The Journal of Systems & Software 203 (2023) 111739

Contents lists available at ScienceDirect

The Journal of Systems & Software

journal homepage: www.elsevier.com/locate/jss

Data-driven agile software cost estimation models for DHS and DoD☆

Wilson Rosa [a,*], Sara Jardine [b]

[a] Alexandria, VA 22315, United States
[b] Galorath, Martinsburg, WV 25403, United States

ARTICLE INFO

Article history:
Received 10 November 2022
Received in revised form 26 April 2023
Accepted 29 April 2023
Available online 6 May 2023

Keywords:
Agile
Effort estimation
Function point
Story point
User story
Functional story

ABSTRACT

Problem: Since the Manifesto for Agile Software Development, the cost analysis community has struggled to find the most appropriate size measures for developing accurate agile software development cost estimates at an early phase to establish baseline budgets and schedules, and for the selection of competitive bidders. At this time, common agile sizing measures such as Story Points are not practical since these are reported months after contract award in the Department of Homeland Security (DHS) and Department of Defense (DoD). The problem is compounded with the lack of data to build estimation models for informed decisions.
Aims: The primary objective is to investigate how well two new size measures (Functional Story and Issues) accurately relate to total effort, and how these compare and rank against four popular software size measures (Story, Story Point, Unadjusted Function Points, and Simple Function Points). The second objective is to rank the six sizing measures based on how well each estimate software development effort at an early phase or after contract award.
Method: The experimental framework relied on an analysis of variance and goodness-of-fit tests to examine and compare the accuracy of effort estimation models using six competing size measures. The analysis is based on data from 17 agile projects implemented between 2014 to 2021.
Results: Our two new size measures (Functional Story, Issues) proved to be good predicters of total software development effort. Functional Story is better at predicting total effort at early phase than Function Points and Story. Functional Story and Function Points are better at predicting total effort after contract award than Story Point and Issues.
Conclusion: The DHS and DoD cost community can choose one or more of these estimation models to evaluate agile software development cost proposals or track agile developer's progress after contract award.

Published by Elsevier Inc.

1. Introduction

1.1. Problem statement

Despite the need to improve agile cost estimation practices across the United States Federal Government, there is yet no consensus on the best estimation methods and most appropriate size measures (Mann and Hoang, 2021). The primary contributor to this issue is the lack of empirical research to examine the effectiveness and accuracy across competing size measures based on the same cross-company dataset, assumptions, and research method. Without this information, the United States Department of Homeland Security (DHS) and Department of Defense (DoD) are unable to standardize cost estimation practices for Agile software development.

The goal of this study is to examine the direct effect of two new size measures and four popular size measures on estimating agile software development effort. This study will also compare and rank the accuracy of each of the six size measures as a predictor to agile software development effort.

1.2. Deficiencies in past studies

Past research studies on agile estimation have either used Story Point, Story, Use Case Point, Function Point, or COSMIC as the primary size measures. These studies, however, had several shortcomings:

- All previous studies have overlooked "Issues" and "Functional Story" as size measures.
- Majority used datasets from a single company.
- Vast majority constrained the dependent variable (effort) to pure software development activity alone and omitted other associated software development labor activities incurred by the contractor.

☆ Editor: Leandro L. Minku.
* Corresponding author.
E-mail addresses: wilymar04@cox.net (W. Rosa), alsaffar.sara@gmail.com (S. Jardine).

https://doi.org/10.1016/j.jss.2023.111739
0164-1212/Published by Elsevier Inc.

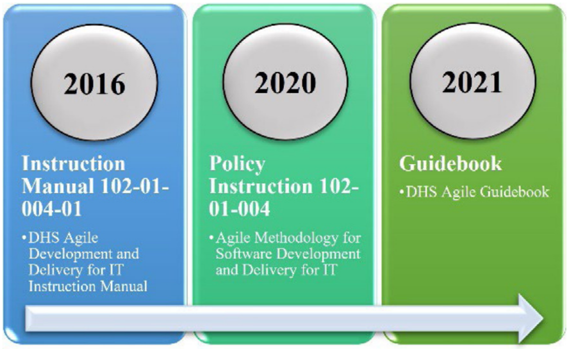
2016
Instruction Manual 102-01-004-01
•DHS Agile Development and Delivery for IT Instruction Manual
2020
Policy Instruction 102-01-004
•Agile Methodology for Software Development and Delivery for IT
2021
Guidebook
•DHS Agile Guidebook

Fig. 1. DHS Agile development policies.

- Vast majority of studies collected data at the sprint level with very few at the release level.
- None of the studies specified the source documents used for counting COSMIC or Function Point.
- Although several studies have examined Story and Function Point as size measures, none have captured total contract effort.

1.3. Significance of this study

This study helps improve the software cost estimating practice in five ways:

1. Breaks new ground by incorporating two new potential size measures, Functional Story and Issues, not previously addressed elsewhere.
2. Provides practical effort estimation models based on four size measures available at an earlier phase and can be used for developing estimates before or after contract award. Those four measures are:
    - Simple Function Point (SiFP),
    - Unadjusted Function Point (UFP),
    - Story, and
    - Functional Story
3. Shows how each of the six size measures accurately relates to total contract effort.
4. Examines whether size measures known at early phase are more accurately related to effort, than those typically known after contract award (i.e., Issues, Story Points).
5. Uses a cross-company dataset and captures total contract effort at the release level. The choice for using total effort is driven by the fact that most agile development contracts in DHS are Firm Fixed Price (FFP) or Time & Materials (T&M), and typically report effort at the total level instead of reporting by lower elements (software development, systems engineering, training, etc.).

2. Background

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

In 2016, the Under Secretary of Homeland Security for Management (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 (DHS, 2016). Five pilot programs were conducted to address challenges associated with IT program overruns and schedule delays, lack of program transparency, and poor requirements development and traceability. An Agile Acquisition Working Group was then stood up to effectively plan and implement the programs, as well as developing appropriate documentation to support program execution (DHS, 2016). The lessons learned from the pilots were used to develop and update DHS policies and procedures for executing these five programs and future agile software acquisitions. The resulting key policies and procedures (DHS, 2016, 2021) are shown in Fig. 1.

In the DHS and DoD, *Agile* is a software development methodology that uses an iterative approach to deliver solutions incrementally through close collaboration and frequent reassessment (DHS, 2021; DoD, 2020). Many different *Agile* approaches (Scrum, Scaled Agile Framework, SecDevOps, etc.) have been implemented to date within DHS (DHS, 2016, 2020, 2021). Since there is no specific set of DHS-approved methodologies, program managers are encouraged to determine the most appropriate *Agile* approaches for their program (DHS, 2021).

3. Literature review

This study reviewed numerous agile software development research papers conducted on effort estimation models. However, only three were related to this study by partially capturing the same size measures (4 out of 6) based on a company dataset. The two sections below compare the key similarities and differences between related works and this study.

3.1. Related works

Fernández-Diego et al. (2020) conducted a systematic literature review (SLR) to report on the state of effort estimation models and practices for agile software development; focusing on what methods, predictors, and datasets that were recently used for software size and effort estimation. Their study analyzed data extracted from 73 peer-reviewed papers. Of these papers, 61.6% (45 of 73) used Story Points, 13.7% (10 of 73) used COSMIC, and only 6.8% (5 of 73) used function points (UFP or SiFP) as the primary predictor for estimating agile project software development effort.

The work by Fernandez-Diego and colleagues is like this study since it confirms the direct effect of Story Point and function points on agile software development effort. Their work, however, differed from this study in three ways. First, their study only captured pure software development effort whereas our study captured the total effort incurred by the contractor agile teams, which also included pure software development. Secondly, the project datasets were from single-companies versus a cross-company dataset is used in this study. Lastly, user stories, issues, and Functional Story/requirements were not cited as predictors to software development effort in the research.

Rosa et al. (2022) introduced an approach for estimating agile software development effort and schedule using functional requirements. Thirty-six delivered cross-company agile projects were examined. The results revealed that functional requirements have a statistically significant effect on agile software development effort, while the model's prediction accuracy improved when the dataset was stratified by domain. Our work is like this study in that it uses a cross-company project dataset and evaluates functional requirements as an independent variable to

Table 1
Summary of related work and our study.

	Metric	Fernández-Diego et al. (2020)	Rosa et al. (2022)	Abrahamsson et al. (2011)	Our Study
Size Metric	Functional req.		X		
	Functional Story				X
	SiFP	X			X
	UFP	X			X
	Story			X	X
	Issue				X
	Story Point	X			X
Dataset Limitations	Cross-company?	no	yes	no	yes
	Number of projects?	nr	36	1	17
	Sample size (n)	71	36	1,338	17
	Reporting level?	Sprint	nr	Sprint	Release
	Data source listed?	no	yes	no	yes
	Data inspected?	no	no	no	yes
	Same app domain?	nr	no	nr	Yes
	Total contract labor?	no	no	no	yes

yes = positive response; no = negative response; nr = not reported.

predict software development effort. However, their work differs from this study in three ways. First, the functional requirements in their study were obtained from the software requirements specification (SRS), where our study obtained functional requirements primarily from the product's backlog. Second, their effort's regression models only captured pure software development activities whereas our study captures total effort incurred by the contractor agile teams, which also included pure software development. Lastly, their study did not address agile software measures nor function points as predicters to software development effort.

Abrahamsson et al. (2011) proposed a method for predicting agile software development effort based on user stories. The study included two case studies. The first case study (A) contained 1,325 user stories from an Italian company and the second case study (B) contained 13 user stories from a non-profit research organization. The regression results showed poor accuracy for both cases where case (A) had a mean magnitude of relative error (MMRE) of 90% and case (B) had a MMRE of 66%. Case study (B) had better accuracy because its user stories were written in a structured way and with a higher level of detail than case study (A). The authors concluded that effort estimation models based on user stories worked reasonably well if stories were written in a structured way. Our work is like this study in that it examines the direct effect of user stories on agile software development effort. However, their work differs from ours in three ways: it is based on single company, only captures pure software development effort, and data was collected at the sprint level.

3.2. Contrast between our study and related works

The main contrast between our study and related works are summarized in Table 1. Prior agile estimation research studies did not specifically consider Issues or Functional Story as primary measures for estimating effort. Previous research efforts were constrained to pure software development effort, while our study captured the total software development effort incurred by the contractor's agile teams. Most past studies collected data at the sprint level, whereas our data was collected at the release level. Also, data from previous studies were company self-reported and were not inspected for quality and consistency.

4. Research approach

4.1. Research questions

Today, DHS and DoD cost communities are still unable to determine the most effective estimation models for agile software development due to the lack of empirical studies on the direct effect of competing size measures in relation to development effort using a relevant project dataset. The following two research questions (RQ) will address this main problem:

RQ 1: **How does each of the new and popular size measures accurately relate to total contract software development effort?** This research question attempts to evaluate how each of the six size measures are statistically significant in relation to agile software development effort when using the same cross-company dataset and same application domain. The experimental framework will rely on analysis of variance and goodness-of-fit tests to examine the effectiveness and reliability of each of the six resulting effort estimation models.

RQ 2: **How do the six size measures compare and rank as accurate predictors of total contract software development effort?** This question is an extension of RQ 1. The purpose is to rank the most accurate size measure for predicting agile software development effort. The analysis will rely on the Mean Magnitude of Relative Error (MMRE) as the indicator of accuracy. The lower the MMRE, the higher the model's accuracy based on a particular size measure.

4.2. 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 17 agile projects across eleven different companies, delivered for the DHS (14) and DoD (3) from years 2014 to 2021. The DHS and DoD projects in this study follow the same agile processes and delivery strategies in accordance with OMB (Kundra, 2010).

4.3. Data collection

The data collected for each agile project included actual effort, final size, agile methods, and project characteristics. The data was extracted from official documents including monthly contractor invoices, product backlog, functional requirements document (FRD), requirements traceability matrix (RTM), acquisition documents, and agile core metrics. The documents shown in Fig. 2 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

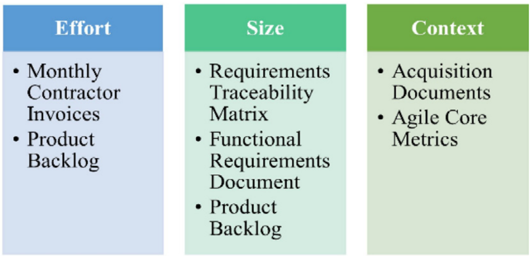
Effort
• Monthly Contractor Invoices
• Product Backlog
Size
• Requirements Traceability Matrix
• Functional Requirements Document
• Product Backlog
Context
• Acquisition Documents
• Agile Core Metrics

Fig. 2. Data sources.

Table 2
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

Table 3
Variable name and definition.

Variable	Type	Definition
Effort	Dependent	Actual hours associated to development activities listed in Table 2
Functional Stories (REQ)	Independent	A subset of user stories aligned to functional requirements. Describes what the software does (Abrahamsson et al., 2011).
Issue (ISSUES)	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, enhancements.
Story (STY)	Independent	A feature unit of business value that can be estimated and tested. Describes work that must be done to create and deliver a product.
Story Point (STY_PTS)	Independent	A unit of measure to express the relative effort of a 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	Sizing method that requires the identification of elementary processes and logic files to approximate a function point count (Boehm et al., 2000).

Table 4
Independent variable comparison.

Element	Variable					
	Functional story	Story point	Issue	Story	SiFP	UFP
Element Type:						
Bug		x	x			
Task		x	x			
Defect		x	x			
Story (Functional)	x	x	x	x	x	x
Story (Non-functional)		x	x	x		
Data Sources:						
Product Backlog	x	x	x	x	x	x
FRD	x			x	x	x
RTM	x			x	x	x
Element Multiplier						
Fibonacci sequence		x				
IFPUG* Multiplier(s)					x	x

*IFPUG = International Function Point Users Group.

is maintained by the contractor's agile teams and government product owner.

The effort hours in this study captures 11 major cost elements incurred by the contractor's agile development teams in accordance with the DHS IT Work Breakdown structure (WBS). These major IT WBS cost elements are also applicable to the DoD programs captured in this study and are identified in Table 2.

The decision to report total effort as opposed to pure software development alone is driven by the fact that most agile development contract types in DHS are FFP or T&M, and therefore, these contracts generally do not break down effort into major cost elements as would traditional cost-plus contracts.

4.4. Variables in the study

The variables and associated definitions used in this study to develop and evaluate effort models are shown in Table 3.

Table 4 shows the similarities and differences between Functional Story and other size measures in terms of element types included, data sources, and whether a multiplier is needed to adjust the raw count. The Functional Story count includes a single element (Functional), does not require a complexity multiplier, and can be obtained from documents available at an early program phase such as an RTM, FRD, or after contract award using the Product Backlog.

4.5. Data normalization

The data normalization process included obtaining agile size measures, counting Functional Story (requirements), and deriving Function Points. The steps of the normalization process will be discussed in this section.

4.5.1. Estimating agile size

The team used a repeatable method to estimate agile size for each project in the dataset. Below is an outline of the four steps followed to determine the total number of Stories, Issues, and Story Points from an agile project's product backlog. Table 5 provides an example excerpt derived from the product backlog output the team used to demonstrate the issue and story point counting process.

Step 1: Determine 100% Complete Issues. Find the column titled, *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 5
Example of Agile backlog items.

Issue ID	Status	Description	Issue type	Story points
0001	Done	As a <user> I need to manually initiate 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**

Table 6
Example functional story.

Issue description	Issue type	Category
As a <user> I need to manually initiate 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

4.5.2. Counting functional story

The team also used a specific counting method to determine Functional Story. Below is an outline of the three steps to determine the total number of Functional Stories from an agile project's product backlog. Table 6 provides an example excerpt derived from the product backlog output the team used to demonstrate the Functional Story counting process.

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

Step 2. Create a column titled, *Category*, then categorize each story as either functional or non-functional. This step is performed by a Certified Function Point Specialist.

Step 3. In the column titled, *Category*, count the rows marked as functional. This example results in a total count of **3 Functional Story** (Table 6).

4.5.3. Computing function points

Once the Functional Story counts 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) team. After the CFPS team completed their counts for each project, the team met to resolve questions about unclear requirements to validate the analysis. Typical questions included which Functional Stories constituted the functions versus logical files, since there was often interpretation differences amongst the CFPS team about what the software is supposed to do, which could lead to computing the wrong UFP/SiFP counts across the Functional Story list. Other questions to reconcile the final count included identifying and removing duplicate Functional Stories from the list.

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. The higher the R^2, the more variability is explained by the model.
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. The higher the R^2 (adj), the more variability is explained by the model.
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. The higher the R^2 (pred), the more variability is explained by the model.
P-value	The probability value of obtaining results at least as extreme as the observed results of a statistical hypothesis test, assuming the null hypothesis is correct. The lower the *p*-value for each independent variable, the more statistically significant the variable is in predicting the dependent variable.
SEE	Standard Error of the Estimate (SEE) is the difference between observed and estimated effort. SEE is to linear models as standard deviation is to sample means. The lower the SEE, the better the regression model fits to the dataset.
F-test	F-test is the square of the equivalent t-test. The larger the value, the smaller the probability that difference could occur by chance.
MMRE	Mean Magnitude of Relative Error (MMRE) is an indicator of a model's accuracy. The lower the MMRE, the higher the accuracy of the model.

4.6. Model selection and validation

Each software development effort equation model was developed using the entire dataset (17) using the Automated Cost Estimating Integrated Tools (ACEIT) Cost Analysis Statistical Package (CO$TAT) (Tecolote Inc., 2020). The following goodness-of-fit measures in Table 7 were used for examining the reliability, accuracy, and fit of each model.

The log-normal equation form using the ordinary least squares (OLS) regression method (Tecolote Inc., 2020) was determined to be appropriate for all effort models in this study since the resulting normal probability plots 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 inferred that each independent variable was highly correlated to effort. The high adjusted R^2 and low MMRE characterized each model's accuracy, while the data points fit the equation line well. The marginal difference between adjusted and predicted R^2 also suggests that each model predicts new observations just as well as fitting the existing data points. Since the sample dataset only included 17 datapoints, only single variable regression was performed.

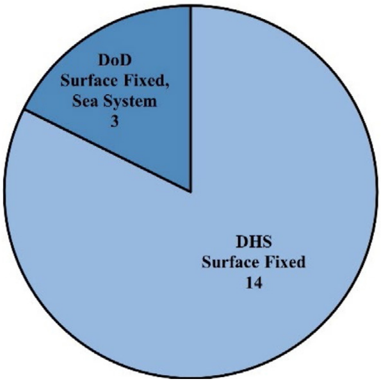

Fig. 3. Operating environment.

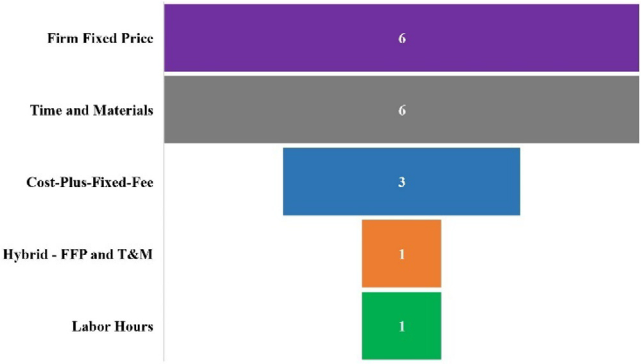

Fig. 4. Contract type.

5. Data analysis

The entire dataset in this study represents projects categorized as *Automated Information Systems* delivered from 2014 to 2021. Thirteen of the 17 projects were hosted on the cloud, while the remaining four projects were hosted on-premises. Of the 13 cloud-hosted projects, 12 used Amazon Web Services (AWS). All DHS projects were from *Surface Fixed* operating environments, while the DoD projects were from *Surface Fixed*, *Sea System* operating environments. *Surface fixed* is software at a fixed site while *surface fixed, sea system* is software installed on a surface or underwater boat or ship (Rosa et al., 2022). Fig. 3 shows the project counts by agency and operating environment.

Fig. 4 shows the project counts by contract type, where most agile projects (12) utilized Firm Fixed Price and Time and Material contract types.

The distribution of Agile frameworks is presented in Fig. 5. The Hybrid Agile projects all used the Scaled Agile Framework (SAFe).

The distribution of the Agile team approach is presented in Fig. 6. Of the 17 projects, ten followed SecDevOps while the remaining seven were DevOps or development only (DevOnly).

Descriptive statistics provide a summary of the characteristics of the dataset. The dependent variable was *Effort Hours*. Fig. 7 is a box-and-whisker plot of the descriptive statistics for Effort Hours. The values shown in the box-and-whisker plot include the first quartile (bottom line), median (line inside), and third quartile (upper line). The median effort hours for the dataset is 42,631 h.

The descriptive statistics for the six size measures (independent variables) in the agile dataset are presented in Table 8. For each variable, the minimum (Min), mean, median, maximum (Max), and standard deviation (StdDev) values are shown.

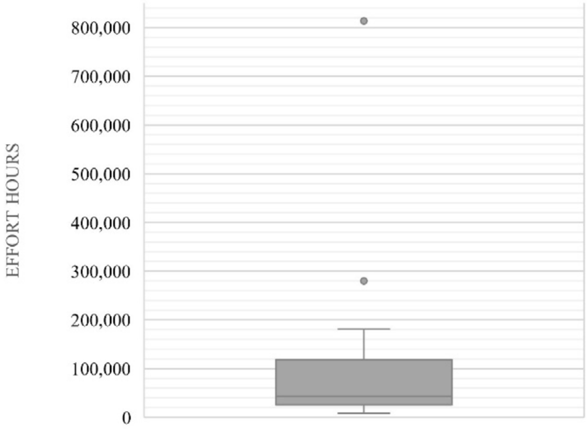

Fig. 5. Effort hours Box-and-Whisker plot.

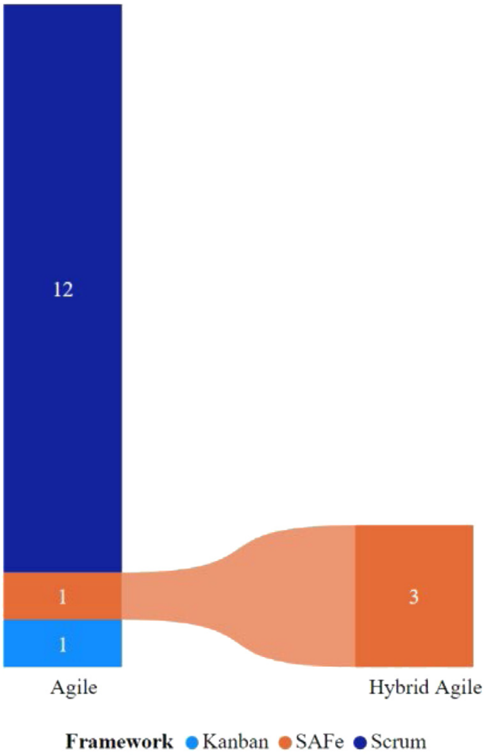

Fig. 6. Agile framework.

Table 8
Independent variable descriptive statistics.

Size Measure	Min	Mean	Median	Max	StdDev
Functional Story	16	241	87	1,881	450
Issues	75	1,456	1,043	5,744	1,565
Stories	27	999	430	4,964	1,289
Story Points	301	5,301	2,915	24,492	6,331
UFP	94	1,613	706	10,712	2,697
SiFP	86	1,481	642	9,628	2,411

6. Results

This section provides the model results organized by the two research questions. A discussion of the key findings and comparison of these findings to related works is also provided.

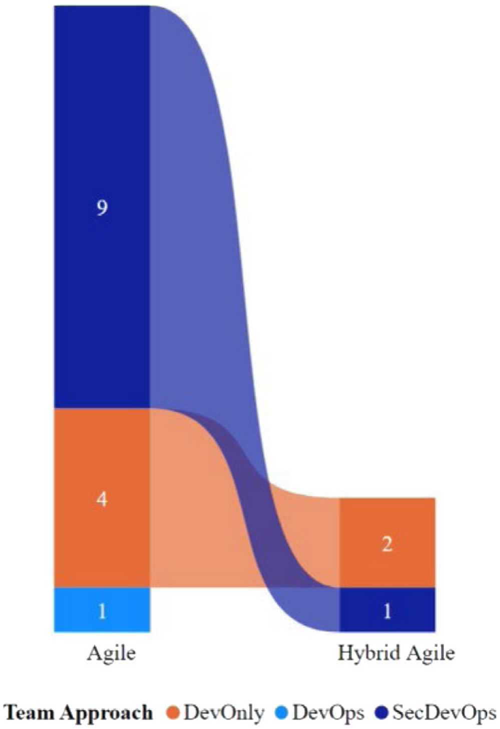

Fig. 7. Agile team approach.

6.1. Results for research question 1

This section provides the results associated with **RQ 1: How does each of the new and popular size measures accurately relate to total contract software development effort?**

The resulting six effort models are described in paragraphs below. The models are applicable to DHS and DoD agile software project sizes ranging approximately 20 to 5,000 stories, ten to 2,000 Functional Story, 80 to 11,000 Function Points, and a peak staff between nine to 200 Full-Time Equivalents (FTE).

6.1.1. Model 1

Eq. (1) predicts effort for agile software development projects as a function of Functional Story.

$$Effort = 558.4 \times REQ^{0.9696} \tag{1}$$

Where,

Effort = total final development hours

REQ = total Functional Story 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 Eq. (1). The high t-statistic and low *p*-value suggest that Functional Story is strongly correlated to total effort. The small difference (3%) between adjusted and predicted R^2 suggest the model predicts new observations as well as it fits the existing data. An adjusted R^2 of 90% and MMRE of 29% signifies that Functional Story is a good predictor of the total effort.

Fig. 8 shows the normal probability plot for Eq. (1). The residuals are close to the straight line, suggesting that loglinear regression is valid for modeling effort vs Functional Story.

6.1.2. Model 2

Eq. (2) predicts effort for agile software development projects as a function of Issues.

$$Effort = 340.1 \times ISSUES^{0.7637} \tag{2}$$

Table 9
Regression analysis results for Eq. (1).

Coefficient Statistics Summary				
Term	Coef	T-Statistic	P-value	
Intercept	6.33	16.44	0.00	
REQ	0.97	12.16	0.00	
Goodness-of-Fit Statistics				
SE	R^2	R$^2_{(adj)}$	R$^2_{(pred)}$	MMRE
0.37	90.80%	90.18%	87.95%	28.61%
Analysis of Variance				
Source	DF	Sum of Sq.	Mean Sq.	F-stat
Regression	1	19.79	19.79	147.97
Residual	15	2.01	0.13	
Total	16	21.80		

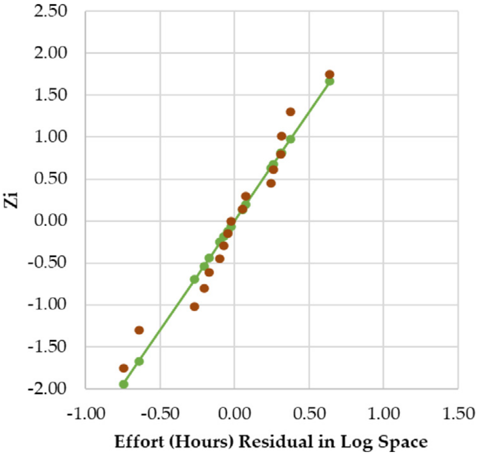

Fig. 8. Normal probability plot for Model 1.

Table 10
Regression results for Eq. (2).

Coefficient Statistics Summary				
Term	Coef	T-Statistic	P-value	
Intercept	5.83	7.96	0.00	
ISSUES	0.76	7.03	0.00	
Goodness-of-Fit Statistics				
SE	R^2	R$^2_{(adj)}$	R$^2_{(pred)}$	MMRE
0.58	76.73%	75.18%	68.40%	48.34%
Analysis of Variance				
Source	DF	Sum of Sq.	Mean Sq.	F-stat
Regression	1	16.72	16.72	49.45
Residual	15	5.07	0.34	
Total	16	21.80		

Where,

Effort = total final development hours

ISSUES = sum of total stories, bugs, tasks, enhancements, or any other tickets

Table 10 provides the regression analysis report of the coefficient statistics, goodness-of-fit statistics, and analysis of variance for Eq. (2). The high t-statistic and low *p*-value suggest that Issues is strongly correlated to total effort. The small difference (6%) between adjusted and predicted R^2 suggest the model predicts new observations as well as it fits the existing data. An adjusted R^2 of 75% and MMRE of 48% signifies that Issue is a moderate predictor of the total effort.

Fig. 9 shows the normal probability plot for Eq. (2). The residuals approximate a straight line, suggesting that loglinear regression is valid for modeling effort vs Issues.

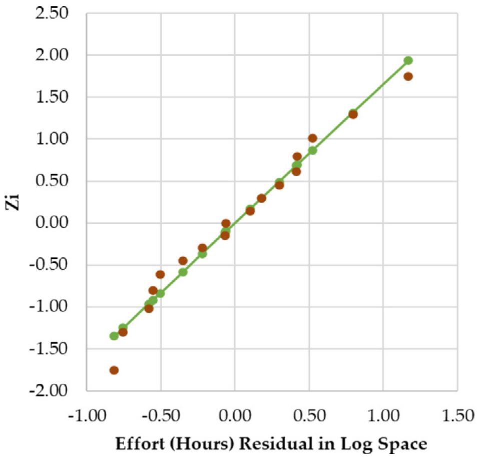

Fig. 9. Normal probability plot for Model 2.

Table 11
Regression results for Eq. (3).

Coefficient Statistics Summary				
Term	Coef	T-Statistic	P-value	
Intercept	6.85	9.55	0.00	
STY	0.67	5.78	0.00	
Goodness-of-Fit Statistics				
SE	R^2	$R^2_{(adj)}$	$R^2_{(pred)}$	MMRE
0.67	69.00%	66.93%	58.43%	55.34%
Analysis of Variance				
Source	DF	Sum of Sq.	Mean Sq.	F-stat
Regression	1	15.04	15.04	33.38
Residual	15	6.76	0.45	
Total	16	21.80		

6.1.3. Model 3

Eq. (3) predicts effort for agile software development projects as a function of Stories.

$$Effort = 939.6 \times STY^{0.665} \tag{3}$$

Where,
Effort = total final development hours
STY = total Stories from backlog in JIRA

Table 11 provides the regression analysis report of the coefficient statistics, goodness-of-fit statistics, and analysis of variance for Eq. (3). The high t-statistic and low *p*-value suggest that Story is strongly correlated to total effort. The small difference (8.5%) between adjusted and predicted R^2 suggest the model predicts new observations as well as it fits the existing data. An adjusted R^2 of 67% and MMRE of 55% signifies that Story is a moderate predictor of the total effort.

Fig. 10 shows the normal probability plot for Eq. (3). The residuals approximate a straight line, suggesting that loglinear regression is valid for modeling effort vs Story.

6.1.4. Model 4

Eq. (4) predicts effort for agile software development projects as a function of Story Points.

$$Effort = 318.4 \times STY\_PTS^{0.6313} \tag{4}$$

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 Eq. (4). The high t-statistic and low *p*-value suggest that Story Points is strongly correlated to total effort. The small difference (5%) between adjusted and predicted R^2 suggest the model predicts new observations as well as it fits the existing data. An adjusted R^2 of 80% and MMRE of 38% signifies that Story Point is a good predictor of the total effort.

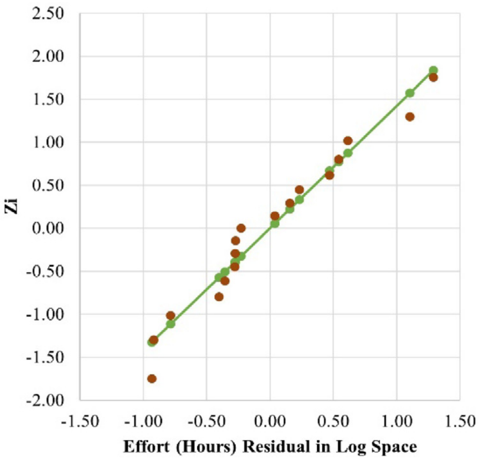

Fig. 10. Normal probability plot for Model 3.

Table 12
Regression results for Eq. (4).

Coefficient Statistics Summary				
Term	Coef	T-Statistic	P-value	
Intercept	5.76	8.85	0.00	
STY_PTS	0.63	7.71	0.00	
Goodness-of-Fit Statistics				
SE	R^2	$R^2_{(adj)}$	$R^2_{(pred)}$	MMRE
0.43	80.92%	79.56%	74.94%	37.72%
Analysis of Variance				
Source	DF	Sum of Sq.	Mean Sq.	F-stat
Regression	1	11.23	11.23	59.37
Residual	14	2.65	0.19	
Total	15	13.88		

Fig. 11 shows the normal probability plot for Eq. (4). The residuals approximate a straight line, suggesting that loglinear regression is valid for modeling effort vs Story Points.

6.1.5. Model 5

Eq. (5) predicts effort for agile software development projects as a function of Unadjusted Function Points.

$$Effort = 122.7 \times UFP^{0.9204} \tag{5}$$

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 Eq. (5). The small difference (3%) between adjusted and predicted R^2 suggest the model predicts new observations as well as it fits the existing data. An adjusted R^2 of 88% and an MMRE of 32% signifies that UFP is a good predictor of the total effort.

Fig. 12 shows the normal probability plot for Eq. (5). The residuals approximate a straight line, suggesting that loglinear regression is valid for modeling effort vs UFP.

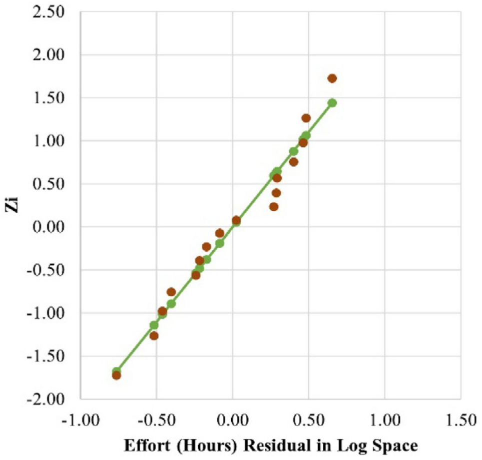

Fig. 11. Normal probability plot for Model 4.

Table 13
Regression analysis results for Eq. (5).

Coefficient Statistics Summary Term	Coef	T-Statistic	P-value	
Intercept	4.81	8.50	0.00	
UFP	0.92	10.89	0.00	
Goodness-of-Fit Statistics SE	R^2	$R^2_{(adj)}$	$R^2_{(pred)}$	MMRE
0.40	88.77%	88.02%	84.81%	32.22%
Analysis of Variance Source	DF	Sum of Sq.	Mean Sq.	F-stat
Regression	1	19.35	19.35	118.53
Residual	15	2.45	0.16	
Total	16	21.80		

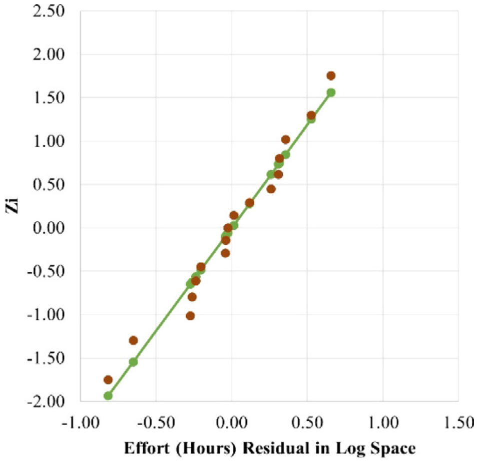

Fig. 12. Normal probability plot for Model 5.

6.1.6. Model 6

Eq. (6) predicts effort for agile software development projects as a function of Simple Function Points.

$$Effort = 110.1 \times SiFP^{0.9426} \quad (6)$$

Where,
Effort = total final development hours
SiFP = total Simple Function Points

Table 14
Regression analysis results for Eq. (6).

Coefficient Statistics Summary Term	Coef	T-Statistic	P-value	
Intercept	4.70	8.65	0.00	
SiFP	0.94	11.53	0.00	
Goodness-of-Fit Statistics SE	R^2	$R^2_{(adj)}$	$R^2_{(pred)}$	MMRE
0.38	89.86%	89.19%	85.62%	30.04%
Analysis of Variance Source	DF	Sum of Sq.	Mean Sq.	F-stat
Regression	1	19.59	19.59	132.95
Residual	15	2.21	0.15	
Total	16	21.80		

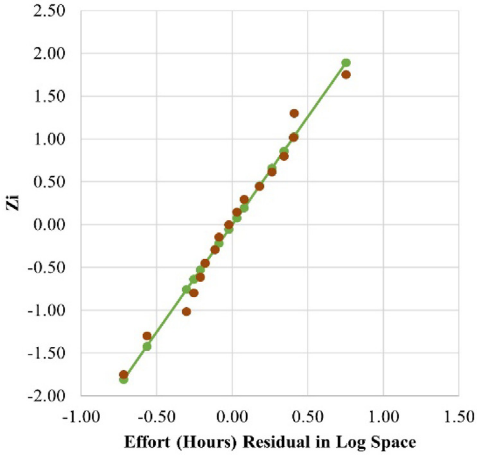

Fig. 13. Normal probability plot for Model 6.

Table 14 provides the regression analysis report of the coefficient statistics, goodness-of-fit statistics, and analysis of variance for Eq. (6). The small difference (4%) between adjusted and predicted R^2 suggest the model predicts new observations as well as it fits the existing data. An adjusted R^2 of 89% and an MMRE of 30% signifies that SiFP is a good predictor of the total effort.

Fig. 13 shows the normal probability plot for Eq. (6). The residuals approximate a straight line, suggesting that loglinear regression is valid for modeling effort vs SiFP.

6.2. Results for research question 2

This section provides the results associated with **RQ 2: How do the six size measures compare and rank as accurate predictors of total contract software development effort?**

Table 15 compares the statistical significance of the resulting effort models using six different software size predictors. The comparative results with a synopsis of the suggested ranking order of the models are summarized below. The models were compared and ranked according to the resulting MMRE, from lowest to highest. The precedence to use the lowest MMRE to rank the models is substantiated by literature (Abrahamsson et al., 2011; Fernández-Diego et al., 2020; Rosa et al., 2022). A low MMRE is indicative of high model predictive power whereas a high MMRE indicates low predictive power.

The most accurate predicters in ranking order from highest to lowest are Functional Story, Simple Function Point, Unadjusted Function Point, Story Point, Issue, and Story. After contract award, all six drivers can be obtained from the Product Backlog, which is continuously refined throughout the project's lifecycle.

Table 15
Model comparison.

Model	Independent Variable	R^2(adj)	R^2 (pred)	MMRE	Rank
(1)	Functional Story	90%	88%	29%	1
(2)	Issue	75%	68%	48%	5
(3)	Story	67%	58%	55%	6
(4)	Story Point	80%	75%	38%	4
(5)	UFP	88%	85%	32%	3
(6)	SiFP	89%	86%	30%	2

Table 16
Comparison of study results.

Effort driver	Metric	Fernández-Diego et al. (2020)	Rosa et al. (2022)	Abrahamsson et al. (2011)	Our study
Functional Story	MMRE				28%
	R^2				90%
Software Req.	MMRE		45%		
	R^2		79%		
SiFP	MMRE	66%			30%
	R^2	nr			89%
UFP	MMRE	58%			32%
	R^2	nr			88%
Story	MMRE			66%	55%
	R^2			nr	66%
Issue	MMRE				48%
	R^2				75%
Story Point	MMRE	84%	66%		37%
	R^2	nr	nr		79%

nr = not reported.

6.3. Discussion of results

Table 16 provides an informative comparison of the results from our present study and other related works. The most significant findings from our study are also discussed below.

Our proposed size measure, Functional Story, revealed to be a very effective predicter of total software development effort, as evidenced by the high accuracy and high model fit. This finding is consistent with Rosa et al. (2022) claiming that a size measurement only containing the "base functional component" from a software requirements list is effective at predicting effort. Certainly, differences in the measurement criteria could contribute to the MMRE differences between the present study and that of Rosa et al. (2022). For example, the criteria used by Rosa et al. (2022) measures "system shall" statements from a non-agile source document titled, "System Requirements Specification".

From the model comparison in Table 15, Functional Story is the best predicter of effort, as evidenced by the lowest MMRE (28%) and highest adjusted R^2 (90%). The comparison also reveals that Functional Story performs slightly better than the two Function Point measures (UFP, SiFP). This finding supports the notion that practitioners should use Functional Story as the preferred size measure throughout the agile program lifecycle since it has the highest predictive power, it is easier to obtain, and takes less time to count than Function Points or Story Points.

Our second proposed size measure, "Issue", revealed to be a moderate predictor of total effort, as indicated by the adjusted R^2(75%) and other statistical significance metric results. This finding is a breakthrough since none of the other related research work (Fernández-Diego et al., 2020; Rosa et al., 2022; Abrahamsson et al., 2011) explored "Issue" as an effort predicter. However, the resulting MMRE (48%) is an indication of a moderate predictive power. This could be attributed to the fact that the "Issue" counts contain Story, Bugs, Tasks, and other element types, while we also assumed the same complexity for all issue types during the counting process. A future study should adjust the "Issue" count based on the element type. Practitioners can use "Issue" size measure to predict software development effort during release planning or whenever a Product Backlog is available. Furthermore, a size growth adjustment factor is highly recommended since the "Issue" count continuously changes from contract start through full operational capability.

The results proved that SiFP and UFP are strong predicters of total software development effort as evidenced by the high adjusted R^2 (88%–89%) and low MMRE (30%–32%). This finding invalidates literature (Fernández-Diego et al., 2020) that substantiates that SiFP and UFP have low predictive power to software development effort. The assertion from literature is inconclusive as effort was collected at the sprint level from a single project, did not specify the source document for counting function points, and did not indicate the activities included in total software development hours. Practitioners can use either of the two Function Point size measures to estimate software development effort after contract award since their predictive power is much stronger than Story Point and Issue, which are typically known after initial operational capability.

Although Story, as a size measure, proved to have a higher adjusted R^2 of 67%, the resulting MMRE (55%) is an indication of its moderate predictive power to total effort. This finding is consistent with literature (Abrahamsson et al., 2011) that substantiates that Story performs well as a predicter when written in a structured way. The moderate MMRE (55%) from our analysis could have been attributed to the fact that the Story count contains functional, quality, and technical component types, while we assumed the same complexity for each story type. A future study should adjust Story counts based on the component types. Despite showing the lowest predictive power among the six size measures, practitioners can still use Story to predict software development effort since it can be obtained throughout the implementation lifecycle and the count is less volatile than Story Point and Issues, which are continuously changing from contract start to end.

This study also revealed that effective effort estimation models can be built when considering the following experimental design constraints:

1. Effort data reported at the total contract level
2. Projects are of the same application domain
3. Effort and size collected at the release level
4. Projects are from cross-companies
5. Effort and size data analyzed at the release level
6. Size measures counted and validated by same team

6.4. Threats to validity

Possible threats to the validity of the resulting effort 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 also 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 was performed by the same Certified Function Point Specialist, using either product backlog, FRD, or the RTM. While these are common artifacts in DHS and DoD acquisition, other organizations may not have access to these artifacts to develop function point counts and moreover, different function point counters may be generating the counts for the dataset.

The models presented in this study proved to be effective in estimating total development effort hours for agile projects reported at the release level for DHS and DoD. However, we cannot generalize beyond this group for several reasons. First, most of the projects were developed using Scrum and SAFe. Second, the total effort includes other activities 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.

The threat to constructive validity is the limited number of datapoints in the sample size of 17 agile projects. With a sample size this small, there is a threat to the statistical conclusions drawn in this study, 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.

7. Conclusion

7.1. Primary findings

Functional Story is an effective predicter of software development effort, as it is easier to obtain and takes less time to count than Function Points. The regression model results add insight into the belief that Functional Story and Function Points (SiFP, UFP) are better predicters of agile software development effort than the more recently popular agile measures such as Story Points, Stories, and Issues. Functional Story and Function Points are also more appropriate when assessing the cost of agile projects from proposal evaluation through first delivery, when mainstream agile sizing metrics are not always readily available.

The findings also support the notion that the size measures (i.e., Functional Story, Function Points) known to us before the agile project started, are far better predicters of total effort than the size measures (i.e., Story Point, Issues) known to us after contract award.

7.2. Future work

Our future work will examine effort estimation models using high-level software development size measures such as Epic. In the DHS context, an Epic is an initial set of high-level requirements, which will be progressively decomposed into user stories (functional and non-functional), whereas a Function Story is a single story categorized as Functional.

The regression models in this paper did not examine the development team's experience and methods used in drafting number of stories, issues, and story points. This poses a threat to external validity as these measures are highly "team" dependent and may be counted differently across projects. A future study should adjust the number of stories, issues, or story points based on development team's experience and methods used in drafting these counts. The team's experience and counting methods should be added as a categorical variable or size adjustment.

Although the dataset used in this study came from private companies developing software for DHS and DoD, the resulting models may not be generalizable to other government or private sector projects. A future study should develop similar regression models using data from private companies developing software for the commercial products. Another study can also be conducted using similar research methods based on a different dataset from other government agencies.

CRediT authorship contribution statement

Wilson Rosa: Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Project administration, Software, Supervision, Validation, Visualization, Writing – original draft, Writing – review & editing. **Sara Jardine:** Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Project administration, Software, Supervision, Validation, Visualization, Writing – original draft, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The data that has been used is confidential

Acknowledgment

The contents of this paper reflect the views of the authors and are not necessarily endorsed by the Department of Homeland Security.

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

References

Abrahamsson, P., Fronza, I., Moser, R., Vlasenko, J., Pedrycz, W., 2011. Predicting development effort from user stories. In: 2011 International Symposium on Empirical Software Engineering and Measurement. pp. 400–403. http://dx.doi.org/10.1109/ESEM.2011.58.

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

Department of Defense, 2020. Software resource data report. https://cade.osd.mil/content/cade/files/csdr/dids/current/dd3026-1_2019.XLSX.

Department of Homeland Security, 2016. Agile development and delivery for information technology instruction manual 102-01-004-01, rev 00. https://dhsconnect.dhs.gov/org/comp/mgmt/ocio/cto/Pages/Agile.aspx.

Department of Homeland Security, 2020. Agile methodology for software development and delivery for information technology, instruction 102-01-004, rev 02. https://dhsconnect.dhs.gov/org/comp/mgmt/ocio/cto/Pages/Agile.aspx.

Department of Homeland Security, Chief Technology Officer Directorate, Office of the Chief Technology Officer, 2021. Department of Homeland Security Agile Guidebook. https://dhsconect.dhs.gov/org/comp/mgmt/ocio/cto/Pages/Agile.aspx.

Fernández-Diego, M., Méndez, E.R., González-Ladrón-De-Guevara, F., Abrahão, S., Insfran, E, 2020. An update on effort estimation in agile software development: A systematic literature review. IEEE Access 8, 166768–166800. http://dx.doi.org/10.1109/ACCESS.2020.3021664.

Kundra, V., 2010. 25 Point Implementation Plan to Reform Federal Information Technology Management. The White House. https://www.dhs.gov/sites/default/files/publications/digital-strategy/25-point-implementation-plan-to-reform-federal-it.pdf.

Mann, K., Hoang, R., 2021. Functional sizing of Agile programs at U.S. department of homeland security. Metricviews, Issue 1. IFPUG. https://www.ifpug.org/wp-content/uploads/2021/07/IFPUG_July_MetricViews_2021.pdf.

Rosa, W., Clark, B.K., Madachy, R., Boehm, B., 2022. Empirical effort and schedule estimation models for agile processes in the US DoD. IEEE Trans. Softw. Eng. http://dx.doi.org/10.1109/TSE.2021.3080666.

Tecolote Inc., 2020. Automated cost estimating integrated tool: CO$TAT. https://www.aceit.com/aceit-suite-home/product-info/costat.

Dr. Wilson Rosa serves as Assistant Director at the Department of Homeland Security Cost Analysis Division. Dr. Rosa has won 7 Best Paper Awards at premier conferences including the IEEE/ACM International Symposium on Empirical Software Engineering and Measurement (1), International Conference on Software Process and Product Measurement (1), and ICEAA Professional Development & Training Workshop (4). He earned a Ph.D. and M.S. in Engineering Management from the George Washington University and a B.S. in Mechanical Engineering from the University of Puerto Rico. Dr. Rosa completed the Senior Managers in Government Executive Leadership Program at the Harvard Kennedy School.

Sara Jardine is a Program Manager and Expert Cost Analyst for Galorath with over 18 years of cost analysis and financial management experience. She has served a broad variety of U.S. federal agencies including the Army, Navy, Secretary of Defense, the Department of Veterans Affairs, and the Department of Home-land Security. Ms Jardine has earned 3 Best Paper Awards at ICE-AA Professional Development & Training Workshops. She earned a M.S. in Project Management from George Washington University and a B.Sc. in Mathematics from the University of Michigan.