Let’s Go Agile: Data-Driven Agile Software Cost and Schedule Models for DHS projects

Wilson Rosa, Sara Jardine, Kimberly Roye, Chad Lucas, and Kyle Eaton

Abstract— This paper offers a comprehensive set of software development effort and schedule estimating models for agile software projects using contractor backlog data, monthly contract reports, and related requirements documents. The regression models in this study predict effort or schedule using six different sizing measures: functional stories (requirements), unadjusted function points (UFP), simple function points (SiFP), issues, stories, and story points. We also offer effort and schedule benchmarks and descriptive statistics of the dataset. The analysis is based on data collected from 18 agile projects implemented within DHS and DoD between years 2014 to 2021. In addition to being able to estimate software development effort and schedule to support DHS and DoD decision reviews of agile programs, the cost analysis community can also use these models to crosscheck vendor proposals and evaluate contract performance.

Index Terms— Agile software development, Cost estimation, Product backlog, Story Point, Function Point, User Story, Issues, Functional requirements, Software acquisition, Software process, Time estimation

1 Introduction

1.1 Problem Statement

Historically, the United States (U.S.) Department of Homeland Security (DHS) has struggled with incorporating software development requirements into their program’s cost estimate and schedule [10]. This circumstance has hindered the ability of DHS Information Technology (IT) programs to develop accurate cost estimates and achievable baselines, resulting in cost and schedule breaches.

In 2017, the DHS Under Secretary for Management (USM) tasked the DHS Cost Analysis Division (CAD) to find ways to improve cost estimates for Agile software development programs. There were two primary objectives [10]:

1. Enhance the credibility and accuracy of a software development estimate, and
2. Decrease the time required to develop the estimate

Two years later, the DHS Acting Chief Financial Officer signed a memorandum requiring 10 active programs to adopt Simple Function Point (SiFP) [9] as the sizing measure to estimate and monitor the cost of Agile software development programs within DHS [10]. Despite the growing acceptance of SiFP, the availability of actual cost data on agile programs continues to hinder DHS' ability to develop accurate cost estimates.

1.2 Significance of This Study

This study offers a set of cost estimating relationships (CERs) and schedule estimating relationships (SERs) based on a dataset collected from completed DHS and DoD Agile projects categorized as automated information systems. The analysis uses a cross-company dataset and captures total effort at the release level incurred by the contractor's Agile teams. The choice for using total effort is driven by the fact that majority of Agile software contracts in DHS are Firm Fixed Price (FFP) or Time & Materials (T&M), and these typically report effort at the total level vice reporting by major elements (software development, etc.). Contractor's product backlog, and Requirements Traceability Matrix (RTM) are the primary document sources for all sizing measures.

This study breaks new ground by incorporating a potential software size measure titled Functional Story, not previously addressed elsewhere ([3], [7], [11], [12], [14]). Another important distinction of this study is comparing and ranking the accuracy and fit of effort and schedule parametric estimation models using six different sizing measures including: SiFP, UFP, story point, stories, issues, and functional story. These sizing measures were chosen since they are the only ones that could be collected or counted from backlog and RTM.

1.3 Research Questions

The following research questions (RQ) are addressed in this paper:

RQ 1: Do functional requirements, defined as functional stories, relate to total agile development effort?

RQ 2: Do software size, defined as issues, relate to total agile development effort?

RQ 3: Do software size, defined as stories, relate to total agile development effort?

RQ 4: Do story points relate to total agile development effort when using cross-company dataset?

RQ 5: Do UFP relate to total agile development effort?

RQ 6: Do SiFP relate to total agile development effort?

RQ 7: How do functional stories, issues, stories, story points, UFP, and SiFP rank as variables for predicting total development effort for agile projects?

RQ 8: Do software size, defined as functional stories, relate to total agile development schedule?

RQ 9: Do UFP relate to total agile development schedule?

RQ 10: Do SiFP relate to total agile development schedule?

RQ 11: How do functional stories, UFP, and SiFP rank for predicting total development schedule for agile projects?

2 Adoption of Agile at the DHS

In 2010, the Office of Management and Budget (OMB) issued a 25-point plan to reform IT projects and called on federal government agencies to implement shorter delivery timeframes [8]. The directive proposed a method consistent with *Agile* best practices when developing IT systems [8]. Soon after, DHS began adopting *Agile* processes for software development and delivery.

In 2016, the DHS USM initiated and conducted pilot efforts to improve the execution and oversight of DHS IT acquisition programs using industry best practices, including Lean and Agile incremental development methodologies [5]. Five pilot programs were conducted to address challenges associate with IT program overruns and schedule delays, lack of program transparency, and poor requirements development and traceability. An Agile Acquisition Working Group was then created to effectively plan and implement the programs, as well as developing appropriate documentation to support program execution [5]. The lessons learned from the pilots were used to develop and update policies and procedures for executing these five programs and future agile software acquisitions. Resulting key policies and procedures ([2], [5], [6]) are shown in Figure 1.

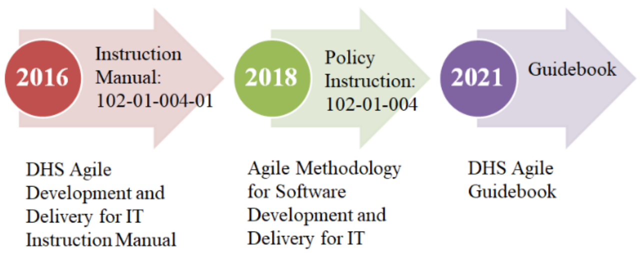

Figure 1 Agile Development Policies in DHS

In the DHS context, *Agile* is a development methodology that uses an iterative approach to deliver solutions incrementally through close collaboration and frequent reassessment [6]. Many different *Agile* approaches (Scrum, Scaled Agile Framework, SecDevOps, etc.) have been employed to date within DHS [2]. Those approaches support the Federal Chief Information Officer's goal [8] to reform IT management and the preference for modular approaches to deliver working functionality every six months [6]. As there are no specific set of DHS-approved methodologies, program managers are encouraged to determine the most appropriate *Agile* approaches for their program [2].

3 Research Method

3.1 Population and Sample

This study captured agile projects categorized as automated information systems; also known as *information technology* projects in the context of DHS, or *business systems* in the context of DoD. The sample dataset includes eighteen agile projects across 11 different companies, delivered for the DHS (15) and DoD (3) from years 2014 to 2021. As shown in Figure 2, most projects (12 of 18) in this study were delivered during the COVID-19 global pandemic that began in March 2020 and is still ongoing.

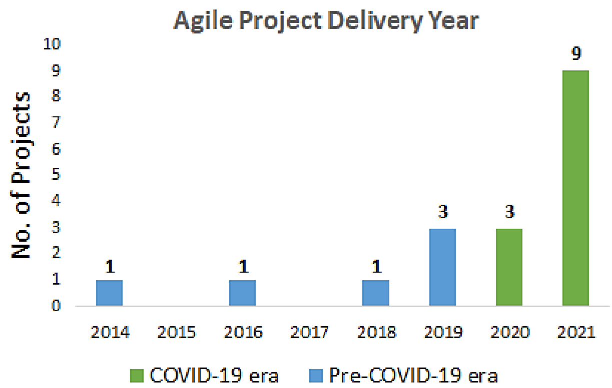

Figure 2 Agile Project Delivery Year

3.2 Data Collection

The data collected for each agile project included actual effort, schedule, final size, and project characteristics. The data collection kick-off and subsequent follow-on meetings with program managers were conducted 100% virtually using Microsoft Teams as the collaborative platform.

The data was extracted from official documents such as monthly contractor invoices, product backlog, functional requirements document (FRD), requirements traceability matrix (RTM), acquisition documents, and agile core metrics. The documents shown in Figure 3 were provided by program managers. Of note, a product backlog is a collection of issues completed or remaining to be addressed. The backlog is generated for the overall software product, each individual release, and each iteration/sprint (identifying the remaining Issues for a specific iteration/sprint). The product backlog is maintained by the contractor Agile teams.

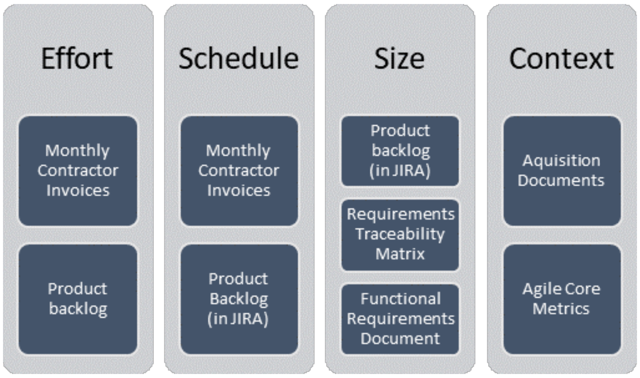

Figure 3 Data Sources

The effort hours in this study captures eleven major cost elements incurred by the contractor's agile development teams in accordance with the DHS IT Work Breakdown structure (WBS). Those major IT WBS cost elements are also applicable to the DoD programs captured in this study and are identified in Table 2. The choice for reporting total effort (as opposed to software development alone), is driven by the fact that most agile development contracts in DHS are FFP or T&M, and therefore, generally do not break out effort by major cost elements as would traditional cost-plus contracts.

Table 1 Agile Project Effort Activities

ID	DHS IT WBS Element
1.i.1	Program Management
1.i.2	Systems Engineering
1.i.4.2	Software Development
1.i.4.3	Data Development & Transition
1.i.4.5	Training Development
1.i.4.6.1	Development Test & Evaluation
1.i.4.6.1	Cybersecurity Test & Evaluation
1.i.4.7	Logistics Support Development
1.i.7	System Level Integration & Test
1.i.8.6.1	Help Desk/Service Desk (Tier 3)
1.i.8.6.4	Software Maintenance

3.3 Data Normalization

The data normalization process included obtaining agile size measures, counting functional stories (requirements), and computing function points. The details of the normalization process will be discussed in this section.

3.3.1 Counting Agile Size Measures

The team used a repeatable method to measure agile size for each project in the dataset. Below is an outline of the four steps to determine the total number of user stories, issues, and story points from an agile project's product backlog. Table 2 provides an example of the backlog output the team used during the counting process.

Step 1: Determine 100% Complete Issues. Find the column titled, *Issue Status* and filter by rows marked as *Done*. By filtering by the issues that were 100% compete, issues that had a status of being *in progress or deferred*, were omitted.

Step 2: Calculate total stories. In the column titled, *Issue Type*, count the rows marked as *story*. This example results in a total count of 5 stories.

Step 3: Calculate total issues. In the column titled, *Issue Type*, count all rows marked as either story, task, bug, or other. This example results in a total count of 8 issues.

Step 4: Calculate total story points. In the column titled, *Story Points*, count the total story points by adding the values for each row. This example results in a total count of 24 story points.

Table 2 Example Agile Size Measures

Issue ID	Issue Status	Issue Description	Issue Type	Story Points
0001	Done	As a <user> I need to manually inititate the<outcome>	Story	2
0002	Done	As a <user> I need to view the <outcome>	Story	5
0003	Done	As a <user> I want to view trend of <information> so I can <outcome>	Story	5
0004	Done	As a <developer> I would like to <function> so that I can <outcome>	Story	2
0005	Done	As a <system administrator> I need to manage certificates so I can <outcome>	Story	5
0006	Done	As a <SCRUM Master> I want to review list of Epic(s) so I can <outcome>	Task	1
0007	Done	The following field name is spelled incorrectly: <Name>	Bug	3
0008	Done	As a <system user> I need to have the selected tools installed…	Other	1

3.3.2 Counting Functional Stories or Requirements

The team also used a specific counting method to determine functional stories (or requirements). Below is an outline of the three steps to determine the total functional stories from an agile project's product backlog. Table 3 provides an example of the backlog the team used during the counting process.

Step 1. Find the column titled, *Issue Type,* and the rows marked as *story* in this column.

Step 2. Categorize each story as either functional or non-functional. This step is performed by a Certified Function Point Specialist.

Step 3. This example results in a total count of three functional stories. The total functional stories equal the number of functional requirements (3) captured in the RTM or FRD.

Table 3 Example Functional Requirements

Issue Description	Issue Type	Category
As a <user> I need to manually inititate the<outcome>	story	Functional
As a <user> I need to view the <outcome>	story	Functional
As a <user> I want to view trend of <information> so I can <outcome>	story	Functional
As a <developer> I would like to <function> so that I can <outcome>	story	Non-functional
As a <system administrator> I need to manage certificates so I can <outcome>	Story	Non-functional

3.3.3 Computing Function Points

Once the functional stories (or functional requirements) were identified for each project in the agile dataset, the final step was to compute the UFP and SiFP for each agile project. To minimize function point counting errors, the UFP and SiFP counts for all agile projects were derived by the same Certified Function Point Specialist (CFPS). After the CFPS and support team completed their counts for each project, the team met to deconflict any issues. This resulted in a single, verified, validated, and agreed upon UFP and SiFP count for each project.

3.4 Variables in the Study

The variables in this study (Table 4) were chosen since these represented all inputs that can be collected from available sources in DHS. Of note, a categorical variable, characterizing whether a project scope was an enhancement or full development, was also evaluated in the regression analysis. A full development versus enhancement scope is important in estimating effort and schedule since the scope of an enhancement is typically less than a one-year effort while a full development effort is typically more than a one-year.

Table 4 Variable Name and Definition

Variable	Type	Definition
Effort	Dependent	Actual labor hours associated to all development activities listed in Table 1
Schedule	Dependent	Actual development time (in months) associated to all activities listed In Table 1. Time reported at the release level or the delivery of software from development into a production environment for use.
Functional Stories	Independent	A subset of requirements describing what the software does in terms of tasks and services [1]
Issue	Independent	A unit of work that will be traced through a workflow, from creation to completion. In the backlog, every row is referred to as an *issue*. Total issue is the sum of stories, bugs, tasks, epics, and others
Story	Independent	A feature and/or unit of business value that can be estimated and tested. Describes work that must be done to create and deliver a feature for a product.
Story Point	Independent	A unit of measure to express the overall size of a user story, feature, or other piece of work in the backlog.
Unadjusted Function Point (UFP)	Independent	A function point count without the assignment of complexity to any of the objects counted
Simple Function Point (SiFP)	Independent	Method for sizing software that requires only the identification of elementary processes and logic files to approximate a function point count (2)
Scope	Categorical	A categorical variable to indicate whether the scope of development project is an enhancement or full development effort

4 Dataset Demographics

The entirety of the dataset in this study represents projects categorized as Automated Information Systems (AIS) delivered from 2014 to 2021. Fourteen of the 18 datapoints are hosted on the cloud, while the remaining four are hosted on-premises. Of the 14 cloud-hosted, 13 used Amazon Web Services. All DHS projects were from *Surface Fixed* operating environments, while the DoD projects are from *Surface Fixed*, *Sea System* environments. Figure 4 shows project counts by agency and operating environment.

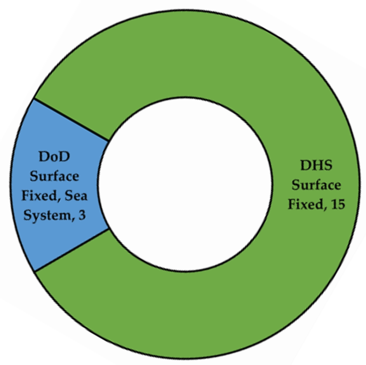

Figure 4 Operating Environment

The project counts by contract type is provided in Figure 5. Most agile projects utilized Firm-Fixed Price (FFP) and Time and Materials (T&M).

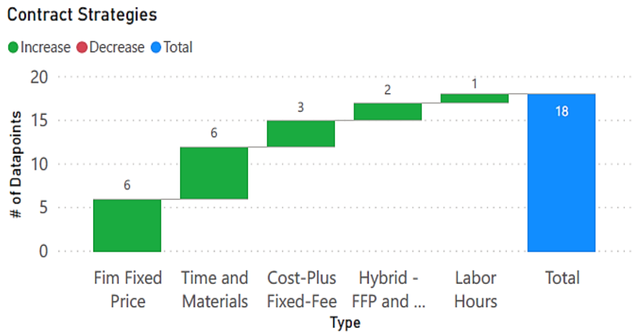

Figure 5 Contract Strategy

The team approach of each project was documented to better understand the development practices of the agile teams. Of the eighteen agile projects, half (9) followed SecDevOps practices while two-thirds of the Hybrid Agile projects were DevOnly. Figure 6 presents the team approach for the dataset.

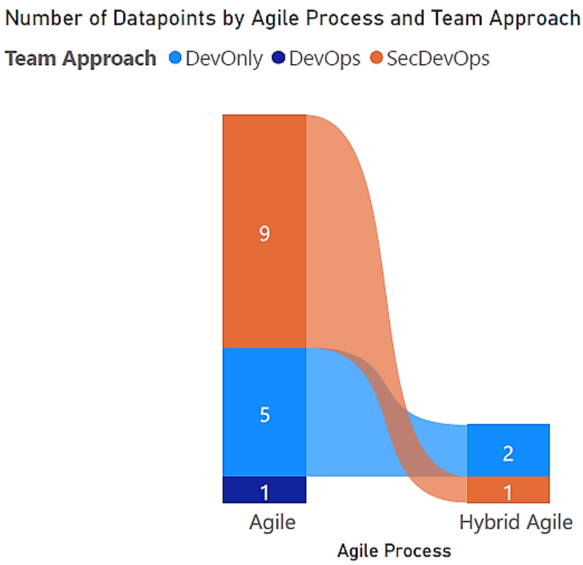

Figure 6 Agile Process and Team Approach

The distribution of Agile frameworks is presented in Figure 7. The Hybrid Agile projects all used the Scaled Agile Framework (SAFe). The majority (13) of the Agile projects were Scrum.

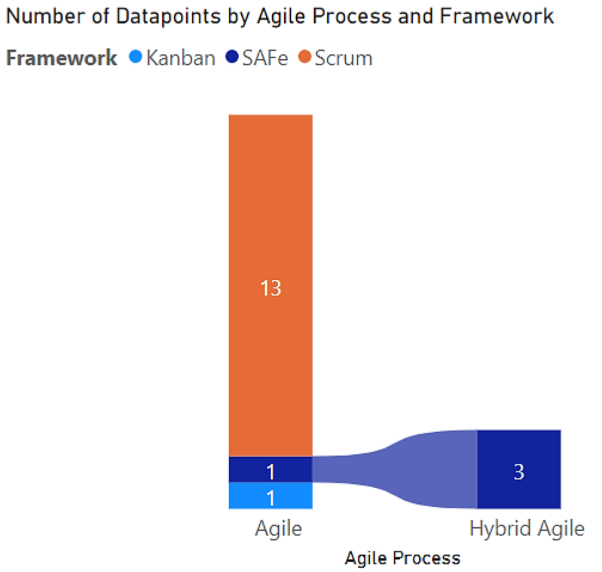

Figure 7 Agile Framework

The schedule duration of the agile projects in the dataset were collected in months and reported at the release level. The number of iteration weeks for the Agile or Hybrid Agile development process in the dataset were between two and four weeks. The distribution of sprint/iteration intervals (in weeks) is presented in Figure 8.

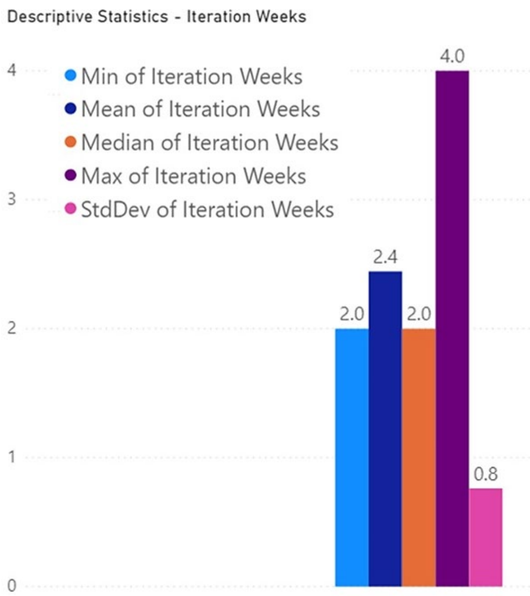

Figure 8 Weeks per Iteration/Sprint

5 Descriptive Statistics

Descriptive statistics provide a summary of the characteristics of the data used during regression analysis. The two variables used as dependent are *Hours* and *Months*. Figure 9 displays the descriptive statistics for Effort Hours and Figure 10 presents those for Schedule Months.

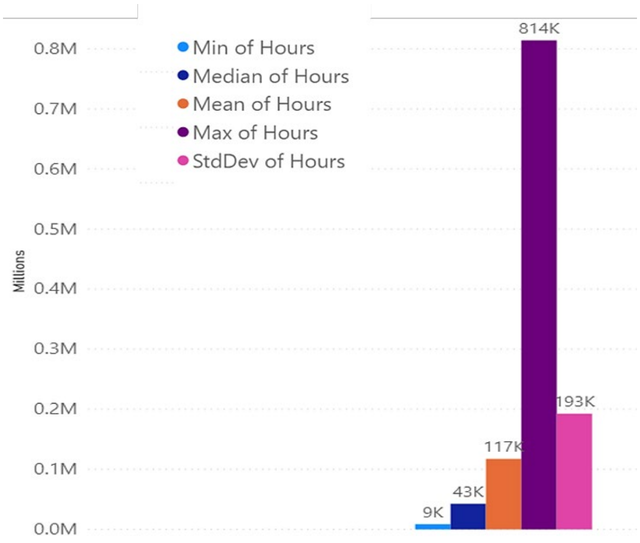

Figure 9 Effort (Hours) Distribution

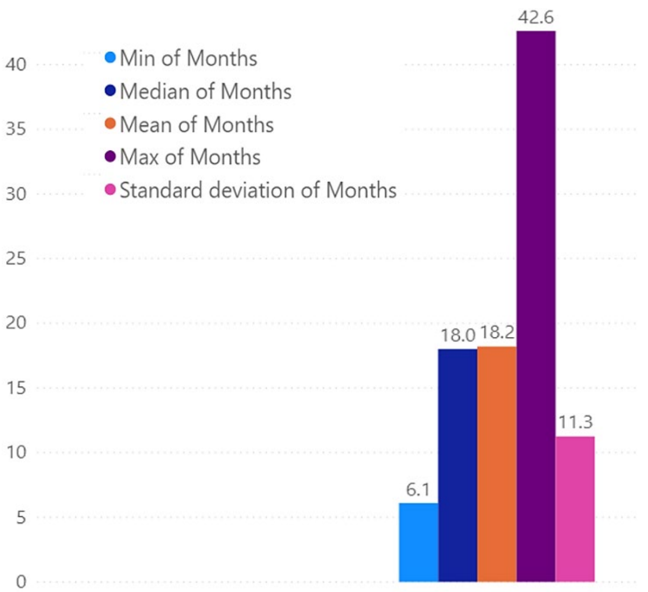

Figure 10 Schedule (Months) Distribution

The descriptive statistics for the independent variables in the agile dataset are presented in Table 5. For each variable, the minimum value, median, maximum value, and standard deviation (StdDev) values are shown.

Table 5 Independent Variable Descriptive Statistics

Size Measure	Min	Median	Max	StdDev
Functional Stories	16	95	1,881	441
Issues	75	842	5,744	1,547
Stories	27	424	4,964	1,274
Story Points	602	2,708	24,492	6,109
UFP	86	659	9,628	2,368
SiFP	94	712	10,650	2,631

1 Functional Story ~ 1 Functional Requirement

6 Benchmarks

This section provides effort and schedule estimation factors that can be used by the cost estimating community to build quick estimates and use these to validate program office estimates or contractor proposals.

The effort benchmarks include (1) Hours per Functional Story, (2) Hours per UFP, and (3) Hours per SiFP. Schedule benchmarks include (1) Functional Story per Peak FTE per Month, (2) UFP per Peak FTE per Month, and (3) SiFP per Peak FTE per Month. These benchmarks are shown in Table 6 below. For each benchmark, the 25th quartile (Q1), median (Q2), 75th quartile (Q3), standard deviation, and coefficient of variation (CV) values are shown.

Table 6 Effort and Schedule Benchmarks

Benchmark	Q1	Q2	Q3	StdDev	CV
Hours/Functional Story	410	494	653	261	47%
Hours/UFP	61	81	107	40	46%
Hours/SiFP	57	71	100	39	47%
Functional Story /FTE/ Month	0.19	0.28	0.32	0.14	47%
UFP / FTE / Month	1.2	1.8	2.1	0.9	50%
SiFP / FTE / Month	1.3	2.1	2.3	1.1	52%

FTE = Peak Full-Time Equivalent

7 Regression Analysis

7.1 Model Selection and Validation

Regression analysis was performed using the Automated Cost Estimating Integrated Tools (ACEIT's) Cost Analysis Statistical Package (CO$TAT) [13]. To assess regression model quality, goodness-of-fit metrics in Table 7 were evaluated. The normal probability plot was evaluated and display for each effort and schedule model in the next section.

Table 7 Goodness-of-Fit Metrics

Metric	Description
R^2	Coefficient of determination is the percentage of total variation in the response variable explained by the model. [25]
R^2 (adj)	Adjusted R^2 is the percentage of the variation in the response explained by the model, adjusted for the number of predictors in the model relative to the number of observations.
R^2 (pred)	Predicted R^2 is a cross validation method that involves removing each observation from the dataset, estimating the regression equation, determining how well the model predicts the removed observation, and repeats this for all data points.
P-value	Statistical significance of the coefficient.
SEE	Standard Error of the Estimate is the difference between observed and the estimated effort. SEE is to linear models as standard deviation is to sample means
F-test	F test is the square of the equivalent t-test; the bigger it is, the smaller the probability that difference could occur by chance.

Metric	Description
MAD	Mean Absolute Deviation of % errors measure the percentage by which the regression overestimates or underestimates the observed actual value.

7.2 Model Results

This section provides the results of the effort and schedule models associated with Research Questions (RQ) 1 through 11. Since the sample dataset included 18 datapoints, only single variable regression was performed. As more data points are collected and the dataset continues to grow, multivariable regression will be evaluated in future research. Of the 18 datapoints, only 15 were used in the regression analysis while three were omitted because of missing data or incomplete scope in relation to the other agile projects.

The log-normal equation form using ordinary least squares (OLS) regression [13] was determined to be the best fit for the resulting effort and schedule models since the normal probability plots for this regression method showed the residuals closest to the fitted line, which validated the use of OLS. The high t-statistics and low p-values of each variable infers that the independent variable is strongly correlated to effort or schedule. The high adjusted R^2 and low MAD characterize each model's accuracy and fits the data well. The marginal difference between adjusted and predicted R^2 also suggest that each model predicts new observations just as well as it fits the existing data. In general, the criteria for model quality and acceptance are shown in Table 8.

Table 8 Criteria for Model Acceptance

Metric	Criteria
R^2	≥ 70%
$R^2_{(adj)}$	≥ 65%
$R^2_{(pred)}$	$(R^2_{(adj)} - R^2_{(pred)}) \leq 10\%$
P-value	p-value ≤ 0.05 for coefficient of Independent variable
MAD	≤ 55%
Normal Probability Plot	Visually, the residuals on the plot approximate a straight line

The following models are applicable to DHS and DoD agile software project sizes ranging approximately 20 to 5,000 stories, 10 to 2,000 functional stories, 80 to 11,000 function points, and a peak staff between 9 to 200 FTEs. Models 1-6 represent the resulting equations that estimate effort in hours while models 8-10 represent the resulting equations that estimate schedule duration in months at the release level and regardless of the number of sprints or iterations. A comparative analysis associated with the resulting effort models is provided in response to RQ 7 and the resulting schedule models is provided in response to RQ 11.

7.2.1 Effort Model 1

RQ 1: Do software functional requirements, defined as **functional stories**, relate to total agile development effort?

Equation (1) predicts effort for agile software development projects using functional stories (requirements) as input.

$$\mathbf{Effort = 935.5xREQ^{0.882}} \qquad (1)$$

Where,

Effort = total final development hours

REQ = functional stories obtained from product's backlog, RTM, or FRD

Table 9 provides the regression analysis report of the coefficient statistics, goodness-of-fit statistics, and analysis of variance for equation (1). The resulting equation is statistically significant and demonstrates that functional requirements is an effective variable to estimate the effort of agile software projects.

Table 9 Regression Analysis Results for Equation (1)

Coefficient Statistics Summary

Term	Coef	T-Statistic	P-value
Intercept	6.84	16.83	0.00
REQ	0.88	10.57	0.00

Goodness-of-Fit Statistics

SE	R^2	$R^2_{(adj)}$	$R^2_{(pred)}$	MAD
0.39	89.57%	88.77%	85.86%	31.28%

Analysis of Variance

Source	DF	Sum of Sq.	Mean Sq.	F-stat
Regression	1	16.93	16.93	111.65
Residual	13	1.97	0.15	
Total	14	18.90		

Figure 11 shows the normal probability plot for Equation (1). The residuals are close to the straight line. This suggests that loglinear regression is valid for modeling effort vs functional stories.

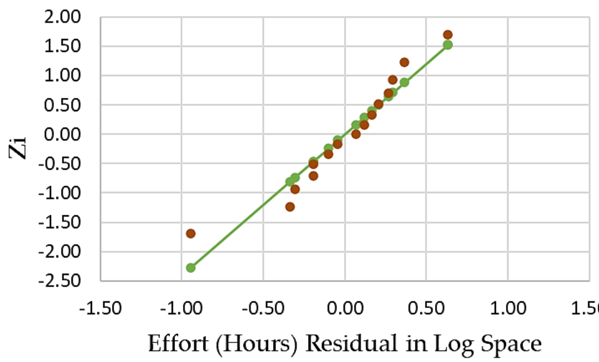

Figure 11 Normal probability Plot for Equation (1)

7.2.2 Effort Model 2

RQ 2: Do software size, defined as **issues**, relate to total agile development effort?

Equation (2) predicts effort for agile software development projects as a function of issues.

$$\mathbf{Effort} = \mathbf{604.3}x\mathbf{ISSUES}^{\mathbf{0.6879}} \qquad (\mathbf{2})$$

Where,

Effort.	=	total final development hours
ISSUES	=	sum of stories, bugs, tasks, epics, or any other fixes

Table 10 provides the regression analysis report of the coefficient statistics, goodness-of-fit statistics, and analysis of variance for equation (2). The resulting equation is statistically significant and demonstrates that issues is an effective variable to estimate the effort of agile software projects.

Table 10 Regression Analysis Results for Equation (2)

Coefficient Statistics Summary

Term	Coef	T-Statistic	P-value
Intercept	6.40	7.80	0.00
ISSUES	0.69	5.72	0.00

Goodness-of-Fit Statistics

SE	R^2	$R^2_{(adj)}$	$R^2_{(pred)}$	MAD
0.64	71.53%	69.34%	59.39%	51.55%

Analysis of Variance

Source	DF	Sum of Sq.	Mean Sq.	F-stat
Regression	1	13.52	13.52	32.66
Residual	13	5.38	0.41	
Total	14	18.90		

Figure 12 shows normal probability plot for Equation (2). The residuals approximate a straight line. This suggests that loglinear regression is valid for modeling effort vs issues.

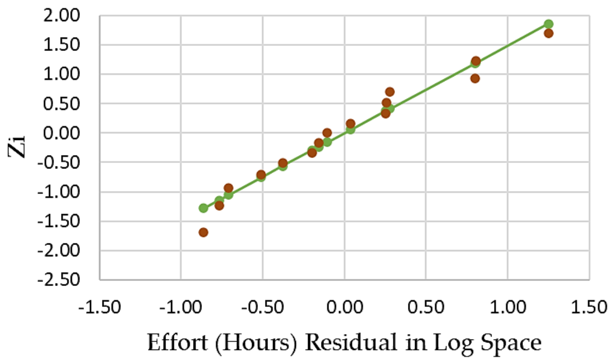

Figure 12 Normal probability Plot for Equation (2)

7.2.3 Effort Model 3

RQ 3: Do software size, defined as **stories**, relate to total agile development effort?

Equation (3) predicts effort for agile software development projects as a function of stories.

$$\mathbf{Effort} = \mathbf{1365}x\mathbf{STORY}^{\mathbf{0.6228}} \qquad (\mathbf{3})$$

Where,

Effort	= total final development hours
STORY	= total stories derived from backlog in JIRA

Table 11 provides the regression analysis report of the coefficient statistics, goodness-of-fit statistics, and analysis of variance for equation (3). The resulting equation is statistically significant and demonstrates that stories is an effective variable to estimate the effort of agile projects.

Table 11 Regression Analysis Results for Equation (3)

Coefficient Statistics Summary

Term	Coef	T-Statistic	P-value
Intercept	7.22	10.25	0.00
STORY	0.62	5.54	0.00

Goodness-of-Fit Statistics

SE	R^2	$R^2_{(adj)}$	$R^2_{(pred)}$	MAD
0.66	70.22%	67.93%	59.04%	54.11%

Analysis of Variance

Source	DF	Sum of Sq.	Mean Sq.	F-stat
Regression	1	13.27	13.27	30.65
Residual	13	5.63	0.43	
Total	14	18.90		

Figure 13 shows the normal probability plot for Equation (3). The residuals approximate a straight line. This suggests that loglinear regression is valid for modeling effort vs story.

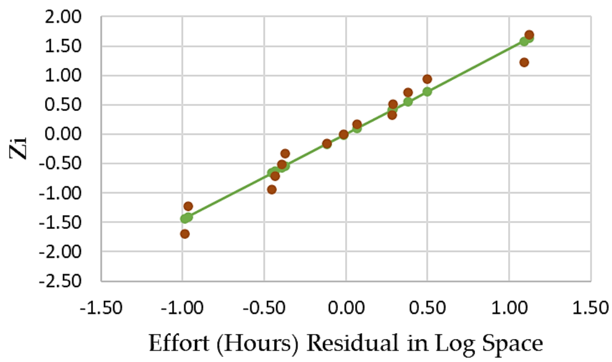

Figure 13 Normal Probability Plot for Equation (3)

7.2.4 Effort Model 4

RQ 4: Do **story points** relate to total agile development effort?

Equation (4) predicts effort for agile software development projects as a function of story points.

$$\mathbf{Effort} = \mathbf{206.5}x\mathbf{STY\_PTS}^{\mathbf{0.6842}} \qquad (4)$$

Where,

Effort = total final development hours

STY_PTS = story Points derived from backlog in JIRA

Table 12 provides the regression analysis report of the coefficient statistics, goodness-of-fit statistics, and analysis of variance for equation (4). The resulting equation is statistically significant and demonstrates that story points is an effective variable to estimate the effort of agile software projects.

Table 12 Regression Analysis Results for Equation (4)

Coefficient Statistics Summary

Term	Coef	T-Statistic	P-value
Intercept	5.33	7.75	0.00
STY_PTS	0.68	8.06	0.00

Goodness-of-Fit Statistics

SE	R^2	$R^2_{(adj)}$	$R^2_{(pred)}$	MAD
0.39	84.42%	83.12%	78.15%	32.68%

Analysis of Variance

Source	DF	Sum of Sq.	Mean Sq.	F-stat
Regression	1	9.80	9.80	65.02
Residual	12	1.81	0.15	
Total	13	11.61		

Figure 14 shows the normal probability plot for Equation (4). The residuals approximate a straight line. This suggests that loglinear regression is valid for modeling effort vs story points.

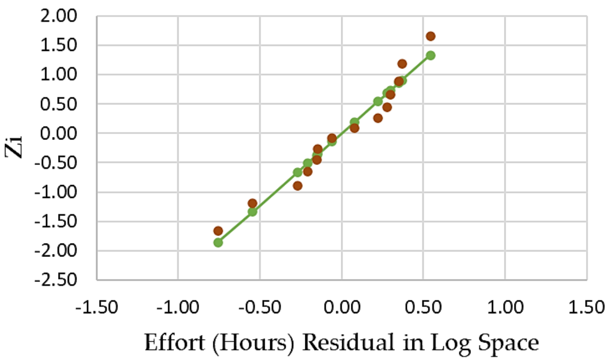

Figure 14 Normal Probability Plot for Equation (4)

7.2.5 Effort Model 5

RQ 5: Do **unadjusted function points** relate to total agile development effort?

Equation (5) predicts effort for agile software development projects as a function of unadjusted function points.

$$\mathbf{Effort} = \mathbf{189.5}x\mathbf{UFP}^{\mathbf{0.8747}} \qquad (5)$$

Where,

Effort = total final development hours

UFP = total Unadjusted Function Points

Table 13 provides the regression analysis report of the coefficient statistics, goodness-of-fit statistics, and analysis of variance for equation (5). The resulting equation is statistically significant and demonstrates that unadjusted function points is an effective variable to estimate the effort of agile software projects.

Table 13 Regression Analysis Results for Equation (5)

Coefficient Statistics Summary

Term	Coef	T-Statistic	P-value
Intercept	5.24	9.71	0.00
UFP	0.87	10.84	0.00

Goodness-of-Fit Statistics

SE	R^2	$R^2_{(adj)}$	$R^2_{(pred)}$	MAD
0.38	90.04%	89.28%	85.63%	31.61%

Analysis of Variance

Source	DF	Sum of Sq.	Mean Sq.	F-stat
Regression	1	17.02	17.02	117.54
Residual	13	1.88	0.14	
Total	14	18.90		

Figure 15 shows the normal probability plot for Equation (5). The residuals approximate a straight line. This suggests that loglinear regression is valid for modeling effort vs UFP.

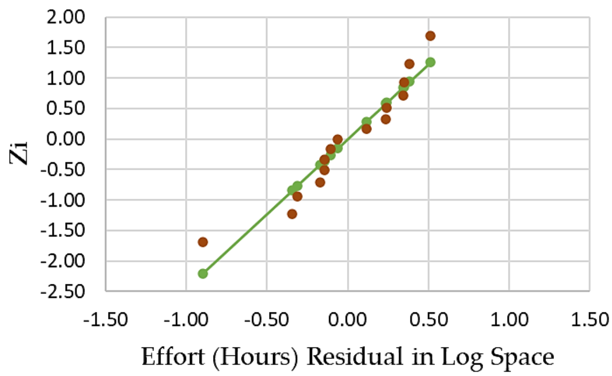

Figure 15 Normal Probability Plot for Equation (5)

7.2.6 Effort Model 6

RQ 6: Do **simple function points** relate to total agile development effort?

Equation (6) predicts effort for agile software development projects as a function of simple function points and a dummy variable associated with scope type.

$$\mathbf{Effort} = \mathbf{261.1}x\mathbf{SiFP}^{0.7708}x\mathbf{1.615}^{D1} \qquad (6)$$

Where,

Effort	=	total final development hours
SiFP	=	Simple Function Point
D1	=	dummy variable associated with scope where full development was assigned a value of 1 and enhancement was assigned a value of 0

Table 14 provides the regression analysis report of the coefficient statistics, goodness-of-fit statistics, and analysis of variance for equation (6). The resulting equation is statistically significant and demonstrates that simple function points and a scope type dummy variable are effective variables to estimate the effort of agile software projects.

Table 14 Regression Analysis Results for Equation (6)

Coefficient Statistics Summary

Term	Coef	T-Statistic	P-value
Intercept	5.56	10.67	0.00
SiFP	0.77	8.96	0.00
D1	0.48	2.14	0.05

Goodness-of-Fit Statistics

SE	R^2	$R^2_{(adj)}$	$R^2_{(pred)}$	MAD
0.35	92.00%	90.67%	86.25%	25.93%

Analysis of Variance

Source	DF	Sum of Sq.	Mean Sq.	F-stat
Regression	2	17.39	8.69	69.02
Residual	12	1.51	0.13	
Total	14	18.90		

Figure 16 shows the normal probability plot for Equation (6). The outlier in the plot was observed and was not removed since further investigation did not reveal that this project was an anomaly or had deficiencies. The residuals approximate a straight line. This suggests that loglinear regression is valid for modeling effort vs SiFP along with scope type (dummy variable).

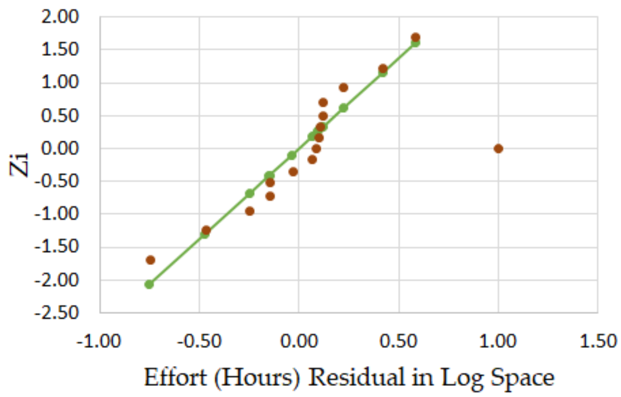

Figure 16 Normal Probability Plot for Equation (6)

7.2.7 Effort Model Comparison

RQ 7: How do **functional requirements, issues, stories, story points, UFP,** and **SiFP** rank as variables to predict total development effort for agile projects?

Table 15 compares the statistical significance of the effort estimation models using six different software size predictors. The comparative results with a synopsis of the suggested ranking order of the models is summarized below.

Table 15 Effort Model Comparison

ID	Model Equation	$R^2_{(adj)}$	$R^2_{(pred)}$	MAD	Rank
1	$E = 935.5x\mathrm{REQ}^{0.882}$	88.8%	85.9%	31.2%	3
2	$E = 604.3x\mathrm{ISSUES}^{0.6879}$	69.3%	59.4%	51.5%	5
3	$E = 1365x\mathrm{STORY}^{0.6228}$	67.9%	59.04%	54.1%	6
4	$E = 206.5x\mathrm{STY\_PTS}^{0.6842}$	83.1%	78.2%	32.6%	4
5	$E = 189.5x\mathrm{UFP}^{0.8747}$	89.3%	85.6%	31.6%	2
6	$E = 261.1x\mathrm{SiFP}^{0.7708}x1.6^{D1}$	90.7%	86.3%	25.9%	1

Based on the comparison of effort models, although all models passed the criteria for statistical significance, **simple function points, unadjusted function points, and functional requirements** are stronger predicters to development effort than **stories, story points, or issues**. Next, we will evaluate the resulting schedule models.

7.2.8 Schedule Model 1

RQ 8 Do software requirements, defined as **functional stories**, relate to total agile development schedule?

Equation (7) predicts schedule for agile software development projects as a function of functional requirements and a dummy variable associated with scope type.

$$\mathbf{Schedule} = \mathbf{2.685}x\mathbf{REQ}^{0.2135}\, x\mathbf{2.718}^{D1} \qquad (7)$$

Where,

Schedule	=	total final development months
REQ	=	functional stories obtained from product's backlog, RTM, or FRD

D1 = dummy variable associated with scope where full development was assigned a value of 1 and enhancement a value of 0

Table 16 provides the regression analysis report of the coefficient statistics, goodness-of-fit statistics, and analysis of variance for equation (7). The resulting equation is statistically significant and demonstrates that functional requirements and a scope type dummy variable are effective variables to estimate the schedule of agile software projects.

Table 16 Regression Analysis Results for Equation (7)

Coefficient Statistics Summary

Term	Coef	T-Statistic	P-value
Intercept	0.99	3.13	0.01
REQ	0.21	2.80	0.02
D1	1.00	5.49	0.00

Goodness-of-Fit Statistics

SE	R^2	$R^2_{(adj)}$	$R^2_{(pred)}$	MAD
0.26	88.39%	86.45%	81.78%	18.83%

Analysis of Variance

Source	DF	Sum of Sq.	Mean Sq.	F-stat
Regression	2	6.28	3.14	45.67
Residual	12	0.83	0.07	
Total	14	7.11		

Figure 17 shows the normal probability plot for Equation (7). The residuals approximate a straight line. This suggests that loglinear regression is valid for modeling schedule vs functional requirements along with scope type.

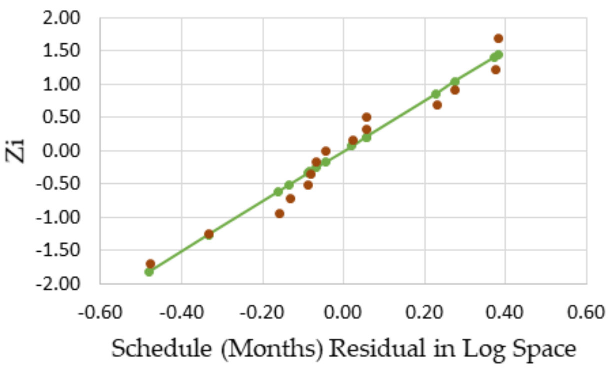

Figure 17 Normal Probability Plot for Equation (7)

7.2.9 Schedule Model 2

RQ 9: Do software program **unadjusted function points** relate to total agile development schedule?

Equation (8) predicts schedule for agile software development projects as a function of unadjusted function points and a dummy variable associated with scope type.

$$\mathbf{Schedule} = \mathbf{1.938}x\mathbf{UFP}^{\mathbf{0.2025}}x\ \mathbf{2.739}^{D1} \qquad (8)$$

Where,

Schedule = total final development months
UFP = Unadjusted Function Points
D1 = dummy variable associated with scope where full development was assigned a value of 1 and enhancement a value of 0

Table 17 provides the regression analysis report of the coefficient statistics, goodness-of-fit statistics, and analysis of variance for equation (8). The resulting equation is statistically significant and demonstrates that unadjusted function points and a scope type dummy variable are effective variables to estimate the schedule of agile software projects.

Table 17 Regression Analysis Results for Equation (8)

Coefficient Statistics Summary

Term	Coef	T-Statistic	P-value
Intercept	0.66	1.51	0.16
UFP	0.20	2.72	0.02
D1	1.01	5.49	0.00

Goodness-of-Fit Statistics

SE	R^2	$R^2_{(adj)}$	$R^2_{(pred)}$	MAD
0.27	88.14%	86.17%	80.83%	18.59%

Analysis of Variance

Source	DF	Sum of Sq.	Mean Sq.	F-stat
Regression	2	6.27	3.13	44.60
Residual	12	0.84	0.07	
Total	14	7.11		

Figure 18 shows the normal probability plot for Equation (8). The residuals approximate a straight line. This suggests that loglinear regression is valid for modeling schedule vs UFP along with scope type (dummy variable).

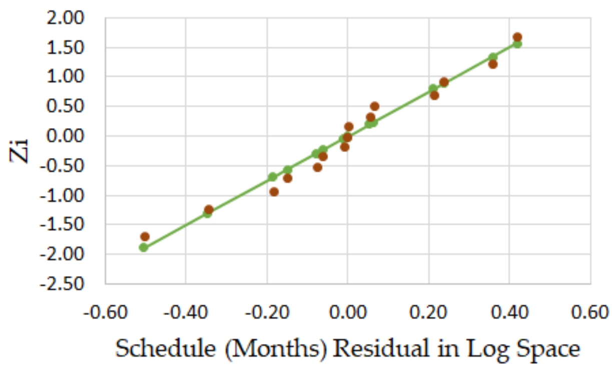

Figure 18 Normal probability Plot for Equation (8)

7.2.10 Schedule Model 3

RQ 10: Do **simple function points** relate to total agile development schedule?

Equation (9) predicts schedule for agile software development projects as a function of unadjusted function points and a dummy variable associated with scope type.

$$\textbf{Schedule} = 2.009x\textbf{SiFP}^{0.1923}x2.826^{D1} \qquad (9)$$

Where,

Schedule = total final development months
SiFP = Simple Function Points
D1 = dummy variable associated with scope where full development was assigned a value of 1 and enhancement was assigned a value of 0

Table 18 provides the regression analysis report of the coefficient statistics, goodness-of-fit statistics, and analysis of variance for equation (9). The resulting equation is statistically significant and demonstrates that simple function points and a scope type dummy variable are effective variables to estimate the schedule of agile software projects.

Table 18 Regression Analysis Results for Equation (9)

Coefficient Statistics Summary

Term	Coef	T-Statistic	P-value
Intercept	0.69	1.62	0.13
SiFP	0.19	2.69	0.02
D1	1.04	5.84	0.00

Goodness-of-Fit Statistics

SE	R^2	$R^2_{(adj)}$	$R^2_{(pred)}$	MAD
0.27	88.01%	86.02%	80.41%	18.35%

Analysis of Variance

Source	DF	Sum of Sq.	Mean Sq.	F-stat
Regression	2	6.26	3.13	44.06
Residual	12	0.85	0.07	
Total	14	7.11		

Figure 19 shows the normal probability plot for Equation (9). The residuals approximate a straight line. This suggests that loglinear regression is valid for modeling schedule vs SiFP along with scope type (dummy variable).

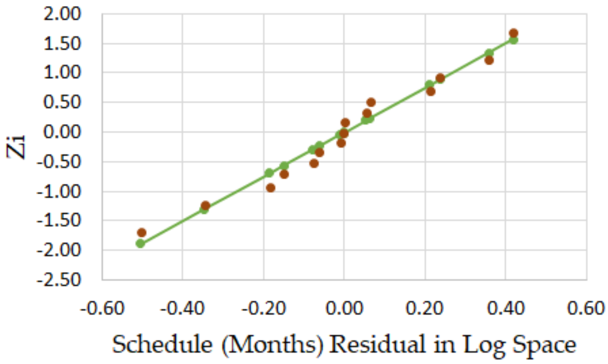

Figure 19 Normal Probability Plot for Equation (9)

7.2.11 Schedule Model Comparison

RQ 11: How do **functional requirements, UFP,** and **SiFP** rank as variables for predicting total development schedule for agile projects?

Table 19 compares the statistical significance of the schedule estimation models using three different software size measures and a scope dummy variable as predictors. The comparison of the results with a synopsis of the suggested ranking order of models is summarized below.

Table 19 Schedule Model Comparison

ID	Equation	$R^2_{(adj)}$	$R^2_{(pred)}$	MAD	Rank
7	$S = 2.685xREQ^{0.2135}2.718^{D1}$	86.5%	81.8%	18.8%	1
8	$S = 1.938xUFP^{0.2025}x\ 2.739^{D1}$	86.2%	80.8%	18.6%	2
9	$S = 2.009xSiFP^{0.1923}x2.826^{D1}$	86.0%	80.4%	18.4%	3

Based on the comparison, **functional requirements, UFP, and SiFP** are strong predicters to agile software development program schedule. When performing regression analysis using stories, story points, and issues as independent variables, each variable failed the statistical significance threshold of p-value ≤ 0.05.

7.3 Discussion of Results

As a result of the regression analysis performed on the software agile dataset, it was determined that **functional requirements, unadjusted function points, and simple function points** are the strongest predicters to both effort and schedule for agile software development programs. Stories, story points, and issues also had statistically significant effort models, however, these variables were not statistically significant for estimating schedule. Table 20 summarizes the nine statistically significant effort and schedule regression models in ranking order from most to least statistically significant.

Table 20 Best Effort and Schedule Models

ID	Model Equation	$R^2_{(adj)}$	$R^2_{(pred)}$	MAD
	Effort Models			
6	$E = 261.1xSiFP^{0.7708}x1.615^{D1}$	90.67%	86.25%	25.93%
5	$E = 189.5xUFP^{0.8747}$	89.28%	85.63%	31.61%
1	$E = 935.5xREQ^{0.882}$	88.77%	85.86%	31.28%
4	$E = 206.5xSTY\_PTS^{0.6842}$	83.12%	78.15%	32.68%
2	$E = 604.3xISSUES^{0.6879}$	69.34%	59.39%	51.55%
3	$E = 1365xSTORY^{0.6228}$	67.93%	59.04%	54.11%
	Schedule Models			
7	$S = 2.685xREQ^{0.2135}x2.718^{D1}$	86.45%	81.78%	18.83%
8	$S = 1.938xUFP^{0.2025}x2.739^{D1}$	86.17%	80.83%	18.59%
9	$S = 2.009xSiFP^{0.1923}x2.826^{D1}$	86.02%	80.41%	18.35%

With the regression output demonstrating statistical significance for these CERs and SERs, we recommend these for use in the cost community when estimating total effort and schedule of future agile programs categorized as automated information systems in the DHS context.

7.4 Model Usefulness

The effort and schedule models based on functional requirements or function points (unadjusted and simple) are program characteristics typically known early in a software program before contract award. Therefore, the models associated with these variables can be used for estimating software development effort and schedule early in the program such as cost estimates supporting earlier milestone decisions, analysis of alternatives (AoAs), and proposal evaluations.

On the other hand, user stories, story points, or issues are program characteristics typically known after contract award when the program is being executed and the product backlog becomes available. Therefore, the resulting models associated with these variables can be used for estimating software development effort later in the program such as lifecycle cost estimates supporting later milestone decisions and analyses supporting post-Initial Operating Capability (IOC). Finally, story points can be used as a secondary crosscheck to evaluate contractor performance.

To account for uncertainty of the CERs and SERs when using them in cost estimating application, uncertainty around the input variable and regression equation should be considered. Since all nine resulting models are log-linear derived using OLS in log-space, the output represents the median estimate, at the 50% confidence level. Applying OLS in log-space yields multiplicative lognormal uncertainty in unit space. Therefore, the uncertainty distribution shape around the log-linear regression is lognormal.

To assess the uncertainty of the regression equation, the confidence interval (CI) can be calculated, which conveys the error around the output. A common CI is calculated at the 95% confidence level. The general formula to calculate the CI is,

$$(PE - CV \; x \; SE, \qquad PE + CV \; x \; SE)$$

Where,

PE = Point Estimate
CV = Critical Value derived
SE = Standard Error

In the CI, $PE - CV \; x \; SE$ value represents the lower bound while $PE + CV \; x \; SE$ value represents the upper bound. Many statistical software packages will calculate a confidence interval based on the model's output (median), standard error (SE), specified confidence level (95%), sample size, and degrees of freedom (DF). Alternatively, the prediction interval can be calculated to model CER and SER output uncertainty. For more information on how the prediction interval is calculated, see the *Joint Agency Cost Estimating Relationship (CER) Development Handbook* [15].

7.5 Threats to Validity

Possible threats to the validity of the resulting effort and schedule models include internal, external, or constructive. A discussion of each threat is summarized below.

Threats to *internal* validity include the dataset timeframe from 2014 to 2021, which raises potential issues where earlier projects (2014-2018) were developed using agile processes tailored to fit the developer's need. It is likely that agile processes have evolved during the 7-year timeframe. The scope of this study covers programs that were classified as *agile*, perhaps loosely, and a focus on only a single development process. This poses a limitation to programs using a different software development process such as waterfall and may produce very different results.

Threats to *external* validity include differences in the way function points may be counted for different programs outside the agile dataset used for this study. During data normalization, the counting process for function points were developed by the same Certified Function Point Specialist, using either backlog, FRD and RTM. These are common artifacts in DHS and DoD acquisition. Other organizations may not have access to these artifacts to develop their function point counts and moreover, different function point counters may be generating the counts for the dataset.

The models presented in this paper proved to be effective in estimating total development hours and duration for agile projects reported at the release level for DHS and DoD. However, we cannot generalize beyond this group for several reasons. First, majority of the projects were developed using Scrum and SAFe. Second, the total effort includes other activities above and beyond those captured in mainstream software cost estimating models. Examples of elements captured in the total effort for our agile dataset included program management, systems engineering, training, security, testing, and operations.

Threats to *constructive* validity is the number of datapoints in the sample size of 18 agile projects. With a sample size this small, there is a threat to the statistical conclusions as they may be subject to overfitting and does not allow for detecting effects with greater power. To address this threat, a larger sample size is needed for confirmatory hypothesis testing.

8 Conclusion

The effort and schedule regression models and benchmark results in this paper offer the software cost estimating community a wider range of sizing measures to estimate future agile software programs in addition to being able to evaluate contractor proposals of agile software programs.

The goodness-of-fit metrics of the effort and schedule regression models add insight to the belief that functional requirements and function points (SiFP, UFP) are more effective predicters of agile software development effort than popular agile measures such as story points, stories, and issues. Additionally, we offer the software cost estimating community alternative sizing measures to estimate software development in lieu of the traditional SLOC and object points sizing measures that have been used for years but have proven to be inconsistent and difficult to obtain.

Functional stories (or requirements) and function points are also more appropriate in assessing cost and effort of agile projects from proposal evaluation through IOC, when mainstream agile sizing metrics are not available.

9 REFERENCES

[1] B. Boehm, C. Abts, W. Brown, S. Chulani, B. Clark, E. Horowitz, R. Madachy, D. Reifer, and B. Steece, Software Cost Estimation with COCOMO II, PrenticeHall, 2000.

[2] Chief Technology Officer Directorate, Office of the Chief Technology Officer, "Department of Homeland Security Agile Guidebook," 31 Jul. 2021.

[3] Dantas, Emanuel & Perkusich, Mirko & Dilorenzo, Ednaldo & Santos, Danilo & Almeida, Hyggo & Perkusich, Angelo. (2018). Effort Estimation in Agile Software Development: An Updated Review. International Journal of Software Engineering and Knowledge Engineering. 28. 1811-1831. 10.1142/S0218194018400302.

[4] Department of Defense, "Software Resource Data Report," Nov. 2020. Available at: https://cade.osd.mil/content/cade/files/csdr/dids/current/dd3026-1_2019.XLSX

[5] Department of Homeland Security, "*Agile Development and delivery for Information Technology*," Instruction 102-01-004, rev 01, 16 Apr. 2018. Available at: https://www.dhs.gov/sites/default/files/publications/agile-development-and-delivery-for-information-technology.pdf

[6] Department of Homeland Security, "Agile Development and delivery for Information Technology Instruction Manual 102-01-004-01," rev 00, 15 Jul. 2016.

[7] M. Fernández-Diego, E. R. Méndez, F. González-Ladrón-De-Guevara, S. Abrahão and E. Insfran, "An Update on Effort Estimation in Agile Software Development: A Systematic Literature Review," in IEEE Access, vol. 8, pp. 166768-166800, 2020, doi: 10.1109/ACCESS.2020.3021664.

[8] V. Kundra, U.S. Chief Information Officer, "25 Point Implementation Plan to Reform Federal Information Technology Management," Office of Management and Budget, The White House, 9 Dec. 2010. Available at:https://www.dhs.gov/sites/default/files/publications/digital-strategy/25-point-implementation-plan-to-reform-federal-it.pdf

[9] L. Lavazza and R. Meli, "An Evaluation of Simple Function Point as a Replacement of IFPUG Function Point," 2014 Joint Conference of the International Workshop on Software Measurement and the International Conference on Software Process and Product Measurement, 2014, pp. 196-206, doi: 10.1109/IWSM.Mensura.2014.28

[10] K. Mann and R. Hoang, "Functional Sizing of Agile Programs at U.S. Department of Homeland Security," MetricViews, Issue 1, IFPUG, Jul 2021. Available at: https://www.ifpug.org/wp-content/uploads/2021/07/IFPUG_July_MetricViews_2021.pdf

[11] W. Rosa, B. K. Clark, R. Madachy and B. Boehm, "Empirical Effort and Schedule Estimation Models for Agile Processes in the US DoD," in IEEE Transactions on Software Engineering, doi: 10.1109/TSE.2021.3080666.

[12] W. Rosa, R. Madachy, B. Clark and B. Boehm, "Early Phase Cost Models for Agile Software Processes in the US DoD," 2017 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM), Toronto, ON, 2017, pp. 30-37. doi:10.1109/ESEM.2017.10, available at: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8170082&isnumber=8169971

[13] TECOLOTE Inc. "Automated Cost Estimating Integrated Tool: CO$TAT," November 2020. Available at: https://www.aceit.com/aceit-suite-home/product-info/costat

[14] M. Usman, E. Mendes, F. Weidt, and R. Britto, "Effort estimation in Agile software development: a systematic literature review," In Proceedings of the 10th International Conference on Predictive Models in Software Engineering (PROMISE '14). ACM, New York, NY, USA, 2014, pp. 82-91.

[15] Naval Center for Cost Analysis, "Joint Agency Cost Estimating Relationship (CER) Development Handbook," 9 Feb 2018. Available at: https://www.ncca.navy.mil/references/CER_Dev_Handbook_Aug2018_Final.pdf