Journal of Theoretical and Applied Information Technology

15th December 2023. Vol.101. No 23

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

AN EXTENSION TO THE TRIANGULATION TECHNIQUE FOR EFFORT ESTIMATION IN SOFTWARE DEVELOPMENT PROCESS

RAID ZAGHAL[1], ODAY AL-WAHSH[2], SAEED SALAH[3], HANA SHABANEH[4]

[1]Ph.D., Department of Computer Science, Al-Quds University, Jerusalem, P.O. Box 20002

[2]M.Sc., Department of Computer Science, Al-Quds University, Jerusalem, P.O. Box 20002

[3]Ph.D., Department of Computer Science, Al-Quds University, Jerusalem, P.O. Box 20002

[4]M.Sc., Department of Computer Science, Al-Quds University, Jerusalem, P.O. Box 20002

Email: zaghal@staff.alquds.edu, sasalah@staff.alquds.edu, oday.alwahsh@students.alquds.edu, hshabaneh@staff.alquds.edu

ABSTRACT

Effort Estimation (EE) is one way to evaluate a software project and understand its schedule and budget. It is one of the most important aspects of the software development life cycle. The failure of not recognizing the accurate effort estimation may lead to increase the financial costs of the companies and their clients which will cause negative impact on their job duties and their future marketing plans besides the client's disappointment and dissatisfaction. The Palestinian IT sector is one of the most developing and promising sectors. However, the studies that investigate the methods and techniques of effort estimation are most likely missing. For that reason, we were motivated to study the status of the software development companies in Palestine to better understand how the technical teams estimate the needed effort of their software projects. The purpose of this study is (1) to survey the existing effort estimation techniques used by prominent Palestinian software development companies and analyze their practices, and (2) to suggest an appropriate effort estimation technique that can suite the nature and needs of these companies and to validate this technique via real application within actual software projects in a selected subset of these companies. Based on our survey and analysis, we have selected an existing effort estimation technique called (Triangulation) as the most appropriate EE method for small companies and for the Palestinian software development industry. After that, we have designed an extension of this technique to (fine-tune) the company's exact needs and provide it with some flexibility to make some changes and adjustments (e.g., decrease project delivery time by increasing resources). Moreover, this technique was applied on the Palestinian software projects to validate its results. We believe this study can be a valuable resource for Palestinian software development companies; as they can use it as a guideline to help them get better and more accurate effort estimates, which in return can reduce costs and provide better and more accurate scheduling and staffing needs.

Keywords: *Software Development, Effort Estimation Technique, Ensemble Effort Estimation, Software Engineering, Agile*

1. INTRODUCTION

The IT sector in Palestine is rapidly growing and holds great promise. Still there is a lack of studies on effort estimation approaches and techniques in the Palestinian IT sector. It has been observed that software development companies in Palestine typically have small to medium-sized development teams, with up to 20 programmers. This motivated us to conduct a case study of Palestinian software development companies to better understand how technical teams approach effort estimation and which challenges they face. Our goal is to provide the Palestinian software development companies with practical recommendations and ideas to improve their effort

estimation and save valuable resources. This study suggests using triangulation, an existing technique, to improve the accuracy of effort estimation for software development companies. Utilizing this technique can assist companies in accurately estimating costs and achieving superior outcomes, particularly in the context of small and medium-sized projects. In addition, we have designed and tested an extension to the triangulation method that enables Palestinian companies to adjust project delivery time by fine-tuning resources and parameters. This can lead to more accurate cost predictions. effort estimation (EE) and accuracy are major challenges for software development projects [1-3]. The Palestinian IT sector is a promising industry with a respectable number of mature and new companies working in software development [4]. This fast-growing sector employs thousands of professionals.

Previous studies have shown that software companies in Palestine face a 25% overrun in project costs due to inaccurate effort estimation for software projects [5]. However, there is a lack of research on how to improve EE in Palestine. Furthermore, many Palestinian companies do not have enough knowledge of the available tools and techniques for estimating effort, which affects their project planning (e.g., scheduling, human resources, budgeting, etc). As a result, these companies often experience delays in delivering their products because they fail to estimate the required effort correctly from the beginning. This leads to increased financial costs for both the companies and their customers, which negatively impacts their businesses and marketing plans. On the other hand, Palestinian engineers who work in agile development methodology using SCRUM need to estimate the time needed to complete their assigned tasks in each sprint. Therefore, they need a reliable way to calculate the time for each task and deliver it on time. We conducted a survey [6] and found that most software engineers in Palestinian companies do not use any formal techniques for estimating effort. Instead, they rely on common knowledge and personal experience. This indicates a lack of knowledge of effort estimation methods among engineers and/or project managers in Palestinian companies. For that reason, this research aims to address this issue and find a suitable effort estimation model for these companies, as well as for other countries with similar status and software industry requirements. Since there are very few published studies on effort estimation methods in Palestine (and similar countries), we conducted this study to improve the accuracy of effort estimation in software projects. We selected a technique that matches the variables and factors of projects in the Palestinian market, verified and modified it to increase its estimation accuracy, and then validated it and linked it to our proposal. In addition, we made some recommendations that may positively affect these companies and the quality of their products and estimations. We have observed a high percentage of inaccurate or improvised estimations that had resulted in cost overruns. We think that no suitable model has been proposed to improve the accuracy of effort estimations for our software industry. Therefore, our goal is to suggest an appropriate effort estimation model/technique based on a well-known model with our own extension/modification. This will help us to obtain accurate results that meet the needs of Palestinian companies.

The main objective of the study described in the paragraph is to improve effort estimation practices in the Palestinian IT sector. The study aims to provide Palestinian software development companies with practical recommendations to enhance their effort estimation techniques, thus saving resources and increasing the accuracy of project cost predictions. It introduces the use of triangulation as a technique to enhance effort estimation for small and medium-sized projects, along with a tested extension to this method to allow companies to adjust project delivery times effectively.

The main contribution of this research is to propose a reference model for the Palestinian companies and software engineers that will help them to estimate the effort required to complete their software development projects accurately before they start. By using the appropriate effort estimation technique and verifying its validity and accuracy level, they can improve their work-schedule and the overall quality of their services/products. Additionally, we added an improvement to the Triangulation method, which will enable Palestinian companies to adjust the project delivery time and calculate the project cost more precisely. The rest of the paper is organized as follows. Section 2 reviews the related work. Section 3 describes the methodology. Section 4 presents and discusses the results. Finally, in Section 5 we conclude the paper and outline some future work.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

2. RELATED WORKS

This section provides an overview of some related research on effort estimation techniques in software development. Following that, we present effort estimation methods to explain the rationale behind choosing a particular technique.

2.1. Effort Estimation Techniques In Software Development

Reza and Chirra [9] presented a survey of various software cost estimation techniques. The authors discussed the importance of accurate cost estimation in software development and reviewed several traditional and modern cost estimation methods. Traditional methods include the COCOMO model, function point analysis, and constructive cost model, while modern methods include artificial neural networks, fuzzy logic, and machine learning. The paper also examined the advantages and disadvantages of each technique and concluded with a discussion on the challenges and future directions of software cost estimation research. Popli, *et al.,* [10] discussed the challenges of cost and effort estimation in agile software development and presented a framework for estimating cost and effort in agile projects. The framework includes using historical data and expert judgment to estimate initial requirements and adjusting the estimates based on feedback and changes throughout the project. The authors also suggested using story points as a unit of estimation and involving the entire team in the estimation process.

Prakash, *et al.,* [11] presented a survey of software estimation techniques in traditional and agile development models. The survey includes various techniques, their strengths and weaknesses, and the factors that influence estimation accuracy. The article concluded that combining techniques and involving multiple stakeholders in the estimation process can lead to more accurate estimations in both traditional and agile development. Zarour, *et al.,* [12] discussed the challenges faced by software development teams in estimating project costs and duration in industrial contexts. The authors conducted an exploratory multiple case study to examine the estimation techniques used in five companies. The study found that estimation techniques varied widely among companies and were influenced by factors such as team experience, project size, and development methodology. The authors suggested that estimation techniques should be tailored to each project and that teams should be trained on the most effective techniques for their specific context. Gumaei, *et al.,* [13] presented an empirical study on software cost estimation in the Saudi Arabian software industry. The study surveyed 40 software companies to understand the current state of software cost estimation practices and the challenges faced by these companies in the estimation process. The results showed that most companies used expert judgment as the primary estimation method and that the accuracy of estimates was affected by various factors such as project size, complexity, and technology used. The study recommended the adoption of formal estimation methods and process improvement to enhance estimation accuracy.

Kafle [14] presented an empirical study on software test effort estimation that investigates the estimation process of software testing in industrial settings. The study concluded that test estimation in practice is mostly based on intuition, heuristics, and experience rather than formal methods, and suggested the need for improved estimation techniques and training. Tailor, *et al.,* [15] conducted a review of 25 studies that compared software cost and effort estimation methods. They found that some methods outperformed others in terms of accuracy, but that the best method varied depending on the specific context and project. They also found that combining multiple estimation methods tended to produce more accurate results. Mahmood, *et al.,* [16] proposed an ensemble model to improve the accuracy of software development effort estimation. The model combines three techniques - Support Vector Regression (SVR), Multilayer Perceptron (MLP), and Random Forest (RF) - to achieve better performance. The study concluded that the proposed model produces more accurate estimates compared to individual techniques.

Table 1. Comparative Analysis of Effort Estimation Methods

Method	Type	Advantages	Disadvantages
Analogy	Non-algorithmic	- Require past experiences. - No need of new resources	- Required large amount of data - Sometimes similar problem pattern not available
Expert based	Non-algorithmic	- Due to expert experiences fast process	- Chances of biased decisions
Bottom-Up	Non-algorithmic	- Stable as the estimation errors in the various components might balance out	- Time - Inaccurate data - Require lot of data
Top-Down	Non-algorithmic	- Faster and easier - Low level costs	- Justifies decisions are less - Less stability
COCOMO	Non-algorithmic	- Repeatable results can be generated - Easily modifying input data - Easy filtered formula - Elear results	- Inaccurate cost estimate - Size uncertainty - Require huge data - Practically not good
Function Point	Non-algorithmic	- Estimation based on requirement designs - Specifications - Tool independent	- Not considered good enough
Neural Network	Machine learning	- Superior cost estimate - Consistent estimate	- Training data required - Not standard guidelines

In software development, researchers and experts aim to determine the most accurate technique for estimating effort and other project attributes. In 2017, the Project Management Institute conducted a survey that found 69% of software projects achieved their original goals and business priorities, but 43% exceeded their initial budgets, 48% were delivered late, and 32% failed due to budget loss [17]. IBM-PMO Consultants conducted a survey of 1,500 change management executives and found that only 40% of projects meet schedule, budget, and quality goals. Including, underestimating project complexity was identified as a challenging factor in 35% of the projects.

2.2. Effort Estimation Methods

Software effort estimation refers to the process of predicting the most accurate amount of time and cost required to develop a software project. It can be roughly estimated using the following formula (Estimation = Duration of task completion + task completion Cost. The absence of a proper estimation process may result in inaccurate results [18]. The process of effort estimation involves the adoption of certain resources and methods to produce inputs and outputs. To estimate effort, quantitative and expertise data are both used, and the quantity and quality of the data are important factors [19]. Effort can be estimated using various methods. The primary methods include algorithmic estimation, in particular is based on mathematical models that produce a function of cost factors (e.g., COCOMO, Puntametc), expertise estimation, which relies on expert judgment when collecting data is complex (e.g., Delphi, Rule-based), and learning-oriented methods, which are more advanced versions of algorithmic and expertise estimation that use conclusions from previous projects, examples, and current knowledge (e.g., Neural and Analogy). Other methods include price to win, bottom-up, and top-down approaches (e.g., Triangulation) [20]. Price to Win: Estimates according to budget of the software. Bottom-up: Estimate and test each component of the system. Top-Down: Estimate and test the entire system. Table 1 shows a comparative analysis of prominent estimation methods.

According to [21], several factors can be used to evaluate estimation techniques, such as the model's structure, cost, ability to handle missing or past data, ability to consider uncertainty, accuracy, use of agile methods, dynamics, ability to handle changing requirements, time consumption, applicability, and trustworthiness of the results. Research on the Palestinian market shows a 25% overrun in project costs due to inaccurate effort estimations for software projects, at the same time there is a lack of knowledge among Palestinian companies on how to estimate effort using published tools and techniques, which affects their project planning. To address this issue, a study was conducted to suggest a technique for estimating

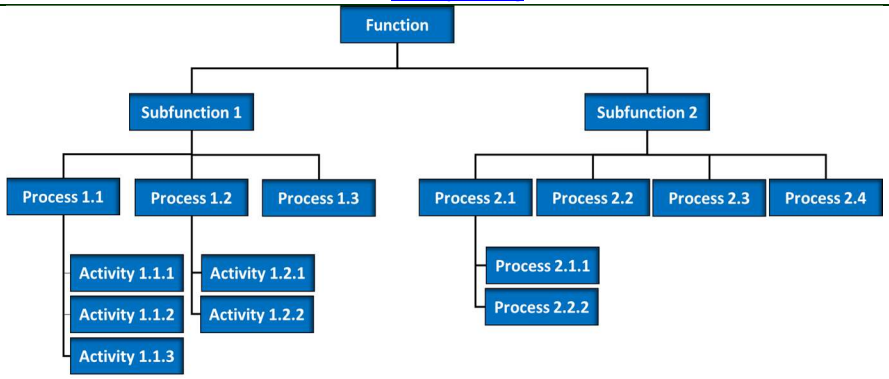

Figure 1. An Example of a Functional Decomposition Diagram

effort, which is triangulation. The Triangulation method aligns with the collected data from Palestinian companies and supports effort and time-consuming approaches, making it suitable for Agile methodology. Other techniques, such as COCOMO, do not consider essential project parts like Front-end, Back-end, and QA, which Triangulation does.

3. METHODOLOGY

3.1. Basic Concepts

This section offers an overview of the fundamental concepts required to comprehend this research. It discusses the basic principles of effort estimation in software engineering and introduces Triangulation, the technique used in this study.

1) Document Analysis: Document analysis is a technique for eliciting business analysis information such as understanding context and requirements. It can be used to analyse various forms of project requirements including RFPs, Tenders, Business requirements and technical specifications. This technique can also validate findings from other elicitation methods like interviews and observations. Data mining is one approach to document analysis that helps identify patterns, categorize data, and discover opportunities for change [7].

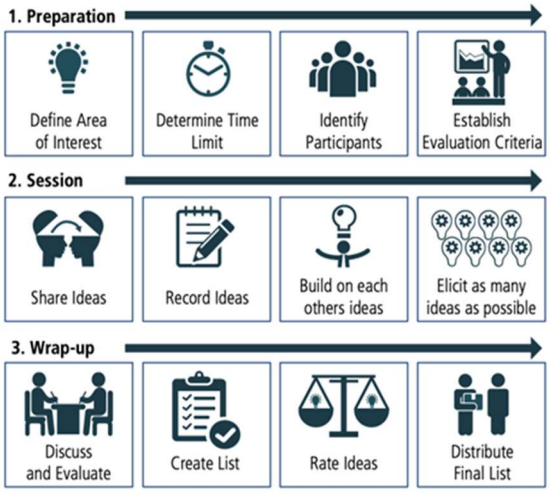

Figure 2. Brainstorming steps used in creative thinking and analysis

2) Benchmarking: Benchmarking involves evaluating previous projects or studies with similar objectives to improve the current study. By conducting benchmarking, researchers can build upon previous work and enhance their project with additional data, effort, and resources. This process also includes reviewing prior risk analysis to estimate the effort required for the current study [8].

3) Functional decomposition: Functional decomposition is a technique that breaks down a process into smaller parts to make it easier for the recipient to understand. Each part is analysed in detail and components may be merged into specific categories. This technique helps the recipient understand the hierarchy of sub-elements and how they relate to the larger element. The diagramming of functional decomposition depends on the hierarchy of sub-components, with each having only one parent component [7]. Figure 1 presents an example of the functional decomposition model.

4) Brainstorming: Brainstorming is a technique that fosters creative thinking and generates numerous new ideas for further analysis. It aims to produce a wide range of solutions to a problem and is most effective when conducted in a group setting, drawing on the experience and creativity of all members. However, individual brainstorming can also be effective in sparking new ideas and enhancing creativity [7]. Figure 2 describes the brainstorming steps.

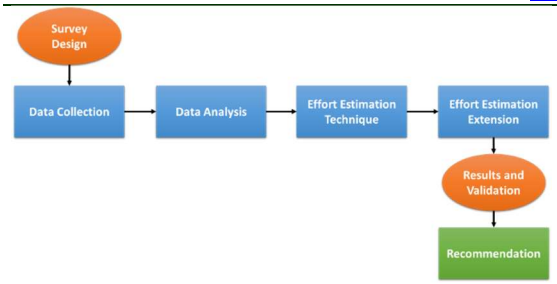

Figure 3. The Research Steps in The Adopted Methodology

5) Calendar day: A calendar day refers to the total number of days according to the calendar, including holidays, weekends and working days. For example, if we consider 12 working days starting from 5 June 2022, then the end of these 12 working days according to the calendar day will be 16 June 2022 with weekend days included.

6) Business day: A business day refers to the total number of working days only, excluding vacation days and weekends. One working day is equivalent to approximately six working hours. For example, if we consider 12 working days starting from 5 June 2022, then the end of these 12 working days according to the business day will be 20 June 2022 without weekend days included. Next, we explain the rest of the methodology, experiments, and the verification phase.

3.2. Research Methodology

This section proposes an effort estimation technique and an extension of the selection technique suitable for Palestinian companies to calculate effort and cost estimates for their software projects. This section is divided into five subsections. Section 4.2.1 examines the state of the Palestinian market by surveying techniques and methods used in estimating software project effort. Section 4.2.2 collects data from previously implemented software projects in Palestinian companies to pilot the selected technique. Section 4.2.3 analyses data collected from Palestinian companies. Section 4.2.4 identifies the best technique for software projects in Palestinian companies. Section 4.2.5 modifies the selected technique to obtain additional information about effort estimation.

1) Current EE methods and techniques used in the Palestinian companies: This stage aims to improve or suggest new methods for estimating effort and cost in software projects by understanding techniques and methods currently used by Palestinian companies. We distributed a Google Forms survey consisting of multiple-choice questions designed to be brief (less than ten minutes) and accessible for respondents to add more information. We have reached nearly 112 people who have filled out the survey on Google Forms. Table 2 provides a summary of relevant survey questions and their corresponding results.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Table 2. Survey Questions and Findings

Question	Findings
How many employees work in your company?	Most companies are small; only 12.5% of the companies had 250+ employees.
Have you ever participated in/run level of effort estimation or participated in/run level of effort estimation as part of your current job?	84.8% reported they did not use any effort estimation methods!
If I suggested to you a technique in calculating the effort estimation, would you use it?	98.9% said they will indeed use it!
In case you did use an EE technique; which one did you use?	35% said they used **Triangulation**, 34% said they used **Benchmarking**; and the rest used a mix of **intuitive** methods and **experience**.
What do you think about your level of effort estimation?	42% said they are **Advanced beginner** level, 24% said they are **Competent** level; while the rest are **Proficient** or **Expert**.
How do you get the system requirements?	41% use **RFPs** (Request for Proposal); 23% use **Tenders**; 76% use **Business Requirements**; and 58% use **technical requirements**.
Have you previously encountered an inaccurate effort estimation in one of the projects? If yes, what is the percentage of deviation from the actual time.	Most of them reported a deviation between 6 – 11%.
What is the unit of measurement for the effort rating output of the system?	64% used **workdays**; 52% used **hours**, while a small percentage used a mix of **weeks** and **months**.
Do you calculate risk factors when calculating the effort estimate? If the answer is yes, what is the percentage?	Most of them reported a percentage between 5 – 12%.
How many hours does your effort estimation technique take?	Most of them reported around one workday (8 hours) of EE overhead per two weeks of development.

Our survey found that 95 out of 112 software engineers and project managers in Palestinian companies do not use any effort estimation techniques. When asked if they were motivated to use a technique to calculate effort estimates, 94 out of 95 responded yes and one responded no. We also gathered information on the techniques used by those who reported using an effort estimation technique. Figure 3. illustrates the methodology model's steps followed to achieve the purpose of this study.

2) Data collection: We collected data from previously implemented software projects in Palestinian companies to apply and verify the suitability of our effort estimation technique. Some companies agreed to provide data while others declined due to privacy concerns. The collected projects varied in size, ideas, programming languages used and company size (start-ups or outsourcing companies such as ASAL and Crypton). The data included project requirements, number of developers, techniques used, project duration and other relevant information for our EE technique.

3) Data analysis: We analysed the data collected from Palestinian companies and extracted relevant information for applying our effort efficiency technique. The analysis was summarized in Table 6 showing information on 14 projects collected from Palestinian companies. Note that we did not obtain permission to display more detailed data.

4) Selection of appropriate effort estimation techniques: In this section, we will identify and select an effort estimation (EE) technique that fits the data collected from Palestinian software companies. The technique we will apply is the Triangulation method. Triangulation is a technique used for estimation in software development. It involves comparing a user story to two previously estimated similar intent user stories. For example, if a new story is bigger than a previously estimated story at six-story points and smaller than one estimated at 10, choosing eight would be a good strategy. This technique was chosen over others for several reasons. Firstly, the data collected from Palestinian IT companies is well-suited for the Triangulation method. Secondly, this technique considers three main parts of a web project: front-end, back-end and Quality Assurance (QA).

Thirdly, it works well with the Agile methodology. The Triangulation method supports a time-consuming approach and is less complex to use than some other techniques. Factors for choosing estimation techniques have also been mentioned in a survey to estimate the cost effect of different programmers [21]. The researchers stated that no

Table 3. Simulation of The Proposed Framework

Item	Estimation	Category
Estimated Time	60 Hours	Misc. Parameters
No. of Junior Developers	1	
No. of Mid-Level Developers	0	
No. of Senior Developers	0	
No. of Junior QA	1	
No. of Mid-Level QA	0	
No. of Senior QA	0	
Actual working hours per day	6.5	
Management overhead	0.2	
Acceptance test factor	0.2	
Risk factor	0.4	
Reduction Factor	0	
Total number of resources	2	
Acceptance test period	12	
Estimated time including acceptance test	72	Cost
Risk hours	28.8	
Estimated time including risk factor	100.8	
Junior Developer man hours	100.8	
Mid-Level Developer man hours	0	
Senior Developer man hours	0	
Junior QA man hours	100،8	
Mid-Level QA man hours	0	
Senior QA man hours	0	
PM man hours	20.16	
Calendar days	23.3	Deadline
Calendar months	0.775	

single method is more accurate than others and each method affects different factors.

5) Extension of the triangulation method: In this section, we will explain the framework that we have added to the Triangulation method. This framework was developed based on our experience and the experiences of project managers in Palestinian companies. It helps companies adjust project delivery time and cost by considering resources such as the number of developers, quality engineers and project managers. The framework is used during the effort estimation phase for a project. After calculating the time required to complete the project using the Triangulation method, if the client wants to complete it earlier than originally planned, our extension can be used by setting certain variables representing vital resources like people and money. For example, adding more software engineers can change the delivery time. These inputs are entered into this framework and mathematical operations are performed to calculate a new time to finish the project and determine its price based on added resources. To better understand how this framework works, we have included a table (Table 3) that shows simulations of how it can be used to calculate new project delivery times and costs.

Here we explain all the variables in the table and the output:

- **Estimated time**: This value is obtained using the Triangulation method and represents the estimated time for the project. It is measured in hours, days, or estimation units and does not include the risk factor.
- **Number of developers**: This refers to the proposed number of developers to be added to the project to reduce delivery time. It includes Junior, Mid-Level, and Senior Developers with varying costs based on their experience.
- **Number of QA engineers**: This refers to the number of QA engineers for the project. Any change in this number affects the project's cost and delivery time. The costs of QA engineers vary based on their experience.
- **Working hours per day**: This refers to the team's daily working hours, usually 6.5 hours based on a business day.
- **Management overhead**: This refers to the daily working hours for the project manager. A project manager can be added for a full or a third of a working day.
- **Acceptance test factor**: Acceptance testing is performed after system testing and before making the system available for use. It is a period during which the client checks that the system meets all requirements. This period often affects the calendar day and must be considered in effort estimation, often affecting 10% of the total estimated time.
- **Risk factor**: This value is obtained using the Triangulation method and is an input for the technique.

- **Reduction factor:** This represents the percentage reduction in project delivery time when additional developers are added. For example, adding two developers does not necessarily reduce the time by half. Adding one developer may only reduce the time by 35%. This value also depends on whether the system is modular or not. Some projects may not be affected by adding another developer if they are not modular. The reduction factor also considers tasks that can be worked on asynchronously and those that depend on each other. The value of the reduction factor decreases as the number of resources increases.
- **Total number of resources:** This value is calculated by adding the number of developers and the number of QA personnel.
- **Acceptance test period:** This value is calculated by multiplying the estimated time value by the acceptance test factor.
- **Estimated time including acceptance test:** This value is calculated by adding the estimated time and the acceptance test period.
- **Risk hours:** This value is calculated by multiplying the estimated time including acceptance test by the risk factor.
- **Estimated time including risk factor:** This value is calculated by adding the estimated time including acceptance test and the risk hours.
- **Total developers man hours:** This value is calculated by determining the number of working hours for developers based on their experience level. The equation for calculating the man hours for junior, mid-level, and senior developers is as follows:

*(Junior/Mid-Level/Senior) developer man hours = Estimated Time Including Risk Factor * Number of (Junior/Mid-Level / Senior) Developers * (1 - Reduction Factor)*

- **Total QA man hours:** This value is calculated by determining the number of working hours for QA engineers based on their experience level. The equation for calculating the man hours for junior, mid-level, and senior QA engineers is as follows:

*(Junior/Mid-Level / Senior) QA man hours = Estimated Time Including Risk Factor * Number of (Junior/Mid-Level/Senior) QA * (1 - Reduction Factor)*

- **PM man hours:** This value is calculated using the following mathematical equation:

*PM Man Hours = Estimated Time Including Risk Factor * Management Overhead * (1 - Reduction Factor)*

- **Calendar days:** This value is calculated using the following mathematical equation:

*Calendar Days = Estimated Time Including Risk Factor * 1,5 * (1 - Reduction Factor) / Actual Working Hours per Day*

- **Calendar months:** This value is divided by the value of the calendar days by 30 days.

Calendar Months = Calendar Days / 30

4. EXPERIMENTAL RESULTS AND ANALYSIS

In this section, we present our experiments and their results, along with a method to validate these results. Section 4.1 details the application of the Triangulation method, including the projects it has been applied to and its results. Section 4.2 explains the application of the extension of Triangulation method, including the projects it has been applied to and its results. Finally, in Section 4.3, we will show the verification of both the triangulation technology and its extension.

4.1. Software Project Inputs

First, we will describe the various forms of inputs for any project. This includes the initial project description, which outlines its goals and objectives and is delivered to a software company for review. Inputs can take the form of a Request for Proposal (RFP), Tender, Business Requirements or Technical Specification. Table 4 shows the inputs for projects and their level of detail. The higher the level of detail, the more accurate the effort estimate will be due to the clarity of the project's details.

Next, we briefly explain these concepts:

- **Request for Proposal (RFP):** An RFP describes a project in detail so potential vendors can submit proposals for the

Table 4. Input Forms of a Software Project.

Input	Level of Detail
Request for Proposal (RFP)	Low
Tender	High
Business Requirements	Moderate
Technical Specifications	High

development process and technology. It allows companies to address product development challenges by collecting ideas from several vendors [22]. A detailed RFP for software development allows issuers to compare vendor proposals and evaluate different ways to solve a problem. Companies use an RFP to exchange project details and set up partnerships. It is beneficial for both issuers and vendors as it allows companies to get several different solutions and decide which would best solve their problem. With a standardized RFP, it becomes easier to explain a project and receive realistic estimates. Overall, the RFP document provides context that vendors can use to provide a more accurate timeframe, project scope, price range, and generate tailor-made offers [22].

- **Business Requirements:** The software development life cycle's business requirements phase makes it possible for end users' needs to inform the design of a future system. Business analysts or product owners who analyze business activity and serve as subject matter experts typically record business requirements. Requirements are essential to ensure that all stakeholders and team members are on the same page as the software development group. They act as a starting point for a project's development process and keep all team members aligned to a single goal. High-quality business requirements documentation helps keep projects within budget and on schedule. Defining a company's business requirements involves considering the tasks and goals of the business and customers' needs. This can be complex and requires input from various people throughout the organization. Tools and steps can be taken during the requirement gathering process to achieve the best results [23].

- **Technical Specification:** A technical specification is a detailed document that describes all technical procedures related to product development. It covers vital information about the development process. A well-written technical specification using a good template serves as a guide for product development [24].

Table 5. The Triangulation Main Steps For Time Estimation

Steps	Notes
Functional decomposition	Missing items increase the risk
Running Triangulation technique benchmarking (optional)	Benchmarking allows for more accurate estimate
Getting the total	Estimated cost as computed by the Triangulation technique
Multiply by a risk factor	10% - 35%
Record assumptions	Record assumptions about the business, the development environment, and other factors.

4.2. Steps to Estimate Time Using The Triangulation Method

In this section, we will present the main steps to estimate time using the Triangulation method. It is important to note that these steps are interdependent. The output of each step serves as the primary input for the next step. Table 5 outlines these steps.

- **Functional Decomposition:** Functional decomposition involves breaking down the input into smaller components. For example, if the input is a technical specification, we can divide the system into smaller parts or items.

  This results in a subdivided output that can be used in the next step. Additionally, breaking down the input into smaller items can help determine the level of effort required for each item and its deliverables. The person responsible for breaking down the input into smaller items should be an expert in the field, such as a team leader or senior engineer. However, there is a risk associated with functional decomposition; important details may be overlooked.

- **Implementing the Triangulation method:** The Triangulation method is based on the rule of thirds. This technique involves three software engineers with different levels of

seniority. Playing cards are distributed according to the Fibonacci sequence, considering the varying seniority levels of the engineers. The Fibonacci sequence starts at 1 and ends at 8. It is not possible to exceed 8 because any items exceeding this number must be further decomposed using functional decomposition. To illustrate the Triangulation method, we consider a scenario involving three engineers for a specific item.

- Scenario (2,2,2): All three engineers select the same card, in this case 2. The estimated time for this item is therefore 2.
- Scenario (2,3,3): Two engineers select 3 while one engineer selects 2. The estimated time for this item is 3.
- Scenario (2,2,3): Two engineers select 2 while one engineer selects 3. In this scenario, the engineer who selected 3 must justify their choice before replaying.
- Scenario (2,3,5): One engineer selects 2, another selects 3, and the third selects 5. In this case, the engineers who selected 2 and 5 must justify their choices before replaying. If the results remain the same after replaying, the estimated time for this item is 5. This is because choosing a higher estimate can help reduce the risk of underestimating the time required for the item.

- **Benchmarking:** Benchmarking is an optional step that can be used to verify the results. This technique involves comparing the results of previous projects with the current one. If there are similar projects, benchmarking can be used to confirm the results. However, if the project is new and there are no similar projects to compare it to, benchmarking cannot be used. This is why it is considered an optional step.
- **Calculating the Total:** After estimating the time required for each item using the Triangulation method, the next step is to sum up all the results to obtain a total. This total represents the estimated time required for the entire project.
- **Risk Factor:** The risk factor refers to conditions that may affect the accuracy of effort estimation when using the Triangulation method. An expert in effort estimation typically determines the proportion of the risk factor, which can range from 10% to 35%. This percentage is based on several criteria or reasons, such as unclear inputs, optimistic or high estimations from senior team members, lack of experience with the technology to be implemented in the project, or the complexity of the system being worked on.
- **Recording Assumptions:** When calculating effort estimation, certain assumptions are made. For example, if the project is a web application, it may be assumed that the backend will be developed using C-Sharp, the front-end using React.js, and the database using MongoDB. The effort estimation is based on these assumptions because often the client does not specify the technology stack in the project inputs. If the technology changes, the estimated effort may also change. For instance, if the development team does not have experience with the required technology.

4.3. Results of The Triangulation Method

In this section, we will present the results of applying the Triangulation method to 14 software projects from Palestinian companies. The results are shown in Table 6.

4.3.1. Experimenting with the extension of the triangulation method

In this section, we applied the extension of the Triangulation method to 3 projects from Palestinian companies. The experiment was conducted over a specific period of the project and the results are shown in Table 7. This table provides details for each project, including the number of developers and QA engineers, as well as other input details. The table also shows the calculated project completion time and cost details for all sources, broken down by the number of hours.

Table 6. Dataset for The Experiment on The Triangulation Method and Verification Results

Brief description of the project	Basis of estimation	Technology stack	Level of certainty	Items added?	Risk factor	Actual effort	Estimated effort	Error margin
Web application for mobile phones sales / compare different options	Mockups, requirements gathering interviews	PHP, MySQL	High	Yes	10%	**180 Hours**	**167 Hours**	**7.22**
Mobile app for university students - Phase I	Design, Technical Specification	Flutter cross platform	High	Yes	15%	**140 Hours**	**120 Hours**	**14.29**
Integration solution with Facebook conversion APIs and Google ads	Business requirements, research	Netcore, MongoDB	Low	Yes	15%	**150 Hours**	**119 Hours**	**20.67**
Mini CRM solution for small businesses	Mockups, database design, requirements gathering interviews	ASP.NET MVC5, SQL Server	High	Yes	10%	**60 Days**	**61 Days**	**1.67**
Online learning platform for short videos - Phase I	Requirements gathering interviews	VueJS, Netcore	Medium	Yes	15%	**40 Days**	**45 Days**	**12.5**
Online learning platform for short videos - Phase II	Requirements gathering interviews	VueJS, Netcore	Medium	Yes	10%	**50 Days**	**48 Days**	**4**
Web application to evaluate business ideas based on Lean canvas	Requirements gathering interviews	PHP, MySQL	Medium	No	10%	**30 Days**	**32 Days**	**6.67**
Web application for internship project ideas & industry professionals	Workshop	ASP.NET MVC5, SQL S.	High	No	10%	**55 Days**	**50 Days**	**9.09**
Web application for students to submit mini research papers & review them	Requirements gathering interviews	PHP, MySQL	Medium	No	15%	**40 Days**	**35 Days**	**12.5**
Website for election campaign	Content, iterative feedback on design	WordPress	High	No	10%	**45 Hours**	**40 Hours**	**11.11**
Archiving system for insurance agency	Requirements gathering interviews	ASP.NET MVC5, SQL S.	High	No	10%	**220 Hours**	**241 Hours**	**9.55**
Automated process to generate study groups	Technical specification	C#	High	No	20%	**25 Hours**	**16.5 Hours**	**34**
Application to receive patients in dental clinics & organize appointments	Requirements gathering interviews	React with Ionic, ASP.NET MVC5	Medium	Yes	15%	**34 Days**	**36 Days**	**5.88**
Application for managing dental clinics, managing appointments, treatment plans and financials	Requirements gathering interviews	React with Ionic, ASP.NET MVC5	Medium	No	15%	**32 Days**	**33 Days**	**3.13**

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Table 7. Dataset and results of the application of our extension

	Project (1): Mobile App for E-School	Project (2): Web App for Online store	Project (3): Web App for car insurance co.
Estimated Time (Hours)	65	130	195
No. of Junior Developers	2	1	1
No. of Mid-Level Developers	0	0	1
No. of Senior Developers	0	1	1
No. of Junior QA	1	0	1
No. of Mid-Level QA	0	1	0
No. of Senior QA	0	0	0
Actual working hours per day	6.5	6.5	6.5
Management overhead	0.2	0.3	0.3
Acceptance test factor	0.1	0.1	0.2
Risk factor	0.3	0.2	0.15
Reduction Factor	0.35	0.35	0.35
Total number of resources	**3**	**3**	**4**
Acceptance test period	**6.5**	**13**	**39**
Estimated time including acceptance test	**71.5**	**143**	**234**
Risk hours	**21.45**	**28.6**	**35.1**
Estimated time including risk factor	**92.95**	**171.6**	**269.1**
Junior Developer man hours	**120.83**	**111.54**	**174.91**
Mid-Level Developer man hours	**0**	**0**	**174.91**
Senior Developer man hours	**0**	**111.54**	**174.91**
Junior QA man hours	**60.41**	**0**	**174.91**
Mid-Level QA man hours	**0**	**111.54**	**0**
Senior QA man hours	**0**	**0**	**0**
PM man hours	**12.083**	**33.46**	**52.47**
Calendar days	**13.94**	**25.74**	**40.36**
Calendar months	**0.46**	**0.85**	**1.34**

4.3.1. Results Verification

The experiments aimed to apply the Triangulation method to projects in the Palestinian market and assess its suitability. An extension of this technique was then developed to improve and assist Palestinian companies in estimating effort and accurately determining project resources and costs. To evaluate the accuracy and validity of the Triangulation method and its extension, we followed the following method. For the Triangulation method, we gathered previous software projects that had been implemented and applied the technique to them. We then compared the actual results with the estimated efforts obtained using the Triangulation method to determine its accuracy. To verify the accuracy of the extension of the Triangulation method, we applied it to three projects that had software and QA engineers. We then simulated the extension of the Triangulation method and compared its results with the actual project results, focusing on the number of resources and working hours for each engineer. This allowed to assess the accuracy of the extension of the Triangulation method when developers were added to a project and its ability to accurately reduce project delivery time.

1) Verification of the triangulation method: In this section, we present the results of verifying the accuracy of the Triangulation method for the 14 software projects used in the experiment phase, as shown in Table 6. These results are given in the last three (shaded) columns of the table: Actual Effort, Estimated Effort by the Triangulation method, and the error margin (percentage) between both estimates. Factors that can affect the accuracy of the estimated time compared to the actual time include the risk factor and level of certainty. These two factors can significantly impact the accuracy of the estimated time, as seen in Project No. 3, where there is a high-risk factor and a decrease in level of certainty due to unclear project requirements from the client. Based on these input factors, we believe that this technique has a high accuracy rate for Palestinian projects.

2) Verification of the extension of the triangulation method: In this section, we discuss the results of evaluating the accuracy of our extension to the Triangulation method. We have applied this extension to three software projects over a specific period, as shown in Figure 4. The same number of developers and QA engineers were added, and the estimated time was calculated using the extension

and compared with the actual time. Knowing the accuracy of time also allows us to determine the accuracy of resource estimates, such as developers and QA engineers.

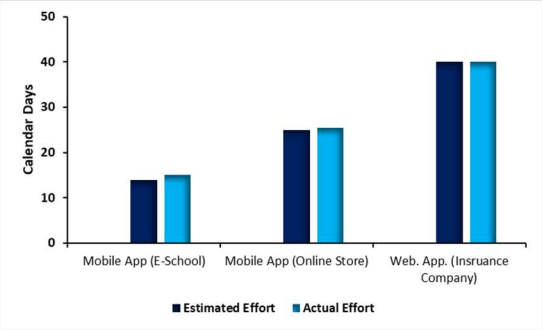

Figure 4. Verification Results of the Triangulation method extension

The results show that the accuracy of our extension is very high; the difference between the actual and estimated cost is almost negligible. The factors that can affect the accuracy of this result include the risk factor, as seen in the first project (Mobile app for E-School). The results for the second and third projects were closer to the actual results because the risk factor was lower.

5. CONCLUSION

We believe that this study provides an important reference model for Palestinian companies to help them calculate the effort required for their software projects in terms of cost, delivery time, and number of resources. This model can be used as a guideline to obtain more accurate effort estimates. We chose the Triangulation method and conducted intensive experimentation and validation using actual data. The results were very promising. We selected this technique for several reasons, including its suitability for small-team/small-business projects in Palestinian IT companies. In addition, we compared this data with other formal techniques, such as the COCOMO technique, and found that they did not meet the needs of the Palestinian IT sector. We designed the extension of the Triangulation method to allow for control and adjustment of project delivery time by varying the allocated resources, such as the number of engineers. Several factors were taken into consideration, including acceptance test factor, risk factor, reduction factor, and their relevant combinations. However, we have faced some challenges while conducting this study that might have affected the generalization of these results and to consider all relevant factor of EE, due to the fact that we could not find enough companies to cooperate and share their data with us.

Future studies could apply the extension of the Triangulation method to other software projects with different specifications and evaluate its accuracy. Other parameters that might affect effort estimation accuracy could also be considered, such as programming language, work environment, type of administrative support, skill levels of engineers, etc. These could serve as the basis for new studies aimed at expanding the applicability of the technique and improving its accuracy and flexibility.

REFERENCES

[1] Lenarduzzi V., "Could social factors influence the effort software estimation?", *Proceedings of the 7th International Workshop on Social Software Engineering;* 2015. p. 21-24.

[2] Dagnino A., "Estimating software-intensive projects in the absence of historical data", *35th International Conference on Software Engineering (ICSE),* 2013. p. 941-950. IEEE.

[3] Tanveer B., "Guidelines for utilizing change impact analysis when estimating effort in agile software development", *Proceedings of the 21st International Conference on Evaluation and Assessment in Software Engineering*; 2017. p. 252-257.

[4] Zein S, Salleh N, Grundy J., "Mobile application testing in industrial contexts: An exploratory multiple case-study", *International conference on intelligent software methodologies, tools, and techniques, Springer, Cham*, 2015. p. 30-41.

[5] Zarour A, Zein S., "Software development estimation techniques in industrial contexts: An exploratory multiple case-study", *International Journal of Technology in Education and Science*. 2019;3(2):72-84.

[6] Survey About Effort Estimation Techniques in the Palestinian IT sector [Internet]. [cited 2023 May 2]. Available from: https://forms.gle/4ZadKz2e89vBLNYg7.

[7] IIBA A., "Guide to the Business Analysis Body of Knowledge," *(BABOK Guide),* Ver 3.0. 2015.

[8] Trendowicz A., "Software Cost Estimation, Benchmarking, and Risk Assessment", *The Software Decision-Makers' Guide to*

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Predictable Software Development. Springer, Heidelberg, 2013.

[9] Chirra SMR, Reza H., "A survey on software cost estimation techniques", *Journal of Software Engineering and Applications,* 2019;12(06):226.

[10] Popli R, Chauhan N. Cost and effort estimation in agile software development, *International Conference on Reliability Optimization and Information Technology* (ICROIT); 2014. p. 57-61. IEEE.

[11] Prakash B, Viswanathan V. A survey on software estimation techniques in traditional and agile development models, *Indonesian Journal of Electrical Engineering and Computer Science*. 2017;7(3):867-876.

[12] Zarour A, Zein S. Software development estimation techniques in industrial contexts: An exploratory multiple case-study, *International Journal of Technology in Education and Science*. 2019;3(2):72-84.

[13] Gumaei A, Almaslukh B, Tagoug N., "An Empirical Study of Software Cost Estimation in Saudi Arabia Software Industry", *International Journal of Soft Computing and Engineering* (IJSCE). 2015;ISSN 2231-2307.

[14] Kafle LP, "An empirical study on software test effort estimation", *Int. J. Soft Comput. Artif. Intell*., 2014;2(2).

[15] Tailor O, Saini J, Rijwani MP, "Comparative analysis of software cost and effort estimation methods: a review" *Interfaces,* 2014;5(7):10.

[16] Mahmood Y, Kama N, Azmi A, Ali M., "Improving estimation accuracy prediction of software development effort: A proposed ensemble model", *International Conference on Electrical, Communication, and Computer Engineering (ICECCE),* 2020. p. 1-6. IEEE.

[17] Langley MA., "Success Rates Rise: Transforming the high cost of low performance" *PMI, editor. Pulse of the Profession*, 2017. p. 32.

[18] Popli R, Chauhan N. "Research challenges of agile estimation", *Journal of Intelligent Computing and Applications,* 2012.

[19] Munialo SW, Muketha GM, "A review of agile software effort estimation methods", 2016.

[20] Trendowicz A, Jeffery R, "Software project effort estimation. Foundations and Best Practice Guidelines for Success", *Constructive Cost Model–COCOMO* , 2014;12:277-293.

[21] A N, Qurashi JA, Kumar S., "A Survey Study of Various Software Cost Effort Estimation in Perspective of India", *International Journal of Computer Sciences and Engineering*. 2019.

[22] Vidjikant S, Nykyforuk L, Romaniv T., "How to Write a Request for Proposal(RFP) for Software Development in 2023", *Softjourn.,* 2023.

[23] Best Practices for capturing Business requirements for your project [Internet]. 2023 [cited 2023]. Available from: https://www.softwaredevelopment.co.uk/blog/best-practices-for-capturing-business-requirements-for-your-software-project/.

[24] How to write technical specifications [Internet]. 2023 [cited 2023]. Available from: https://monday.com/blog/rnd/technical-specification/.