points (PRP_n) = $150 + \sum_{k=1}^3(15)$, So that $PRP_3 = 165$
- (3) Release points completed at end of sprint three: $RPC_3 = 45$
- (4) Velocity is equal to 15 story points per sprint: $v = \frac{RPC_3(45)}{n(3)}$
- (5) The actual percent complete using both formulas (story points and EAC):
 - a. Story points: $APC_3 = \frac{RPC_3(45)}{PRP_0(150)}$, So that $APC_3 = 30\%$
 - b. Estimate at completion: $APC_3 = \frac{AC_3(\$30,000)}{EAC_3(\$110,000)}$, So that $APC_3 = 27.27\%$

The two equations for calculating the actual percent complete can no longer be interchanged. Since we wanted to keep the original plan static, we used PRP_0 instead of PRP_n . Therefore, the plan remains at 150 and does not take into account the additional 15 points of effort added. The EAC, however, does include the additional work. As defined in Chapter I, a program’s EAC is the actual cost of work performed (ACWP) plus an estimate to complete (ETC). A program’s ACWP and ETC can include costs for work

that are not part of the original plan (such as overtime labor) which is necessary to fulfil the contractual obligation but causes the program to go over budget. For this exercise we assumed that this added work is necessary and would be included within the EAC. The difference between the EAC and BAC would then become a variance at completion explained on the contractor performance reports.

(6) Number of total sprints required using both formulas (story points and EAC):

a. Story points: $N = \frac{n(3)}{APC_3(30\%)}$, So that $N = 10$

b. Estimate at completion: $N = 3 * \frac{EAC_3(\$110,000)}{AC_3(\$30,000)}$, So that $N = 11$

The difference between the two results (calculated in 5) for actual percent complete (27.27% and 30%) directly impact the calculation of total sprints required. By using only the planned points of 150, the 10 sprints remaining are understated because we have ignored the 15 points of effort. As a result, since we are at the end of sprint three, this formula predicts only seven sprints remaining until the software is delivered, compared to the eight sprints remaining when the additional work is added.

(7) The earned value using APC_{SP} versus APC_{EAC} (where the subscript represents the use of story points (SP) or the estimate at completion (EAC) in the equation), is now impacted. The BAC is at \$100,000, as it has not been increased for the additional \$10,000.

a. $EV_{SP} = APC_{SP} * BAC$, So that $EV_{SP} = 30\% * \$100,000 = \$30,000$

b. $EV_{EAC} = APC_{EAC} * BAC$, So that $EV_{EAC} = 27.27\% * \$100,000 = \$27,270$

(8) The number of total sprints calculated in (6) now impacts the planned percent complete.

a. $PPC_{SP} = \frac{n}{N_{SP}}$, So that $PPC_{SP} = \frac{3}{10} = 30\%$

b. $PPC_{EAC} = \frac{n}{N_{EAC}}$, So that $PPC_{EAC} = \frac{3}{11} = 27.27\%$

(9) The planned value (or BCWS) is subsequently impacted by the difference in the planned percent complete. As noted in (7), the BAC is at \$100,000 as it is not able to absorb the additional \$10,000.

a. $PV_{SP} = PPC_{SP} * BAC$, So that $PV_{SP} = 30\% * \$100,000 = \$30,000$

b. $PV_{EAC} = PPC_{EAC} * BAC$, So that $PV_{EAC} = 27.27\% * \$100,000 = \$27,270$

(10) The schedule variance is zero in either case:

a. $SV_{SP} = EV_{SP} - PV_{SP}$, so that $SV_{SP} = \$30,000 - \$30,000 = \$0$

b. $SV_{EAC} = EV_{EAC} - PV_{EAC}$ so that $SV_{EAC} = \$27,270 - \$27,270 = \$0$

To conclude, we assumed in this example that the BAC remained equal to the original plan.

While the additional work was added to the estimate at completion, it would be reported as a variance to plan rather than as the “new plan” we applied in the EVM_{Sprint} example.

Summary - Results of Examples

In both examples we started with the same story point plan and the same budget. In the EVM_{Sprint} example, we assumed that the points added adjusted the development plan (or BAC). In the $EVM_{Program}$ example, we held the development plan at the original planned number of story points and held our BAC at \$100,000 in order to illustrate the difference between an EAC and the BAC when reporting earned value. A summary of the results of our calculations are depicted in Table 6.

Table 6: Results of EVM_{Sprint} and $EVM_{Program}$ Examples Summarized

Eq'n Ref	Name	EVM_{Sprint}	$EVM_{Program}$
----------	------	----------------	-----------------

Eq'n Ref	Name	EVM_{Sprint}	$EVM_{Program}$
(1)	Original Planned Release Points	$PRP_0 = 150$ $BAC_0 = \$100,000$	$PRP_0 = 150$ $BAC_0 = \$100,000$
(2)	Revised Planned Release Points	$PRP_3 = 165$ $BAC_3 = \\$110,000$ $EAC_3 = \$110,000$	$PRP_3 = 150$ $BAC_3 = \\$100,000^4$ $EAC_3 = \$110,000$
(3)	Release Points Completed	$RPC_3 = 45$	$RPC_3 = 45$
(4)	Velocity	$v_3 = 15$	$v_3 = 15$
(5)	Actual Percent Complete $= \frac{RPC_3}{PRP_3}$	$APC_{SP} = 27.27\%$ $APC_{EAC} = 27.27\%$	$APC_{SP} = 30\%$ $APC_{EAC} = 27.27\%$
(6)	Total Sprints	$N_{SP} = 11$ $N_{EAC} = 11$	$N_{SP} = 10$ $N_{EAC} = 11$
(7)	Earned Value $= APC_3 * BAC$	$EV_{SP} = \$30,000$ $EV_{EAC} = \$30,000$	$EV_{SP} = \$30,000$ $EV_{EAC} = \\$27,270$
(8)	Planned Percent Complete	$PPC_{SP} = 27.27\%$ $PPC_{EAC} = 27.27\%$	$PPC_{SP} = 30\%$ $PPC_{EAC} = 27.27\%$
(9)	Planned Value	$PV_{SP} = \$30,000$ $PV_{EAC} = \$30,000$	$PV_{SP} = \$30,000$ $PV_{EAC} = \$27,270$
(10)	Schedule Variance	$SV_{SP} = \$0$ $SV_{EAC} = \$0$	$SV_{SP} = \$0$ $SV_{EAC} = \$0$

In order for both calculations (story point formulas and EAC formulas) to result in the same release date, as Sulaiman et al. intended, the planned work (PRP_n) has to be reflected in both the BAC and the EAC. We changed this assumption in the $EVM_{Program}$ example and as a result ended up with different earned values (EV_{EAC}) between EVM_{Sprint} and $EVM_{Program}$.

EVM_{Sprint} reported an earned value of \$30,000 and $EVM_{Program}$ reported an earned value of \$27,270. The difference is a direct result of using a BAC of \$110,000 versus a BAC of

⁴ In $EVM_{Program}$ - the additional 15 story points required are not removed from the backlog, but the budget to cover the work is added only to the EAC since it was not considered part of the original plan for EVM reporting.

\$100,000. As we mentioned earlier, Sulaiman et al. (2006) used a “plan” that was true as of the last plan meeting and hence equal to their estimate at completion. In our effort to prove that the BAC and the EAC are two distinctive calculations for reporting earned value, we coincidentally introduced another fallacy. When we altered our $EVM_{Program}$ example to reflect a static plan, we did not change the rest of the formulas that rely on the EAC. For example, the total number of sprints in step (6) wherein $N = 11 \left(3 * \frac{EAC_3(\$110,000)}{AC_3(\$30,000)} \right)$ derives the planned percent complete in step (8) of 27.27% $\left(\frac{n(3)}{N_{EAC}(11)} \right)$. The resulting planned value now includes an estimate for the additional work, which *we said we did not plan for and we excluded from our BAC*. In conclusion, in an attempt to illustrate that Sulaiman et al. had “mixed apples with oranges” in their calculations, we inadvertently compared “bananas and coconuts.” However, the intent of the exercise was to demonstrate that the earned value should not be calculated using the actual percent complete (APC) because this percentage will inherently include changes made to the backlog during development. Whereas earned value reflects work completed, it should be as a reflection against the original plan. We continued through the entire set of formulas in order to complete the comparison, but also to illustrate that a wrong earned value affects the other performance measures.

The \$10,000 difference in our example, while a minor dollar amount, was to highlight the criticality for contractors to report an earned value based upon the original plan, and allowing the EAC (and perhaps other supporting agile reports) to reflect the movement of story points or requirements. If we were to model our examples using “real-world” data, then we could extend EVM_{Sprint} and $EVM_{Program}$ through a series of subsequent sprints. It is likely that our programs would experience additional changes in their requirements. Perhaps we add more requirements,

take some away, or swap-out requirements so that some are postponed and others brought forward from a different release. As more changes occur and as time progresses, it becomes increasingly important to track the revisions in context with the original plan. It becomes critical when we consider that DoD programs are not developed independent of one another. Another program or system is most often reliant upon our software to integrate into its system. As such, programs in the DoD will continue to be held accountable to explain variances to an original budget and an original schedule. If the contractor bases its earned value data on what Alleman et al. referred to as “yesterday’s weather metrics,” (2003) or what Sulaiman et al. referred to as “true at each sprint boundary” (2006), then the contractor (as well as the DoD) loses sight of the original plan and the original schedule to which it should be reporting earned value. In turn, the contractor has provided little insight into the “real” progression of the program.

In addition to understanding the plan in order to report “real” progress, there is a fundamental concern that the terms “*plan* and *EAC*” would be used interchangeably in reference to earned value management. Since the Sulaiman et al. article was discussed at an Agile conference in 2006, *AgileEVM* has been referenced as an adaptation of the “traditional project management practice...against a baseline plan using Earned Value Management (EVM)” (Sulaiman, 2007). These “traditional EVM calculations... and traditional EVM metrics” allow program managers to “effectively re-baseline after every iteration” (Sulaiman and Smits, 2007). As we defined in Chapter I, the BCWP and the BCWS are measurements against a BAC to provide insight into the program’s performance. The guidelines outlined in the *EIA 748-C Earned Value Management System (EVMS) Standard* include best practices for when and how to manage changing budgets on a program, reflecting the understanding that plans are never static. However, plans are not so flexible that they equal the EAC, nor do we feel these plans should be

re-baselined every iteration. The estimate at completion and the budget at completion are two different measurements. When contractor performance reports (CPRS) are submitted, the earned value (BCWP) reported is expected to be in relation to the BAC. If a contractor uses the EAC (or some other updated projection that is not the BAC) to report the earned value, then they have changed the definition of earned value. In an effort to emphasize this difference, we have summarized in Table 7 the *AgileEVM* calculations compared to *traditional* EVM calculations.

Table 7: *AgileEVM* vs Traditional EVM

	<i>AgileEVM</i>	Traditional EVM	Difference
Plan	$PRP_n = PRP_0 + \sum_{k=1}^n PA_k$	BAC_0	If $PRP_0 = BAC_0$ <u>Then</u> $BAC_0 + \sum_{k=1}^n PA_k = \text{Revised Estimate}$ <u>And</u> A Revised Estimate $\neq BAC_0$
Work Completed	$RPC_n = \sum_{k=1}^n PC_k$	$BCWP_n$	$RPC_n \neq BCWP_n$ when work completed includes tasks outside of plan
Percent Complete	$APC_n = \frac{RPC_n}{PRP_n} = \frac{AC_n}{EAC_n}$	$\frac{BCWP_n}{BAC_0}$	Since A Revised Estimate $\neq BAC_0$ and $RPC_n \neq BCWP_n$: $\frac{BCWP_n}{BAC_0} \neq \frac{RPC_n}{PRP_n}$ <u>And</u> $\frac{BCWP_n}{BAC_0} \neq \frac{ACWP_n}{EAC_n}$
Earned Value	$EV = APC_n * BAC$	$BCWP$	$APC_n * BAC \neq BCWP$

Part II: Earned Value Measurements - Data Selection

The second part of our analysis uses EVM data to determine if there are inconsistent and/or atypical measures of progress reported between agile development contracts and non-agile development contracts. Data selection began by first identifying programs known to use agile software development methods. Discussions with individual program offices led to identification of two programs, both within the Electronic Systems and Software commodity. We used this commodity as our population for programs likely to include agile software development, and in querying CADE we found twenty-five programs with available EVM data. A program can have one to several contracts assigned, which in turn can have one to many tasks assigned. Contract Performance Reports (CPRs) are completed at the task level. Table 8 depicts the initial twenty-five programs identified and their associated seventy contracts with EVM data on eighty-six tasks.

Table 8: Data Pool Extracted from CADE

Electronic Systems and Software Commodity by Lead Service			
	Number of Programs	Contracts with EVM Reporting	EVM Task-Level Data Available
Data Available			
Air Force	6	19	33
Army	6	11	11
DoD	1	2	0
MDA	2	3	0
Navy	10	35	42
Total	25	70	86

In order to compare earned value reporting on agile development programs to that of non-agile development programs, we obtained additional information regarding the development method (i.e. agile, waterfall, or incremental) using the software resources data reports (SRDRs) visual analysis tool for these programs and program tasks. These reports allowed us to identify five programs within the electronic systems and software commodity whose primary software development process was either waterfall or incremental. Additional programs' development methods were verified via internet searches that directed us to either contractor or government documentation explicitly identifying the development method applied. Ultimately, we were able to verify the primary software development method for nine of the twenty-five programs within the Electronic Systems and Software Commodity depicted in Table 9.

Table 9: Final Data Selection by Service

Electronic Systems and Software Commodity			
	Number of Programs	Contracts with EVM Reporting	EVM Task-Level Data Available
Development Method Identifiable			
Air Force	4	6	9
Army	3	3	3
DoD	0	0	0
MDA	0	0	0
Navy	2	5	3
Total Data Used	9	14	15
Percent of Total	36%	20%	17%

These nine programs include 14 contracts with 15 EVM task-level data available.

Table 10 is a summary of these same tasks by development method and includes the ranges of EAC, time-periods, and percent complete further describing the data set being analyzed.

Table 10: Final Data Selection Details

Summary by Development Method by Service for Data Used in Analysis				
	EVM Task-Level Data Available	Sum of Latest EAC (\$mil)	Range of Reporting Periods Captured	Range of Percent Complete (BCWP/BAC)
Agile				
Air Force	3	\$ 39.26	01/31/2011 - 10/19/2015	80%-100%
Army	1	\$ 946.32	11/24/2010 - 10/27/2015	76%
Navy	3	\$ 781.99	4/15/2011 - 10/21/2015	58%-100%
Incremental				
Air Force	3	\$ 1,819.13	10/26/2006 - 10/02/2015	92%-99%
Waterfall				
Air Force	3	\$ 1,131.92	12/18/2007 - 10/26/2015	98%-100%
Army	2	\$ 1,075.75	04/08/2008 - 10/26/2015	72%-99%

Table 11 depicts the summary statistics on the number of months under contract by development method. The average contract length for a waterfall and incremental program is 81.42 and 86.03 months respectively. The average length of the agile contracts is 45.26 months.

Table 11: Summary Statistics of Months under Contract by Development Method

Months Under Contract	Agile	Incremental	Waterfall
Mean	45.26	86.03	81.42
Standard Error	1.97	4.59	1.42
Median	33.00	111.00	73.00
Standard Deviation	29.18	44.27	27.77
Sample Variance	851.59	1,959.49	770.97
Minimum	9.00	24.00	11.00
Maximum	118.00	134.00	173.00
N	220.00	93.00	385.00

The data extracted includes 698 CPR entries, across all contracts. Figure 4 highlights the distribution of these 698 entries reflecting the larger data set is waterfall development contracts.

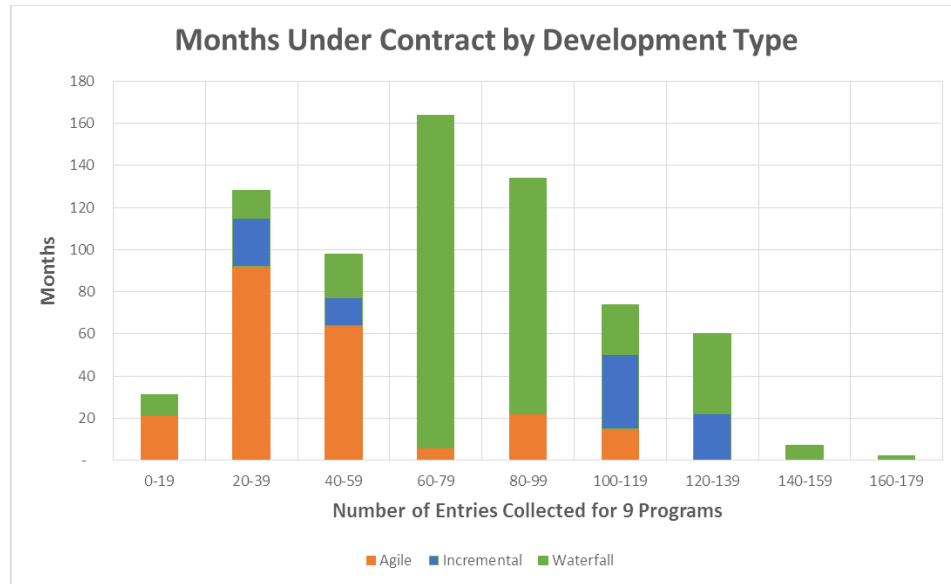


Figure 4: Months under Contract by Development Method

Changes to the Budget at Completion

As we have discussed previously, should contractors be able to adjust their budgets to reflect requirement changes then we should see additional volatility in the budgets of agile programs. Depicted in Table 12 and Figure 5, the contracts identified as agile saw an average growth of 1.11% in the first quartile compared to only 0.28% for incremental and negative 0.13% for waterfall. By the time the agile contracts are within 50-75% of their contract dates, they had incurred an average increase in their budget of only 1.31% compared to the waterfall contract's average increase of 3.2%.

Table 12: Cumulative Growth in BAC over Time

Growth in BAC	0-25% Contract Period	25-50% Contract Period	50-75% Contract Period	75-100% Contract Period
Agile	1.11%	2.11%	1.31%	1.69%
Incremental	0.28%	0.18%	0.40%	0.82%
Waterfall	-0.13%	0.15%	3.20%	1.94%

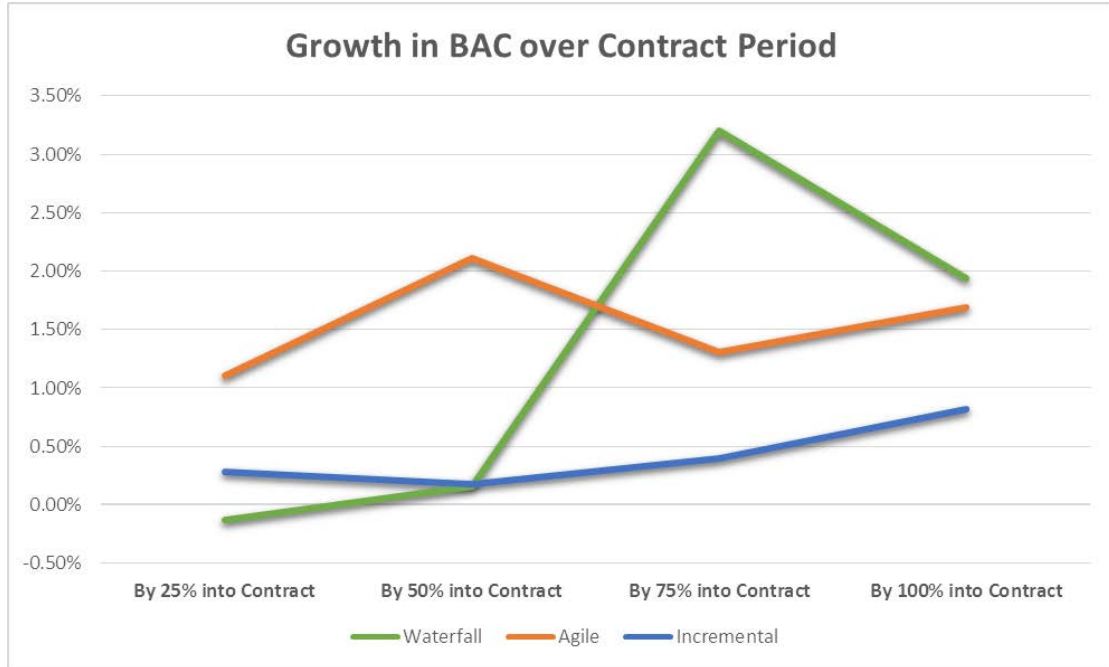


Figure 5: Cumulative Growth in BAC over Time

The growth itself does not appear to be significant but the pattern is interesting, as it appears that most of the growth in the budget is identified earlier in agile programs than in waterfall or incremental development programs. Analyzing each quartile independently, the agile programs reported a larger range of budget changes in the first 25% of the contract period than the other two development methods (Table 13 and Figure 6).

Table 13: Growth in BAC during 0-25% of Contract Period

Growth in BAC	Min	Mean	Max	Standard Deviation	Contract Tasks	Data Points
Agile	-4.57%	1.11%	21.67%	4.60%	6	28
Incremental	0.00%	0.28%	1.38%	0.55%	1	5
Waterfall	-7.43%	-0.13%	1.19%	1.50%	3	26

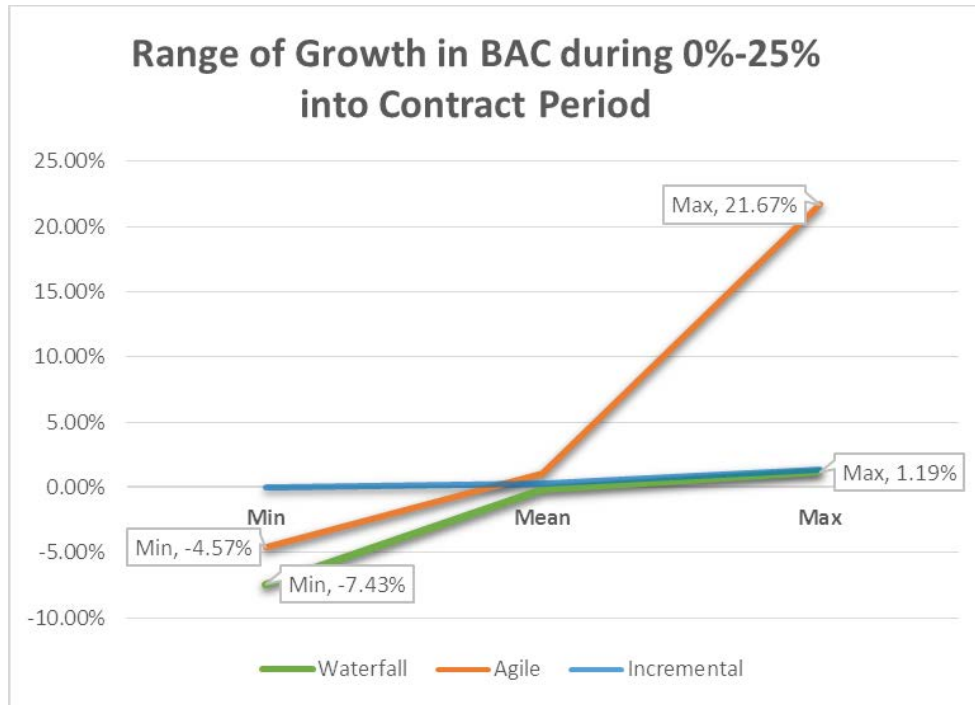


Figure 6: Growth in BAC during 0-25% of Contract Period

The range in budget changes is less extensive with incremental or waterfall development programs until later into the contract period. Between the 25-50% period, the range in BAC growth reported by agile programs is between a decrease of 1.66% and an increase of 44.67% as shown in Table 14 and Figure 7.

Table 14: Growth in BAC during 25-50% of Contract Period

Growth in BAC	Min	Mean	Max	Standard Deviation	Contract Tasks	Data Points
Agile	-1.66%	2.11%	44.67%	7.42%	6	45
Incremental	-0.02%	0.18%	0.69%	0.23%	2	12
Waterfall	-29.74%	0.15%	29.06%	6.36%	4	48

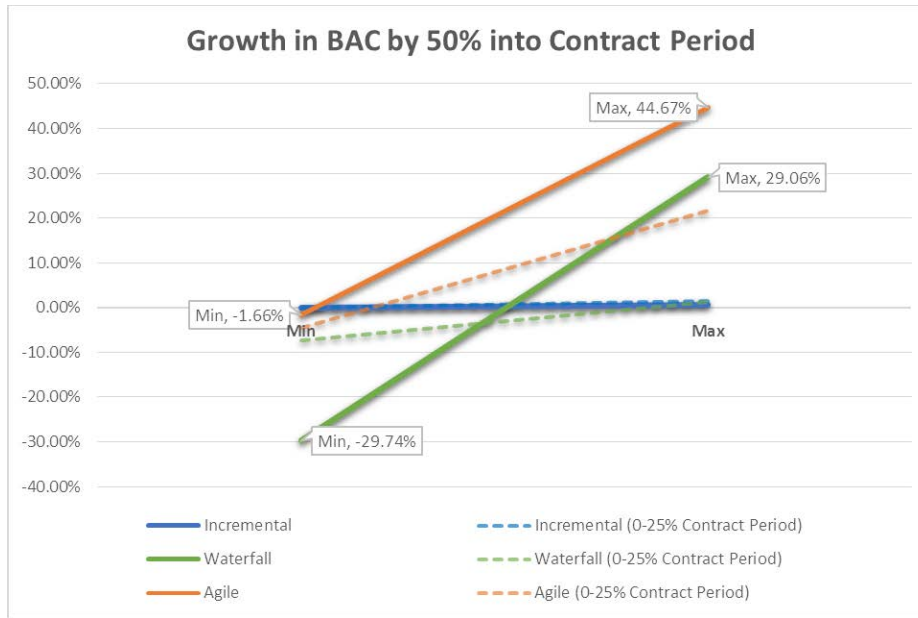


Figure 7: Growth in BAC by 50% into Contract Period

Lastly, we counted the number of times the BAC is adjusted as a percentage of the total number of CPR submissions in our data set. As shown in Table 15, the number of changes is high for all contracts regardless of the development method. The agile contracts changed their BACs roughly 70% of the time, while waterfall and incremental contracts change their BAC with almost each CPR submission. Given that agile development encourages changing requirements, we expected to see more changes with the agile contracts. However, our results suggest that a decision to adjust the BAC has little to do with the development method employed.

Table 15: Percent of CPR submissions that included a BAC Adjustment

Percent of Entries that included a change in BAC	0-25% Contract Period	25-50% Contract Period	50-75% Contract Period	75-100% Contract Period	Total
Agile	60.71%	75.56%	71.64%	67.50%	69.55%
Incremental	40.00%	91.67%	97.62%	100.00%	94.62%
Waterfall	61.54%	91.67%	97.17%	97.56%	94.29%

We initially looked to conduct an analysis of variance to determine if these measures had similar means. Using JMP[®] to run the ANOVA, our resulting p -value was 0.6482; however, our data failed the test for normality. We then ran the Kruskal-Wallis as our non-parametric test, which resulted in a p -value 0.7548. This was greater than our alpha, so we failed to reject the null hypothesis that the populations between our groups are equal. The complete output from JMP[®] is included in the Appendix in Figure A-1 through Figure A-4. Based on the trend we saw in Figure 5, we ran the same analysis on each of the quartiles. The results are similar, all with p -values of greater than 0.05. Therefore, we conclude that while the different patterns of BAC growth between different development methods is interesting, there is not statistical significance to support the differences seen based upon the development method employed.

Percent Completed Each Month

As discussed in Chapters I and II, agile development supports quick delivery of software that is of the most value to the customer. To look at the rate at which our programs record earned value, we took the monthly BCWP as a percent of the BAC. The average percent complete each month for agile contracts is 3.20%, 2.06% for incremental and 1.85% for waterfall contracts as depicted in Figure 8.

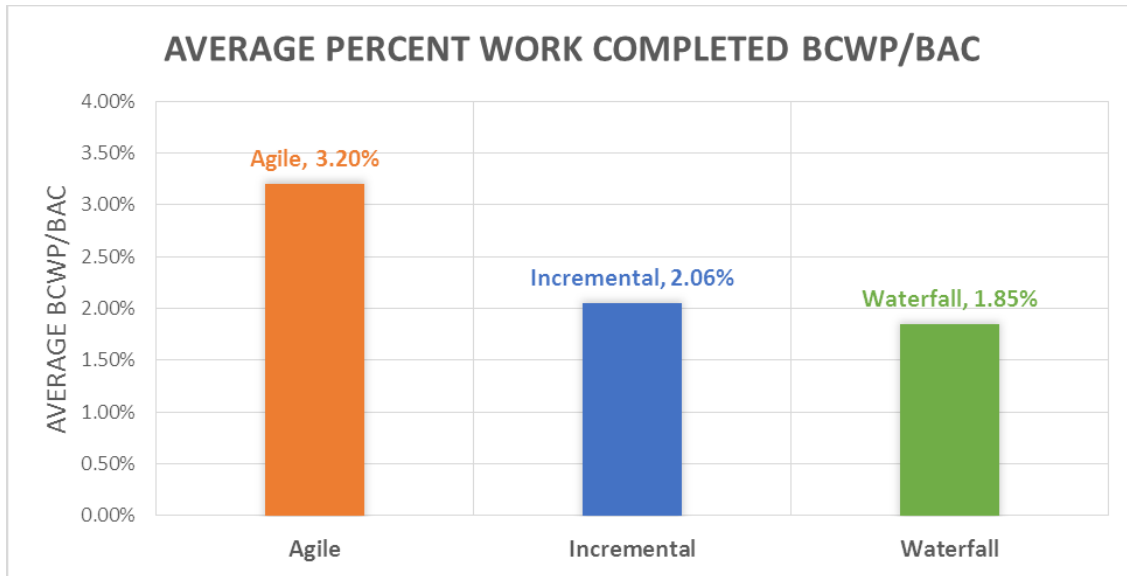
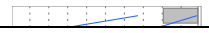


Figure 8: Average Percent of Work Completed Each Month

Initially, the ANOVA results to compare the means of these groups resulted in a p -value less than 0.0001, suggesting a statistical difference. However, we failed the Shapiro-Wilk test for normality (p -value < 0.0001) while passing the Levene test for constant variance (p -value equal to 0.2138). We conducted a Kruskal-Wallis test whose p -value was less than 0.0001, suggesting a difference between the medians somewhere. Therefore, we ran the Steel-Dwass non-parametric multiple pair test to highlight those differences. The p -values shown in Table 16 suggest that the median (since constant variance appears to be maintained) percent completed each month is different between each of the development methods.

Table 16: Percent Completed Each Month (BCWP/BAC) Nonparametric Comparisons All Pairs Using Steel-Dwass Method

Level	-Level	Score Mean Difference	Std Err Dif	Z	p-Value
Waterfall	Incremental	-38.889	15.96001	-2.4366	0.0393*
Incremental	Agile	-72.307	11.19349	-6.4597	<.0001*
Waterfall	Agile	-153.818	14.77268	-10.4123	<.0001*
Level	-Level	Hodges-Lehmann	Lower CL	Upper CL	
Waterfall	Incremental	-0.003038	-0.005307	-0.000133	
Incremental	Agile	-0.010375	-0.015428	-0.005707	
Waterfall	Agile	-0.012861	-0.016239	-0.010163	

The results support our expectations that the rate of work performed for agile contracts is higher than non-agile development contracts. The data also suggests that there is also a difference between waterfall and incremental contracts, which we had not included as part of our hypothesis. Similar to the process we used with the percent change in BAC, we reviewed the percent completed each month by quartile. As shown in Figure 9, the trend appears to be similar for agile and waterfall contracts, while the incremental contracts record more earned value at the front end of the contract period.

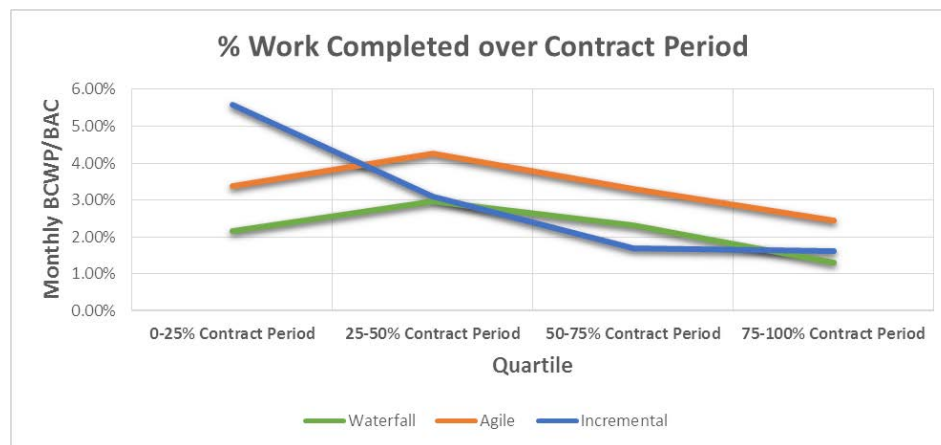


Figure 9: Percent of Work Completed by Development Method Over Time

The ANOVA results of each quartile resulted in p -values less than our alpha, suggesting a statistical difference. We proceeded to run the Shapiro-Wilk, Levene, and Kruskal-Wallis, and Steel-Dwass multiple pair tests (results are summarized in Table 17). The results suggest that the median percent completed each month is different between waterfall and agile contracts throughout the contract period. The median percent completed between incremental and agile contracts is significantly different during the last half of the contract and significantly different between waterfall and incremental programs during the last 25% of the contract period. (The full JMP[®] output by quartile is in appendix Figure A-10 through Figure A-25).

This again supports our expectation that agile development contracts would report a higher rate of work performed when compared to the other contracts. However, we were not anticipating a statistical difference in the last quartile between waterfall and incremental development contracts. Rather, we had expected that a statistical difference was more likely to be identified earlier in the contract as opposed to later. We also originally anticipated a greater volatility between periods for agile development contracts, if indeed the work performed was being measured against an “updated plan” or EAC rather than the original BAC. This however was not the case.

Table 17: Summary of JMP® Results for Percent Completed Each Month by Quartile

P-Values Reported	0-25% Contract Period	25-50% Contract Period	50-75% Contract Period	75-100% Contract Period
ANOVA	0.0011	0.0030	0.0009	<.0001
Shapiro-Wilks for Normality	<.0001	<.0001	<.0001	<.0001
Levene for Constant Variance	0.0002	0.2203	0.0368	0.7235
Kruskal-Wallis Non Parametric	0.0083	0.0010	<.0001	<.0001
Steel-Dwass for Each Pair				
Incremental – Agile	0.4651	0.1792	<.0001	0.0001
Waterfall – Incremental	0.1222	0.9612	0.4497	0.0003
Waterfall – Agile	0.0185	0.0006	<.0001	<.0001

Schedule Variances

Given the expectation of an increased rate of development and possible volatility in the measure of earned value reported, we expected the schedule variances on agile programs to mirror this volatility. Reviewing the average schedule variances by quartile, we noticed that the trend in the first half of the contract period appears to be significantly different for agile contracts (Figure 10).

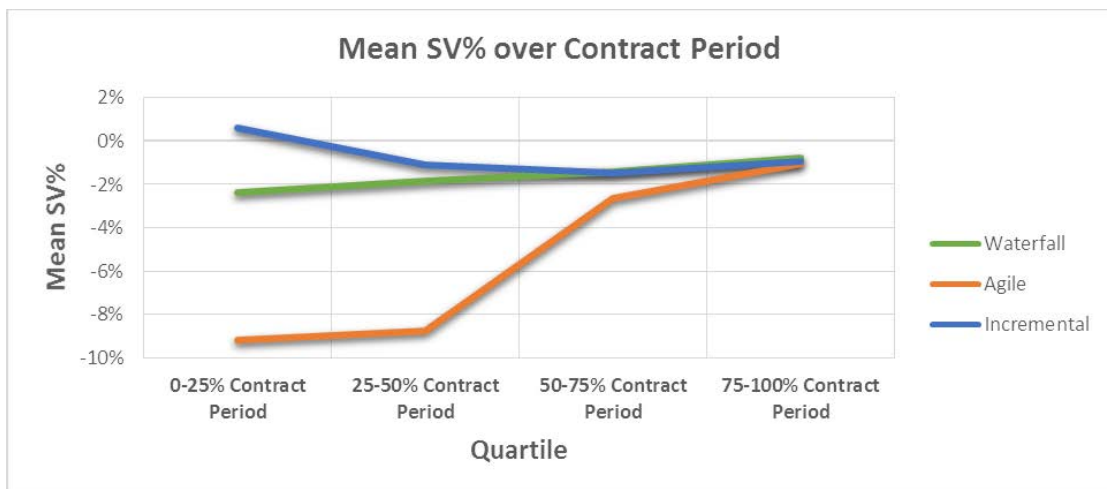


Figure 10: Mean Schedule Variance Percentage over Contract Period

Initially, the ANOVA results comparing the means of these groups resulted in a p -value less than 0.0001, suggesting a statistical difference. However, we failed the Shapiro-Wilk test for normality (p -value < 0.0001) as well as the Levene test for constant variance (p -value < 0.0001). Therefore, we have to ignore the p -value of the ANOVA. We conducted a Kruskal-Wallis test that resulted in a p -value less than 0.0001, suggesting a difference between the populations somewhere. Therefore, we ran the Steel-Dwass non-parametric multiple pair test to highlight those differences. The p -values shown in Table 18 signify that there is a difference between the waterfall and agile development methods' populations with a p -value of less than 0.0001, however we cannot speak specifically to the medians since we failed our test for constant variance.

Table 18: Percent Schedule Variance Nonparametric Comparisons All Pairs Using Steel-Dwass Method

Level	-Level	Score Mean Difference	Std Err Dif	Z	p-Value
Waterfall	Agile	73.18214	14.77268	4.953884	<.0001
Waterfall	Incremental	27.14073	15.96001	1.700546	0.2050
Incremental	Agile	20.16295	11.19349	1.801310	0.1691
Level	-Level	Hodges-Lehmann	Lower CL	Upper CL	
Waterfall	Agile	0.0051691	0.002622	0.0086104	
Waterfall	Incremental	0.0021591	-0.000847	0.0047124	
Incremental	Agile	0.0036958	-0.001036	0.0107610	

This difference in the schedule variances reported between development contracts did not directly address the trend we saw in Figure 10. Therefore, we ran the same tests for each quartile, the details of which can be seen in the Appendix (Figure A-30 through Figure A-45). Table 19 summarizes the p -values for our non-parametric tests for each quartile. While the test

in Table 18 indicated no significant differences in the schedule variances reported between incremental and agile or incremental and waterfall development contracts, when ran for each quartile individually, the results indicate a possible significance in the differences between these contracts in fourth quartile, which was not apparent from reviewing Figure 10. Overall, we can conclude that the schedule variances reported by the contracts are significantly different between development methods.

Table 19: Nonparametric Comparisons by Quartile for Percent Schedule Variance

P-Values Reported	0-25% Contract Period	25-50% Contract Period	50-75% Contract Period	75-100% Contract Period
Kruskal-Wallis Non Parametric	<.0001	<.0001	0.0166	0.3312
Steel-Dwass for Each Pair				
Incremental – Agile	0.0198	0.0066	0.8707	0.0416
Waterfall – Incremental	0.0133	0.2880	0.6993	0.0243
Waterfall – Agile	0.0004	0.0002	0.3292	0.5088

Additional Considerations in Schedule Variances

In addition to the differences already discussed, we want to highlight the range of variances reported within the first two quartiles. While we have identified that agile development contracts are reporting software variances that are significantly different in population, we have said little about the variability in the first half of the contract period. Reflected in Table 20 are the means, ranges, and standard deviations for the schedule variances reported by these contracts for each of the first two quartiles.

Table 20: Summary Statistics on SV Percentage during First Two Quartiles

SV %	Min	Mean	Max	Standard Deviation	Contract Tasks	Data Points
25-50% Contract Period						
Agile	-41.27%	-9.21%	49.32%	16.57%	6	28
Incremental	-0.41%	0.60%	1.55%	0.67%	1	5
Waterfall	-5.36%	-2.39%	1.48%	2.14%	3	26
25-50% Contract Period						
Agile	-40.11%	-8.78%	7.97%	10.00%	6	45
Incremental	-1.90%	-1.09%	-0.50%	0.44%	2	12
Waterfall	-5.18%	-1.86%	0.70%	1.46%	4	48

Further investigating the ranges seen in the first quartile, we found that the largest variances occur within the first ten CPR submissions for agile development contracts shown in Figure 11.

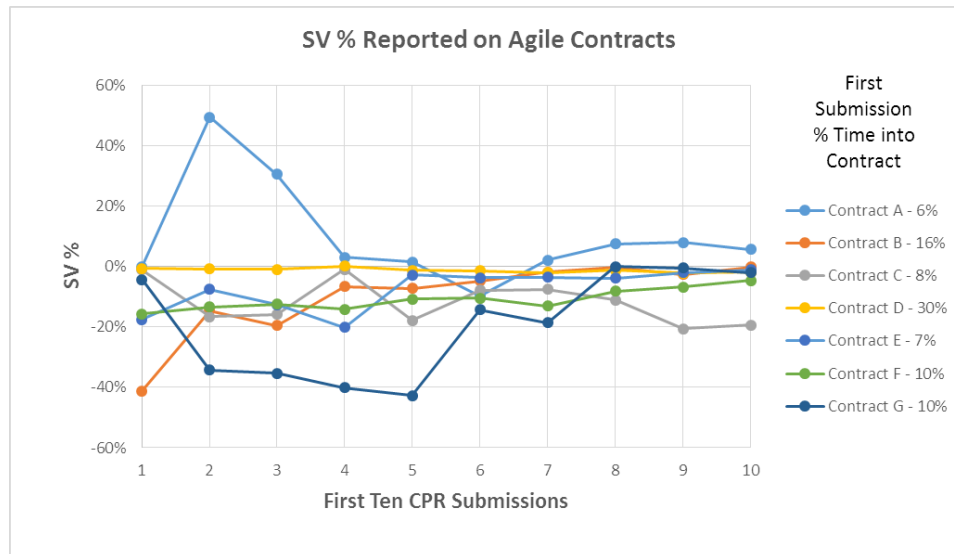


Figure 11: Agile SV Percentages Reported in First Ten CPR Submissions

These schedule variances range from greater than 40% over schedule to 40% under schedule.

Contrast this variability with Figure 12, which reflects the first ten CPR submissions from our

waterfall development contracts. The difference between the two graphs support the statistical findings that there is indeed a difference between the two populations, but it does not explain why they are so very different.

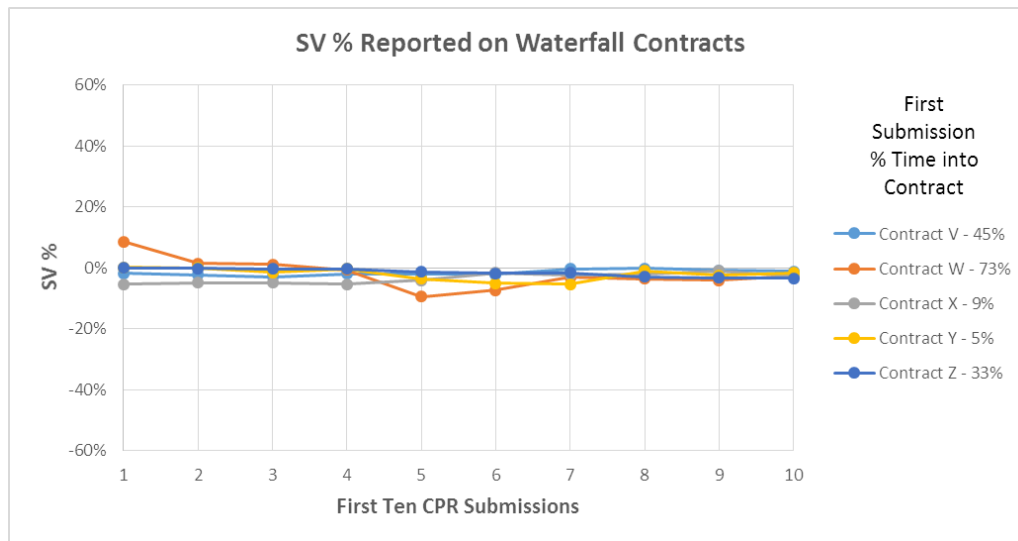


Figure 12: Waterfall SV Percentages Reported in First Ten CPR Submissions

One reason for the difference could be that perhaps agile development contractors have more timely CPR submissions, which is noted by the legends in each of the two graphs. The legend displays for each “contract” the percentage of time into the contract for which a CPR was first recorded in CADE. For example, in Figure 11, “Contract A” had a CPR submission reporting a schedule variance at 6% into the contract period while “Contract D” did not have any EVM data available until it was 30% into its’ contract period. Similarly, Figure 12 displays the five waterfall contracts reported earned value as early as 5% into the contract and as late as 73% into the contract. Another reason for these large schedule variances, so early in the agile development contract, could be that the definition of earned value is being altered because the

contractor is relating measures of work performed to a continuous changing list of requirements. We do not have enough data to support a firm conclusion one way or another, but regardless of the reason, these variances should be better understood if they are to be beneficial to the program manager.

Cost Variances

Similar to schedule variances reported, we expected that the increased rate of development and possible volatility in the measure of earned value reported, we would expect to find that cost variances reported would also be volatile. The cost variance for agile programs appears to take a much sharper decline during the second quartile (Figure 13) as compared to waterfall and incremental reported cost variances.

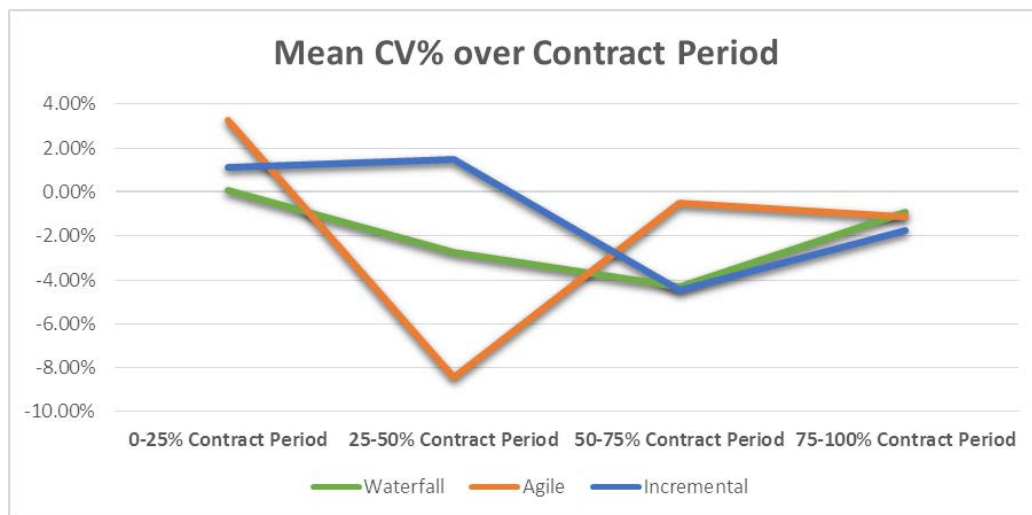


Figure 13: Mean Cost Variance Percentage over Contract Period

Our ANOVA resulted in a p -value equal to 0.8752, suggesting that we failed to reject the null hypothesis that the means were equal among the groups. However, we failed the Shapiro-Wilk

test for normality (p -value < 0.0001) as well as the Levene test for constant variance (p -value < 0.0001). Therefore, we have to ignore the p -value of the ANOVA. We conducted a Kruskal-Wallis test that resulted in a p -value equal to 0.6237 indicating that we have failed to reject the null hypothesis for equal medians. The details of the JMP® output is included in the Appendix (Figure A-46 through Figure A-49).

Additional Considerations in Cost Variances

In addition to the differences already discussed, we want to highlight as we did with the schedule variances, the range of values reported within the first two quartiles. While we have identified that we have no statistical significance between the cost variances reported on agile development contracts versus non-agile development contracts, there is still a large range of data reported that needs addressed. Reflected in Table 21 are the means, ranges, and standard deviations for the cost variances reported by these contracts for each of the first two quartiles.

Table 21: Summary Statistics on CV Percentage during First Two Quartiles

CV %	Min	Mean	Max	Standard Deviation	Contract Tasks	Data Points
25-50% Contract Period						
Agile	-34.67%	3.27%	47.68%	20.12%	6	28
Incremental	-0.04%	1.12%	1.91%	0.69%	1	5
Waterfall	-9.27%	0.08%	7.60%	4.14%	3	26
50-75% Contract Period						
Agile	-44.59%	-8.44%	25.88%	15.81%	6	45
Incremental	-1.85%	1.47%	4.66%	2.31%	2	12
Waterfall	-8.76%	-2.76%	11.06%	3.37%	4	48

Further investigating the range of variances within the first ten CPR submissions, we see that the agile development contracts in Figure 14 visually support the large variance reported in Table 21.

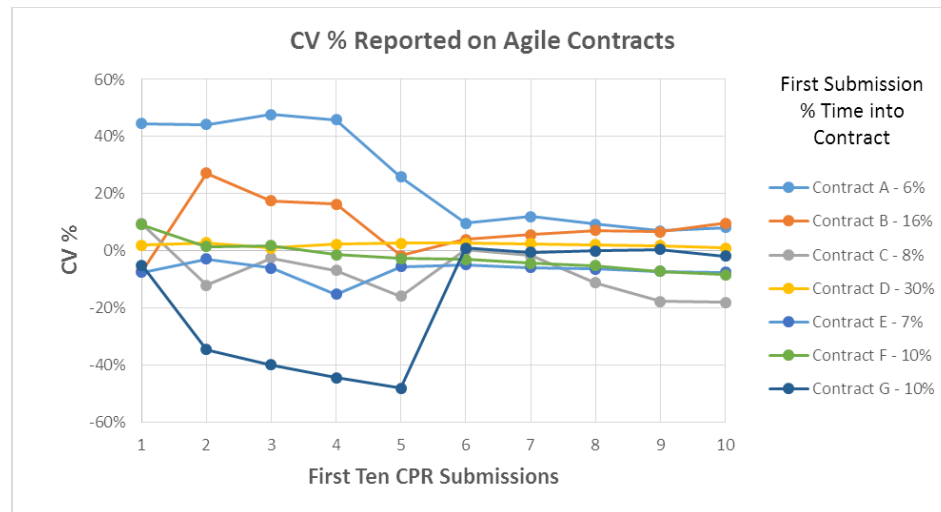


Figure 14: Agile CV Percentages Reported in First Ten CPR Submissions

As we had depicted in the legends of the schedule variances, the legend displays for each “contract” the percentage of time into the contract for which a CPR was first recorded in CADE. Contrast the first ten agile cost variances reported in Figure 14 with the first ten reported on waterfall development contracts in Figure 15.

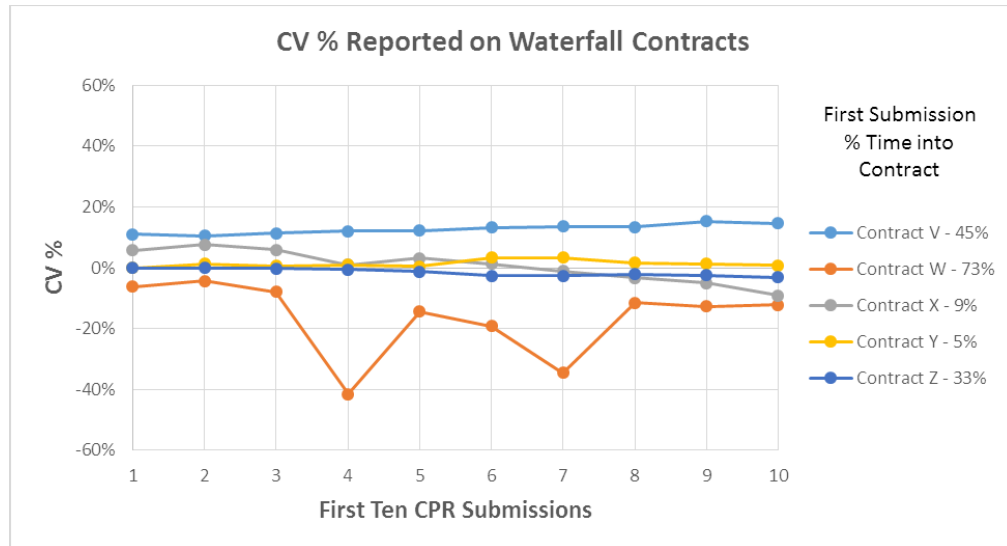


Figure 15: Waterfall CV Percentages Reported in First Ten CPR Submissions

Perhaps these differences are a result of work being performed so quickly on agile development contracts that earned value is recognized for months before the actual cost to perform the work has caught up in the accounting systems. However if this was the case then we would not see the large negative cost variances. It is therefore our presumption that the contractors implementing agile development are incorrectly measuring earned value, which is resulting in such volatile metrics such as these large cost variance percentages.

Agile development encourages requirement changes that we expected would increase the variances reported in cost and schedule as the requirements were changed or shifted between releases. However, a schedule or cost variance is only valuable if the program can use it to correct a course of action and to get back on track. Schedule variances, as those shown in Figure 11, that begin as a negative 40% and subsequently report close to 0% are concerning if they become commonplace for agile contracts. Such wide ranges of reported schedule variances call into question the validity of EVM data reported on agile development contracts. The cost

variances also show signs of extreme cost growth or extreme cost savings in the initial EVM submissions that after time merge towards zero. Given that we do not have the actual story points or quantifiable backup data from the contractors to compare with these variances, we can only speculate about the reasons for these variances over the first several reporting periods. Based on our data set, a likely reason we see less volatility with the waterfall contracts is that these programs are more established by the time EVM data is reported. The counter-argument is that waiting too long into a program to report schedule or cost variances, provides no value at all. Perhaps the recent introduction of agile development to EVM reporting is exaggerating these cost and schedule variances that are a result of measuring earned value against an updated list of requirements that is different from the original BAC.

Summary

The purpose of this chapter was to review and document the results of our data to support the inference that agile development programs are changing the way they measure progress for earned value reporting. Our analysis was conducted by first applying hypothetical examples to the calculation of release dates using earned value data, as originally proposed by Sulaiman et al. (2006) as *AgileEVM*. We concluded that in order for the earned value reported by development contractors to be meaningful and useful to the DoD program manager, the earned value must be in relation to the BAC and not to the updated requirements from a recent sprint review. The second part of our analysis was the review of EVM data for DoD Electronic Systems and Software programs identified as implementing agile development methods by comparing the means, ranges, and standard deviations of several earned value metrics. We identified differences in the rate at which work is completed, and inconsistencies in the schedule and cost

variances. Chapter V continues the discussion of the results and considerations necessary for agile development to report earned value.

V. Conclusions and Recommendations

Review

This thesis examined the impact of agile software development on earned value reporting and the possible interpretations of the work completed. We set out to answer the following questions:

1. Are contractors changing how they measure progress when reporting earned value on software contracts that implement agile development methods? Specifically:
 - a. Is the budgeted cost for work performed (BCWP) being incorrectly measured against an estimate at completion (EAC) rather than the budget at completion (BAC)?
 - b. If so, how would this affect the interpretation of earned value measurements reported this way?
2. Is there evidence from current agile development programs of inconsistent and/or atypical earned value reporting? Specifically:
 - a. Is the average budget at completion (BAC), rate of work performed (BCWP/BAC), the schedule variance percentage (SV %), and the cost variance percentage (CV %), different for contracts implementing agile development versus incremental or waterfall contracts?
 - b. Is the range and standard deviation in the budget at completion (BAC), rate of work performed (BCWP/BAC), the schedule variance percentage (SV %), and

the cost variance percentage (CV %), different for contracts implementing agile development versus incremental or waterfall contracts?

Conclusions of Research

Our answer to the first question of whether contractors were changing how they measure progress when reporting earned value was inconclusive. We demonstrated, through the use of hypothetical examples, why the EAC should not be used to measure earned value and the potential risks in portraying the EAC as a “revised plan” for purposes of recognizing work performed. If a contractor employing agile development uses the formulas found in Sulaiman et al. (2006) as an instructional translation guide between the EVM and agile story points, then that contractor risks reporting an incorrect earned value. As mentioned previously, Sulaiman et al. (2006) provided an appropriate caveat regarding the planning nature of agile programs. Therefore, we are not calling into question their calculations, but rather the potentially incorrect interpretations that can be drawn from them. While an EAC is useful for calculating revised completion dates, it is not the same as the BAC, which is used to measure the progress recorded in earned value. While these examples were illustrative of the issues that could arise, they by themselves did not confirm that contractors implementing agile development are indeed adhering to these potential fallacies.

The EVM data analyzed in Chapter IV provided evidence that the earned value reported on agile software development contracts is “different” from those same measures reported on non-agile development contracts. We found that the average rate of work completed (BCWP/BAC) was higher for agile development contracts compared to waterfall and

incremental development contracts. We were not surprised by this finding given that one of the twelve principles of the *Agile Manifesto* (Table 1) is to “deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale” (2001). In spite of this, we also found that schedule and cost variances, which are also calculated using the BCWP, resulted in inconsistent interpretations of a program’s progress. In some instances, the schedule and cost variances were so large that it appeared the program would never be completed. These large variances diminish over time leading us to believe that the nature of agile to make requirement changes during development is leading to incorrect EVM entries at the beginning of contract performance, specifically within the first quartile. While we cannot confirm this interpretation with our analysis, given the relatively recent introduction of earned value reporting on agile development contracts, it is probable that these variances are a result of an incorrect comparison of story points to earned value or even budget (BAC) to revised plan (EAC).

While we found no significance to the percent change in the BAC between development methods, the range of changes seen in the BAC for agile programs was far greater than that of non-agile programs. The BACs in our agile programs were adjusted anywhere from down 4.57% to an increase of 44.67% (Table 13 and Table 14) during the first half of the contract period. Referencing back to Figure 5, the agile and waterfall development contracts had distinct patterns to the growth recorded in their BACs. Agile programs showed growth earlier in the contract period, while waterfall programs showed growth later in the program. While this pattern could appear prevalent because of our small sample of programs, it is conceivable that because software developers, in applying this agile mindset to their development process, are able to flag necessary BAC increases (or even other issues) earlier in the development process.

However, a BAC that increases over 40% in the first half of the contract causes us to question if the contractors are changing the way they record earned value.

Recommendations for Future Research

The conclusions reached in this research are limited to theorizing that any variability or commonality between the earned value measures can be attributed solely to the development method on that contract. Future research could support our findings by obtaining the story point measures and quantifiable back up data used by the contractor to compare alongside the EVM data. A comparison between the two could then identify if our findings for agile development contracts were indeed attributable to the development method's use of story points and continual refinement of the requirements. Such future research could then support recommendations or best practices for reporting earned value on agile software development.

In addition, future research could be done to address agile development's compatibility with earned schedule. Earned schedule is an extension of EVM that focuses on schedule analysis and in units of time. This research also did not compare the earned value metrics with the integrated master schedule (IMS). We recommend any future research address the traceability between earned schedule and the IMS with the metrics of agile development. Perhaps agile development identifies more with earned schedule, and supported by the IMS the tools will then better support the premise that agile can deliver quicker than non-agile development methods.

Finally, we did not address in our research the contracting considerations and contract vehicles that best support agile software development. Continuous interaction with the customer is one of the twelve values of the *Agile Manifesto* (Table 1) and a cornerstone for agile

development to be effective. Such interaction allows the customer to modify scope, which in turn can imply contract modifications, and we do not address this complication in our research.

Summary

During the course of our research, we uncovered a couple of interesting findings that were published regarding DoD programs. One such finding was that the percent of functions performed by software in combat aircraft increased from 8% in 1960 to 80% in 2000 (DSB, 2000:11). While this was a finding reported over a decade ago, it is our presumption that the DoD's reliance on software has not only increased for combat aircraft, but across all services and commodities. Given the impact software has on the DoD's performance and future capabilities, it seems a realistic expectation that the DoD will continue to require EVM as a program management tool to help ensure that these programs are successful. The second interesting finding was from a study of major DoD programs where 37 out of the 67 programs studied had delays of more than three years, some of those as high as 15 years (Gilmore, 2011). Granted, many of these likely reported EVM and still suffered large delays, however, it is interesting to note the extent of the delays to which the DoD has grown accustomed. In conclusion it seems that the DoD has both a growing reliance on software and a multitude of programs that are unable to deliver on schedule. Perhaps agile software development could be a process to which we can support the rapid growth of software as well as support "less delayed" programs. If this is indeed the case, then reporting earned value on such agile contracts will need to improve beyond what we have displayed in the results of this research.

Earned value management is a powerful program management tool, supported by years of application, education, industry standards, and DoD directives. If the use of agile

development increases for DoD programs, then perhaps additional education and similar, consistent standards will need put into practice for agile contracts. EVM and agile development metrics are inherently different tools that “operate under different assumptions and their metrics are designed to work within those constraints” (Christoph, Eric, 2015:20). It is not the intent of this paper to suggest that *agile* needs to “change” in order to report earned value, or to suggest that the DoD change EVM requirements and guidelines. However, we do recommend that additional collaboration and education between contractors and the DoD program managers continue so that each tool can function as they were designed.

Appendix

JMP® Output for Percent Change to the BAC

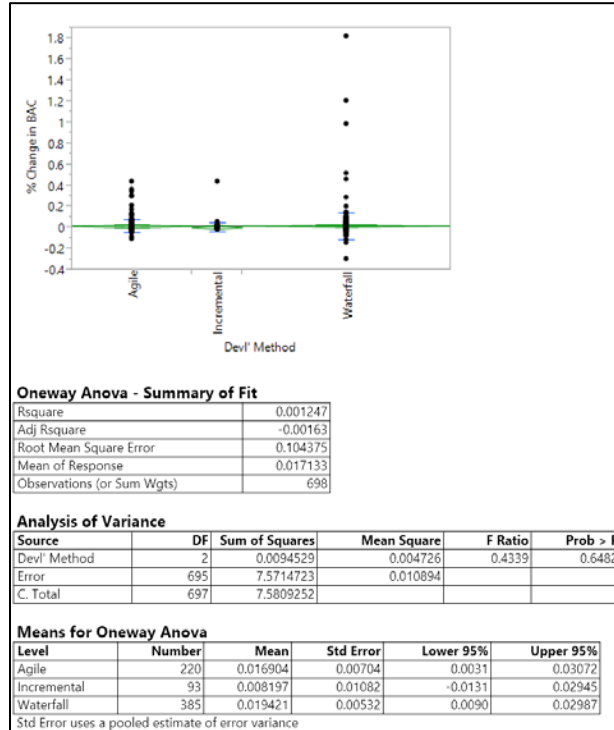


Figure A-1: Percent Change in BAC - ANOVA

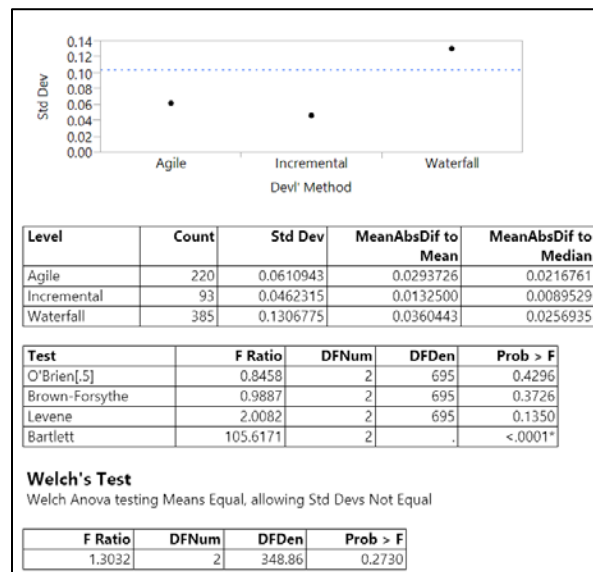


Figure A-2: Percent Change in BAC - Test for Constant Variance

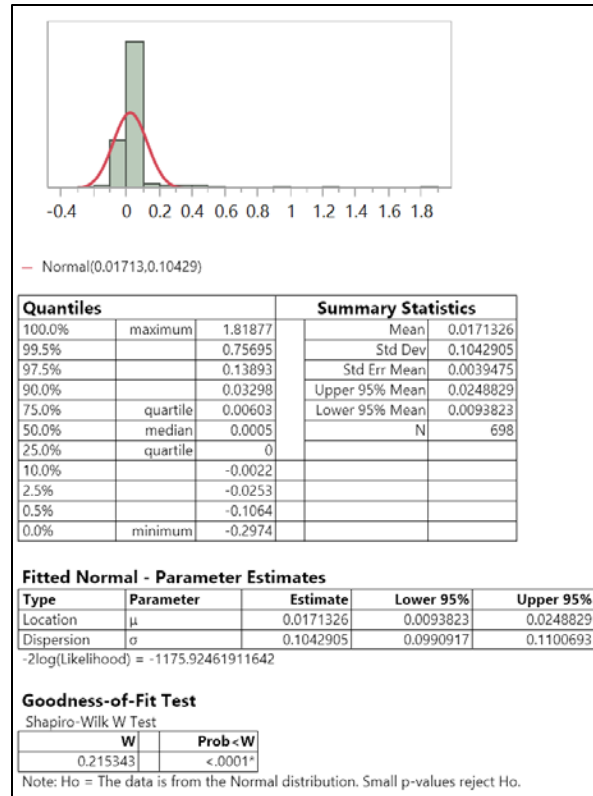


Figure A-3: Percent Change in BAC - Test for Normality

Wilcoxon / Kruskal-Wallis Tests (Rank Sums)					
Level	Count	Score Sum	Expected Score	Score Mean	(Mean-Mean0)/Std0
Agile	220	76914.5	76890.0	349.611	0.010
Incremental	93	31191.5	32503.5	335.392	-0.725
Waterfall	385	135845	134558	352.844	0.486

1-way Test, ChiSquare Approximation		
ChiSquare	DF	Prob>ChiSq
0.5626	2	0.7548

Level	- Level	Score Mean Difference	Std Err Dif	Z	p-Value
Waterfall	Incremental	13.3768	15.95857	0.838220	0.4019
Waterfall	Agile	2.0321	14.74915	0.137780	0.8904
Incremental	Agile	-4.7271	11.12518	-0.424904	0.6709

Level	- Level	Hodges-Lehmann	Lower CL	Upper CL	
Waterfall	Incremental	0.000137	-0.000181	0.0007063	
Waterfall	Agile	0.000000	-0.000253	0.0002746	
Incremental	Agile	-4.027e-7	-0.000533	0.0002066	

Figure A-4: Percent Change in BAC - Non-Parametric Test

JMP® Output for Percent Complete Each Month

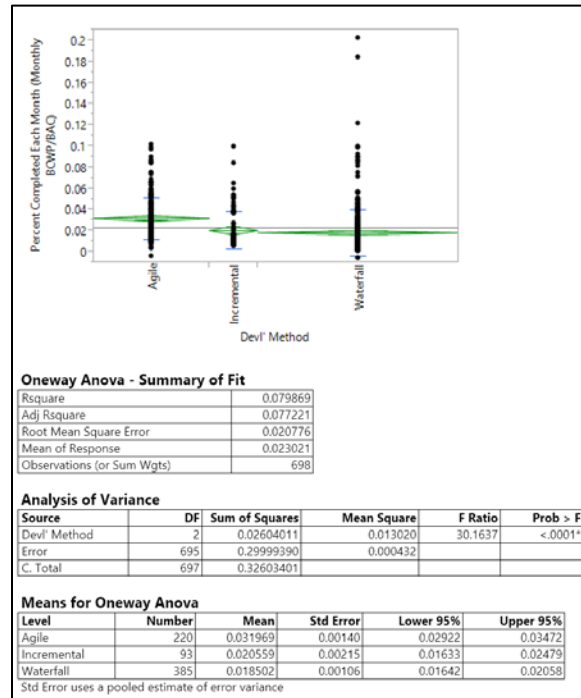


Figure A-5: Percent Complete Each Month - ANOVA

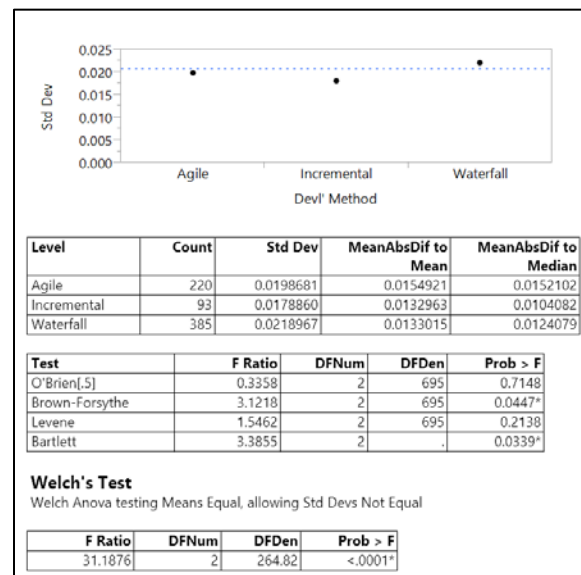


Figure A-6: Percent Completed Each Month - Test for Constant Variance

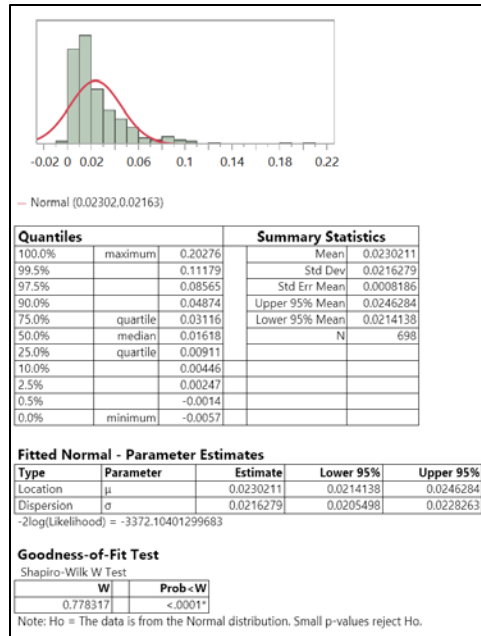


Figure A-7: Percent Completed Each Month - Test for Normality

Wilcoxon / Kruskal-Wallis Tests (Rank Sums)					
Level	Count	Score Sum	Expected Score	Score Mean	(Mean-Mean0)/Std0
Agile	220	103152	76890.0	468.873	10.611
Incremental	93	30690.0	32503.5	330.000	-1.001
Waterfall	385	110109	134558	285.997	-9.228

1-way Test, ChiSquare Approximation		
ChiSquare	DF	Prob>ChiSq
116.1596	2	<.0001*

Figure A-8: Percent Completed Each Month – Kruskal-Wallis Non-Parametric Test

Nonparametric Comparisons for All Pairs Using Steel-Dwass Method					
q*		Alpha			
2.34370		0.05			
Level	- Level	Score Mean Difference	Std Err Dif	Z	p-Value
Waterfall	Incremental	-38.889	15.96001	-2.4366	0.0393*
Incremental	Agile	-72.307	11.19349	-6.4597	<.0001*
Waterfall	Agile	-153.818	14.77268	-10.4123	<.0001*
Level	- Level	Hodges-Lehmann	Lower CL	Upper CL	
Waterfall	Incremental	-0.003038	-0.005307	-0.000133	
Incremental	Agile	-0.010375	-0.015428	-0.005707	
Waterfall	Agile	-0.012861	-0.016239	-0.010163	

Figure A-9: Percent Completed Each Month - Non-Parametric Test

JMP® Output for Percent Completed Each Month by Quartile Period of Time

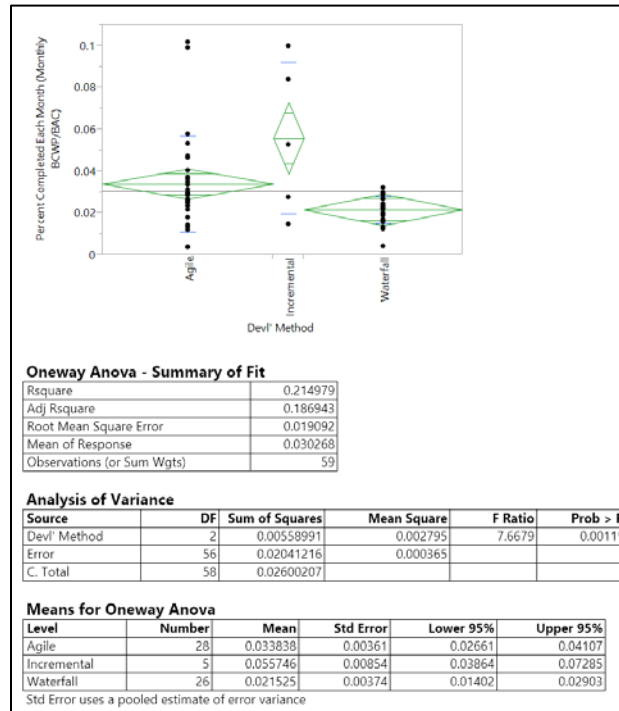


Figure A-10: Percent Complete During 0-25% Quartile - ANOVA

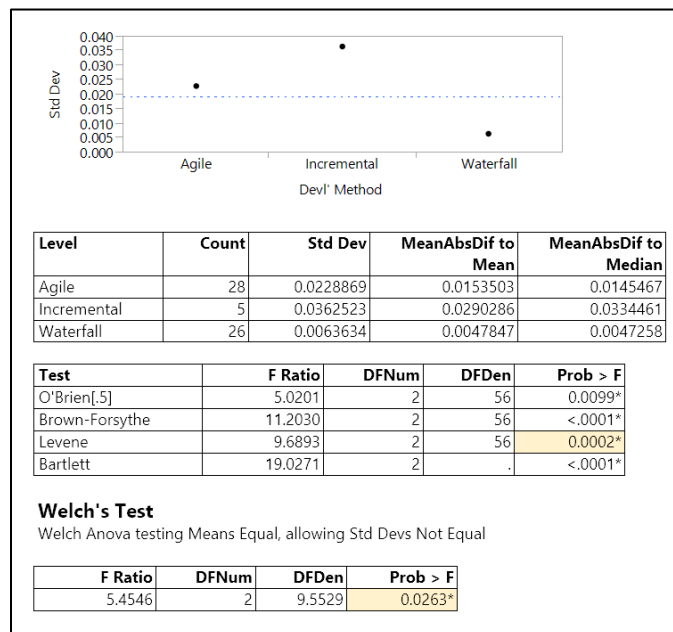


Figure A-11: Percent Complete During 0-25% Quartile - Test for Constant Variance

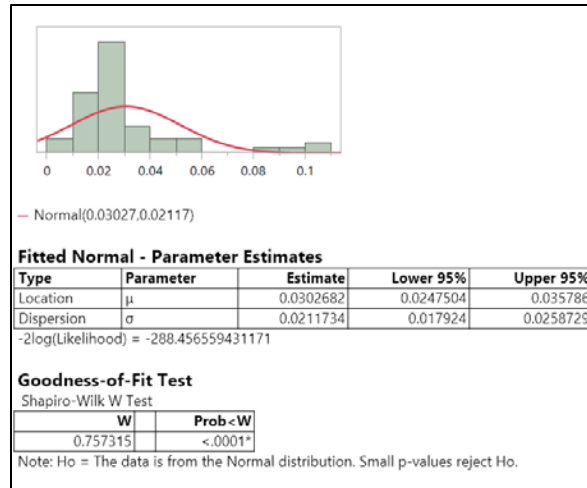


Figure A-12: Percent Complete During 0-25% Quartile - Test for Normality

Nonparametric Comparisons for All Pairs Using Steel-Dwass Method					
q*		Alpha			
2.34370		0.05			
Level	- Level	Score Mean Difference	Std Err Dif	Z	p-Value
Incremental	Agile	5.5393	4.694209	1.18003	0.4651
Waterfall	Incremental	-8.7038	4.439912	-1.96036	0.1222
Waterfall	Agile	-11.6085	4.284602	-2.70936	0.0185*
Level	- Level	Hodges-Lehmann	Lower CL	Upper CL	
Incremental	Agile	0.022064	-0.015144	0.068993	
Waterfall	Incremental	-0.030569	-0.075728	0.004023	
Waterfall	Agile	-0.007824	-0.015728	-0.001192	

Figure A-13: Percent Complete During 0-25% Quartile - Non-Parametric Test

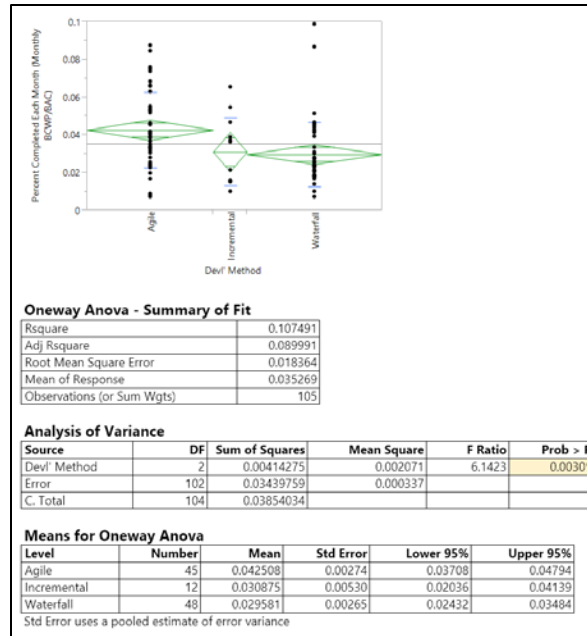


Figure A-14: Percent Complete During 25-50% Quartile - ANOVA

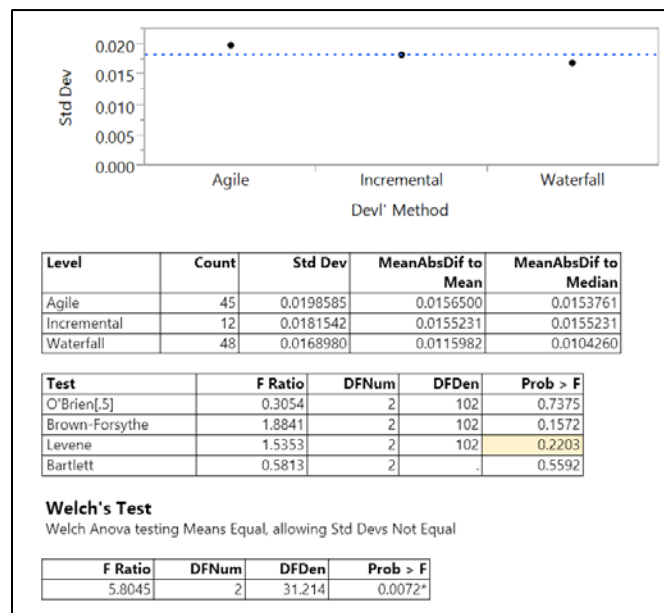


Figure A-15: Percent Complete During 25-50% Quartile - Test for Constant Variance

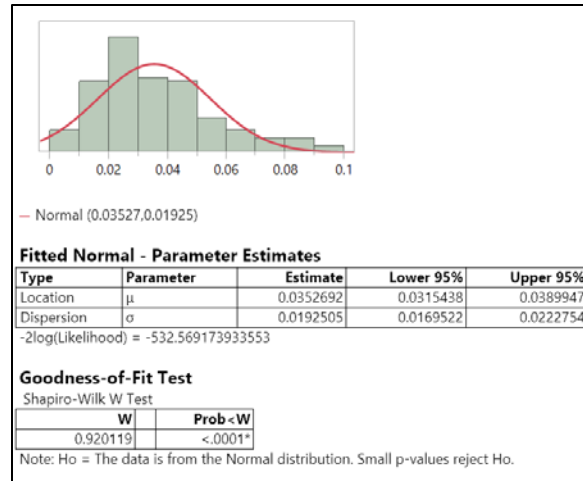


Figure A-16: Percent Complete During 25-50% Quartile - Test for Normality

Nonparametric Comparisons for All Pairs Using Steel-Dwass Method					
q*		Alpha			
2.34370		0.05			
Level	- Level	Score Mean Difference	Std Err Dif	Z	p-Value
Waterfall	Incremental	1.5104	5.636562	0.26797	0.9612
Incremental	Agile	-9.5528	5.392639	-1.77145	0.1792
Waterfall	Agile	-20.7313	5.600533	-3.70166	0.0006*
Level	- Level	Hodges-Lehmann	Lower CL	Upper CL	
Waterfall	Incremental	0.002020	-0.015997	0.009040	
Incremental	Agile	-0.012406	-0.026611	0.004887	
Waterfall	Agile	-0.013377	-0.021334	-0.005647	

Figure A-17: Percent Complete During 25-50% Quartile - Non-Parametric Test

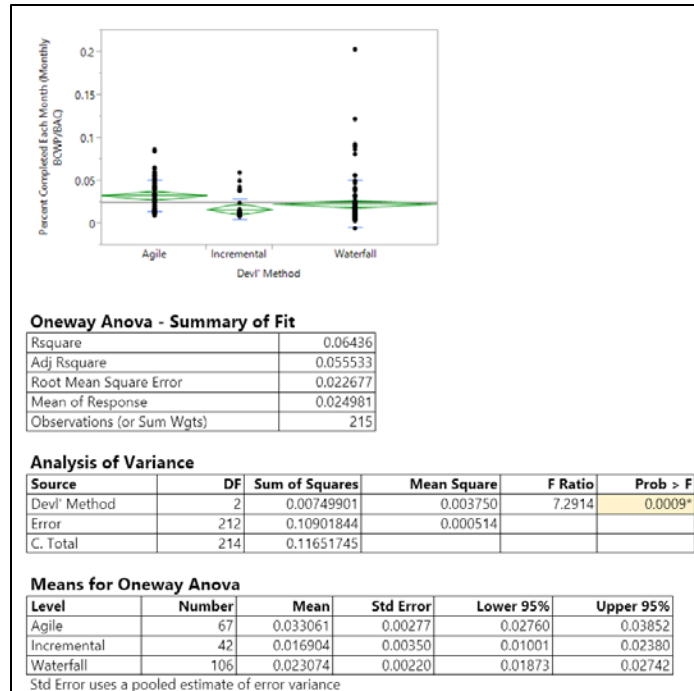


Figure A-18: Percent Complete During 50-75% Quartile – ANOVA

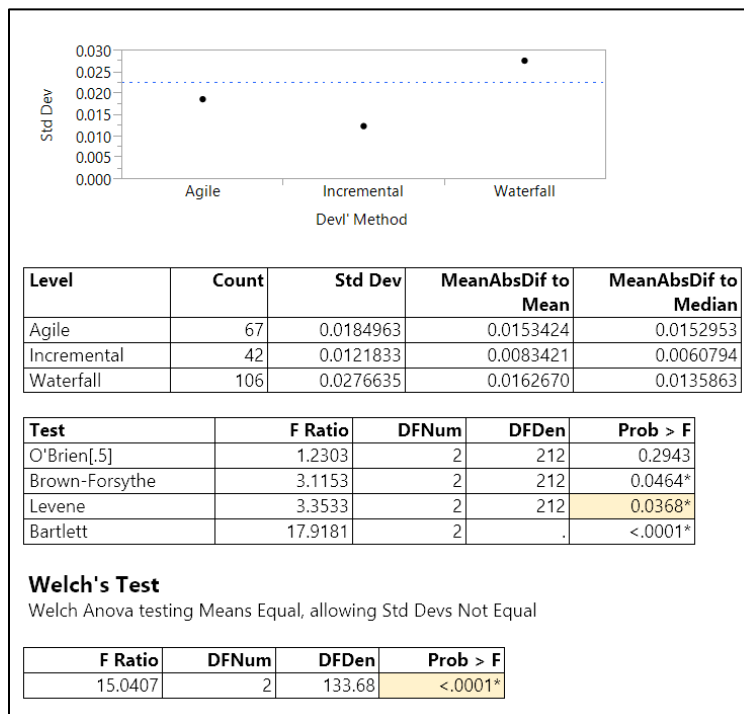


Figure A-19: Percent Complete During 50-75% Quartile - Test for Constant Variance

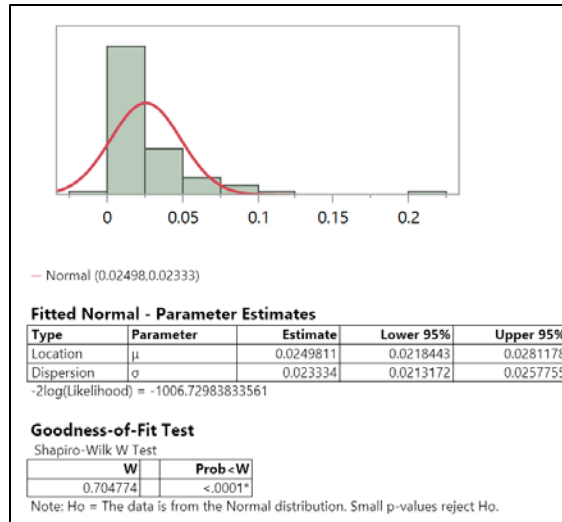


Figure A-20: Percent Complete During 50-75% Quartile - Test for Normality

Nonparametric Comparisons for All Pairs Using Steel-Dwass Method					
q*		Alpha			
2.34370		0.05			
Level	- Level	Score Mean Difference	Std Err Dif	Z	p-Value
Waterfall	Incremental	9.4245	7.816038	1.20579	0.4497
Incremental	Agile	-35.1907	6.221143	-5.65662	<.0001*
Waterfall	Agile	-39.6692	7.816994	-5.07474	<.0001*
Level	- Level	Hodges-Lehmann	Lower CL	Upper CL	
Waterfall	Incremental	0.001272	-0.001189	0.004493	
Incremental	Agile	-0.014047	-0.023237	-0.005345	
Waterfall	Agile	-0.011448	-0.019305	-0.005718	

Figure A-21: Percent Complete During 50-75% Quartile - Non-Parametric Test

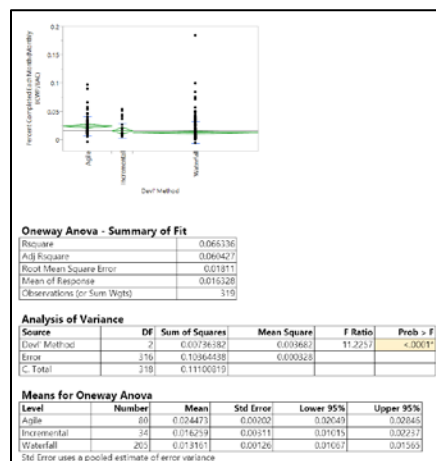


Figure A-22: Percent Complete During 75-100% Quartile - ANOVA

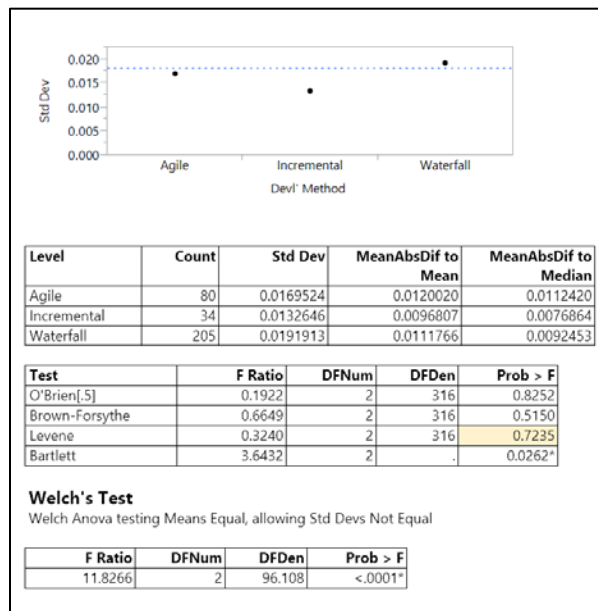


Figure A-23: Percent Complete During 75-100% Quartile - Test for Constant Variance

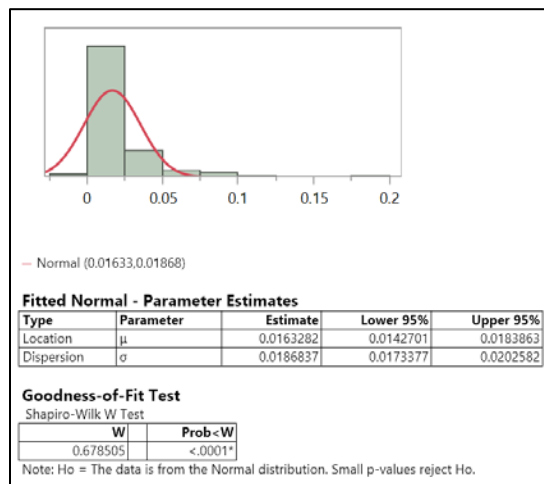


Figure A-24: Percent Complete During 75-100% Quartile - Test for Normality

Nonparametric Comparisons for All Pairs Using Steel-Dwass Method					
q*		Alpha			
2.34370		0.05			
Level	- Level	Score Mean Difference	Std Err Dif	Z	p-Value
Incremental	Agile	-27.9761	6.76673	-4.13436	0.0001*
Waterfall	Incremental	-49.6002	12.80255	-3.87424	0.0003*
Waterfall	Agile	-86.5514	10.86464	-7.96634	<.0001*
Level	- Level	Hodges-Lehmann	Lower CL	Upper CL	
Incremental	Agile	-0.006522	-0.011638	-0.003116	
Waterfall	Incremental	-0.004350	-0.006466	-0.002099	
Waterfall	Agile	-0.010837	-0.013589	-0.008576	

Figure A-25: Percent Complete During 75-100% Quartile - Non-Parametric Test

JMP® Output for Percent Schedule Variances (SV)

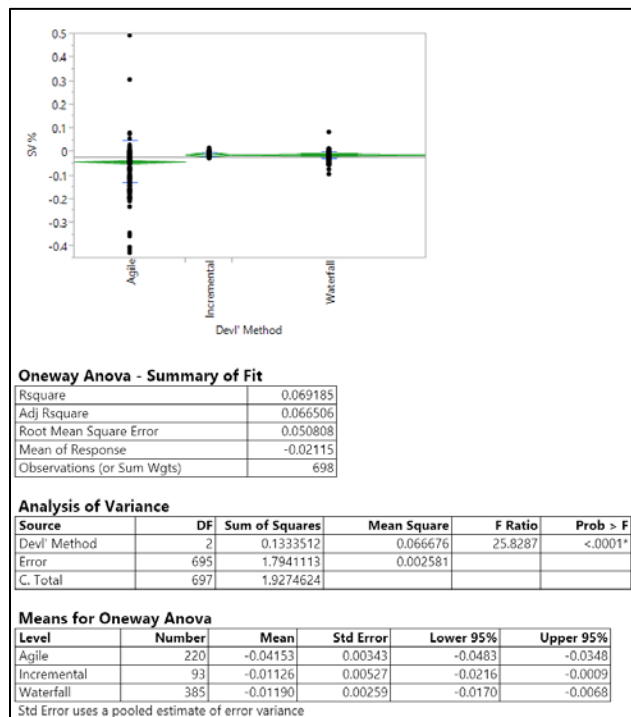


Figure A-26: SV Percentages – ANOVA

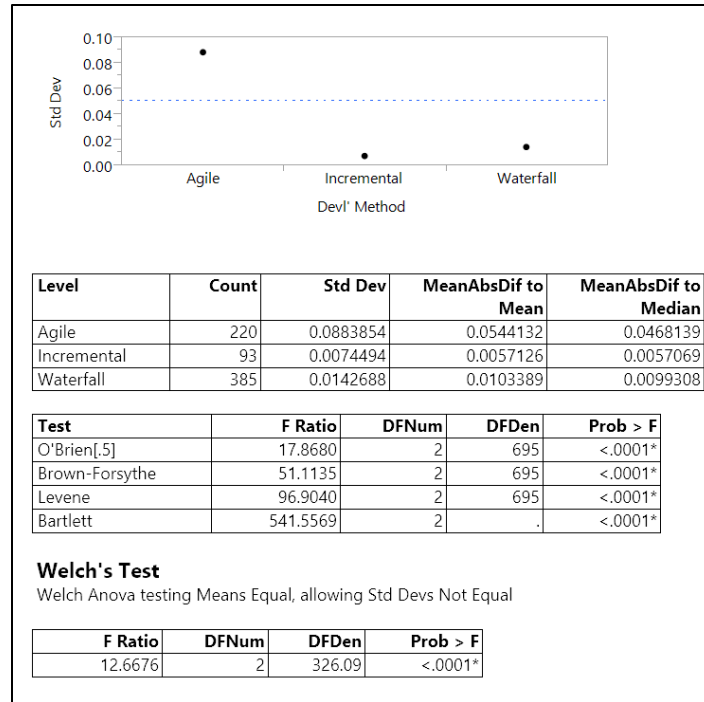


Figure A-27: SV Percentages - Test for Constant Variance

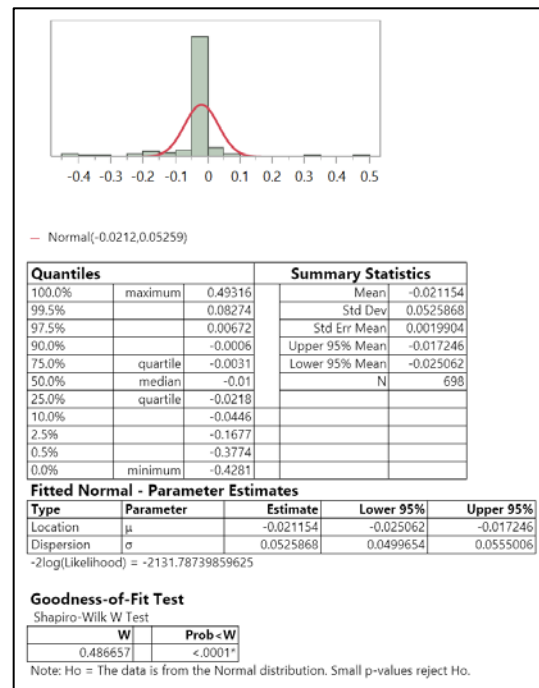


Figure A-28: SV Percentages - Test for Normality

Wilcoxon / Kruskal-Wallis Tests (Rank Sums)					
Level	Count	Score Sum	Expected Score	Score Mean	(Mean-Mean0)/Std0
Agile	220	65325.5	76890.0	296.934	-4.672
Incremental	93	31788.5	32503.5	341.812	-0.395
Waterfall	385	146837	134558	381.395	4.635

1-way Test, ChiSquare Approximation		
ChiSquare	DF	Prob>ChiSq
24.7193	2	<.0001*

Figure A-29: Percent Schedule Variances - Non-Parametric Test

JMP® Output for Percent Schedule Variance (SV) by Quartile Period of Time

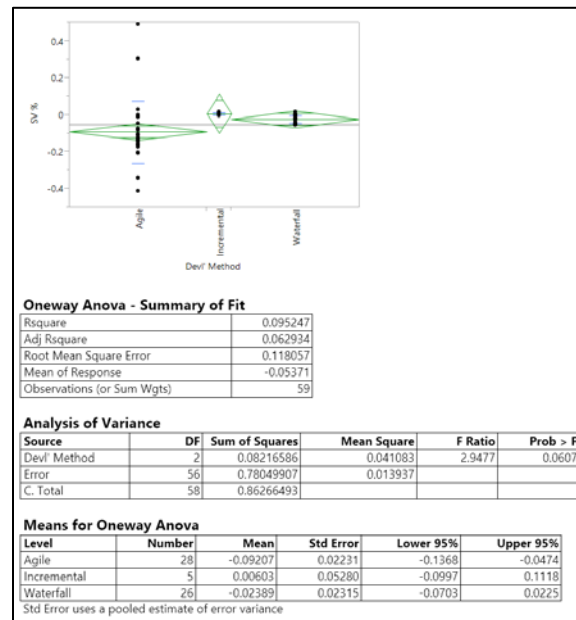


Figure A-30: SV Percentages during 0-25% Quartile - ANOVA

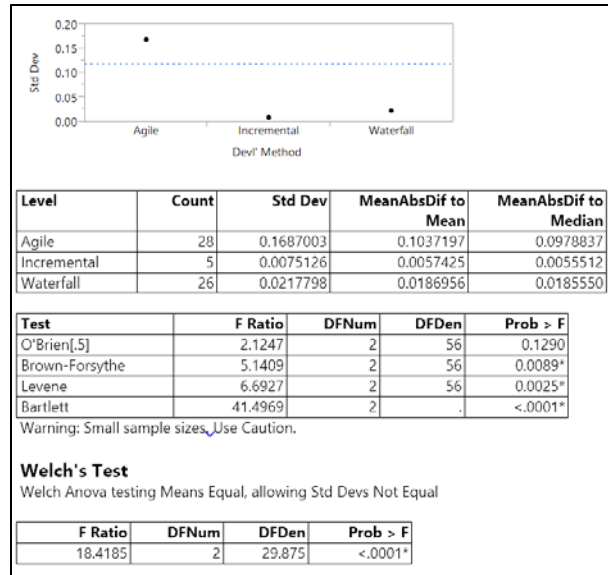


Figure A-31: SV Percentages during 0-25% Quartile - Test for Constant Variance

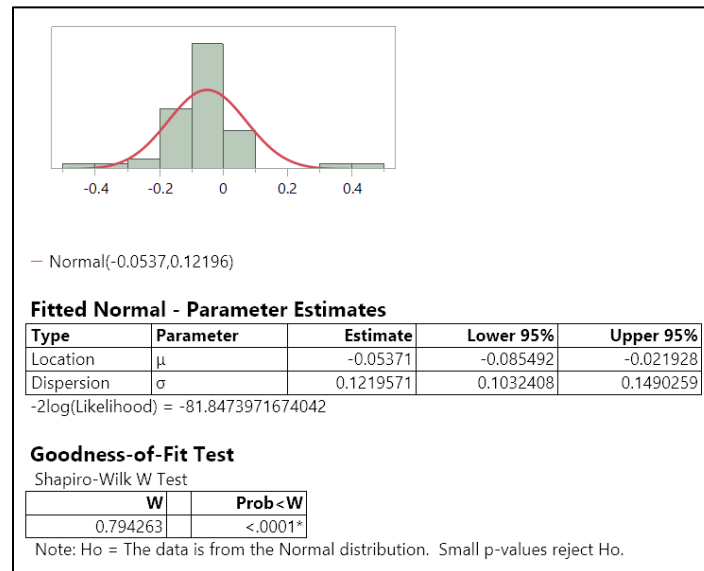


Figure A-32: SV Percentages during 0-25% Quartile - Test for Normality

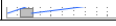
Nonparametric Comparisons for All Pairs Using Steel-Dwass Method					
q*		Alpha			
2.34370		0.05			
Level	- Level	Score Mean Difference	Std Err Dif	Z	p-Value
Waterfall	Agile	16.3558	4.284602	3.81734	0.0004*
Incremental	Agile	12.6107	4.694209	2.68644	0.0198*
Waterfall	Incremental	-12.5192	4.439912	-2.81970	0.0133*
Level	- Level	Hodges-Lehmann	Lower CL	Upper CL	
Waterfall	Agile	0.097159	0.055564	0.127812	
Incremental	Agile	0.129711	0.012910	0.186186	
Waterfall	Incremental	-0.028127	-0.057128	-0.004619	

Figure A-33: SV Percentages during 0-25% Quartile – Non-Parametric Test

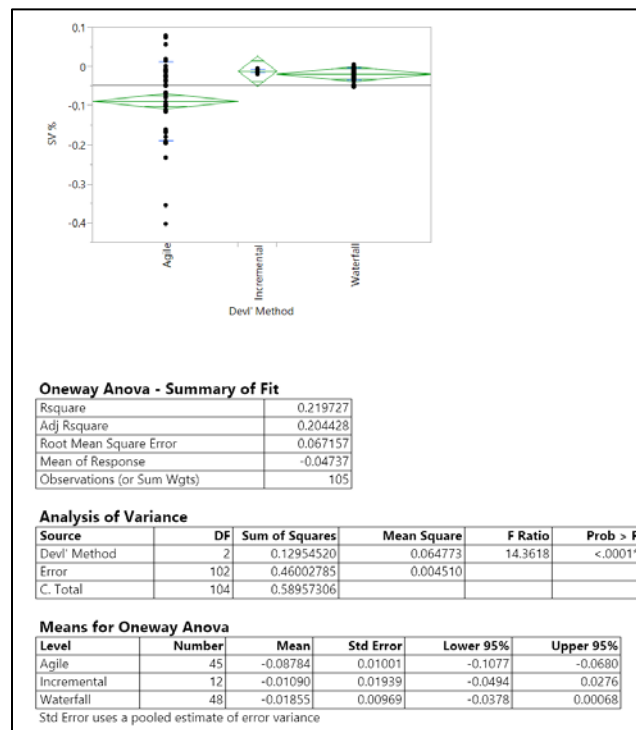


Figure A-34: SV Percentages during 25-50% Quartile - ANOVA

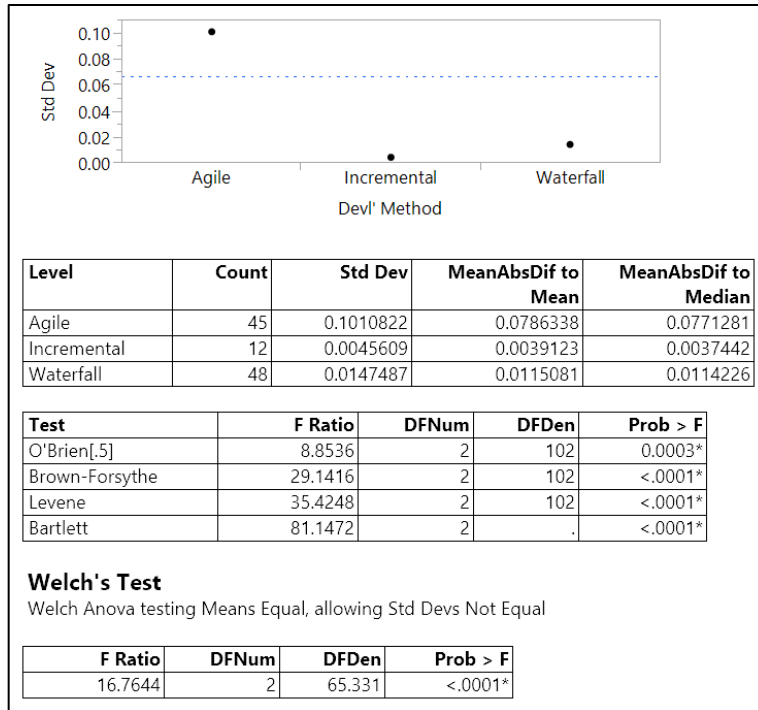


Figure A-35: SV Percentages during 25-50% Quartile - Test for Constant Variance

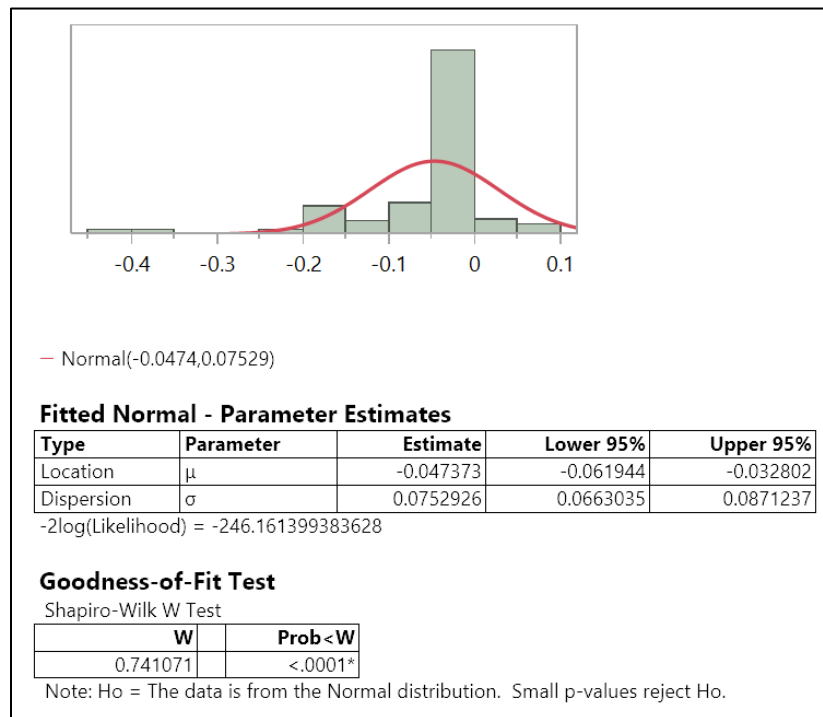


Figure A-36: SV Percentages during 25-50% Quartile - Test for Normality

Nonparametric Comparisons for All Pairs Using Steel-Dwass Method					
q*		Alpha			
2.34370		0.05			
Level	- Level	Score Mean Difference	Std Err Dif	Z	p-Value
Waterfall	Agile	22.5826	5.600533	4.03223	0.0002*
Incremental	Agile	16.4139	5.392639	3.04376	0.0066*
Waterfall	Incremental	-8.4896	5.636562	-1.50616	0.2880
Level	- Level	Hodges-Lehmann	Lower CL	Upper CL	
Waterfall	Agile	0.051394	0.016506	0.0896867	
Incremental	Agile	0.059385	0.007556	0.1481635	
Waterfall	Incremental	-0.005454	-0.014497	0.0027164	

Figure A-37: SV Percentages during 25-50% Quartile – Non-Parametric Test

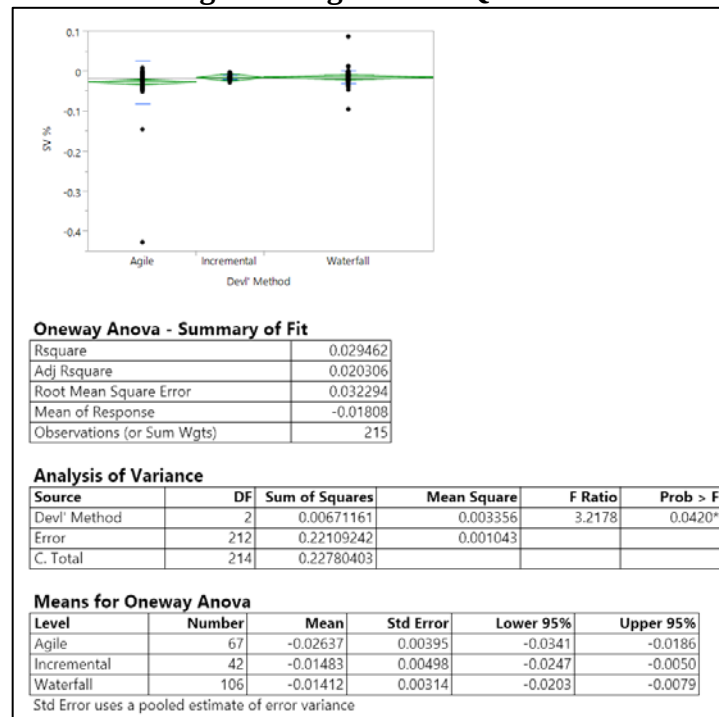


Figure A-38: SV Percentages during 50-75% Quartile – ANOVA

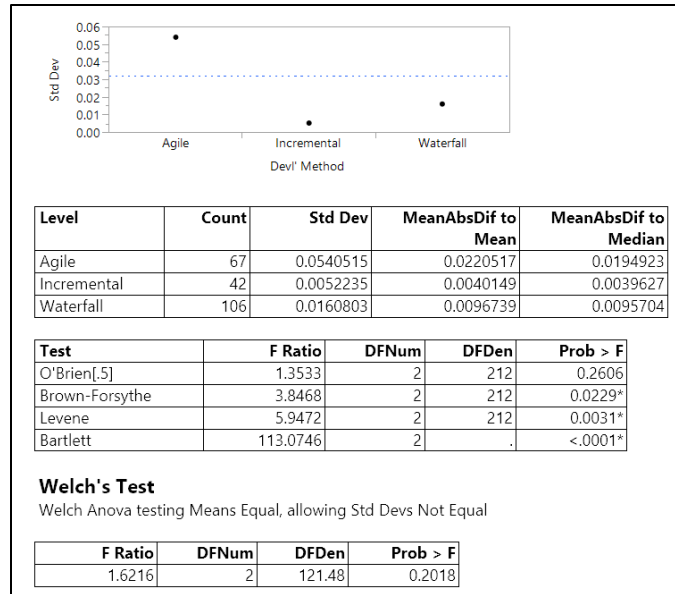


Figure A-39: SV Percentages during 50-75% Quartile - Test for Constant Variance

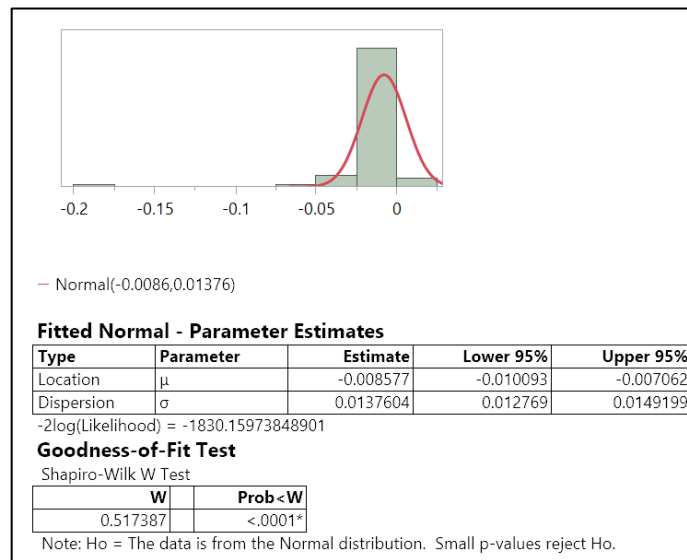


Figure A-40: SV Percentages during 50-75% Quartile - Test for Normality

Nonparametric Comparisons for All Pairs Using Steel-Dwass Method					
q*		Alpha			
2.34370		0.05			
Level	- Level	Score Mean Difference	Std Err Dif	Z	p-Value
Waterfall	Agile	11.12004	7.816994	1.422546	0.3292
Waterfall	Incremental	6.29964	7.816038	0.805989	0.6993
Incremental	Agile	3.11816	6.221143	0.501220	0.8707
Level	- Level	Hodges-Lehmann	Lower CL	Upper CL	
Waterfall	Agile	0.0026627	-0.001575	0.0079686	
Waterfall	Incremental	0.0013075	-0.002767	0.0049025	
Incremental	Agile	0.0011160	-0.004061	0.0069725	

Figure A-41: SV Percentages during 50-75% Quartile – Non-Parametric Test

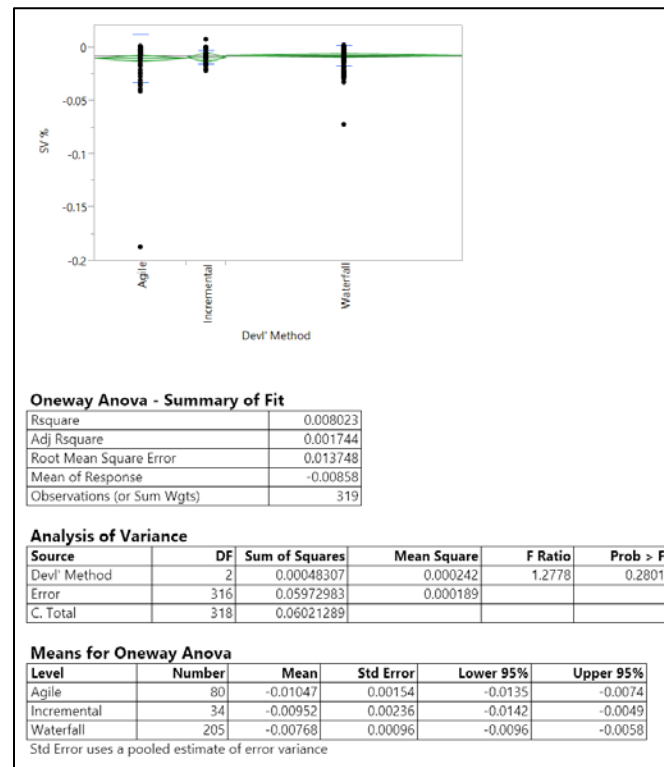


Figure A-42: SV Percentages during 75-100% Quartile - ANOVA

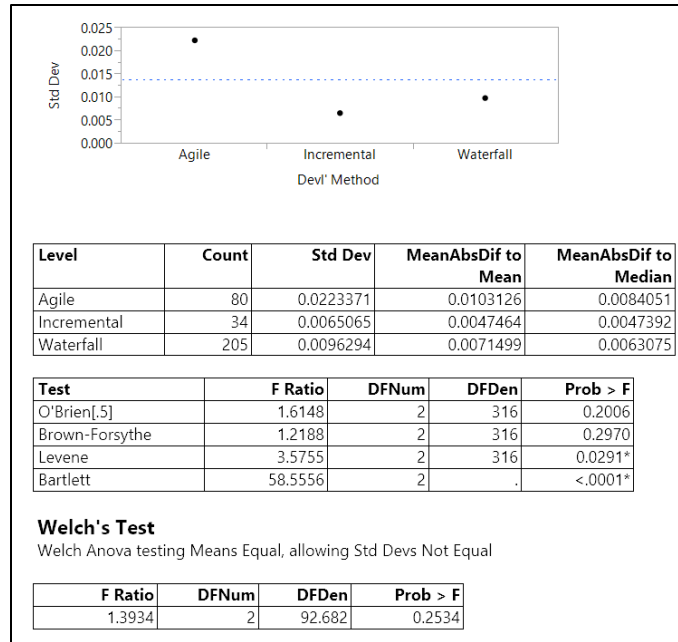


Figure A-43: SV Percentages during 75-100% Quartile - Test for Constant Variance

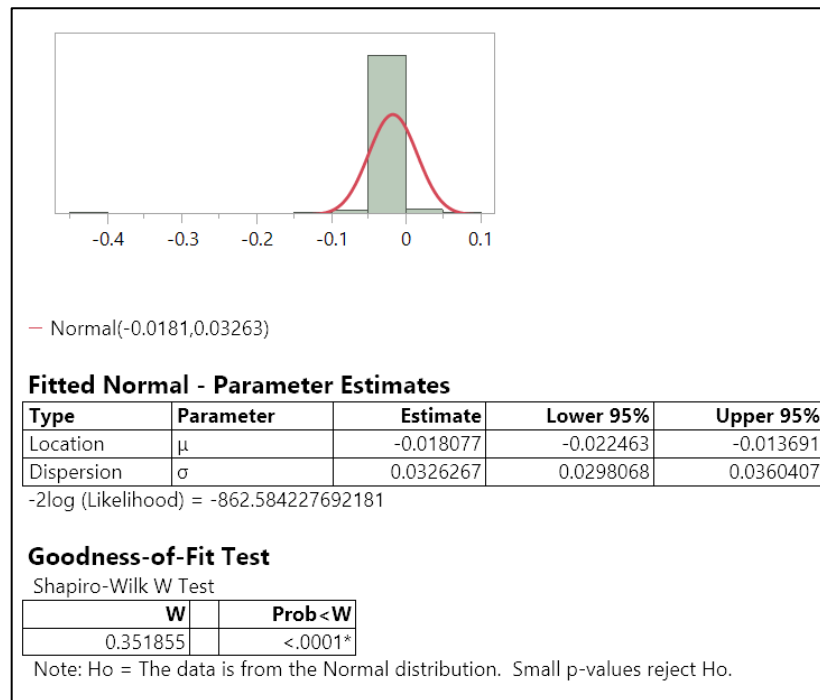


Figure A-44: SV Percentages during 75-100% Quartile - Test for Normality

Nonparametric Comparisons for All Pairs Using Steel-Dwass Method					
q*		Alpha			
2.34370		0.05			
Level	- Level	Score Mean Difference	Std Err Dif	Z	p-Value
Waterfall	Incremental	33.4669	12.80255	2.61408	0.0243*
Waterfall	Agile	12.0430	10.86464	1.10846	0.5088
Incremental	Agile	-16.3456	6.76671	-2.41559	0.0416*
Level	- Level	Hodges-Lehmann	Lower CL	Upper CL	
Waterfall	Incremental	0.004334	0.000524	0.006882	
Waterfall	Agile	0.000720	-0.000752	0.002204	
Incremental	Agile	-0.003654	-0.006524	-0.000093	

Figure A-45: SV Percentages during 75-100% Quartile - Non-Parametric Test

JMP® Output for Percent Cost Variances (CV)

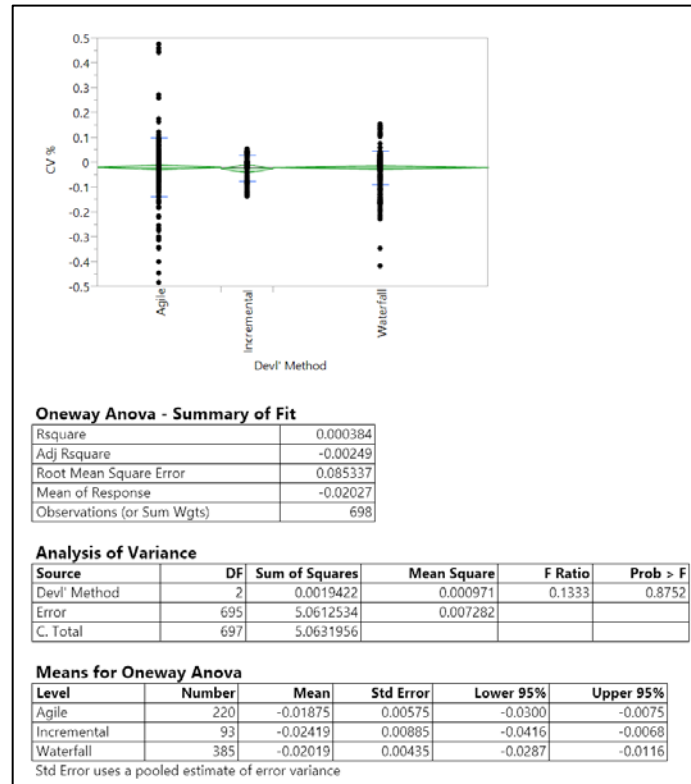


Figure A-46 CV Percentages – ANOVA

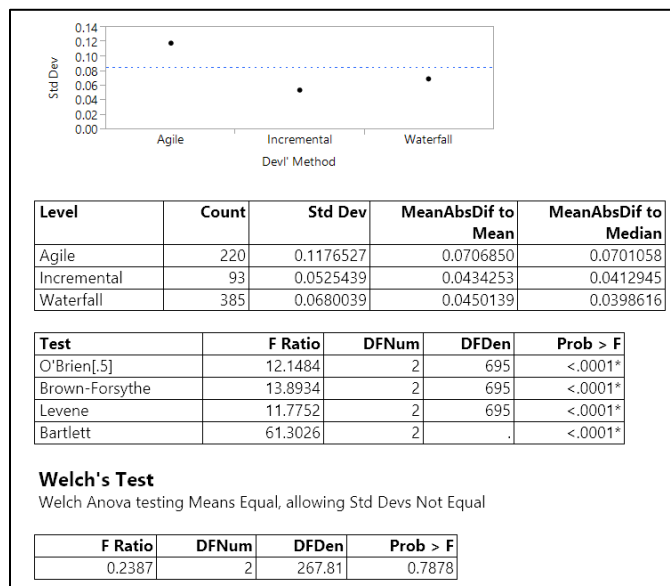


Figure A-47: CV Percentages - Test for Constant Variance

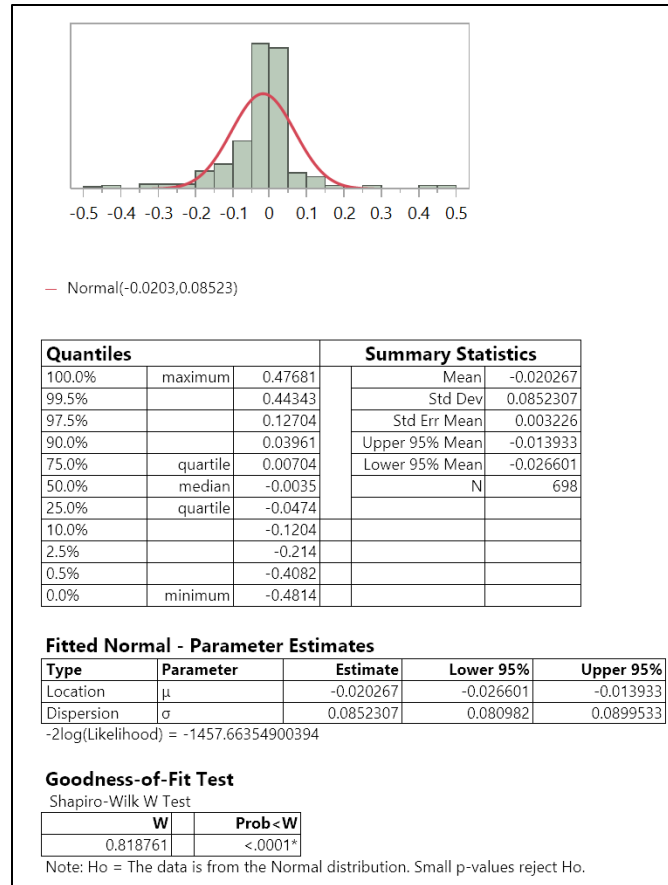


Figure A-48: CV Percentages - Test for Normality

Wilcoxon / Kruskal-Wallis Tests (Rank Sums)					
Level	Count	Score Sum	Expected Score	Score Mean	(Mean-Mean0)/Std0
Agile	220	76515.0	76890.0	347.795	-0.151
Incremental	93	30901.5	32503.5	332.274	-0.885
Waterfall	385	136535	134558	354.635	0.746

1-way Test, ChiSquare Approximation		
ChiSquare	DF	Prob>ChiSq
0.9441	2	0.6237

Figure A-49: CV Percentages - Non-Parametric Test

Bibliography

- Abba, W. (2001). How earned value got to prime time: A short look back and glance ahead. *The Measurable News*.
- Alleman, G. B., Henderson, M., & Seggelke, R. (2003). "Making agile development work in a government contracting environment-measuring velocity with earned value," *Proceedings of the Agile Development Conference*. IEEE Computer Society, Salt Lake City, Utah.
- Blackburn, T. (2015). *Key Performance Measures in a Lean Agile Program* [PowerPoint slides]. Retrieved from <http://www.acq.osd.mil/evm/resources/EVM-Agile%20Meeting.html>
- Christensen, D. S., & Ferens, D. V. (1995). Using earned value for performance measurement on software development projects. *Acquisition Review Quarterly*, (Spring), 155–171.
- Christoph, E. (2015). Earning Value the Agile Way: Using Story Points to Generate metrics. In R.W. Stratton (Ed.), *A Compendium on the Application of EVM to Agile Development and the Application of Agile Development to EVM* (14-20). The Measurable News.
- Cohn, M. (2014, November 11). Agile Needs to Be Both Iterative and Incremental. *Mountain Goat Software*. Message posted to: <http://www.mountaingoatsoftware.com/blog/agile-needs-to-be-both-iterative-and-incremental>
- Defense Acquisition University (DAU). (2015). *Gold CARD*. Retrieved from: www.dau.mil/default.aspx
- Department of Defense (DoD). (2015). *Instruction: Operation of the Defense Acquisition System*. DoDI 5000.02-154, 7 January.
- Defense Science Board (DSB). (2000). *Report on Defense Science Board Task Force on Defense Software*. Washington, DC.
- Eagle One Solutions. (2015). *Exceptional or Expected : Negative Performance Reporting on Agile Programs*. [PowerPoint slides]. Retrieved from <http://www.acq.osd.mil/evm/resources/EVM-Agile%20Meeting.html>
- Executive Office. (2012). *Federal Information Technology Shared Services Strategy*. 2 May 2012. Washington DC.
- Government Accountability Office, United States (GAO). (2013). *Information Technology: OMB and Agencies Need to More Effectively Implement Major Initiatives to Save Billions of Dollars*. (Vol. GAO-13–796T). Before the Subcommittee on Government Operations, Committee on Oversight and Government Reform, House of Representatives. Washington DC.

- Government Accountability Office, United States. (GAO). (2015a). *Defense Major Automated Information Systems: Cost and Schedule Commitments Need to Be Established Earlier*. (Vol. GAO-15-282). Report to Congressional Committees. Washington DC.
- Government Accountability Office, United States. (GAO). (2015b). *GAO Scheduling Best Practices Applied to an Agile Setting*. [PDF Presentation Document]. 15 April 2015.
- General Dynamics. (2015). *IBR Considerations for Agile Projects with EVM* [PowerPoint slides]. Retrieved from <http://www.acq.osd.mil/evm/resources/EVM-Agile%20Meeting.html>
- Ghosh, S. (2012). Systemic Comparison of the Application of EVM in Traditional and Agile Software Project. *PM World Today*, 14(2), 1–14.
- Gilmore, J. (2011). Key issues causing program delays in defense acquisition. *ITEA Journal*, 32(December), 389–391.
- KM Systems Group. (2015). *Agile EVM Lessons Learned* [PowerPoint Slides]. Retrieved from <http://www.acq.osd.mil/evm/resources/EVM-Agile%20Meeting.html>
- House Armed Services Committee. (2010). *House Armed Services Committee Panel on Defense Acquisition Reform Interim Findings and Recommendations*. Washington DC.
- Leidos. (2015). *EVMS / AGILE Management Approach* [PowerPoint Slides]. Retrieved from <http://www.acq.osd.mil/evm/resources/EVM-Agile%20Meeting.html>
- Lockheed Martin. (2015). *Lessons Learned Applying EVMS on Agile Programs* [PowerPoint Slides]. Retrieved from <http://www.acq.osd.mil/evm/resources/EVM-Agile%20Meeting.html>
- Manifesto for Agile Software Development. (2001). Retrieved July 27, 2015, from <http://agilemanifesto.org>
- Niwot Ridge LLC. (2015). *Is Earned Value + Agile A Match Made In Heaven?* [PowerPoint Slides]. Retrieved from <http://www.acq.osd.mil/evm/resources/EVM-Agile%20Meeting.html>
- Office of Management and Budget. (2010). *25 Point Implementation Plan to Reform Federal Information Technology Management*. Washington DC.
- Office of Management and Budget (OMB). (2012a). *Contracting Guidance to Support Modular Development*. Washington DC.

- Office of Management and Budget (OMB). (2012b). *Guidance on Exhibits 53 and 300 - Information Technology and E-Government*. Washington DC.
- Office of the Secretary of Defense, Cost Assessment and Program Evaluation Office (OSD CAPE). (n.d.). *Cost Assessment Data Enterprise*. Retrieved from <http://cade.osd.mil/>
- Office of Under Secretary of Defense for Acquisition, Technology and Logistics (OSD AT&L), Performance Assessments and Root Cause Analyses (PARCA). (2015a). *Department of Defense Earned Value Management System Interpretation Guide*. Washington DC.
- Office of Under Secretary of Defense for Acquisition, Technology and Logistics (OSD AT&L), Performance Assessments and Root Cause Analyses (PARCA). (2015b). *Initial Considerations for Earned Value Management for Agile Software Development in the Department of Defense*. Washington DC.
- Raytheon. (2015). *EVMS Planning and Execution for Agile Projects* [PowerPoint Slides]. Retrieved from <http://www.acq.osd.mil/evm/resources/EVM-Agile%20Meeting.html>
- Schwaber, K. (1994). SCRUM Development Process. Managing, April 1987: 23. Retrieved from DOI: http://dx.doi.org/10.1007/978-1-4471-0947-1_11
- Software Engineering Institute (SEI). (2010). *Considerations for Using Agile in DoD Acquisition*. Technical Note: CMU/SEI-2010-TN-002. Hanscom: Carnegie Mellon University.
- Software Engineering Institute (SEI). (2011). *Agile Methods: Selected DoD Management and Acquisition Concerns*. Technical Note: CMU/SEI-2011-TN-002. Hanscom: Carnegie Mellon University.
- Software Engineering Institute (SEI). (2014). *Potential Use of Agile Methods in Selected DoD Acquisitions : Requirements Development and Management*. Technical Note: CMU/SEI-2013-TN-006. Hanscom: Carnegie Mellon University.
- Sulaiman, T., Barton, B., & Blackburn, T. (2006). "AgileEVM-Earned Value Management in Scrum Projects," *Proceedings of the Agile Development Conference*. IEEE Computer Society.
- Sulaiman, T. (2007 October 5). AgileEVM: Measuring Cost Efficiency Across the Product Lifecycle. *InfoQueue*. Posted on: <http://www.infoq.com/articles/agile-evm>
- Sulaiman, T., Smits, H. (2007). Measuring Integrated Progress on Agile Software Development Projects. *Methods & Tools: Fall*. Retrieved from: <http://www.methodsandtools.com/archive/archive.php?id=61>

Takeuchi, H., & Nonaka, I. (1986). The new new product development game. *Harvard Business Review*, 64(1), 137–146.

Under Secretary of Defense for Acquisition, Technology and Logistics (USECDEF AT&L). (2015a). Memorandum: *Better Buying Power 3.0*. 09 April 2015. Washington DC.

Under Secretary of Defense for Acquisition, Technology and Logistics (USECDEF AT&L). (2015b). *Observations of Agile/Earned Value Management (EVM) and Beyond* [PowerPoint Slides]. Retrieved from <http://www.acq.osd.mil/evm/resources/EVM-Agile%20Meeting.html>

REPORT DOCUMENTATION PAGE			<i>Form Approved</i> <i>OMB No. 0704-0188</i>	
The public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number. PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.				
1. REPORT DATE (DD-MM-YYYY) 24-Mar-2016		2. REPORT TYPE Master's Thesis		3. DATES COVERED (From — To) September 2014 – March 2016
4. TITLE AND SUBTITLE Earned Value Reporting on Agile Software Development Programs within the Department of Defense			5a. CONTRACT NUMBER	
			5b. GRANT NUMBER	
			5c. PROGRAM ELEMENT NUMBER	
6. AUTHOR(S) Hodson, Elizabeth, G, Civilian GS-12			5d. PROJECT NUMBER	
			5e. TASK NUMBER	
			5f. WORK UNIT NUMBER	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Air Force Institute of Technology Department of Systems Engineering and Management 2950 Hobson Way WPAFB OH 45433-7765			8. PERFORMING ORGANIZATION REPORT NUMBER AFIT-ENV-MS-16-M-158	
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) Air Force LCMC Senior EV Analyst, GS-13, Victoria Brown 11 Eglin St Bldg. 1600 Hanscom AFB, MA 01731-1700 Victoria.brown.5@us.af.mil			10. SPONSOR/MONITOR'S ACRONYM(S) AFLCMC/FZC	
			11. SPONSOR/MONITOR'S REPORT NUMBER(S)	
12. DISTRIBUTION / AVAILABILITY STATEMENT Distribution Statement A. Approved for Public Release; Distribution Unlimited				
13. SUPPLEMENTARY NOTES				
14. ABSTRACT The correct measurement of earned value on agile software development programs has recently been identified as a concern by software developers implementing agile development methods. This research attempts to address the difficulties in reporting earned value on agile development programs, and the differences between the measures of completion in agile development versus earned value reporting. The major difference was identified to be in translating between the work originally planned and the work that was "recently" planned as part of the agile development process. This difference, if not transparent, leads to different measurements of earned value and inconsistent reports of a program's progress.				
15. SUBJECT TERMS Agile Software Development, Earned Value Management (EVM)				
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT UU	18. NUMBER OF PAGES 108
a. REPORT U	b. ABSTRACT U	c. THIS PAGE U		
			19a. NAME OF RESPONSIBLE PERSON R. David Fass, PhD, Lt Col, USAF (Ret.)	
			19b. TELEPHONE NUMBER (Include Area Code) (937)-255-3636 X4388 robert.fass@afit.edu	

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. Z39.18