Jestr

Journal of Engineering Science and Technology Review 13 (6) (2020) 187 - 196

JOURNAL OF
Engineering Science and Technology Review

Research Article

www.jestr.org

An Effort Estimation Method for Service-Oriented Architecture

Samson Wanjala Munialo[1,*], Geoffrey Muchiri Muketha[2] and Kelvin Kabeti Omieno[3]

[1]*Department of Information Technology, Meru University of Science and Technology, Kenya*
[2]*Department of Computer Science, Murang'a University of Technology, Kenya*
[3]*Department of Information Technology and Informatics, Kaimosi Friends University College, Kenya*

Received 24 April 2020; Accepted 16 October 2020

Abstract

Determining size and effort of SOA systems is critical for managing SOA projects. As a consequence, a number of methods have been proposed to estimate effort of building SOA projects but the problem of estimating SOA development effort still remains largely unresolved mainly because there is limited attempt in using size metrics to estimate SOA development effort. To address this problem, this study proposed an effort estimation method for SOA centred on size metrics and effort factors. The proposed method enables estimation of effort factors using fuzzy logic technique to improve on estimation accuracy. The method was automated into a tool to facilitate entry of parameters and display of results. The study employed experiment research design based on 15 SOA projects developed by computer science undergraduate students to validate the proposed estimation method. To complement the experiment, we used a survey study involving 20 programmers from the industry to confirm the relevance of effort estimation factors proposed in this study. Result from the experiment revealed that the proposed method is more accurate and returned a lower Mean magnitude of relative error (MMRE). Response from the survey showed that the proposed effort factors are valid and they have influence on SOA development effort.

Keywords: Service-oriented architecture, software metrics, software effort estimation, effort estimation model, effort multiplier factors

1. Introduction

Software effort estimation is the process of predicting human effort required to build a software project. The bulk of the cost of software development is due to human effort estimated in person-months [1]. To stay competitive, software developers need to deliver software products on time, within the budget and to the agreed level of quality [2]. Most projects fail due to planning issues such as cost, effort, time and requirements specifications [3]. Consequently, there is need for reliable effort estimation method to enable adherence to schedule and budget for successful resource allocation and software project implementation.

SOA is a software system comprising of various communicating services working in synergy to achieve a defined objective. A service thus can be viewed as a reusable component that represents a business process. It is a course-grained, discoverable and self-contained software entity that interacts with applications and other services through a loosely coupled, asynchronous, message-based communication model [4]. SOA defines an interaction model between functional units, in which the consumer of the service interacts with the service provider to find out a service that matches its needs through a registry.

Earlier size metrics and effort estimation methods including Function point analysis [5] and COCOMO [6] have had the interest of estimating accurately the effort for developing software. However, these methods were challenged when estimating SOA effort due to SOA features such as integration among services within and outside the organization, service internal structure and types of services [7]. Various software effort estimation methods have incorporated Artificial Neural Network (ANN) while others have used fuzzy logic to improve on estimation quality. However, due to limited SOA projects data sets, no estimation method to date has used ANN to estimate SOA development effort. On the other hand, according to our knowledge, there is no existing fuzzy logic effort estimation method for SOA. A number of research studies have attempted to introduce effort estimation methods for SOA [8][9] [10] [11] [12] [4] [13]. Despite the fact that these methods are promising, the problem of estimating SOA development effort still remains largely unresolved mainly because there is limited attempt in using size metrics and relevant factors to estimate SOA development effort. In addition, so far there is no attempt to use fuzzy logic in SOA effort estimation method.

The main objective of this study was to develop a more accurate fuzzy logic effort estimation method for SOA applications based on size metrics and SOA effort factors. The proposed method was automated and hosted in www.vsoft.co.ke/tool/ for entry of parameters and display of results. Due to limited number of SOA UML SOA datasets from the industry, research investigation was based on a controlled laboratory experiment in the context of Meru University of Science and Technology, 3rd year Computer science undergraduate students. Furthermore, a survey was done in the context of industry programmers to replicate the experiment and determine the relevance of the identified effort factors in estimating SOA development effort.

2. Related Work

Most effort estimation methods including earlier methods such as Function Point Analysis (FPA) [5] make use of

*E-mail address: sammunialo@gmail.com
ISSN: 1791-2377
doi:10.25103/jestr.136.25

software size as the main indicator when estimating software development effort. Albrecht [5] introduced FPA which measures number of functionality in software by counting the number of functional components. Furthermore, a number of modified FPA metrics versions including 3D Function Points, COSMIC full Function Points and International Function Point User Group (IFPUG) were introduced. However, they did not measure SOA key features.

This prompted COSMIC [14] to introduce COSMIC-SOA [14] which counts data movement among service providers and users. However, COSMIC-SOA only focused on data movement across services to measure SOA size, disregarding other SOA attributes including service internal structures and dependency among services. Munialo et al. [15], defined SOA size metrics (SOASM) which takes into account SOA internal structure, data movement, interaction and relationship among services as key attributes for defining SOA size metrics. The metrics were validated theoretically based on Briand's property framework

To estimate software development effort, there is need to package software size and software development effort factors into a method or model to estimate effort more accurately. Recent literature on SOA effort estimation methods classified software development effort estimation methods into traditional effort estimation methods and SOA effort estimation methods. Traditional software development effort estimation methods include Basic Constructive Cost Model (COCOMO), Intermediate COCOMO, COCOMO-II [6], Artificial Neural Network (ANN) methods and Fuzzy logic methods. Boehm [6] introduced COCOMO methods which compute software effort as a function of program size expressed in thousands lines of codes (KLOC). COCOMO methods take into account effort coefficient (a), economy of scale constant (b) and Effort Adjustment Factors (EAF) to compute software development effort ss shown in Eq. 1.

$$Effort = a(KLOC)^b * EAF \quad (1)$$

EAF (Effort adjustment factors) are subjective assessment of products, hardware, personnel and project factors. COCOMO methods are the most validated method by researchers and they are the most adopted method by the industry. However, attributes related to SOA applications are not captured among COCOMO effort factors. Hence, all versions of COCOMO are inadequate in estimating SOA development effort.

Lately, various research studies have incorporated Artificial Neural Network (ANN) in their estimation methods with an aim of acquiring facts from previous software projects and use the facts to predict software development effort more accurately [16] [17] [16]. Neural network methods are preferred when there is enough previous project data to train the ANN method. However, due to limited data on previous SOA projects, SOA project effort estimation data has not matured to be subjected to ANN method.

On the other hand, a number of research studies on Software development effort estimation have integrated fuzzy logic in their estimation methods to yield more accurate results as compared to traditional algorithmic methods [18] [19] [20] [21] [22] [23] [24]. However, according to our knowledge, no research to date has proposed fuzzy logic effort estimation method for SOA applications.

Attempts to estimate SOA development effort have been discussed in various studies [8] [9] [10] [12] [13] [25]. O'Brien [8] introduced SOA effort estimation method known as SMAT-AUS framework which recognize types of service, technical factors and social factors as key inputs when determining scope, cost and effort of Service oriented Architecture (SOA) projects. However, he excluded SOA size as an effort factor and only provided a framework with no details of the framework's computation.

Akkiraju & Geel [10] proposed SOA effort estimation method based on business process model and linguistic analysis approach to reveal business objects. Likewise, Mishra & Kumar [13] used Business Process Modelling Notations (BPMN) constructs to compute development effort of business process SOA applications by counting the number of processes, events, queries, links and tasks. One advantage of estimating by considering business objects is the ability to estimate effort at an early stage of software development. However, at an early stage, key service attributes such as structural attributes and message movement cannot be captured for the purpose of measuring SOA size more effectively.

Li & O'Brien [9] proposed an effort classification matrix for web service composition by considering context and technology aspects of service composition. The method used qualitative effort estimation hypotheses to identify effort factors that influence web service composition. However, they focused on qualitative analysis with no emphasis on empirical analysis and validation on the proposed method.

Li & Keung [25] defined a framework for costing SOA using work breakdown structure approach by decomposing SOA into sub-problems (services). They classified services into available service, migrated service, new service and combined service. Similarly, Farrag et al [11] also classified services into Available service, migrated service, new service and Composite service. However, the aspect of SOA size factor was not captured.

Gupta [12] proposed a model that takes service operation as the unit of measurement whose complexity forms the basis of computing service size. The model [12] provided a clear and detailed analysis of SOA attributes focusing on service internal structure complexity, technical complexity and environment complexity. On the other hand, Verlaine, Jureta & Faulkner [4] also introduced an effort estimation method based on service structural complexity. The two methods considered structural complexity metrics when computing SOA size and effort but they didn't include service dependency and movement of data as key size attributes.

Based on SOA effort estimation literature, researchers have attempted to identify factors that have influence on SOA development. All these approaches are promising but they did not capture all relevant factors for SOA development effort exhaustively. To bridge this gap, our research study focused on consolidating SOA size, SOA Type Factors (STF) and SOA Effort Multiplier Factors (EMF) to compute SOA development effort more accurately.

3. Proposed SOA effort estimation method

The proposed SOA effort estimation method used SOA size, Service type factors, Product factors, Development environment factors, Requirements specification factors and Personnel factors to estimate SOA development effort. The proposed method was automated and hosted at www.vsoft.co.ke/tool/. The tool provides facilities for entry of parameters for size computation and effort estimation.

3.1 SOA size

Our proposed effort estimation method considered SOA size as the main attribute that determines SOA development effort. We adopted SOA size metrics (SOASM) [15] to compute SOA size with minimal revision guided by quantitative validation. SOASM defined Weighted Operation Count (WOC) in Eq. 2, Service Dependency Count (SDC) in Eq. 3, Weighted Message Count (WMC) in Eq. 4 and Weighted Service Count (WSC) metrics in Eq. 5 as indicators of SOA size as shown in Tab.1.

Table 1. SOASM Size Metrics [15]

Eq.	Metric	Description
(2)	$WOC(S) = \sum_{i=1}^{n} (O_i + P_i)$ Weighted Operation Count (WOC)	WOC counts the number and complexity of operations (O_i) and parameters (P_i) contained in a service as captured in UML interface diagram.
(3)	$SDC(X) = i + \sum_{i=1}^{n} a + \sum_{i=1}^{n} g + \sum_{i=1}^{n} t$ Service Dependency Count (SDC)	SDC counts the number and complexity of dependency by considering fan-in dependency (i), fan-out dependencies classified as atomic (a), lighter aggregation (g) and strong aggregation (t) as captured in UML interface diagram.
(4)	$WMC(X) = \sum_{i=1}^{n} s + \sum_{i=1}^{n} a + \sum_{i=1}^{n} r$ Weighted Message Count (WMC)	WMC counts the number of messages between services based on message type which include synchronous (s), asynchronous (a) and reply (r) messages as revealed in UML sequence diagram
(5)	$WSC = WOC + SDC + WSC$ Weighted Service Count (WSC)	WSC counts the number of services based on the sum of WOC, SDC and WMC

After we subjected the metrics to quantitative validation, we revised WOC metric simple, average and complex operation weights to 2, 3 and 4 respectively while parameter weight remained 1. Secondly, we revised the equation for computing SDC by eliminating fan-in dependency (i) from the equation but maintained the weights as defined in SOASM [15]. Lastly, we revised the weights for WMC to 3, 2 and 1 for synchronous message, asynchronous message and reply message respectively.

Likewise, we included Intermediate COCOMO constant A and constant B for different project types to compute effort as a function of size as indicated in Eq. 6. According to Boehm [6], constant A represents effort coefficient scale (exponential) factor and B accounts for relative economies of scale for software of different size and type [6] as shown in Tab. 2.

$$\text{Effort (PM)} = A * (\text{SOA Size})^{B} \quad (6)$$

Table 2. Intermediate COCOMO effort coefficients

Project Type	Coefficient constant (A)	Exponential Scale factor (B)
Organic (Small)	3.2	1.05
Semi-detached (Medium)	3.0	1.12
Embedded (Large)	2.8	1.20

Various studies have related average LOC to Function Point for various programming languages based on historical data [23]. Programming languages including PHP, Java, Perl, JavaScript and C++ Function Point have an average of 53 Lines of Codes (LOC) per Function Point [26]. Based on the relationship between Function Point and Web service point with regard to the use of functional aspect to measure size, our method also used 53 LOC or 0.053 KLOC to be equivalent to 1 Web service point. Therefore, for organic projects, Effort is computed as shown in Eq. (7).

$$\text{Effort (PM)} = 3.2 * (\text{SOA size} * 0.053)^{1.05} \quad (7)$$

Our proposed method estimates the final SOA development effort by including Service Type Factors (STF) and 13 Effort Multiplier Factors (EMF) in the computation.

3.2 Service Type Factors (STF)

This study introduced Service Type Factor (STF) and classified STF into Service Construction Type (SC) and Service Architectural Style (SA). STF is determined at service level rather than at SOA application level because services in SOA application may have different service types.

3.2.1 Service construction type (SC)

Service construction (SC) types are classified into available (Discovered) service, migrated service and new service centred on how the service was realized [25] [11]. Based on previous research [11] on effort distribution, more effort is spent at construction phase compared to other phases when developing a new service as shown in the Tab. 3. According to Tab. 3, effort to develop the three types of services is constant at requirements and analysis, design, testing and integration phases but varies at construction phase. The variation of effort at construction phase resulted to 100 %, 80% and 60% of total effort to develop a new service, migrated service and available service respectively. Consequently, this study allocated weights of 1.00, 0.8 and 0.6 to new, migrated and available services respectively.

3.2.2 SOA architectural style (SA)

SOA architectural style defines the communication protocol and style for developing web services [9]. The two most common communication architectural style or protocols used in SOA applications are REST (Representational State Transfer) and SOAP (Simple Access Protocol). Basically SOAP and REST are not directly comparable given that

SOAP is a protocol that make use of WS* technologies while REST is an architectural style designed to communicate via HTTP protocol [9]. However, this study compared SOAP and REST with regard to service development effort.

Table 3. Service type Effort distribution (%) among development phases [11]

Phase	New service (%)	Migrated service (%)	Available service (%)
Requirements and Analysis	16	16	16
Design	15	15	15
Construction	40	20	0
Testing	22	22	22
Integration	7	7	7
Total Effort	100	80	60

Li & O'Brien [9] compared development effort based on REST and SOAP qualitatively and concluded that services built using SOAP technology have more information and are difficult to build as compared to REST services. They assigned a factor of 2 to SOAP and a factor of 1 to REST services based on development effort. In this regard, our method allocated 1 to REST service and 1.2 to SOAP.

Service Type Factor (STF) is computed by multiplying the product of SC and product of SA of all services in SOA application as indicated in Eq. 8.

Service Type Factor (STF) = Service construction (SC) * Service architecture (SA)

$$\text{STF} = \prod_{i=1}^{n} SC * \prod_{i=1}^{n} SA \quad (8)$$

Therefore, Effort for developing SOA with inclusion of STF in Eq. 9.,

$$\text{Effort (X)} = \text{STF} * A * (\text{service size})^{B} \quad (9)$$

SFT has a tremendous impact on software development effort which may be a decreasing or increasing effect on SOA development effort.

3.3 SOA Effort Multiplier Factors (EMF)

SOA development effort is also determined by effort factors also known as cost drivers which are proportional to the amount of effort employed and whose values either increase or decrease effort. This study proposed 13 SOA effort factors also known as Effort Modifier Factors (EMF) grouped into 4 categories namely SOA product factors, development environment factors, Requirement specification factors and Personnel factors as shown in Tab. 4.

Table 4. SOA effort modifier factors (EMF)

S/N	SOA Effort Factor categories	SOA Effort factors
1	Product factors	Database complexity (DC) Data Size (DS) User interface complexity (UIC) Integration complexity (IC)
2	Development environment factors	Infrastructure capabilities (ICA) Development tool support (DT)
3	Requirements specification factors	Requirement elicitation (RE) Business value /risk (BVR) Security requirements (SR)
4	Personnel factors	SOA experience (SE)

Each EMF factor is classified into its respective categories and weighted accordingly based on its influence on development effort. When EMF is classified as normal it is assigned a weight of 1 which has no effect on software development effort. On the other hand, EMF that is assigned a value that is less than 1 has a decreasing effect on software development effort while EMF with a value greater than 1 has an increasing impact on development effort. EMF is applied at the SOA application level when computing development effort. After allocation of weights to each category, our research study used fuzzy logic to compute EMF values for the purpose of accurate estimation, smooth transition from one category set to another and to provide a realistic way of representing effort attributes.

3.3.1 Product factors

Product factors include elements that add functional value to SOA application with regard to the product structure [12]. Product factors proposed in our method are database complexity, database size, interface complexity and integration complexity.

Database complexity (DC) [12] includes database constraints that affects the complexity of a service. Another product factor is database size (DS) factor which simply counts the number of tables contained in a database. Thirdly, user interface complexity factor (UIC) carries both functional and non-functional features of the service application [4]. On the other hand, integration complexity (IC) factor is inherently the amount of effort used to configure a service to integrate with other services and resources such as databases. This study proposed Product complexity factors with their description as shown in Tab. 5.

Table 5. Product factors description

Factor	Normal (1.00)	High (1.10)	Very High (1.20)
Database Complexity factor (DC)	Simple objects including tables and views	Security features such as user roles and rights.	Use of procedures and triggers.
Database size weight factor (DS)	Less than 50 tables and views.	Between 50 and 100 tables and views.	Above 100 tables and views.
User interface complexity (UIC)	Simple form elements	Form with client-side validation e.g. JavaScript	Interface with touch input, media and security features such biometrics
Integration Complexity (IC)	Service to service and service to database integration	service to services outside the organization integration	Service to legacy application integration

We adopted COCOMO data factors classification of product factors as normal, high and very high. This study rounded off the COCOMO values to 1.00, 1.10 and 1.20 for normal, high and very high respectively. The values were then represented as 3 fuzzy sets namely normal, high and very high linguistic variables.

3.3.2 Service development environment

Service development environment include hardware and software required to support development and implementation of SOA application. Service development environment determines the amount of effort with regard to efficiency, constraint and capability of available hardware and software tools. This study focused on tool support and Hardware/software capabilities. Less effort is spend when developing a web service supported by tools and framework rather than writing codes from scratch. The productivity of a software development team is directly proportional to the development tools employed by programmers in developing the web service [12].

This study classified software development tools (DT) into three categories namely lowly automated, normal and highly automated. The categorization is based on existing development tools for programming languages used to develop web service applications. Tools that support only coding and compilation are classified as lowly automated allocated a weight of 1.10, tools with code line assistant and are user friendly are classified as normal allocated weight of 1 while fully automated tools are classified as highly automated allocated of 0.9.

On the other hand, Infrastructure capabilities (ICA) include hardware, networking and software infrastructure capacity. When facilities have low capacity and capabilities to host and enable service development then more effort is required as compared to when facilities are capable. Hardware in this case includes storage infrastructure, processor and hardware configuration issues. Networking infrastructure includes data communication infrastructure, server and network configuration issues. On the other hand, software capabilities include software integration issues, operating systems compatibility and configuration issues. Infrastructure capabilities are classified as very low with a weight of 1.2, Low with a weight of 1.1 and Normal with a weight of 1.

3.3.3 Requirement factors

Requirements are demands or needs defined by stakeholders outlining what must be accomplished by software developers. Without requirements, you cannot measure success or failure of system development and implementation [11]. Critical issues captured as requirement factors in this study include Requirement elicitation factors, business risk and value, and security requirements.

Requirement elicitations (RE) provide a framework for ensuring software product compliance with users' needs and demands. When requirements are clear and unambiguous less effort is used to develop an application as compared to when requirements are unclear and ambiguous. The proposed requirements elicitation factors weights are 1.30, 1.15, 1 and 0.85 for very ambiguous, ambiguous, clear and very clear requirements respectively.

Secondly, business value is the perception on the need for software product for organization's improvement, survival and image. On the other hand, business risk is also related to business value in the sense that a system whose failure will have great impact to an organization is valued more. Business risk in relation to software development is a possible negative event that may occur in a business as a result of software implementation failure.

More effort is required to build a system that is highly valued and is of high risk to the organization as compared to a system that is lowly valued and low risk. Thirdly, security requirement (SR) is a condition or capability needed by stakeholders to ensure confidentiality, integrity, availability, authenticity and authorization of an application system [27]. The degree of security integration in the system determines the amount of effort required to develop the application. Description of business value/risk factor and security factor, weights of 0.70, 0.85, 1, 1.15 and 1.30 for very low, low, normal, high and very high respectively are as shown in Tab. 6

Table 6. Business value and security requirements description effort factor

Factor	Very low	Low	Normal	High	Very high
Business value/Risk (BVR)	application an organization can do without	application to perform non-core functions	Application to perform a core function	Critical systems	Very Critical systems
Security requirements (SR)	No security feature required in a service	Low security feature	confidentiality and authenticity features	biometric features	Use of encryption algorithm

3.3.4 Personnel factors

People or personnel factors are personnel attributes that contributes to SOA development effort. Personnel factors proposed in this study include web service development experience, Programming language experience, application experience and team cohesion.

Developers' experience in web service application factor is determined by how long developers have worked with web service applications. On the other hand, application experience factor defines a programmers' knowledge on the type of an application e.g. banking application system. Thirdly, programming experience factor is a measure of how long developers have worked with a programming language. Team cohesion factor takes into consideration the team members shared vision, teamwork and consistency of members' objectives. The more experienced developers' are with SOA, the application, programming language and are more cohesive the team is the less effort the developers' will use to develop a web service system. Personnel factors weights are as shown in Tab. 7.

Table 7. Web service developer's experience effort multiplier

Personnel factors	Very low	Low	Normal	High	Very High
SOA Experience (SE)	0 - 6 months	6 - 9 months	1- 2 years	2- 4 years	4 years and above
Application Experience (AE)	0 - 6 months	6 - 9 months	1- 2 years	2 - 4 years	4 years and above
Programming Language Experience (PE)	0 - 6 months	6 - 9 months	1- 2 years	2 - 4 years	4 years and above
Team Cohesion (TC)	Highly intolerable team	Intolerable team	accommodate opinions	Intolerable& Consistency of objectives	Shared long term vision and objectives

SOA experience, application experience, programming experience and team cohesion are allocated 1.30, 1.15, 1, 0.85 and 0.70 weights for very low, low, normal, high and very high respectively. To reduce subjectivity and to be more realistic with EMF classification and weights, we introduced fuzzy logic technique to manipulation EMF values to more accurate results.

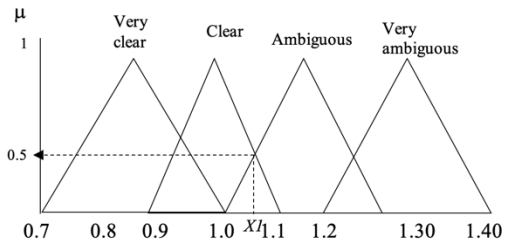
3.4 Fuzzy logic application to EMF

In this study, EMF factors are exposed to subjective judgement and thus we proposed to apply fuzzy logic to EMF factors. Fuzzy logic provides a better way of representing data in fuzzy sets to express data that is unclear and vague in nature. A case in point is requirements elicitation factor (RE) which may be subjective from one developer from the other. Secondly, representing data in a class or category provides a wider representation. For example requirements elicitation factor (RE) ambiguous value represented as 1.15 is represented as a range from 1.00 to 1.1 in fuzzy sets. Fuzzy logic processes include Initialization, Fuzzification, Inference system rules and Defuzzification [18] [19] [20] [21] [22] [23] [24].

i) Initialization: Initialization is the process of defining linguistic variables which are words from a natural language replacing values. The linguistic variable in this case is requirements elicitation with 4 variables namely very ambiguous, ambiguous, clear and very clear. Represented in fuzzy set as follows:

Requirements elicitation = {very ambiguous, ambiguous, clear, very clear}

ii) Fuzzification – Is a technique of using membership function to convert crisp data to fuzzy values. It determines the degree to which inputs belong to a particular fuzzy set. In this study we used Triangular membership function to determine the degree of membership of input x which belongs to a fuzzy set. For requirements elicitation factor, fuzzification for a value x1, will give 0.5 clear and 0.5 ambiguous as shown in Fig. 1.



$\mu_{(x1 = \text{Very clear})} = 0$, $\mu_{(x1 = \text{Clear})} = 0.5$ $\mu_{(x1 = \text{Ambiguous})} = 0.5$, $\mu_{(x1 = \text{Very ambiguous})} = 0$

Given examples of crisp input x1 at 1.05,

Fig. 1. Requirements elicitation factor

Therefore, crisp data x1 = 1.05 falls under clear requirements fuzzy set by 0.5 degree of membership and 0.5 degree of membership in ambiguous requirements fuzzy set.

iii) Fuzzy Inference System – It is a fuzzy logic component that evaluates rules in the rule base to determine the outcome of set conditions. Fuzzy inference system employed in this study is Mamdani System. Rules for requirements elicitation factor are as follows:

$$IF\ (requirements = Very\ clear)\ THEN\ effort = Low$$
$$IF\ (requirements = clear)\ THEN\ effort = Normal$$
$$IF\ (requirements = ambiguous)\ THEN\ effort = high$$
$$IF\ (requirements = very\ ambiguous)\ THEN\ effort = very\ high$$

The result from the above rules are effort modifiers including Low = 0.9, Normal = 1, High =1.1 and very High = 1.2 of SOA development effort.

iv) Defuzzification - Defuzzification is the process of converting output data to crisp output value using a defuzzification method such as Centre of gravity method as in Eq. 10.

$$Center\ of\ gravity\ (requirements\ elicitation) = \frac{\sum_{i=1}^{n} \mu . w}{\sum_{i=1}^{n} \mu}$$

$$COG = \frac{(0.5*1.0)+(0.5*1.1)}{0.8+0.2} = 1.05 \text{ of SOA development effort} \quad (10)$$

The result of COG is a crisp output 1.05 which has a normal influence on SOA development effort. The value is multiplied with other modifiers then the product is multiplied with the effort size function and service type to give the final effort.

3.5 Final effort estimation

The Final effort for developing the entire system is calculated by taking into consideration the SOA application size, product of service type factors and product of EMF as indicated in Eq. 11.

Final Effort (X) = STF * A * (SOA Application size) ^{B} * $\prod_{13}^{n} EMF$ (11)

Where,

A and B are constants derived from COCOMO while SOA Application size in web service point is computed by the identified SOA size metrics known as SOASM as shown in Eq.7.

Effort (PM) = A * (SOA size * 0.053) ^{B}

Service Type Factor (STF) is as shown in Eq. 8,

$$STF = \prod_{i=1}^{n} SC * \prod_{i=1}^{n} SA \quad (11)$$

EMF is the product of all Effort Multiplier Factors as shown in Eq. 12.

$\prod_{13}^{n} EMF$ = DC *DS* UIC * IC * DT * ICA * RE * BR * SR * SE * AE * PE * TC (12)

Based on our proposed effort estimation method, EMF can be computed directly by assigning allocated weights or through application of fuzzy logic.

4. Method

Experiment process involved 15 groups of Computer Science 3rd year students who developed SOA application known as web service applications in groups of 5 students per project. First of all, each group defined their concepts which were corrected and validated. After which they were required to develop software requirement specification (SRS) documents and Software design documents (SDD) for their respective SOA based projects. Each group submitted their SRS documents which were verified and errors were corrected before designing the projects.

The groups designed their SOA applications which were documented in SDDs. The SDDs comprised of UML interface diagrams and UML sequence diagrams which revealed SOA size attributes to expose to SOA size metrics (SOASM). Later on, they constructed SOA based projects based on SDD after which integration and testing were done. Individual developers recorded the actual time taken in each phase which was summed to give the actual time spend by a group. The developed SOA based projects and documentations were presented and submitted by the groups for verification and validation. Each project size computed, development effort estimated and actual time used to develop each project was recorded for further analysis. The entire experiment process was done in approximately 3 months including development of the projects and validation of the proposed method.

Preparation and planning for the expert survey was done appropriately before the survey was conducted to ensure validity and reliability of the instrument. Fifty questionnaire were sent to experts requiring them to indicate if they had worked with SOA, 46 were returned with 27 responded positively having engaged in SOA applications while 19 said they had never participated in developing SOA applications. Random sampling was used to select 20 programmers out of 27 who had worked with SOA applications before. The sampled programmers were taken through the proposed effort estimation method annex. Upon satisfactory understanding of the proposed effort estimation method, the sampled programmers were issued with questionnaires accompanied with annex documentation describing in detail the proposed estimation method.

5. Results and discussions

This study further validated the proposed effort estimation method through a laboratory experiment in the context of 3rd year computer science students and expert opinions were gathered through a survey.

5.1 SOA size metrics results

Data was collected from the 15 SOA based projects as shown in data reference [a] and they were subjected to SOA size metrics (SOASM) [15] as indicated in Tab. 8. Details captured from each project included project name, WOC, SDC and WMC to compute SOA project size (WSC) in web service points. Being small projects developed by students, the biggest project had 44 web service points and the smallest project had 30 web service points.

Table 8. Data analysis for the 15 SOA projects subjected to SOA size metrics

ID	Project Name	WOC	SDC	WMC	SOA size (WSC)
1	Online carpool system	31	7	6	44
2	Online doctors' appointment system	24	3	3	30
3	SACCO management system	32	4	6	42
4	Online event & catering system	25	7	6	38
5	Bus service online reservation system	27	5	5	37
6	Online furniture purchase system	27	4	7	38
7	Construction material online purchase systems	29	4	7	40
8	Prime freelance systems	30	7	7	44
9	Real estate online property management system	28	7	6	41
10	Tourism and accommodation online system	23	6	3	32
11	Apartment rental online system	27	6	6	39
12	Online horticulture sales information system	30	8	3	41
13	CDF disbursement management system	25	5	3	33
14	Online pharmaceutical management system	27	5	6	38
15	Online event ticketing system	24	5	3	32

5.2 Proposed effort estimation method descriptive analysis

The product of EMF per project was multiplied to SOA size and product of STF to compute effort estimation for each project which was compared with actual effort to Magnitude of Relative Error (MRE) as shown in Tab. 9. More details on the data used in this study are captured in data repository http://dx.doi.org/10.17632/sp or http://data.mendeley.com/datasets/spdxkdry2s/1.

According to Tab. 9, the measured size and SOA development effort factors were used to estimate effort for each project based on organic projects as defined in COCOMO given that each of the SOA based projects developed by students were small, predictable and in a stable environment.

Table 9. Effort estimation analysis based the proposed method

ID	Project Name	SOA Size	STF	EMF	Estimated Effort (P/M)	Actual Effort (P/M)	MRE
1	Online Carpool System	44	1.2	1.294	12.813	9.54	-0.343
2	Online doctors' appointment system	30	1	1.294	7.142	6.32	-0.130
3	SACCO management system	42	1.2	1.125	10.611	8.84	-0.200
4	Online Event & Catering system	38	1	1.176	8.322	8.12	-0.025
5	Bus service online reservation system	37	1.44	1.294	12.818	8.31	-0.542
6	Online furniture purchase system	38	0.8	1.125	6.368	7.14	0.108
7	Construction Material online purchase systems	40	1	1.294	9.661	7.06	-0.368
8	Prime freelance systems	44	0.72	1.238	7.354	8.43	0.128
9	Real Estate online property management	41	0.72	1.294	7.138	7.86	0.092
10	Tourism and accommodation online system	32	1	1.294	7.643	6.21	-0.231
11	Apartment rental online	39	0.8	1.294	7.526	6.53	-0.152
12	Online Horticulture Sales Information system	41	1	1.294	9.914	8.84	-0.122
13	CDF disbursement management system	33	1	1.294	7.894	6.23	-0.267
14	Online Pharmaceutical management system	38	0.864	1.238	7.565	6.62	-0.143
15	Online Event ticketing	32	1	1.238	7.310	5.73	-0.276
	Mean Magnitude of relative error MMRE						-0.165

Effort (SOA application) = STF * A * (SOA size) ^{B} * $\prod_{13}^{n} EMF$

Where a = 3.2, b = 1.05 for organic project (Small-scale and predictable projects) SOA size was multiplied by a factor of 0.053 to convert web service point to KLOC based on programming languages used in the experiment.

Therefore, Effort = STF * 3.2 * (SOA size * 0.053)$^{1.05}$ * $\prod_{13}^{n} EMF$

EMF had tremendous effect on effort due to SOA experience among all groups which was at 1.30 points, application experience factor at 1.15 points and Programming experience factor at 1.15 points. On the other hand, business value/risk and security requirements factors had 0.7 and 0.85 respectively for all groups given that the projects were done for academic and research purpose. Database complexity and database size were awarded 1 for each project due to similarity in students' projects based on these factors. Factors that experience variance among different projects in the experiment were user interface complexity, development tool support, infrastructure capacity and requirements elicitation.

The most common measures for effort estimation methods accuracy according to literature are Magnitude of Relative Error (MRE) and Mean Magnitude of relative error (MMRE) [28].

$MRE = \frac{y-\check{y}}{y}$ where y is actual effort and $\check{y}$ is the estimated effort as in Eq. 13.

$$MMRE = \frac{1}{n} X \sum_{i=1}^{n} MRE\, i \quad (13)$$

MMRE is the average MRE for all projects in the experiment where n is the number of projects and MRE_i is for each project.

The accuracy of the proposed effort estimation method was -0.165 MMRE which is within the acceptable margin of -0.25 and +0.25. Therefore, the accuracy of the proposed effort estimation method as revealed in the experiment shows that the method is more accurate when dealing with SOA based applications.

5.3 Expert opinion survey

A survey was conducted to gather expert opinions on the validity of the proposed SOA effort estimation method. Expert opinion survey was also meant to complement laboratory experiment done by students.

5.3.1 Demographic summary of the respondents

All the 20 questionnaires were returned successfully with no outlier data. According to response on academic qualification, 2 of the respondents had MSc. Degree in computing related field and the remaining 18 respondents had BSc. Degree in computing related field as shown in Tab. 10. From the analysis, it was confirmed that experts had enough experience to assist in validating SOA size metrics and effort estimation method.

Table 10 .Experts' experience in Software development

Experience	Below 1 year	Between 1 and 3 years	Above 3 years
Software development	2	8	10
SOA application development	5	6	9

5.3.2 Survey results

5.3.2.1 Experts' response on SOA size effect on effort

Respondents believed that SOA size has influence on SOA development effort with 10 respondents strongly agreed and 10 agreed to the fact.

5.3.2.2 Experts' response on influence of service type on SOA development effort

Selected experts were asked to rate the influence of service type to SOA development effort. Most experts sampled agreed with our study on service type contribution to SOA development effort as shown in Tab. 11.

Table 11. Experts' response on influence of service type to SOA development effort

Service type	Strongly agree	Agree	Disagree	Strongly disagree
Available service	4	16	0	0
Migrated service	10	10	0	0
New service	4	16	0	0
SOAP	8	12	0	0
REST	9	11	0	0

5.3.2.3 Response on influence of SOA effort modifier factors (EMF) to effort

Expert confirmed that SOA effort modifier factors (EMF) identified in this study are relevant when included in the method to estimate SOA effort as shown in Tab. 12. According to Tab. 12, all sampled experts agreed that the proposed EMFs are relevant in estimating SOA effort.

Table 12. Experts' response on influence of EMF on SOA development effort

SOA effort factor	Description	Strongly agree	Agree	Disagree	Strongly disagree
Product factors	Database complexity	15	5	0	0
	Database size	10	10	0	0
	Integration complexity	11	9	0	0
Service development environment factors	Development tool support	13	7	0	0
	Hardware/Software capabilities	16	4	0	0
Requirements specification factors	Requirement elicitation	6	14	0	0
	Business risk/value	5	15	0	0
	Security requirements	14	6	0	0
Personnel factors	Service developers' experience	17	3	0	0
	SOA Application experience	16	4	0	0
	Team cohesion	14	6	0	0

6. Conclusion and future work

The experiment tested the accuracy of the proposed effort estimation method which was proved to more accurate and within the agreed MRE. Due to the fact that the laboratory experiment in this study was done by students, there was need to subject the proposed effort estimation method to the industry for further validation. In this regard, this research conducted a survey involving 20 sampled experts to validate the proposed effort estimation method. Based on the experiment and expert survey results, the proposed effort estimation method is relevant and valid for SOA based applications. The research recommends further validation of the SOA size metrics and SOA development effort estimation method in laboratory experiments or case studies through the use of industry based projects including medium-scale and large-scale SOA projects.

References

[1] S. Ramacharan and K. Venu Gopala Rao, "Software effort estimation of GSD Projects Using Calibrated parametric estimation models," *ACM Int. Conf. Proceeding Ser.*, **vol. 04**, pp. 113–125 (2016).
[2] F. Sarro and A. Petrozziello, "Linear Programming as a Baseline for Software Effort Estimation," **vol. 1**, (2018).
[3] M. Usman, R. Britto, L. O. Damm, and J. Börstler, "Effort estimation in large-scale software development: An industrial case study," *Inf. Softw. Technol.*, **vol. 99**, pp. 21–40, (2018).
[4] B. Verlaine, I. J. Jureta and S. Faulkner, "A Requirements-Based Model for Effort Estimation in Service-Oriented Systems," pp. 82–94, (2014).
[5] G. Albrecht, A., Gaffney, "No Title," *A Softw. Sci. Validation, IEEE Trans Softw. Eng.*, (1983).
[6] Boehm B.W., "Software Cost Estimation with COCOMO," *Prentice-Hall*, (2000).
[7] T. Urbanek, Z. Prokopova, R. Silhavy, and A. Kuncar, "Using improved analytical programming algorithm for effort estimation in software engineering," *MATEC Web Conf.*, **vol. 76**, pp. 4–7, (2016).
[8] L. O. Brien, , "Cost and Effort Estimation for Service Oriented Architecture (SOA) Projects," *2009 Australian Software Engineering Conference A Framework for Scope*, pp. 101–110, (2009).
[9] Z. Li and L. O'Brien, "Towards effort estimation for web service compositions using classification matrix," *Int. J. Adv. Internet Technol.*, **vol. 3**, pp. 245–260, 2011.
[10] R. Akkiraju and H. Van Geel, "Estimating the cost of developing customizations to packaged application software using service oriented architecture," in *ICWS 2010 - 2010 IEEE 8th International Conference on Web Services*, pp. 433–440, (2010).
[11] E. A. Farrag, R. Moawad, and I. F. Imam, "An Approach for Effort Estimation of Service Oriented Architecture (SOA) Projects," *J. Softw.*, **vol. 11**, pp. 44–63, (2016).
[12] D. Gupta, "Service Point Estimation Model for SOA Based Projects," **Vol no. Lxxviii**, pp. 1–17, (2013).
[13] S. Mishra and C. Kumar, "Estimating development size and effort of business process service-oriented architecture applications," in *2014 2nd International Conference on Systems and Informatics, ICSAI 2014*, pp. 1006–1011, (2015).
[14] COSMIC, *Guideline for Sizing SOA Software*. 2015.
[15] O. K. . Munialo W.S, Muketha M.G., "Size Metrics for Service-Oriented Architecture," *Int. J. Softw. Eng. Appl.*, **vol. 10**, pp. 67–83,(2019).
[16] P. Rijwani and S. Jain, "Enhanced Software Effort Estimation Using Multi Layered Feed Forward Artificial Neural Network Technique," *Procedia Comput. Sci.*, **vol. 89**, pp. 307–312, (2016).
[17] I. Kaur, G. Singh, N. Ritika, and W. Vishal, "Neuro fuzzy — COCOMO II model for software cost estimation," *Int. J. Inf. Technol.*, (2018).
[18] A. Hamdy, "Genetic Fuzzy System for Enhancing Software Estimation Models," **vol. 4**, (2014).
[19] V. Sharma and H. K. Verma, "Optimized Fuzzy Logic Based Framework for Effort Estimation in Software Development," **vol. 7**, pp. 30–38, (2010).
[20] S. Tarannum and M. Arif, "Software Effort Estimation Using Fuzzy Approach," pp. 255–257, (2016).
[21] R. P. S. Bedi and A. Singh, "Software Cost Estimation using Fuzzy Logic Technique," *Indian J. Sci. Technol.*, **vol. 10**, (2017).
[22] P. Reddy, S. K., and R. Sree, "Application of Fuzzy Logic Approach to Software Effort Estimation," *Int. J. Adv. Comput. Sci. Appl.*, **vol. 2**, pp. 87–92,(2011).
[23] N. Shivakumar, N. Balaji, and K. Ananthakumar, "A Neuro Fuzzy Algorithm to Compute Software Effort Estimation," *Glob. J.*

Comput. Sci. Technol. C Softw. Data Eng., **vol. 16**, (2016).

[24] H. P. Patra and K. Rajnish, "A Fuzzy based Parametric Approach for Software Effort Estimation," *Int. J. Mod. Educ. Comput. Sci.*, vol. 10, pp. 47–54, (2018).

[25] Z. Li and J. Keung, "Software cost estimation framework for service-oriented architecture systems using divide-and-conquer approach," in *Proceedings - 5th IEEE International Symposium on Service-Oriented System Engineering, SOSE 2010*, pp. 47–54, (2010).

[26] A. L, "Function Point Analysis FPA on A Team Planning Website Based on PHP and MYSQL," *J. Inf. Technol. Softw. Eng.*, **vol. 08**, (2018).

[27] H. Assal and S. Chiasson, "' Think secure from the beginning ': A Survey with Software Developers." In CHI Conference on Human Factors in Computing Systems Proceedings, ACM, (2019).

[28] S. Bilgaiyan, S. Sagnika, S. Mishra, and M. Das, "A systematic review on software cost estimation in Agile Software Development," *Journal of Engineering Science and Technology Review*, **vol. 10**, pp. 51–64, (2017).