





Enhanced Service Point Approach for Microservices Based Applications Using Machine Learning Techniques

Vinay Raj¹ and Sadam Ravichandra²

¹ BVRIT Hyderabad College of Engineering for Women, Telangana, India
vinay.raj@bvrithyderabad.edu.in

² National Institute of Technology Warangal, Telangana, India
ravic@nitw.ac.in

Abstract. The migration of service oriented architecture (SOA) based applications to microservices architecture is a current research trend in the domain of software engineering. Estimating the effort required for migration is a challenging task as the traditional methods are not suitable for this new architectural style of microservices. Service Points (SP) is one new approach proposed by us for estimating the effort required for the migration of SOA based applications to microservices architecture. However, the use of machine learning techniques gives promising benefits in software effort estimation. To improve the accuracy of the service points approach, multiple regression analysis with the Leave-N-Out policy is applied. The standard service points approach, service points approach with Karner's ratings and proposed machine learning based approach are considered for comparison with actual efforts of the chosen dataset of applications. The accuracy of the models is evaluated using different measures such as MRE, RMSE, MAE, etc. It is clear that the effort estimation using regression analysis gives higher accuracy. Using machine learning techniques improves the accuracy of the effort estimation and helps software architects in better planning and execution of the migration process.

Keywords: Microservices · Effort estimation · Machine learning · Regression

1 Introduction

To over the challenges in existing software architectures such as monolithic and SOA, microservices emerged as a new design style using cloud-based containers for deployment. It is a style of designing applications where each service is a small, loosely coupled, scalable and reusable service that can be designed and deployed independently [1]. Each service should perform only one task and should have its own database and independent deployment architecture. Microservices uses communication protocols like HTTP/REST and JSON for data exchange

between the services [2]. Unlike SOA, microservices can be deployed independently as there is no centralized governance and no dependency on middleware technologies. It is effortless to scale on-demand microservices with the use of cloud-based containers. Microservices architecture suits well with the DevOps style as every task is to be broken into small units, and complete SDLC is to be done independently [3]. DevOps and agile methodologies require the fast design of applications and deployment to production.

With the various benefits of microservices, software architects have started migrating their existing legacy applications to microservices architecture [4]. Many companies, including Netflix, Amazon, and Twitter, have started building their new applications with this style of architecture [5]. As microservices has emerged recently, there is a huge demand in both industry and academia to explore the tools, technologies, and programming languages used in this architecture. However, some software architects are in chaos, whether to migrate to this new style or not, as they are unaware of the pros and cons of using microservices. The major challenge is estimating the effort required to migrate the existing applications to microservices [6, 7].

Effort estimation helps software architects in the proper execution and management of the project. Effective estimation helps in proper scheduling of the software engineering activities. Software effort is given by the formula $\text{effort} = \text{people} * \text{time}$ [8]. It has to be done during the early stage of the application design as it gives insights on the effort and cost required to complete the application. Moreover, estimating the accurate effort required for the migration process is a challenging task. Underestimation and overestimation of the effort required may lead to serious project management issues. Software effort estimation techniques are divided into four types: empirical, regression theory-based, and machine learning techniques based estimation [9]. The empirical way of estimating is very popular as it gives a clear picture of the effort required numerically, and few of the models include function point, use case point, and analogy based techniques. Moreover, these techniques are not suitable for measuring the effort for service-based systems as they are designed for procedural object-oriented systems.

All the traditional approaches available for effort estimation cannot be used directly for service-based systems. Approaches need to be modified and extended to cope with these service-based systems like service oriented architecture and microservices architecture [10, 11]. Service points [12] is one such approach proposed by us in our earlier works to estimate the effort required for migration. It is a formal approach recasted from use case points approach in which the service graph [22] is used to calculate the effort instead of use case diagram that is used in use case points.

Machine learning models have been widely applied in software effort estimation and it has given promising benefits [13, 14]. In order to validate the efficiency of the service points method, N applications of SOA are chosen which are migrated to microservices and the regression analysis is performed on the

chosen datasets. The results of the proposed techniques are compared with the actual efforts, and the accuracy of each technique is also evaluated.

The remaining paper is organized as follows. The technical details of service points approach are presented in Sect. 2. The proposed machine learning based approach is discussed in Sect. 3 and the corresponding experimental results are presented in Sect. 4. Section 5 concludes the paper.

2 Service Points Approach

In this section, the technical details of service points approach are presented. The steps for effort estimation using the service point technique are illustrated in Fig. 1. The brief description of each step is also discussed in this section.

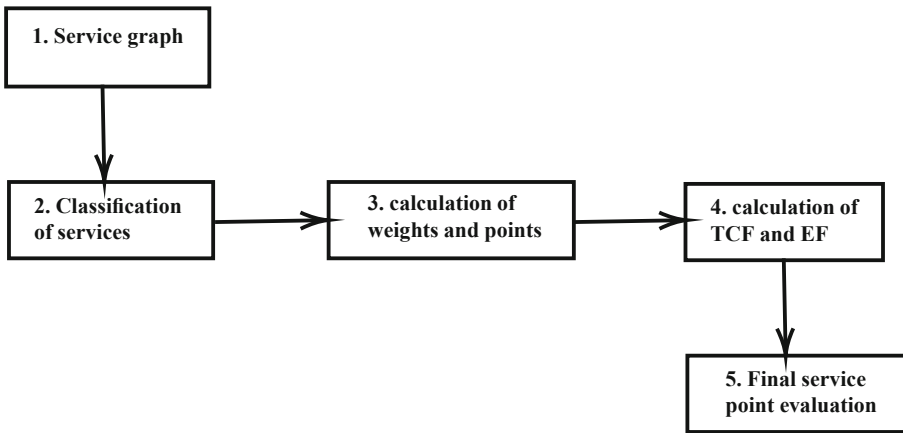


Fig. 1. Service point calculation steps

2.1 Classification of Services

The first step of the service point approach is to classify the services based on the interactions it has with other services. Each service's dependencies on other services are considered and classify them as simple, average, and complex. A service is classified as simple if it interacts with less than four services, average if it interacts with less than eight services, and service is treated as complex if it interacts with more than or equal to eight services [15].

2.2 Calculation of Weights and Points

The next step is to calculate the unadjusted service points based on the weights assigned to each service type. It is calculated by summation of number of services of each type multiplied by weight assigned to corresponding service type.

Unadjusted Service Points (USP) is calculated as shown in Eq. (1).

$$USP = \sum_{i=1}^3 S_i \times W_i \quad (1)$$

where S_i is the number of services of type i and W_i is the corresponding weight of the service of type i where $i=\{\text{simple, average, complex}\}$.

2.3 Technical and Environmental Factors

The final value of the service point depends on 21 technical and environmental factors which contribute to the complexity and the efficiency of the system. Each factor has a value assigned between 0 and 5 depending on the importance and impact the factor has on the system. The rating of each factor between 0 and 5 for each factor is collected through online survey.

Calculation of Technical Complexity Factor (TCF). To calculate the TCF, total weight of the factors is calculated which is obtained by multiplying the value assigned to each factor between 0 to 5 and weights assigned to each factor. Calculation of TFactor is given by Eq. (2).

$$TFactor = \sum_{i=1}^{13} TF_i \times W_i \quad (2)$$

where TF_i is the rating of the technical factor i and W_i is the weight assigned to corresponding factor. Technical Complexity Factor (TCF) is calculated by the below Eq. (3).

$$TCF = 0.6 + (0.01 \times TFactor) \quad (3)$$

Calculation of Environmental Factor (EF). Similarly, the impact of environmental factors in the final service point is evaluated by finding the EF score. To calculate the EF value, the weight of each factor is multiplied with the rating assigned to each factor. It is given by Eq. (4).

$$EFactor = \sum_{i=1}^8 EF_i \times W_i \quad (4)$$

where EF_i is the rating of the environmental factor i and W_i is the weight assigned to the corresponding factor. Environmental Factor (EF) is calculated by the below Eq. (5).

$$EF = 1.4 + (-0.03 \times EFactor) \quad (5)$$

2.4 Final Service Point Evaluation

The final Service Points (SP) is calculated by multiplying the unadjusted service point with both technical and environmental factor values. It is given by the below Eq. (6).

$$SP = USP \times TCF \times EF \quad (6)$$

According to Karner, [15], the effort required to implement each use case point is 20 h. Hence, we do consider the same 20 h for each service point. Therefore, to estimate the final man-hours, the calculated service point should be multiplied by 20 to get the effort required for migration.

We define the naming convention for different approaches used for comparison in this paper. The service points approach with the ratings collected through the online survey proposed by Vinay Raj et al. [12] is denoted as *SP-Vinay Approach*; the service points approach with the Karner's default value as *SP-Karner Approach* and the SP-Vinay approach with regression analysis as *SP-Regression Approach*. These notations are used throughout this paper.

3 Proposed Approach

In this section, the regression approach, description of the datasets and the measures to predict the accuracy of the proposed models are discussed.

3.1 Regression Analysis

It is one of the popular analysis methods to study the relationship between dependent and independent variables and present the relationship in the form of a model [16]. If we have more than two variables, then it is referred as multiple regression and it is the most preferred and applied method for cost estimation [17]. Since the proposed service point approach is based on the multiple factors including USP, TCF and EF, we define the multiple regression based effort estimation which is represented based on the below equation.

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \epsilon \quad (7)$$

where y represents the effort calculated, x_1 represents the size metric calculated with USP of the chosen application and TCF, x_2 represents the adjustment factor (AF) considered as an independent variable in this multiple regression and the coefficients β_1 , β_2 , and β_0 represents the constant values. Here the additional ϵ represents the error induced during the calculation of the effort.

$$Size(x_1) = USP \times TCF \quad (8)$$

Adjustment Factor is calculated as the product of environment factor (EF) and the productivity factor (PF). The value of PF can be considered as 20 h as proposed by Karner, if the projects do not have any historical data. We consider

only the EF in the calculation of x_2 as TCF is already included in the first variable x_1 .

$$AdjustmentFactor(x_2) = {}_pPF \times EF \quad (9)$$

However, we calculate the productivity factor ${}_pPF$ by dividing the actual effort by the SP as it gives the accurate effort required to implement each service point.

$${}_pPF = \frac{ActualEffort}{SP} \quad (10)$$

3.2 Datasets

The use of microservices architecture has just started and there are very few projects which are migrated from SOA. The authors in [18] state that data collection is more important for validation of effort estimation techniques. Due to the inability to access SOA projects developed in the industry and the unavailability of datasets based on UML artifacts in the industry, the study research investigation is collected from [19]. The dataset is represented a dataframe and the size of the dataset is 7 rows with 5 columns. So, gathering the information such as number of services and coupling/dependencies between the microservices of the real time projects was a difficult task. Out of the 7 applications we gathered, 5 applications were collected from Indian IT organizations and one application from a research centre in UK where a team is working on the best approaches for migration of SOA based applications to microservices. The remaining one application is developed at our university by a team of Post Graduate (PG) students which is related to the online exam portal during the pandemic time. The details of the data collected are presented in Table 1. The Unadjusted Service Points (USP) is also calculated for the applications and is presented in Table 1.

Table 1. Characteristics of 7 applications

Application	Total No. of services			USP
	Simple	Average	Complex	
A1	6	5	2	22
A2	2	2	2	12
A3	6	10	2	32
A4	23	15	4	65
A5	0	7	3	23
A6	3	14	0	31
A7	20	15	21	113

3.3 Evaluation Criteria

To evaluate the accuracy of the estimated approach, several frequently used measures [18, 20] are considered such as magnitude relative error (MRE), mean of MRE (MMRE), root mean square error (RMSE), and prediction within 25% of the actual value. The definitions of the measures are discussed below.

- The magnitude of relative error (MRE) is calculated as given below.

$$MRE = \frac{ActualEffort - EstimatedEffort}{ActualEffort} \quad (11)$$

- MMRE is the mean of the MREs of all the applications and it is used to evaluate the prediction performance of the model. The Mean of MRE is calculated as

$$MMRE = \frac{1}{n} \left(\sum_{i=1}^n MRE_i \right) \quad (12)$$

- The Root Mean Square Error (RMSE) evaluates the difference between actual effort and the estimated effort. It is used to find the standard deviation of the errors which occur after applying the proposed method on the datasets. RMSE is calculated as given below.

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (ActualEffort - EstimatedEffort)^2} \quad (13)$$

- To verify whether the predicted values are within m% of the actual values, we use a measure and in software engineering, the value of m is typically set to 25%. The measure PRED(25) is calculated as

$$PRED(25) = \frac{k}{n} \quad (14)$$

where k is the number of observations for those whose MRE is less than or equal to 0.25 and n is the total number of applications.

- The above discussed measures are criticised and behave differently when evaluating prediction models, hence two other measures are suggested [21]. Mean Absolute Error (MAE) is unbiased and it is calculated as given below.

$$MAE = \frac{1}{n} \sum_{i=1}^n | ActualEffort_i - EstimatedEffort_i | \quad (15)$$

where n is the number of applications chosen for evaluation of performance.

- Standardized accuracy (SA) is one the recommended measures for comparing the performance of prediction models which is based on MAE. It is defined as follows:

$$SA = 1 - \frac{MAE_{P_j}}{MAE_{guess}} \times 100 \quad (16)$$

where MAE_{P_j} is the MAE of the proposed approach and MAE_{guess} is the value of MAE of a random guess. The standard accuracy represents the impact of MAE_{P_j} when compared to any random guess. For effort estimation techniques, the value of MAE should be less and SA should be maximum.

4 Experimental Results

The SP-Vinay approach, SP-Karner approach and SP-Regression approach are applied on the 7 applications and the results obtained by comparing these three methods are presented in this section.

4.1 Application of Service Points Method

The service points approach is applied on the 7 applications and the effort is calculated by considering the TCF and EF values. The effort is calculated by considering the collected ratings and also with the Karner's default value. The PF value for calculating the effort is taken as 20 man-hours. The magnitude of relative error (MRE) is also calculated with the help of actual efforts of the applications. The results of the efforts calculated are presented in Table 2.

Table 2. Applying service point approach to applications.

Application	Actual effort [h]	SP-Vinay approach		SP-Karner's approach	
		Estimated effort [h]	MRE	Estimated effort [h]	MRE
A1	396	326	0.176	305	0.229
A2	140	178	-0.271	166	-0.185
A3	587	474	0.192	444	0.243
A4	1205	962	0.201	902	0.251
A5	518	340	0.343	319	0.384
A6	502	459	0.08	430	0.143
A7	2034	1673	0.177	1567	0.229

4.2 Application of Proposed Regression Model

The proposed regression method is applied on the same dataset and the function for effort estimation is obtained. First, the variables x_1 and x_2 are calculated with the Eqs. (8) and (9). The calculated values of the variables are presented in Table 3. These values x_1 and x_2 will be used in the calculation of the effort using the regression model. We consider the TCF value which is calculated using the ratings collected through online survey in the regression analysis.

Using the calculated values of variables, we applied multiple linear regression on the applications. However, to improve the accuracy we apply Leave-N-Out policy where we train the model only on first 5 applications to calculate the

coefficients and test with the remaining two applications. The values of the coefficients calculated are presented in the effort estimation function, given in the below equation.

$$\hat{y} = -274.10 + 15.017x_1 + 17.136x_2 \tag{17}$$

Table 3. Variable values for multiple regression analysis.

Application	Size (x_1)	AF (x_2)
A1	23.43	16.8
A2	12.78	10.9
A3	34.08	17.1
A4	69.22	17.3
A5	24.49	21.1
A6	33.01	15.1
A7	120.34	16.8

Using the calculated coefficient values $\beta_0 = -274.10$, $\beta_1 = 15.017$, and $\beta_2 = 17.136$, we test the remaining two applications. The efforts calculated using the generated coefficients are presented in the Table 4. It is clear from the table that the values are very close to the actual efforts of the applications. The proposed regression model works as recommendation system for estimating the effort required for migration of SOA applications to microservices. The coefficients generated are used to calculate the efforts for all the other applications which are used for training the model as well.

Table 4. Application of regression model to testing data

Application	Actual effort [h]	Estimated effort (Regression) [h]	MRE
A6	502	480.36	0.043
A7	2034	1940.99	0.045

4.3 Comparison

The estimation function is calculated for all the 7 applications by regression analysis on the 6 applications and leaving one application everytime. The values of the coefficients are calculated and the effort by regression analysis are generated for all the applications. The results of the comparison of all the proposed methods are presented in Table 5. For each approach, the estimated effort in hours and estimation success values are given in the table. It is clear that the efforts

calculated through the regression analysis give better values closer to the actual efforts required for migration. For application A2, the efforts estimated through SP-Vinay approach and SP-Karner’s approach are more than the actual efforts. Hence the estimation success percentage is more than 100 which is marked as *. Generally success percentage more than 100 does not makes sense. So, only the SP-Regression model gives better result for the application A2. The efforts estimated by the standard SP-Vinay approach, SP-Karner approach, SP-Regression approach and the actual efforts are compared and presented as a line graph as shown in Fig. 2. The x -axis represents the 7 applications and the y -axis represents the efforts in man-hours for each application. From the graph, it is clear that the effort estimated with the regression model is close to the actual efforts of all the applications.

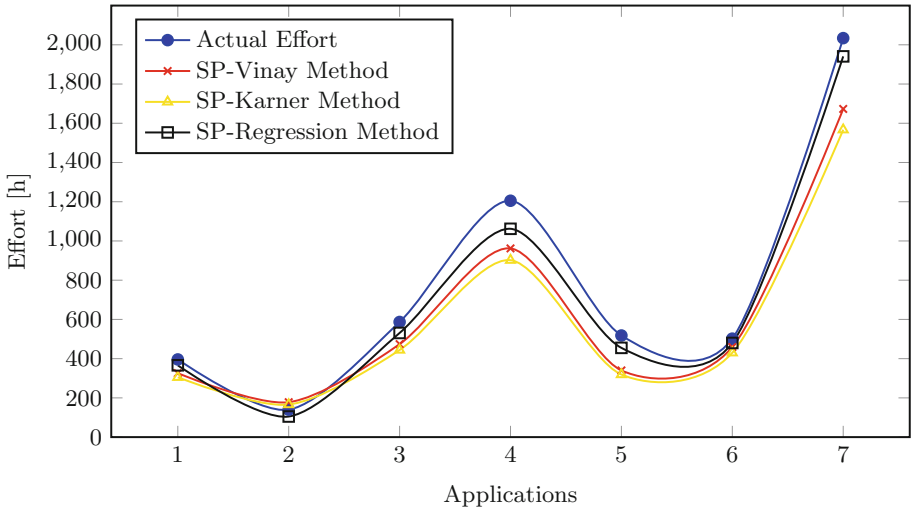


Fig. 2. Comparison of efforts estimated by proposed methods.

The accuracy of the proposed methods are evaluated using the measures discussed in Sect. 3.3 and the values are presented in Table 6. From the results, the MAE value is very less and SA value is also better for effort estimated through regression model. Though the MMRE and PRED values are close to each other, the RMSE value is better only for the regression model.

Table 5. Comparison of proposed effort estimation techniques

Application	Actual effort [h]	SP-Vinay approach		SP-Karner's approach		SP-regression approach	
		Estimated effort [h]	Estimation success (%)	Estimated effort [h]	Estimation success (%)	Estimated effort [h]	Estimation success (%)
A1	396	326	82.32	305	77.02	366	92.42
A2	140	178	127*	166	118*	105	75.0
A3	587	474	80.74	444	75.63	531	90.45
A4	1205	962	79.83	902	74.85	1062	88.13
A5	518	340	65.63	319	61.58	455	87.83
A6	502	459	91.43	430	85.65	480	95.61
A7	2034	1673	82.25	1567	77.04	1941	95.42

Table 6. Accuracy of the proposed methods using different measures.

Approach	MMRE	RMSE	PRED(25)	MAE	SA
SP-Vinay approach	0.128	185.95	0.857	138.57	86.143
SP-Karner approach	0.184	234.24	0.714	178.42	82.158
SP-regression approach	0.107	74.55	0.857	63.24	93.67

5 Conclusion

Effort estimation is an important software engineering activity that helps project managers and architects effectively schedule the project. With the evolution of microservices, companies are migrating existing legacy applications to microservices architecture. Service points is an approach to estimate the effort required for the migration. To improve the accuracy of the service points approach, a machine learning model using multiple linear regression with Leave-N-Out policy is proposed, where the model is trained with N applications and tested with the remaining all-N applications. We have taken 7 applications designed with SOA and migrated to microservices, and the efforts are calculated using the regression function. The accuracy of the proposed models is evaluated using different metrics such as MRE, RMSE, PRED, MAE, and SA. The comparison results show that the efforts estimated using the proposed regression model are close to the actual efforts, and the error rate is very low compared to the SP standard approach and SP-Karner's approach.

References

1. Thönes, J.: Microservices. *IEEE Softw.* **32**(1), 116 (2015)
2. Raj V, Ravichandra S. Microservices: a perfect SOA based solution for enterprise applications compared to web services. In: 2018 3rd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), pp. 1531–1536. IEEE, 18 May 2018

3. Salah, T., Zemerly, M.J., Yeun, C.Y., Al-Qutayri, M., Al-Hammadi, Y.: The evolution of distributed systems towards microservices architecture. In: 2016 11th International Conference for Internet Technology and Secured Transactions (ICITST), pp. 318–325. IEEE, 5 December 2016
4. Raj, V., Sadam, R.: Patterns for migration of SOA based applications to microservices architecture. *J. Web Eng.* **10**, 1229–46 (2021)
5. Raj, V., Sadam, R.: Evaluation of SOA-based web services and microservices architecture using complexity metrics. *SN Comput. Sci.* **2**(5), 1–10 (2021). <https://doi.org/10.1007/s42979-021-00767-6>
6. Taibi, D., Lenarduzzi, V., Pahl, C.: Processes, motivations, and issues for migrating to microservices architectures: an empirical investigation. *IEEE Cloud Comput.* **4**(5), 22–32 (2017)
7. Soldani, J., Tamburri, D.A., Van Den Heuvel, W.J.: The pains and gains of microservices: a systematic grey literature review. *J. Syst. Softw.* **1**(146), 215–232 (2018)
8. Pendharkar, P.C., Subramanian, G.H., Rodger, J.A.: A probabilistic model for predicting software development effort. *IEEE Trans. Softw. Eng.* **31**(7), 615–24 (2005)
9. Subramanian, G.H., Pendharkar, P.C., Wallace, M.: An empirical study of the effect of complexity, platform, and program type on software development effort of business applications. *Empirical Softw. Eng.* **11**(4), 541–53 (2006)
10. Canfora, G., Fasolino, A.R., Frattolillo, G., Tramontana, P.: A wrapping approach for migrating legacy system interactive functionalities to service oriented architectures. *J. Syst. Softw.* **81**(4), 463–80 (2008)
11. Siddiqui, Z.A., Tyagi, K.: A critical review on effort estimation techniques for service-oriented-architecture-based applications. *Int. J. Comput. Appl.* **38**(4), 207–16 (2016)
12. Raj, V., Ravichandra, S.: A novel effort estimation approach for migration of SOA applications to microservices. *J. Inf. Syst. Telecommun. (JIST)*. **3**(36) (2021)
13. Sehra, S.K., Brar, Y.S., Kaur, N., Sehra, S.S.: Research patterns and trends in software effort estimation. *Inf. Softw. Technol.* **1**(91), 1–21 (2017)
14. Wen, J., Li, S., Lin, Z., Hu, Y., Huang, C.: Systematic literature review of machine learning based software development effort estimation models. *Inf. Softw. Technol.* **54**(1), 41–59 (2012)
15. Karner, G.: Resource estimation for objectory projects. *Objective Syst. SF AB.* **17**(17), 1–9 (1993)
16. Montgomery, D.C., Peck, E.A., Vining, G.G.: *Introduction to Linear Regression Analysis*. Wiley, Hoboken 9 Apr 2012
17. Shepperd, M., Schofield, C.: Estimating software project effort using analogies. *IEEE Trans. Softw. Eng.* **23**(11), 736–43 (1997)
18. Sarro, F., Petrozziello, A., Harman, M.: Multi-objective software effort estimation. In: 2016 IEEE/ACM 38th International Conference on Software Engineering (ICSE), pp. 619–630. IEEE, 14 May 2016
19. Munialo, S.W., Wanjala, S.: A size metric-based effort estimation method for service oriented architecture systems (Doctoral dissertation, MMUST) (2020)
20. Menzies, T., Yang, Y., Mathew, G., Boehm, B., Hihn, J.: Negative results for software effort estimation. *Empirical Softw. Eng.* **22**(5), 2658–2683 (2016). <https://doi.org/10.1007/s10664-016-9472-2>

21. Port, D., Korte, M.: Comparative studies of the model evaluation criterions MMRE and pred in software cost estimation research. In: Proceedings of the Second ACM-IEEE International Symposium on Empirical Software Engineering and Measurement, pp. 51–60, 9 October 2008
22. Raj, V., Sadam, R.: Performance and complexity comparison of service oriented architecture and microservices architecture. *Int. J. Commun. Netw. Distrib. Syst.* **27**(1), 100–17 (2021)