Empirical Study on Software Estimation Techniques

Ravi Kiran Mallidi
School of Computer Applications
Lovely Professional University
Punjab, India

Manmohan Sharma
School of Computer Applications
Lovely Professional University
Punjab, India

***Abstract*—The project estimation of cost and effort is a critical task in any development environment. Organizations used different techniques to estimate for different environments. The effort estimations are different from person to person. Organizations adopt estimation templets to provide the estimations for the given requirement by the vendor. Proper estimations help the development team to develop the project with quality and stability. The main aim of this paper is to present different estimation techniques and templets used in development environments. Different estimation models like Work Break Down, Simple-Medium-Complex (SMC), Use Case Point, Functional Point Analysis, Agile Story Point estimations elaborated. Moreover, we proposed simple estimation templates for estimation models described. The templates help the project teams to come up estimates accurately concerning time and cost.**

Keywords—Estimation, Use Case Point, Functional Point, Agile, Story Point

I. INTRODUCTION

Organizations are facing common problems over-budget, late delivery due to the accuracy of estimations for years. Proper cost estimates avoid losing money and effective planning with optimized resource usage. Sometimes the estimations are depending on development methodology recommended specific estimation approach or practice. Various methods of estimation for Development and Maintenance Projects with the nature of the project and delivery model. Software organizations are using a variety of estimation models such as

- Work Breakdown Structure
- Simple-Medium-Complex (SMC)
- Use Case Point
- Functional Point
- Wideband Delphi
- Agile SCRUM Story Point
- Expert Base Estimation approach

Estimation starts from the start of the software development life cycle (SDLC) and refined to all the phases in the project start from the Planning phase to Maintenance Phase. Estimates determine the time, effort, resources, and money of the specific project or system. Estimations based on the given requirement from the Business teams with includes assumptions, risks, uncertainty, technology, and team composition.

The software project estimate includes size, effort in person-months / hours, schedule, and cost. Effort estimation accuracy would depend upon input data, historical data, and predictability of the Organization Software Development Life Cycle (SDLC) processes. Project managers play a crucial role in predicting accurate estimates by the given input or requirements. The strategy and execution mechanism would change according to the SDLC process adopted by the project. In this study, we compare effort estimation models SMC, Use Case Point, Functional Point, Agile Story Point.

The paper organized as follows: Section 2 describes Literature Review, Section 3 describes Different Estimation Models, and Section 4 concludes the paper.

II. LITERATURE REVIEW

There are studies about effort estimations modes in literature. Some of the studies in the novel are briefly summarized below.

Hoda et al. [10] focused on Agile historical overview and its trends, evolution over the last two decades. Scrum is a prominent agile method from 40% in survey 2007 to 70% IN SURVEY 2018 from the State of Agile. eXtream Programming places 2nd with 23% usage. Still, 84% of organizations are maturing agile practice.

Munialo et al. [16] The Standish report in 2012, 43% project challenged, and 18% of projects are failing. Paper presented Traditional and Agile estimation methods and comparisons of each of them. Non-Algorithm estimation methods like Expert Judgement, Analogy technique Wideband Delphi, and Algorithm based estimation methods like Source Line of Code (SLOC), Functional Point, Object Point, Constructive Cost Model (COCOMO).

Kaur et al. [13] critical factor of estimation is accurate to evade budget overrun. The paper presents a Common Software Measurement International Consortium (COSMIC) and Function Point Measurement (FMS) methods to estimate mobile applications. Developed five types of Functional process measurements (based on requests). Types are View Functionality, Data Manipulation Function, Inquiry Function, User Support Function, and Specific Function.

Shimoda et al. [21] suggested to use a combination of Agile and Waterfall methods by the trade-off between two approaches. The hybrid approach proposed to achieve better realization of low cost. Observed three models Water Fall, Agile and Hybrid and compared the characteristics of development methods include Large Scale, High reliability, High productivity, High estimation accuracy, Early realization, and Easy to change.

Ahmed et al. [2] compared the Traditional and Agile models in terms of Adaptability and Teamwork. Defined Risk of Agile as Lack of Understanding Requirement, Team Meeting, Visit, Training, Poor Communication, and Number of project Personnel.

Adnan et al. [1] Proposed an approach to improve the effort estimation with compared the multiagent component effort system with ontology knowledge-base, knowledge creation, knowledge tree maintenance, and facility.

Arifin et al. [3] emphases Effort-Time is more relevant, more fitted than the Effort-size model. Improve the accuracy of effort estimates by using expert-based, along with Effort-Time or Effort-Size to use.

Bik et al. [4] constructed a reference model for estimations by examining the story point life cycle. Studied the story point life cycle and combine Process-Deliverable-Diagram (PDD) to capture most commonly used story points

Hasan et al. [9] presented a study on different Software Development Methods and their properties. Studied from Waterfall to Agile and provided advantages and disadvantages of each development approach.

Zozas et al. [25] proposed two maintenance drivers for estimating the change request of the Agile projects. Predictive and discriminating power for high accuracy evaluated.

Khatri et al. [14] introduced the Algorithm to estimate by considering environmental and technical factors to achieve accuracy. The proposed approach considers Total Technical Factors (TTF), Total Number of Use Case Points (TUCP), Total Environmental Factors (TEF) weights in calculating the Total Estimated Cost (TEC).

Prakash et al. [17] presented survey results on Agile software development projects. Studied different modules and compared various estimation models like Use Case Point, Functional Point, COCOMO, Algorithm-Based Model, Expert Judgement Model and Estimation by Analogy.

Yadav et al. [24] presented project estimations in Functional Point and development on Agile. Calculate Functional Point for a given requirement with Weighted and Complexity Factors. Add the infusing factors for the FP estimates. The Algorithm proposed to convert Functional Point Estimates to Story Point Estimates.

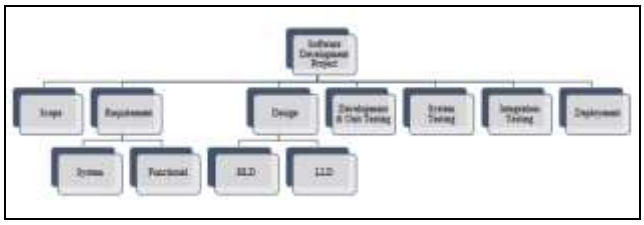
III. DIFFERENT ESTIMATION MODELS

Estimation techniques or model's usage is different from Organizations in projects includes software domains like Banking, Automobiles, B2B, B2A, governance, Now a day projects running in Agile methodology mainly. Scope,

Time, Cost, Quality, Resources, and Risk to estimate accurately are the significant challenges for Project manager. Below are the some of the estimation models described for better cost estimates.

A. Work Breakdown Structure

Break the task into a straightforward task-based in SDLC life cycle Project Management Body of Knowledge (PMBOK) defined Work Breakdown Structure (WBS) as "*deliverable oriented hierarchical decomposition of the work to be executed by the project team*." Tree representation is straightforward to understand the entire project tasks by the team. Various project life cycle includes Scope, Software Requirement Specification (SRS), Design, Coding and Unit testing, System Testing, Integration Testing, and Deployment. WBS provides the basic framework for cost estimation and schedule development tasks. Below are the tree structures of activities.



B. Simple Medium Complex (SMC)

Simple medium Complex is a fundamental estimation technique used by the Organization if they don't have the proper information in-hand. The ballpark estimation would be arrived based on the standard estimate for each requirement. Small templet must be defined to achieve the evaluation of the given requirement. For each complexity describes the person-days (PD's) for the organizational standard. The complexity data must arrive from old project data and organization experience to deliver the piece of work time. Below is the sample case study for estimation using SMC

TEMPLATE FOR ESTIMATES - Version 1.0	.NET or Java / JSP		
Classification of Units			
Effort required for Coding and Unit Testing	Simple	Medium	Complex
Person Days	2.5	4	8
The number of units that fall into the following category	11	7	3
Total Number of Units	21		
Effort required for Coding and Unit Testing	27.5	28	24
Total Effort for Coding and Unit Testing for project	79.5		
Phases		P Days	Measurement
Familiarization & Analysis	10	17.67	10%
Analysis and Design	25	44.17	25%
Coding	45	79.50	45%
Integration and System Testing	10	17.67	10%
Deployment Support	10	17.67	10%
Effort required for Project Phases		176.67	
Others (as applicable)			
Performance Testing(PPROD)		5.00	
Environment Set up		5.00	
Overall System Testing		5.00	
Pilot and Full-Scale Deployment support		0.00	
Non -functional Requirements		5.00	
Subtotal		196.67	
With Complexity Factor		196.67	1
Project Management and Co-ordination		19.67	10%
Contingency		19.67	10%
Final offshore effort for the Project PDS		236.00	

In the above template shows how to do estimates using SMC. Marked the predicted Simple, Medium and Complex factors and number of units to be considered for the requirement or project. The base Development and Unit testing effort derived. Based on the efforts, the other efforts would be calculated based on the standard effort percentage. In the above case study, we have taken Analysis for 10%, Design for 25%, Development and Unit testing for 45% (arrived from standard values), Testing for 10%, and Deployment Support for 10%.

C. Use Case Point

Use Case Points (UCP) derives the application size and effort from its use cases defined in the requirement phase. If the requirement is clear, the efforts would be more accurate. Gustav Karner, in 1993 done work on UCP, UCP analyzes and provides estimates based on the actors, scenarios, and technical and environmental factors.

Defined the variables and computed using perceived values and various constants. Formula to calculate the Use Case Point

UCP = TCF * ECF * UUCP * PF

Technical Complexity Factor (TCF), Environment Complexity Factor (ECF), Unadjusted Use Case Points (UUCP), and Productivity Factor (UF) used for UCP calculation.

Below are the steps to generate UCP estimations:

- Technical Factors compute and calculation
- Environmental Factors compute and calculation
- Unadjusted Use Case Points compute and calculation
- Productivity Factor computes and calculation
- Product of the variables

Technical Complexity Factor – Productivity of Technical Complexity Factor calculated with various application technical issues. Each factor weighted from 0-5 with relative impact. A 0 weight indicates the factor is irrelevant, and the 5 weights indicate the factor has the most impact. The factors consist of Distributed systems, Complex internal Processing, Performance, End-User Efficiency, Reusability, Portable, Easy to install, Easy to use, Easy to change, Concurrent, Special security features, direct access to third parties, User training facilities.

TCF factors calculate with productivity impact that various technical issues in an application. A 0 weight indicates the factor is irrelevant, and the 5 weights indicate the factor has the most impact. The factors are Familiarity with Unified Modeling Language, Object-Oriented Experience, Application Experience, Lead analyst capability, Motivation factors, Requirements stability, Working as Part-time, Programming language difficulty.

- Unadjusted Use Case Points (UUCP) - Two components used to calculate the Unadjusted Use Case Points
- Unadjusted Use Case Weight (UUCW) - Calculate the Number of scenarios or activities in certain use cases.
- Unadjusted Actor Weight (UAW) - Complexity of all the Actors in use cases combined.

Productivity Factor – Ratio of the Number of person-hours per use case point based on past projects. Industry-standard in-between 15-30. The value depends on the Organization productivity values.

D. Function Point Estimation

Functionality understanding of the project depends on how much smaller Functional Point Analysis made at the time of requirements gathering. Project Stakeholders, users, developers, and program managers easily understand the functionality of the system if the Functional Point Analysis correctly done. Mainly Transactional Functions and Data Functions:

Transactional Functions – Mainly contains three **components**: they are External Inputs (EI), External Outputs (EO), and External Inquiries (EQ). The transaction defined as to adds, modifies, deletes, retrieves, or processes information contained of these Transactional Functions components.

- External Inputs – Data retrieval from external applications to internal applications. This data from the outside used for maintaining internal logical files.
- External Outputs – Data contribution from internal applications to external applications. The data used to update the internal logical file values. Reports generated using the internal and external interface data and the reports used for other applications as input - these reports generated from internal valid files or external interface files.

- External Inquiry –Data retrieval from external interface files and internal logical files for output and input components. The input does not update the internal valid files or external interface files. Output data does not contain derived data.

Data Functions - Mainly contains two components: they are Internal Logical Files (ILF) and External Interface Files (EIF).

- Internal Logical Files – Logical data within the application boundary and maintained through external inputs.
- External Interface Files – Logical data uses for reference purposes only. The data available is entirely outside the application, and a different application maintains the data. Means that an external interface file is an internal logical file of another application

Low, Average, or High. File Types Referenced, Data Element Types, and Record Element Types used in Ranking calculation used for the classification of five significant components and factors mentioned. Total number of Internal Logical Files (ILFs) maintained, read, or referenced called as File Types Referenced (FTRs). EI / EO transaction called external interface files read or referenced. Unique user-recognizable non-recursive fields, including different vital attributes that maintained on ILF / EIF, called Data Element Type (DET). Record Element Type (RET) is in the subgroup of the ILF / EIF.
Transactional Functions ranking notated as Number of files updated or referenced (FTRs) and number of data element types (DETs) used.
Sample case study described for estimation using Functional Point Estimation. Unadjusted Factor (UAF) is generated based on Transactional Function and Data Functions.

Component Type	Component Complexity			
	Low	Average	High	Total
External Inputs	__*3=EI1	__*4=EI2	__*6=EI3	EI1+EI2+EI3
External Outputs	__*4=EO1	__*5=EO2	__*7=EO3	EO1+EO2+EO3
External Inquiries	__*3=EQ1	__*4=EQ2	__*6=EQ3	EQ1+EQ2+EQ3
Internal Logical Files	__*7=ILF1	__*10=ILF2	__*15=ILF3	ILF1+ILF2+ILF3
External Interface Files	__*5=EIF1	__*7=EIF2	__*10=EIF3	EIF1+EIF2+EIF3
Total UAF	(EI1+EI2+EI3) + (EO1+EO2+EO3) + (EQ1+EQ2+EQ3) + (ILF1+ILF2+ILF3) + (EIF1+EIF2+EIF3)			

	ILF			EIF			EI			EO			EQ		
Line Item	L	A	H	L	A	H	L	A	H	L	A	H	L	A	H
	7	10	15	5	7	10	3	4	6	4	5	7	3	4	6

After calculating the UAF, The Value Adjustment Factor (VAF) calculated on General System Characteristics (GSC's). The degrees of influence range between scale of 0-5. Data communication, Distributed functions, Performance objectives, heavily used configuration, Transaction rate, Online data entry, End-user efficiency, Online update, Complex processing, Reusability, Installation ease, Operational ease, Multiple sites, and Facilitate change are part of GSCs.

VAF = 0.65 + (Sum of degrees of the Influence of the fourteen GSCs / 100

Multiplying VAF to Unadjusted Function Point (UAF) to calculate final Function Points

FP = UAF * VAF

Functional Point Estimations are suitable, while the requirements are very much in detail. All the internal and external interfaces are evident along with database interactions.

E. Agile Story Point Estimation

Agile requirements are volatile. The requirement can change at any time in the sprint cycle. Estimating Agile is very complex due to the nature of the required change. Any project success is completing the project within the Budget and Time. According to The Standish Group 2018 Chaos Report [22], Agile projects success rate is more than

Waterfall (around 2X more), and 42% Successful, 50% Challenged, and 8% Failing for Agile projects. The agile projects success rate is 60% greater than non-agile projects.

Agile software development estimations are based on the relative size and measured in the Fibonacci scale of user story points. Agile Scrum is typically two to three weeks long and iteratively where the requirements are continuously improved, and development work carries out in the next subsequent sprints. The Fibonacci sequence consists [0,1,3,5…] number sequence. Agile used the Fibonacci sequence to achieve better estimates and planning by reducing complexity, effort, and planning of the project.
The most widely used Story Point Estimation is Planning Poker. Three methods of estimation techniques combined in Planning Poker

- Expert Opinion – The Expert is providing an opinion on how much time the task takes. The Expert provides an estimate in product backlog meeting with his / her experience or gut feel or intuition. Estimates generally done in the planning of the sprint. The Expert provides the forecast effort; the team accepted after a detailed discussion on the requirement. The process was accurate compared to other analytical methods in Agile.
- Analogy – It mainly for user story comparison. Compared the User story with undersized user story estimation implemented earlier or from the repository and provides the estimate correctly. Predicts accurate results as compared to other options. Sometimes the forecast estimate may go wrong due to the interpretation of data from earlier or history.
- Desegregation – Splitting the user stories into tiny user stores and estimate. The sprint contains two weeks, and hardly the resource gets a development time of five to seven days to develop. So, the bigger stores splits into n-number of smaller story points. This approach ensures many comparable stories.

Scrum recommends the Stories are estimated using Story Points. The team delivers the specific Story points for each sprint to achieve the working model. At the end of each sprint, the working model shown to the end-users for feedback. The provided feedback incorporates in consequent sprint as a new requirement.

The Story point estimation model introduced to avoid the person-hours estimation in Agile. Story point estimates have a more significant advantage on the other hand disadvantages due to lack of experience to effort estimate. As a result, the story point estimates are varying from team to team. By the multiple teams, the estimates vary, and more / less compared to other estimation techniques.

Simple formula proposed for estimating Agile projects with a combination of Team Productivity, Complexity Factor, Iterative velocity, Test Driven Development Factor (TDD) instead of following the Poker man Fibonacci scale. In this template-based approach, any person can calculate the estimations accurately. Below is the case study for the Agile Estimation

Step 1: Determine the Raw Efforts considering team productivity

Yrs of Experience	> 5	3 to 5	< 3
JSP,JS	6	8	12
MVC	4	5	6
Business Object	8	9	12
DAO	4	4	6
DB Changes	2	2	2
Build on Server	1	1	1

Step 2: Determine the Complexity factor

Technology	Known	New Version	Unknown
Requirement Complexity			
Low	0.1	0.2	0.3
Medium	0.3	0.4	0.5
High	0.6	0.7	0.8

Step 3: Determine the Iterative Velocity

Project Duration (in Months)	Iteration velocity	Pair Programming followed
0 - 4	2	Yes
4 - 8	1.6	Yes
> 8	1.25	Yes
NA	1	No

Step 4: Determine the factor for Test Driven Development (TDD factor) - Factor Considered: 0.12

Step 5: Calculate the Adjusted Estimation using the formula

Adjusted Estimates = Raw Effort * (1 + Complexity Factor + TDD factor)

Step 6: Calculate Actual Efforts and Determine the Story Points

Actual Efforts = Iteration Velocity * Adjusted Estimates

IV. CONCLUSION AND FUTURE WORK

Research work carried out to study on various popular estimation techniques used in Organizations for the last decade. Also derived templates for every estimation model explained. The Organization adopts different estimation models according to customer goals. Some customer asks the project to execute in an Agile or Iterative methodology. So, the project team selects an Agile estimation template to estimate the plan provided and come up with accurate estimates. If the customer wants to execute the project in Waterfall, use any of the models like Work Break Down, Use Case Point, Simple Medium Complex Factor, and Functional Point. The benefit of the proper estimates is that it reduces the risk of failing the project.

Due to Organizations are moving towards Agile adoption, further work has to carry out to arrive at different Effort Estimation Model for Agile projects using various complex factors into consideration. Provide developer friendly for estimating the task level in Agile projects.

REFERENCES

[1] Adnan, M., & Afzal, M. (2017). Ontology based multiagent effort estimation system for scrum agile method. *IEEE Access*, 5, 25993-26005.

[2] Ahmed, M., Malik, B. H., Tahir, R. M., Perveen, S., Alvi, R. I., Rehmat, A., ... & Asghar, M. (2018, July). Estimation of Risks in Scrum Using Agile Software Development. In *International Conference on Applied Human Factors and Ergonomics* (pp. 111-121). Springer, Cham

[3] Arifin, H. H., Daengdej, J., & Khanh, N. T. (2017, March). An Empirical Study of Effort-Size and Effort-Time in Expert-Based Estimations. In *2017 8th International Workshop on Empirical Software Engineering in Practice (IWESEP)* (pp. 35-40). IEEE

[4] Bik, N., Lucassen, G., & Brinkkemper, S. (2017, September). A reference method for user story requirements in agile systems development. In *2017 IEEE 25th International Requirements Engineering Conference Workshops (REW)* (pp. 292-298). IEEE.

[5] Canedo, E. D., & Costa, R. P. D. (2018). Methods and metrics for estimating and planning agile software projects. *Twenty-fourth Americas Conference on Information Systems, New Orleans, 2018*

[6] Choetkiertikul, M., Dam, H. K., Tran, T., Pham, T., Ghose, A., & Menzies, T. (2018). A deep learning model for estimating story points. *IEEE Transactions on Software Engineering*, *45*(7), 637-656.

[7] Hacaloglu, T., & Demirors, O. (2019, August). Measureability of functional size in Agile software projects: Multiple case studies with COSMIC FSM. In *2019 45th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)* (pp. 204-211). IEEE.

[8] Hannay, J. E., Benestad, H. C., & Strand, K. (2018). Agile Uncertainty Assessment: For Benefit Points and Story Points. *IEEE Software*.

[9] Hasan & Khan. (2019). Software Development Methods – Properties and Advances. *International Journal of Computer Applications Volume 178 – No. 53*

[10] Hoda, R., Salleh, N., & Grundy, J. (2018). The rise and evolution of agile software development. *IEEE software*, *35*(5), 58-63

[11] Ibrahim, K. S. K., Yahaya, J., Mansor, Z., & Deraman, A. (2019, July). The Emergence of Agile Maintenance: A Preliminary Study. In *2019 International Conference on Electrical Engineering and Informatics (ICEEI)* (pp. 146-151). IEEE.

[12] Iqbal, J., Omar, M., & Yasin, A. (2019). The impact of agile methodologies and cost management success factors: an empirical study. *Baghdad Science Journal*, *16*(2), 496-504.

[13] 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*.

[14] Khatri, S. K., Malhotra, S., & Johri, P. (2016, September). Use case point estimation technique in software development. In *2016 5th international conference on reliability, infocom technologies and optimization (trends and future directions)(ICRITO)* (pp. 123-128). IEEE.

[15] Moyo, S., & Mnkandla, E. (2020). A Novel Lightweight Solo Software Development Methodology With Optimum Security Practices. *IEEE Access*, *8*, 33735-33747.

[16] Munialo, S. W., & Muketha, G. M. (2016). A review of agile software effort estimation methods. International Journal of Computer Applications Technology and Research Volume 5–Issue 9, 612-618, 2016

[17] Prakash, B., & Viswanathan, V. (2017). A, "Survey on Software Estimation Techniques in Traditional and Agile Development Models". *Indonesian Journal of Electrical Engineering and Computer Science*, *7*(3), 867-876.

[18] Raunak, M. S., & Binkley, D. (2017, September). Agile and other trends in software engineering. In *2017 IEEE 28th Annual Software Technology Conference (STC)* (pp. 1-7). IEEE.

[19] Sarwar, A., Hafeez, Y., Hussain, S., & Yang, S. (2020). Towards Taxonomical-Based Situational Model to Improve the Quality of Agile Distributed Teams. *IEEE Access*, *8*, 6812-6826.

[20] Shams, A., Bohm, S., Winzer, P., & Dorner, R. (2019, July). App Cost Estimation: Evaluating Agile Environments. In *2019 IEEE 21st Conference on Business Informatics (CBI)* (Vol. 1, pp. 383-390). IEEE.

[21] Shimoda, A., & Yaguchi, K. (2017, July). A Method of Setting the Order of User Story Development of an Agile-Waterfall Hybrid Method by Focusing on Common Objects. In *2017 6th IIAI International Congress on Advanced Applied Informatics (IIAI-AAI)* (pp. 301-306). IEEE

[22] The Standish Group 2018 Chaos Report. URL: https://vitalitychicago.com/blog/agile-projects-are-more-successful-traditional-projects/ Retrieved on April 04, 2020

[23] Vyas, M., Bohra, A., Lamba, D. C., & Vyas, A. (2018). A review on software cost and effort estimation techniques for agile development process. *International Journal of Recent Research Aspects*, *5*(1).

[24] Yadav, A., & Sharma, A. (2018, May). Function Point Based Estimation of Effort and Cost in Agile Software Development. In *Proceedings of 3rd International Conference on Internet of Things and Connected Technologies (ICIoTCT)* (pp. 26-27).

[25] Zozas, I., Bibi, S., Ampatzoglou, A., & Sarigiannidis, P. (2019, August). Estimating the maintenance effort of JavaScript Applications. In *2019 45th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)* (pp. 212-219). IEEE