

AGILE ESTIMATIONS: EMPIRICAL STUDY USING WEIGHTED COMPLEXITY FACTORS

Ravi Kiran Mallidi* and Manmohan Sharma

Received: March 09, 2023; Revised: May 19, 2023; Accepted: June 12, 2023

Abstract

Software time and cost estimates are always challenging due to the quickly changing software process. Especially Agile project estimations are complex, and the story points differ from person to person. Due to the person-to-person variations, project managers have difficulty arriving at the project cost for development projects with different sizes or complexity of systems/software. This paper proposes a new estimation model named the WEighted COmplexity Factor Estimation (WECOFE) Model to reduce the complexities of Agile story point estimations. Weighted complexities factors identify Low, Moderate, and High Project Level Factors and Story Level Factors. For each level, multiple factors were defined. Derived Weighted complexity based on Project, Requirement, and Resource Level factors. Each story's point estimation is explained based on weighted complexity defined and assigned factors values. Conducted study on various industry projects, showing that the WECOFE approach is promising by observing the Magnitude of Relative Error (MRE) and Mean Magnitude of Relative Error (MMRE). Also, performed a t-test for one sprint project with 51 stories, and paired sample means are equal for WECOFE with actual story points spent by the team.

Keywords: Agile; Estimations; Story point estimation; Time estimations; Weighted Complexity

Introduction

Scrum methodology is challenging in an agile context, especially for effort estimation. Estimations are a difficult feat in modern software projects. Most of the project's failures are caused due to improper effort estimates and noneffective resource utilization in projects. Organizations use two effective methods to identify project estimations - measuring hours and story points. Story point estimations are popularly used in agile methodology effectively. There are several ways to define the agile project estimates – Planning poker, Bucket System, Dot Voting, T-Shirt sizing, Ordering

Method, and dividing until the maximum size or less described in an earlier paper by Mallidi and Sharma (2021) along with challenges. The base principle for identifying story point estimation is the principle of relativity. Ahmed *et al.* (2018) compared traditional and Agile estimation models regarding Adaptability and Teamwork. They identified Agile Risk as a Lack of Understanding of Requirements, Team meetings, visits, Training, Poor Communication, and the number of project Personnel.

For example, the complexity level of Task 1 is decided by the complexity level of Task 2. The

School of Computer Applications, Lovely Professional University, Punjab, India. E-mail: ravikiran.mallidi@gmail.com*, manmohan.sharma71@gmail.com

* Corresponding author

DOI: <https://doi.org/10.55766/sujst-2023-04-e0737>

Suranaree J. Sci. Technol. 30(4):010237(1-11)

teams look at each aspect of the project relative to the other elements of the project/story. The unit of measure is based on the teams' difficulty and complexity of the story. The Scrum master asks the group to estimate each story's points using Fibonacci projections and the developer estimates. Each developer provides story points to each activity and validates the story points; if there is a considerable difference among each developer, the second round of effort estimation is performed with justifications. Most of the time process of story point estimation makes it difficult for the developer to judge the points due to the relative sizing and complexity of the software-variation of estimates observed in projects due to the non-experience of the developers to identify the story points. Most of the situation's developers equate agile story issues to hours. For example, two developers work on the same feature with two story points estimation. However, the time spent completing the task differs from one resource to another. For example, two developers have agreed on estimating two story points for a work item even though their time estimations are different. All the developers are on the same page regarding this story point estimations despite hours being different from one resource to another. Below are some of the disadvantages of story point estimations.

- Take time to understand and master estimation skills.
- Different from one resource to another and their justification.
- A stable team is required.
- Business teams are unable to estimate things to allocate budgets.
- Possibility of underestimating/overestimating by the developers by providing some justifications. Smart developers provide the higher side of estimations by giving reasons, but the task may be completed with fewer story points.

Defined Weights for various technical factors and Environment factors using UCP based on the weights of the elements the effort estimates derived by Rak *et al.* (2019). The research aims to define a new estimation model using the Weighted Complexity of each story. Base values are determined by the Organization every year. Organizations conduct yearly wise hackathons and identify the efforts required to complete one task. Based on the results, each work item's story points are set in a pre-defined format with proper justification. Projects must use the base values and add Weighted Complexity factors to arrive at appropriate story point estimates. Input factors were

identified and defined in this research and incorporated into a new WECOFE model.

Related Work

Various studies were conducted on Agile estimations and proposed different models for estimates. Adnan and Afzal (2017) presented an approach to improve the effort estimation in a project by comparing the five core components – multiagent effort estimation system, ontology knowledge base, knowledge creation, knowledge tree maintenance, and facility and proposed Multiagent Effort Estimation System. Altaieb and Gravell (2019) investigated using a mix of different Agile estimation techniques, Expert Judgment, and Planning Poker to avoid underestimation of project estimations based on a survey of 63% of projects using Poker, 47% using Analogy, and 38% in Expert Judgement. Iqbal *et al.* (2019) research that 42.31% of projects use extreme programming, 32.69% use scrum, and 25% use Kanban. Gandomani *et al.* (2019) chose case studies from two companies; one adopted Planning Poker for two years, and another used Scrum and XP for software development. Thirty-four projects have taken for analysis and finally analyzed with 29 projects. Compared Estimated and Actual Size for Estimation Types: Average and Consensus. Fernández-Diego *et al.* (2020) studied 73 papers on Agile estimations from 2014 to 2020 and suggested that combining existing data and expert-based estimates is promising. Altaieb *et al.* (2020) conducted a comprehensive analysis and concluded various factors impacting the effort estimation on mobile projects with the help of multiple questions to 20 participants across 18 Organizations. Arora *et al.* (2021) applied different regression models to the Agile projects for effort estimations and concluded that Categorical boosting (CatBoost) regression performed accurately. Compare the results of the Estimated Velocity and Actual Velocity of Story Point analysis and calculate the Project's variance (underestimated/overestimated). Canedo and Costa (2018) observed that Functional and Story point estimates are more appropriate for the case study. Gupta and Mahapatra (2019) proposed a Hybrid Agile Software Development estimation protocol to identify the estimations and validate the results against different models. Hacaloglu and Demirors (2019) studied the applicability of Functional Size Measurement (FMS) in Agile projects to improve the estimation process. Hannay *et al.* (2018) studied and derived three-point-based estimates: Benefit, Cost, and Benefit / Cost. Each requirement is a category to each point base, assigning the estimates

and helping to eliminate the EPICs from Agile Project. Hasan and Khan (2019) conclude that agile estimation is more promising based on Agile methodology with eXtreme Programming (XP) along with Test Driven Development (TDD). Ibrahim *et al.* (2019) studied maintenance challenges: Iterative development, focused work objective, teamwork, customer involvement, face-to-face communication, light documentation, frequent testing, collective ownership, and knowledge transfer. Kaur and Kaur (2019) collected 222 mobile application data and converted the specifications to COSMIC FMS.

Used Simple Linear Regression to validate the relationship between test efforts and COSMIN FMS size and observed promising results. Mahmood *et al.* (2022) evaluated 28 selected studies using MMRE and PRED (25) and found Machine learning-based Agile estimations are more frequently implemented. Moyo and Mnkandla (2020) Developed Secure Solo Software Development Model (Secure-SSDM) and evaluated it with different projects. They recommended various tools and technologies in every stage of Secure-SSDM. Romero-Chacón *et al.* (2019) present their experience with the Costa Rican Information System on Disability System (SICID) platform developed by the Costa Rica Institute of Technology (TEC) and the Adopted Accessibility test. Accessibility corrections and Accessibility reviews are part of the SCRUM process to achieve the requirement. Rosa *et al.* (2021) proposed a sizing measure estimation model for agile projects considering 36 Department of Defence projects. Sarwar *et al.* (2020) proposed 18 hypotheses to support the SADD model. Each hypothesis was validated using a statistical method with a Z test to calculate the hypotheses' confidence by comparing the probability 0.05 value was set. The SADD model incorporated a situational enhancement process in software development. Shams *et al.* (2019) propose combining Work Breakdown Estimates and Story Points helps the best estimates of Mobile Application Development. Zozas *et al.* (2019) suggest seven factors in JavaScript estimation are considered – Number of Corrective tasks (COR_ACT), Cumulative Activity (ACT), Lines of Code (LOC), and Number of Open Bugs-developed maintenance indices like Maintenance Changes Index and Maintenance Effort Index.

Model for Weighted Complexity Factors Estimation (WECOFE)

Currently, estimating agile projects is purely based on the resource/developers' experience. Most of the

time, the estimations are oversized or undersized. Rak *et al.* (2019) proposed a use case reuse model for considering Simple, Medium, and Complex actors and weights for each use case. Functional and technical complexities are defined for each use case. Related work has been done with different authors, and multiple methods are derived for size and effort for a given task. Numerous studies have provided empirical evidence demonstrating that agile effort estimations are based on Artificial Intelligence (AI) / Machine Learning (ML) and traditional ways of estimates not using AI / ML. However, little research was carried out to identify effort estimations to cover the aspects of Functional / Data complexity, Architecture Complexity, External / Internal System Integrations, and Resource Experience for the task.

Considering the above points proposed, the WECOFE model for story point estimations by defining the complexity factors. The Organization determines raw estimated values annually by conducting hackathons between groups. Organizations run yearly wise hackathons and identify the efforts required to complete one task. Based on the results, each work item's story points are set in a pre-defined format with proper justification. Projects must use the base values and add Weighted Complexity factors to arrive at appropriate story point estimates. Input factors were identified and defined in this research and incorporated into a new WECOFE model. However, in the proposed WECOFE model, Project Level Factors (PLF) and Story Level Factors (SLF) are critical factors in identifying the story points for a given story/requirement, as described in Figure 1. This model helps the developers define each factor's values and the remaining story point calculated by the model.

Figure 2, describes the model for complexity factors for agile effort estimations. For each element, define the values concerning Low, Moderate, and High and add this to Raw estimates.

Project Level Factors (PLF)

Project Level scope refers to factors common for all the story/requirements. For each sprint or the

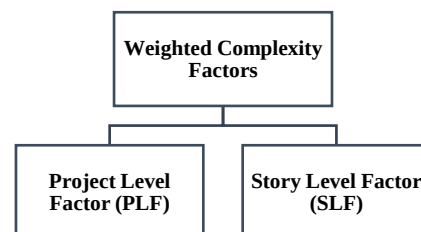


Figure 1. Factors for Agile Estimations

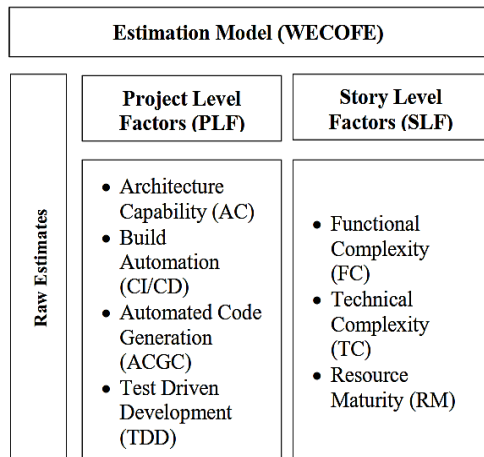


Figure 2. WECOFE Model

Table 1. Complexity Definition for PLF

PLF	Low	Moderate	High
AC	1.2	1	0.9
CI/ CD	1.2	1.1	1
ACC	1	0.8	0.6
TDD	1	1.1	1.12

Table 2. Complexity Definition for SLF

SLF	Low	Moderate	High
FC	1	1.1	1.2
TC	0.8	1	1.2
RM	1.2	1	0.8

total Project, the complexity factors are fixed. Below are the four elements that are related to the project level. For each project, the factors considered for effort estimations

- Architecture Capability (AC)
- Build Automation (CI/ CD)
- Automated Code Generation (ACC)
- Test Driven Development (TDD)

For each factor, defined the complexity based on Low, Moderate, and High, and the complexity values are assigned. Table 1 describes each factor with complexity factor definition for WECOFE calculation. Different weights are considered and substituted in the WECOFE model for calculating MRE and MMRE. After comparing MRE and MMRE values, the values defined in Table 1 came up.

For each factor, the complexity is defined based on Low, Moderate, and High and explained below, and the complexity values are assigned as

Architecture Capability (AC):

- Low - The team has low experience in Architecture defined by the Organization. The team must spend time understanding the architecture defined for the project.

- Moderate - The team has good experience in Architecture defined for the project
- High - The team has excellent expertise in Architecture represented for the project

Automation (CI/ CD):

- Low - CI / CD is not configured or set up for the project. All deployments are manual.
- Moderate - CI / CD has been configured, and the team uses Source Repository and Build tools for deployments. Not fully utilizing CI / CD
- High - CI / CD has been configured, and the team is using Source Repository, Build tools, and auto configurations for deployments

Automated Code Generation (ACC):

- Low - Generate very minimal code using Development Tools (like generating setter and getter)
- Moderate - Generate code using application code generator tools like no-code / low-code with low utilization
- High - Generate code using application code generator tools like no-code / low-code with effective utilization
- *Test-Driven Development (TDD):*
- Low - No Test-Driven Development
- Moderate - Test-Driven Development in the project with 40-50% coverage
- High - Test-Driven Development in the project with > 80% coverage

Story Level Factors (SLF)

Story Level scope refers to factors that are factored into the story/requirement. Below are the three elements that are related to each story level. For each story, the factors considered for effort estimations.

- Functional Complexity (FC)
- Technical Complexity (TC)
- Resource Maturity (RM)

For each factor, defined the complexity based on Low, Moderate, and High, and the complexity values are assigned. Table 2 describes each factor with complexity factor definition for WECOFE calculation. Different weights are considered and substituted in the WECOFE model for calculating MRE and MMRE. After comparing MRE and MMRE values, the values defined in Table 2 came up.

For each factor, the complexity is defined based on Low, Moderate, and High and explained below, and the complexity values are assigned as

Functional Complexity (FC):

- Low - Not having many business validations to develop the function. Very straightforward procedure to implement.
- Moderate - The requirement has Business validations.
- Ex: Banking application validates like fraud detection, account checks
- High - The requirement has complex Business validations. In the function, there are more than 3 - 4 validations.

Technical Complexity (TC):

- Low - Simple technology for the implementation of the given function does not require any complex validations/logic/algorithms to implement.
- Moderate - Moderate technology is used to implement the given function. Moderate validation/logic/algorithm to implement for the function.
- High - Complex technology is used to implement the given process. Complex validation/logic/algorithm to implement for the function.

Resource Maturity (RM):

- Low - The resource has been working on the same technology for the past 2-3 years and requires support from other senior resources.
- Moderate - The resource has been working on the same technology for the past 3-5 years and does not require support from other senior resources. Having the capability of good coding and optimization techniques.
- High - The resource has worked on the same technology for 5+ years and can write new innovative programs. Having the capability of excellent coding and optimization techniques.

Linear Proposition of Weights

The weights of factors for Project Level and Story Level factors are calculated as linearly proportional (L refers to attribute Low, M refers to attribute Moderate, and H refers to attribute High)

$$L = x + 1, M = 1 \text{ and } H = 1 - x$$

$$\text{OR}$$

$$L = x - 1, M = 1 \text{ and } H = x + 1$$

For Project Level factors – AC, CI/ CD, and ACC factors and Story Level factors – RE applies that “Low” requires more effort than attribute “High” ($L > H$). Therefore, the weight of attribute “Low” is higher than the weight of attribute “High.”

For Project Level factors – TDD factors and

Story Level factors – FC and TC applies that “Low” requires less effort than attribute “High” ($L < H$). Therefore, the weight of the attribute “Low” is less than that of “High.”

Raw Estimation Calculation

Calculate Raw estimations for each Organization for their project needs. The base efforts are different from Organization to Organization. Each Organization adopts various processes to identify the base estimation for a work packet. They have proposed the six steps below to identify raw estimations/efforts.

Step 1: Identify the moderate work packet (ex: UI design, DAO implementation, etc.) Definitions are provided in Table 4.

Step 2: Identify the various resources from the Organization with different work experiences.

Step 3: Create a moderate work packet for the specific feature and give it to 3-5 (minimum 3) developers identified to complete the task. Taking 3-5 developers gives more accurate results due to not relying on one developer.

Step 4: Identify the actual stories spent on the work packet for each resource from 3-5 developers.

Step 5: Average the efforts from the 3-5 developers spent stories to complete the work packet (identified to work on the prescribed work packet)

Step 6: Note the average Story Point identified for the work packet. The work packet would be classified as a Moderate work packet.

Table 3 describes the raw estimations (moderate efforts) from three Organizations for their work/projects performed.

Table 4 describes the definitions of each work item worked by the Organizations. Based on the description, the teams must complete a particular Story Point task-the story point as defined in Table 3.

Project Level Adjusted Factors Calculation

Calculate the project-adjusted factor for each factor identified level of maturity at the time the sprint starts. The maturity might change from sprint to sprint. For example, the team's maturity is low if working on a new architecture methodology at Sprint-1, and in later sprints, the maturity increases to Moderate / High.

- Calculate AC factor as – AC complexity factor -1
- Calculate CI/ CD factor as – CI/ CD complexity factor -1
- Calculate ACC factor as – ACC complexity factor -1
- Calculate TDD factor as – TDD complexity factor -1

Table 3. Moderate Efforts from three Organizations

Work Item	Moderate Estimation		
	Org-1	Org-2	Org-3
UI Page	2	2	2
Business Logic	3	3	3
Rest Service	3	3	3
SOAP Service	3	3	3
External Integration	3	3	3
DAO	2	1.5	1.5
DB Changes	1	1	1
Administrative Activities	NA	1	NA
Reports	NA	4	NA
Onboarding New Software	NA	3	NA
UI + Business Logic	5	5	5
UI + Business Logic + DAO	7	6.5	6.5
Business Logic + Rest Services	6	6	6
Business Logic + DAO	5	4.5	4.5
Business Logic + External Integration	6	6	6
UI + Business Logic + Rest Services	8	8	8
UI + Business Logic + Rest Services + External Integration	11	11	11
UI + Business Logic + Rest Services + DAO	10	9.5	9.5
Bug - UI	1	NA	NA
Bug - Business Logic	1	NA	NA
Bug - DAO	1	NA	NA
Bug - Services	1	NA	NA
Bug - External Integration	1	NA	NA

Step 1 calculates the Total Unadjusted Project Level Complexity (UPLC) measured using the formula below.

$$UPLC = \sum_{i=1}^n (P_i (\text{Complexity}) - 1)$$

For $i = 1, 2, \dots, n$ (n represents the total number of Project Level Factors, in this case, four factors) and P_i represents each Project factors value.

Story Level Adjusted Factors Calculation

Calculate the Story level adjusted factor for each factor identified level of maturity at the time of sprint start. The maturity might change from task to task. For example, the maturity of the job is Low if working on a highly complex task, and the same charge may be Moderate in later Sprints if the same resource has experience.

- Calculate FC factor as – FC complexity factor -1
- Calculate TC factor as – TC complexity factor -1
- Calculate RM factor as – RM complexity factor -1
- Calculate TDD factor as – TDD complexity factor -1

Step 2 calculates the Total Unadjusted Story Level Complexity (USLC) measured using the formula below.

$$USLC = \sum_{i=1}^n (S_i (\text{Complexity}) - 1)$$

Table 4. Definitions of Work Item

Work Item	Description
UI Page	UI Screen having 8 - 10 form fields. Transactions - with Business / JSON / AJAX Objects
Business Logic	Business Logic involves transactions with Database, Business Objects / JSON / AJAX fields (8-10 elements). Server-side validations with 8-10 elements, data from UI screen / external component.
Rest Service	Business Logic involves transactions with 8-10 elements, along with validations. For example, receive JSON object with 8-10 elements / Create SOAP with 8-10 elements with business logic.
SOAP Service	Business Logic involves transactions with 8-10 objects along with validations. For example, receive SOAP object with 8-10 elements / Create SOAP with 8-10 elements with business logic (Bottom-up approach).
External Integration	Create an Object with 8-10 elements and connect it to an external system for sending data / receiving data.
DAO	Create / Update / Delete operation with 2-3 database tables. Prepare statements and execute procedures if required.
DB Changes	Altering the existing table structure (adding, deleting, and altering constraints) and related changes in the code base.
UI + Business Logic	UI and Business Logic class implementation with 8-10 objects
UI + Business Logic + DAO	UI, Business Logic, and DAO class implementation with 8-10 objects
Business Logic + Rest Services	Business Logic and Rest Service implementation with 8-10 objects
Business Logic + DAO	Business Logic and DAO class implementation with 8-10 objects
Business Logic + External Integration	Business Logic class and External Service Integration implementation with 8-10 objects
UI + Business Logic + Rest Services	UI, Business Logic class, and Rest Service implementation with 8-10 objects
UI + Business Logic + Rest Services + External Integration	UI, Business Logic class, Rest Service, and External Service integration implementation with 8-10 objects
UI + Business Logic + Rest Services + DAO	UI, Business Logic class, Rest Service, and DAO implementation with 8-10 objects

For $i = 1, 2, \dots, n$ (n represents the total number of Story Level Factors, in this case, three factors), and S_i represents each story factor's value.

Calculate WECOFE Adjusted Factors

The above two sections calculated “Project Level Adjusted Factors” and “Story Level Adjusted Factors.” Using this value calculate the “*Total Unadjusted Complexity Factor (TUCF)*”. for a story to identify the final/accurate story point to develop. **Step 3** calculates the Total Unadjusted Complexity Factor (TUCF) by adding up UPLC and USLC values.

$$TUCF = 1 + UPLC + USLC$$

Finally, calculate the estimation by multiplying the “Raw Estimations” with the “TUCF” value. These values give us a valid Story point compared to other estimation models.

Calculate the *WEighted COMplexity Factor Estimation (WECOFE)* by Raw efforts multiplied by the TUCF value. The derived value is in decimal format, then rounded to the higher side of the number.

$$\text{WECOFE} = \text{Raw Estimates} * \text{TUCF}$$

Conducted core work via an experimental base for existing data from different projects received from various Organizations. The purpose of conducting a series of experiments conducted to systematically evaluate the complexity factors of forecasting techniques and their variants (MRE and MMRE). This enabled the comparative analysis of existing methods used in Agile, like Expert Judgment / Story Point analysis of estimation, along with the novel estimation model proposed. Finally, the model was applied to and calibrated on several sprints with different Projects and sizes.

Data Collection and Model (WECOFE) Validation

In this envisaged study, multiple sources were passed through to gather Agile projects to experiment with the WECOFE model. Data collected from three software organizations involves project managers and software architects to identify the Story work packet belonging to and assign them the factors defined in the new model. The techniques used for data collection are direct,

real-time data collected directly from JIRA boards and indirect methods, which involve the supply of raw data without interacting with project teams. Ethical approvals have been taken to start collecting the data from the JIRA boards and masking the data into the readable format without impacting the end customer security constraints. Acceptance of the concerned team has been taken before collecting the raw data from the JIRA board. As per the concern from the project owner, the data can be used after masking, the same way used in this study.

The *Magnitude of Relative Error (MRE)* is the commonly used parameter to identify the error in estimations. Below is the formula for MRE as

$$\text{MRE} = \frac{|\text{Actual Effort} - \text{Estimated Effort}|}{\text{Actual Effort}}$$

Calculate the MRE value for each story/requirement, which means the difference between the Actual and Estimated efforts identified. Calculate the *Mean Magnitude of Relative Error (MMRE)* value for the sprint level and the average MRE value for each sprint.

$$\text{MMRE} = \frac{\sum_{i=1}^n \text{MRE}}{N}$$

JIRA sprint planning data was collected from the four projects and masked the data to obey ethical processes. Table 5 contains project sprint data spanning from 2 to 14 sprints. The sprint cycle comprises 2–3-week time. Data received projects use Java, J2EE, Spring, and Spring Boot as core software, Architectures as Microservices / N-Tier / Streaming, and Domains as Banking and Trading Systems. Table 5 heading definition is PS – Project Sprint, ASP – Actual Story Point, ESP – Estimated Story Point, WSP – WECOFE model Story Points, ME – MMRE Existing, and MW – MMRE WECOFE.

The MMRE indicates the calculated relative value by which the estimated effort is an underestimate or overestimate in comparison to the actual effort. N represents the number of MRE occurrences in the sprint. WECOFE applied for the existing four projects executed for the last two years and projects. Table 5 calculates the MMRE values for existing and new models for 28 sprints for four projects, and each sprint spans from 2 and 3 weeks. Finally, compare and show accurate results for the new WECOFE model approach. P1 to P4 represent the different project names, and S1 to S[n] represents the sprint cycle. Observed that the project's estimated story point (ESP) estimates are higher/lower than the actual story points (ASP). In most cases, the MMRE value is more than 0.10

Table 5. Story Point, MMRE – Existing VS WECOFE

PS	ASP	ESP	WSP	ME	MW
P1-S1	77	62	78	0.19	(0.03)
P1-S2	213	168	211	0.20	0.01
P1-S3	96	84	94	0.11	0.02
P1-S4	151	130	152	0.11	(0.01)
P2-S1	59	49	60	0.17	(0.01)
P2-S2	105	87	106	0.22	(0.02)
P3-S1	15	15	15	(0.05)	0.00
P3-S2	12	16	12	(0.45)	0.00
P3-S3	10	8	10	0.22	0.00
P3-S4	10	12	10	(0.33)	0.00
P3-S5	11	14	11	(0.20)	0.00
P3-S6	4	4	4	0.00	0.00
P3-S7	6	7	6	(0.17)	0.00
P3-S8	16	15	16	(0.04)	0.00
P4-S1	29	21	24	0.22	0.10
P4-S2	52	47	50	0.10	0.03
P4-S3	119	113	121	0.03	(0.01)
P4-S4	75	64	76	0.11	(0.01)
P4-S5	54	42	56	0.20	(0.06)
P4-S6	91	84	89	0.12	0.02
P4-S7	79	69	72	0.13	0.07
P4-S8	59	51	59	0.12	(0.01)
P4-S9	68	61	69	0.10	(0.01)
P4-S10	73	65	74	0.14	(0.01)
P4-S11	59	62	58	(0.04)	0.01
P4-S12	94	77	86	0.22	0.06
P4-S13	90	84	89	0.12	0.01
P4-S14	63	57	62	0.11	0.01

(ME > 10%) for comparing existing estimated story points, meaning the deviation is more than 10% of existing efforts. In the WECOFE model, the variation is less than 0.10 (MW < 10%) for all the cases in Table 5. Therefore, the new WECOFE model is more efficient in estimating the Agile Project. The usage of the model is also simple for the developer. First, the developer assigns the Complexity attributes with Low, Moderate, and High ranges. Figure 3 shows the Actual, Estimated, and WECOFE model story points and the results show WECOFE is closer to actual story points than the existing model.

Parametric testing (paired t-test)

Consider one sprint P1-S2 from Table 5 to perform parametric testing using a t-test paired two samples for means measured as described in Table 5. For the Project within the validation set, story points effort estimated by Existing and WECOFE are listed in Table 6. The null hypothesis states that there is no difference in estimation accuracy (measured by story points) of estimated project effort between the application of the Existing and WECOFE model. The alternative hypothesis states that there is a difference.

From Table 7, perform the t-test: Paired Two Samples for means for hypothesis testing-the two samples (Actual Vs. Existing Estimates and Actual

Vs. WECOFE estimates). The number of degrees of freedom is $f = n - 1 = 51 - 1 = 50$, and the P value is $4.4824133438423E-10$. Therefore, rejecting the null hypothesis at the 0.05 level for the data set Actual Estimates Vs. Existing Estimations set. Figure 4 compares the Mean and variance for Existing and Actual story points, clearly showing the difference in Mean.

The detailed t-test analysis is provided in Table 8 for Actual Vs. WECOFE model estimations. The number of degrees of freedom is $f = n - 1 = 51 - 1 = 50$, and the P value is 0.322125645100244 . Therefore, accepting the null hypothesis at the 0.05 level for the data set Actual Estimates Vs. WECOFE model estimations. Figure 5 compares the Mean and variance for WECOFE and the Actual story point, clearly showing the means are equal. While comparing, the graphs indicate that the WECOFE is more promising results. Presented on details sprint activities in Table 9 with heading definition as FC – Functional Complexity, TC – Technical Complexity, RM – Resource Maturity, RE – Raw Estimations, CF – Final Calculated Complexity Factor, ASP – Actual Story Points, ESP – Estimated Story Points, WSE – WECOFE Story Points, ME – MRE for Actual Story Points, and MW – MRE for WECOFE story points. Table 9 indicated that the

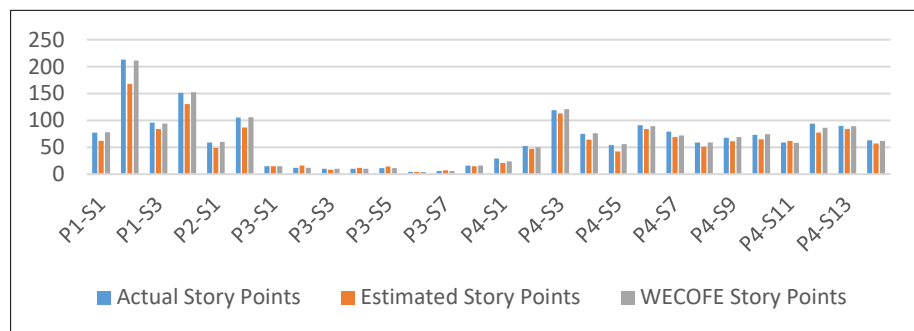


Figure 3. Actual Vs. Estimated Vs. WECOFE Story Points

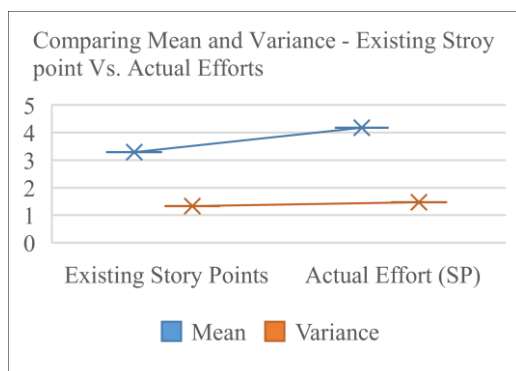


Figure 4. Comparing Mean and Variance

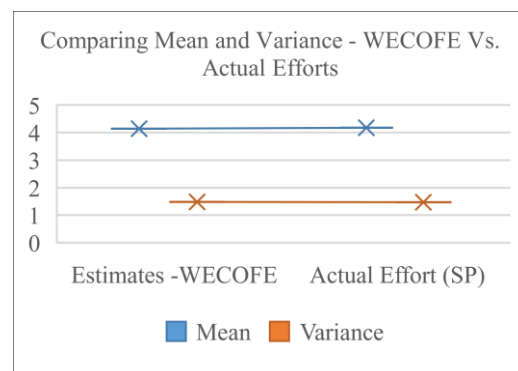


Figure 5. Comparing Mean and Variance

MRE values for Existing efforts against Actual efforts are more than 15 % in most stories. Out of 51 stores, effort deviation is, in most cases (>80%). While comparing the efforts calculated from the WECOFE model, the effort deviation is “0.00” for more than 80% of cases. This table shows that the newly defined model gives better results than expert judgment estimation in most cases used in the IT industry for story point estimations.

Conclusions

For the last decade, various Agile estimations models have been derived by Organizations and adopted in estimations. However, every resource in the Project may have its way of estimating the stories and observing the non-consistency between them. The paper introduces the Complexity Factor technique, the WECOFE model, for Story Point estimation for Agile projects. Adopted/experimented with the model for various actual projects with various sizes and showed better estimation results for Agile projects. The MMR and MMRE values are noted for 28 sprints with 1790 story points spent by the developers. As part of further work, there is the possibility of taking some more projects/sprints with a high number of technical and functional stores and applying statistical parametric testing to calculate optimal deviation for the WECOFE. The t-test has been performed and clearly shows that the variance between WECOFE is more accurate while comparing the null hypothesis is rejected while Existing estimation. Further, the t-test has to perform the n-number of projects to validate the Mean and variance difference between the data sets.

Table 6. Calculation of paired t-test

Item	Description Input
Input	Paired Samples (x1, y1), (x2, y2)
Where x1 represents Estimates story points and y1 represents Actual story points for a Project i	
Ho	$\mu_d = 0$ where $d_i = x_i - y_i$ The expectation of the Mean of the difference is 0
Criteria	Two sides if H_1 : $\mu_d \neq 0$, Reject the H_1

Table 7. T-test Existing efforts Vs. Actual effort

t-Test. Paired Two Sample for Means		
	Existing Story Points	Actual Effort (SP)
Mean	3.294117647	4.176470588
Variance	1.331764706	1.468235294
Observations	51	51
Pearson Correlation	0.76309179	
Hypothesized Mean Difference	0	
Df	50	
t Stat	-7.721980009	
P(T<=t) one-tail	2.24312E-10	
t Critical one-tail	1.675905025	
P(T<=t) two-tail	4.49E-10	
t Critical two-tail	2.008559112	

Table 8. T-test WECOFE efforts Vs. Actual efforts

t-Test. Paired Two Sample for Means		
	Estimates -WECOFE	Actual Effort (SP)
Mean	4.137254902	4.176470588
Variance	1.480784314	1.468235294
Observations	51	51
Pearson Correlation	0.973413069	
Hypothesized Mean Difference	0	
Df	50	
t Stat	-1	
P(T<=t) one-tail	0.161062823	
t Critical one-tail	1.675905025	
P(T<=t) two-tail	0.322125645	
t Critical two-tail	2.008559112	

Spring Name: Squirtle Sprint 2, 3 Weeks sprint

Project Level Factor Values			
Architecture Capability (AC)	Moderate	Automated Code Generation (ACC)	Low
Build Automation (CI/ CD)	High	Test Driven Development (TDD)	Low

Table 9. Sprint activities for project P1-S2 (1)

Key	Work Item	FC	TC	RM	RE	CF	ASP	ESP	WSP	ME	MW
Squirtle-148	Business Logic + Rest Services	Low	Moderate	Moderate	6	1	6	5	6	0.17	0.00
Squirtle-149	UI + Business Logic	Moderate	High	High	5	1.1	6	5	6	0.17	0.00
Squirtle-150	UI + Business Logic	Low	Low	Moderate	5	0.8	4	3	4	0.25	0.00
Squirtle-151	Business Logic + Rest Services	Low	Low	High	6	0.6	4	3	4	0.25	0.00
Squirtle-152	Business Logic	Low	Moderate	Moderate	3	1	3	3	3	0.00	0.00
Squirtle-153	Business Logic	Moderate	Moderate	Moderate	3	1.1	4	3	4	0.25	0.00
Squirtle-154	Business Logic	Moderate	Moderate	Moderate	3	1.1	4	3	4	0.25	0.00
Squirtle-156	Business Logic	Moderate	Moderate	High	3	0.9	3	3	3	0.00	0.00
Squirtle-159	Rest Service	Moderate	Low	Moderate	3	0.9	3	2	3	0.33	0.00
Squirtle-203	UI + Business Logic	Moderate	Moderate	High	5	0.9	5	4	5	0.2	0.00
Squirtle-208	External Integration	Moderate	Moderate	Moderate	3	1.1	4	3	4	0.25	0.00
Squirtle-209	External Integration	Low	Low	Moderate	3	0.8	3	3	3	0.00	0.00
Squirtle-212	External Integration	High	Moderate	High	3	1	3	3	3	0.00	0.00
Squirtle-213	External Integration	Low	Moderate	Moderate	3	1	3	3	3	0.00	0.00
Squirtle-214	External Integration	Moderate	Moderate	Moderate	3	1.1	4	3	4	0.25	0.00
Squirtle-217	External Integration	Moderate	Moderate	Moderate	3	1.1	5	3	4	0.40	0.20
Squirtle-233	Business Logic + External Integration	Low	Moderate	Moderate	6	1	6	4	6	0.33	0.00
Squirtle-234	Business Logic + External Integration	Low	Moderate	Moderate	6	1	6	4	6	0.33	0.00

Table 9. Sprint activities for project P1-S2 (2)

Key	Work Item	FC	TC	RM	RE	CF	ASP	ESP	WSP	ME	MW
Squirtle-247	UI + Business Logic	Low	Moderate	Moderate	5	1	5	4	5	0.20	0.00
Squirtle-248	UI + Business Logic	Low	Moderate	Moderate	5	1	5	4	5	0.20	0.00
Squirtle-256	UI + Business Logic	Low	Moderate	Moderate	5	15	5	2	5	0.60	0.00
Squirtle-155	Business Logic + Rest Services	Low	Moderate	High	6	0.8	5	3	5	0.40	0.00
Squirtle-157	External Integration	Low	Moderate	Moderate	3	1	3	2	3	0.33	0.00
Squirtle-158	External Integration	Low	Moderate	Moderate	3	1	3	3	3	0.00	0.00
Squirtle-160	UI + Business Logic + DAO	Moderate	Moderate	Moderate	7	1.1	8	9	8	-0.13	0.00
Squirtle-161	External Integration	Moderate	Moderate	Moderate	3	1.1	5	4	4	0.20	0.20
Squirtle-162	External Integration	Low	Moderate	Moderate	3	1	3	2	3	0.33	0.00
Squirtle-163	External Integration	Low	Moderate	Moderate	3	1	3	3	3	0.00	0.00
Squirtle-164	Business Logic + External Integration	Moderate	Low	Moderate	6	0.9	6	4	6	0.33	0.00
Squirtle-165	Business Logic + Rest Services	Low	Moderate	Moderate	6	1	6	4	6	0.33	0.00
Squirtle-207	External Integration	Low	High	High	3	1	3	2	3	0.33	0.00
Squirtle-210	Business Logic	Moderate	High	High	3	1.1	4	4	4	0.00	0.00
Squirtle-211	External Integration	Moderate	Moderate	High	3	0.9	3	2	3	0.33	0.00
Squirtle-215	External Integration	Moderate	Low	Moderate	3	0.9	3	2	3	0.33	0.00
Squirtle-216	External Integration	Low	Moderate	Moderate	3	1	3	3	3	0.00	0.00
Squirtle-218	External Integration	Moderate	Moderate	Moderate	3	1.1	4	3	4	0.25	0.00
Squirtle-219	UI + Business Logic	Low	Moderate	High	5	0.8	4	4	4	0.00	0.00
Squirtle-220	UI + Business Logic	Low	Moderate	Moderate	5	1	5	4	5	0.20	0.00
Squirtle-221	Rest Services	Low	Moderate	Moderate	3	1	4	4	3	0.00	0.25
Squirtle-222	UI + Business Logic	Low	Moderate	Moderate	5	1	5	5	5	0.00	0.00
Squirtle-223	UI + Business Logic	Low	Moderate	Moderate	5	1	4	3	5	0.25	-0.25
Squirtle-224	UI + Business Logic	Low	Moderate	Moderate	5	1	5	3	5	0.40	0.00
Squirtle-225	UI + Business Logic	Low	Moderate	High	5	0.8	4	3	4	0.25	0.00
Squirtle-232	External Integration	Low	Moderate	High	5	0.8	3	3	3	0.00	0.00
Squirtle-235	External Integration	Moderate	Moderate	High	3	0.9	3	3	3	0.00	0.00
Squirtle-236	Business Logic	Low	Low	Moderate	3	0.8	3	3	3	0.00	0.00
Squirtle-237	Business Logic	Low	Low	Moderate	3	0.8	3	2	3	0.33	0.00
Squirtle-238	Business Logic + External Integration	Low	Moderate	High	6	0.8	5	4	5	0.20	0.00
Squirtle-239	Business Logic + External Integration	Low	High	High	6	1	6	3	6	0.50	0.00
Squirtle-246	External Integration	Low	Moderate	High	3	0.8	3	2	3	0.33	0.00
Squirtle-249	External Integration	Low	High	High	3	1	3	2	3	0.33	0.00

References

- A. Altaieb, M. Altherwi and A. Gravell. (2020). An Industrial Investigation into Effort Estimation Predictors for Mobile App Development in Agile Processes. 2020 9th International Conference on Industrial Technology and Management (ICITM), Oxford, UK, p. 291-296. <https://doi.org/10.1109/ICITM48982.2020.9080362>
- Adnan, M., and Afzal, M. (2017). Ontology-Based Multiagent Effort Estimation System for Scrum Agile Method. in IEEE Access, vol. 5, p. 25,993-26,005. <https://doi.org/10.1109/ACCESS.2017.2771257>
- Ahmed, M., Malik, B.H., Tahir, R.M., Perveen, S., Alvi, R.I., Rehmat, A., Ain, Q.T., and Asghar, M. (2018). Estimation of Risks in Scrum Using Agile Software Development. In: Ahram, T. (eds) Advances in Artificial Intelligence, Software and Systems Engineering. AHFE 2018. Advances in Intelligent Systems and Computing, vol 787. Springer, Cham. https://doi.org/10.1007/978-3-319-94229-2_12
- Altaieb, A., and Gravell, A. (2019). An empirical investigation of effort estimation in mobile apps using agile development process. Journal of Software, 14(8):356-369 <https://doi.org/10.17706/jsw.14.8.356-369>
- Arora, M., Sharma, A., Katoch, S., Malviya, M., and Chopra, S. (2021). A state-of-the-art regressor model's comparison for effort estimation of agile software. In 2021 2nd International Conference on Intelligent Engineering and Management (ICIEM), London, United Kingdom, p. 211-215. <https://doi.org/10.1109/ICIEM51511.2021.9445345>
- Canedo, E.D., and Costa, R.P.D. (2018). Methods and metrics for estimating and planning agile software projects. Twenty-fourth Americas Conference on Information Systems, New Orleans.
- Fernández-Diego, M., Méndez, E. R., González-Ladrón-De-Guevara, F., Abrahão, S., and Insfran, E. (2020). An Update on Effort Estimation in Agile Software Development: A Systematic Literature Review. in IEEE Access, vol. 8, p. 166768-166800, <https://doi.org/10.1109/ACCESS.2020.3021664>
- Gandomani, T. J., Faraji, H., and Radnejad, M. (2019). Planning Poker in cost estimation in Agile methods: Averaging Vs. Consensus. In 2019 5th Conference on Knowledge-Based Engineering and Innovation (KBEI), Tehran, Iran, p. 066-071. IEEE <https://doi.org/10.1109/KBEI.2019.8734960>
- Gupta, N., and Mahapatra, R.P. (2021). An effective agile development process by a hybrid intelligent effort estimation protocol. Journal of Ambient Intelligence and Humanized Computing, 1-10. <https://doi.org/10.1007/s12652-021-03088-x>
- Hacaloglu, T., and Demirors, O. (2019). Measureability of Functional Size in Agile Software Projects: Multiple Case Studies with COSMIC FSM. 2019 45th Euromicro Conference on Software Engineering and Advanced Applications (SEAA), Kallithea, Greece, p. 204-211. <https://doi.org/10.1109/SEAA.2019.00041>
- Hannay, J.E., Benestad, H.C., and Strand, K. (2018). Agile Uncertainty Assessment for Benefit Points and Story Points. in IEEE Software, 36(4):50-62. <https://doi.org/10.1109/MS.2018.2875845>
- Hasan, S.H., and Khan, U.A. (2019). Software Development Methods - Properties and Advances. International Journal of Computer Applications. 178(53). <https://doi.org/10.5120/ijca2019919436>
- Ibrahim, K.S.K., Yahaya, J., Mansor, Z., and Deraman, A. (2019). The Emergence of Agile Maintenance: A Preliminary Study. 2019 International Conference on Electrical Engineering and Informatics (ICEEI), Bandung, Indonesia, p. 146-151. <https://doi.org/10.1109/ICEEI47359.2019.8988815>
- Iqbal, J., Omar, M., and Yasin, A. (2019). The impact of agile methodologies and cost management success factors: an empirical study. Baghdad Science Journal, 16(2):496-504. [https://doi.org/10.21123/bsj.2019.16.2\(SI\).0496](https://doi.org/10.21123/bsj.2019.16.2(SI).0496)
- Kaur, A., & Kaur, K. (2019). A COSMIC function points based test effort estimation model for mobile applications.

- Journal of King Saud University-Computer and Information Sciences, 34(3):946-963.
- Mahmood, Y., Kama, N., Azmi, A., Khan, A.S., and Ali, M. (2022). Software effort estimation accuracy prediction of machine learning techniques: A systematic performance evaluation. *Software: Practice and Experience*, 52(1):39-65. <https://doi.org/10.1002/spe.3009>
- Mallidi, R.K., and Sharma, M. (2021). Study on agile story point estimation techniques and challenges. *International Journal of Computer Applications*, 174(13):9-14. <https://doi.org/10.5120/ijca2021921014>
- Moyo, S., and Mnkandla, E. (2020). A Novel Lightweight Solo Software Development Methodology With Optimum Security Practices. in *IEEE Access*, vol. 8, p. 33735-33747. <https://doi.org/10.1109/ACCESS.2020.2971000>
- Rak, K., Car, Ž., and Lovrek, I. (2019). Effort estimation model for software development projects based on use case reuse. *Journal of Software: Evolution and Process*, 31(2):e2119. <https://doi.org/10.1002/smr.2119>
- Romero-Chacón, V., Muir-Camacho, H., Rodríguez-González, J., Gómez-Blanco, A., and Chacón-Rivas, M. (2019). Adapting SCRUM Methodology to Develop Accessible Web Sites. 2019 International Conference on Inclusive Technologies and Education (CONTIE), San Jose del Cabo, Mexico, p. 112-1124. <https://doi.org/10.1109/CONTIE49246.2019.00029>
- Rosa, W., Clark, B.K., Madachy, R., and Boehm, B. (2021). Empirical Effort and Schedule Estimation Models for Agile Processes in the US DoD. in *IEEE Transactions on Software Engineering*, 48(8):3,117-3,130. <https://doi.org/10.1109/TSE.2021.3080666>
- Sarwar, A., Hafeez, Y., Hussain, S., and Yang, S. (2020). Towards Taxonomical-Based Situational Model to Improve the Quality of Agile Distributed Teams. in *IEEE Access*, vol. 8, p. 6812-6826. <https://doi.org/10.1109/ACCESS.2020.2964432>
- Shams, A., Bohm, S., Winzer, P., and Dorner, R. (2019). App Cost Estimation: Evaluating Agile Environments. 2019 IEEE 21st Conference on Business Informatics (CBI), Moscow, Russia, p. 383-390. <https://doi.org/10.1109/CBI.2019.00050>
- Zozas, I., Bibi, S., Ampatzoglou, A., and Sarigiannidis, P. (2019). Estimating the Maintenance Effort of JavaScript Applications. 2019 45th Euromicro Conference on Software Engineering and Advanced Applications (SEAA), Kallithea, Greece, p. 212-219. <https://doi.org/10.1109/SEAA.2019.00042>