EffortEst- An Enhanced Software Effort Estimation by Analogy Method

Soulakshmee D. Nagowah[1],Shaheema S.B. Rumjaun[1], Kheshan A. Gutteea[1], Leckraj Nagowah[1]

[1]Computer Science & Engineering Department,
University of Mauritius,
Reduit, Mauritius.
s.ghurbhurrun@ uom.ac.mu,
shaheema.rumjaun@umail.uom.ac.mu,
kheshan1090@hotmail.com
l.nagowah@ uom.ac.mu,

Abstract: ***Over the past few years, large-scale software project development has become the point of growing interest to many organizations and thus, predicting the size, cost and effort of software projects has become a very significant task to project managers. Often inaccurate prediction results into software projects exceeding budget as well as being out of schedule. Therefore, software project managers have been introduced to numerous software tools and methods in recent years to automate their tasks. The paper presents some existing analogy-based software estimation tools used by project managers and these tools are critically analyzed to identify shortcomings. Finally an enhanced software effort estimation method is proposed. A system prototype named EffortEst has been implemented and evaluated based on the enhanced method. EffortEst provides the near-best estimation of software project effort with limited user intervention.***

Keywords: Software Effort Estimation, Analogy, Case- Based Reasoning, Prototype

(Article history: Received 16 September 2016 and accepted 9 December 2016)

1. Introduction

One of the most challenging and fundamental activities in software development is effective software project estimation. Without sound and reliable estimate, proper project planning and control is far from being possible. Many companies are therefore opting for automated estimation software which is considered to be more reliable nowadays. Various techniques are used to automate the process of effort estimation. Analogy-based technique is considered as one of the most promising technique which outperforms the algorithmic methods according to Jingyue et al. [1-3] and by the most recent work of Angelis et al. [4, 5]. Compared to the algorithmic methods, analogy-based prediction is firstly, easier to understand and apply and secondly, it is said to provide a more accurate result [6, 7]. Aamodt and Plaza [9] stated that analogy-based reasoning is an approach to incremental,sustained learning as each time a problem is resolved, a new experience is retained making it available immediately for new problems[6,8].

Estimation by analogy involves firstly, the characterization of the proposed project, secondly the selection of the most similar completed projects whose characteristics have been stored in a historical database and lastly, the derivation of the estimate for the proposed project from the most similar completed projects [10].The current study has introduced a newly developed software effort estimation tool known as EffortEst based on the idea of analogy. Moreover, in this particular study, previous researches about estimation by analogy are analyzed and an evaluation has been carried out to determine how close and accurate the prediction of the new software is, to that of the existing tools. The remaining part of the paper is structured as follows: Section 2 elaborates on some existing analogy-based software estimation tools. A critical analysis of the existing analogy-based software is done in Section 3. Section 4 presents the system architecture for EffortEst. Implementation of a system prototype is discussed in Section 5. An evaluation of the system prototype is dealt in Section 6. Finally, conclusion is discussed in Section 7.

2. Related Work

This section describes some existing analogy-based software estimation tools namely ANGEL, ESTOR and ACE which are most commonly used to predict software effort.

2.1 ANGEL

One of the most popular tools in the literature of software estimation by analogy using case-based reasoning is ANGEL tool developed in the late 90s [6, 11]. ANGEL is popular for its completeness in implementation and does not assume the use of a particular project dataset by the estimators. Instead the estimator can configure ANGEL to use whatever project data set is available.

Moreover, in order to be able to proceed with the ranking process to determine a best matching analogue, ANGEL considers the use of Euclidean distance to perform a comparison between the target project and the analogue project. Furthermore, it focuses heavily on the performance metric defined as MMRE (Mean Magnitude Relative Error) which is known to be one of the most common measure to predict accuracy in software effort estimation studies [3,6,12,13].

MRE = |Eact_i –Epred_i)/ Eact_i| (1)[14]

Where,

Eact is the actual effort and *Epred* is the estimated effort.

MMRE [10] use the Magnitude of Relative Error (MRE)in their calculations, which is defined as:

$$\frac{1}{n}\cdot\sum_{i=1}^{i=n}\left(\frac{(|Epred-Eact|)}{Eact}\right)_i \quad \textbf{(2)}[15]$$

ANGEL then ranks the potential analogues according to their distances from the target project. The estimators specify which metrics are to be used when ANGEL searches for the analogues and also they determine the best metrics subset to be used when searching for a particular dataset. The tool considers all possible subsets of metrics and selects the subsets that minimize the MMRE for the dataset. ANGEL basically has no specific algorithm to derive the effort value for the target project but instead the principle of this tool is to consider the effort of the closest analogue as being the effort value for the target project [13].

2.2 ESTOR

ESTOR is an early implementation of an analogy-based tool to estimate software project effort [6]. It was developed as a proof-of-concept system by Mukhopadhyay et al. [16] in order to evaluate the feasibility of case-based reasoning in software effort estimation. Similar to ANGEL, this approach makes use of the Euclidean distance metric to measure similarity between the actual project and the analogues as well as to retrieve the best analogues upon which the prediction of effort for the new case will be based [6]. However, in accordance to the concept of case- based reasoning, ESTOR makes use of the basic case-based reasoning algorithm and makes use of the three best analogues to compute effort based on the inverse rank weighted mean [17].

EFFORT= [(3*effort of closest analogue) + (2* effort of second closet analogue) + (1*effort of last closet analogue)] /6 (3)[17]

2.3 ACE

Developed by Emilie Mendes et al. [17] at the Centre for Advanced Empirical Software Engineering Research Group (CAESAR) in the late 90s, ACE (Algorithmic Cost Estimator) aims at exploring the benefits of analogy-based estimation. ACE is an algorithm which computes the difference between the target project and each completed project in the database. ACE principles involve the use of similarity functions which should be defined to be able to compute the similarity distance of each analogue with respect to the target project. Also the similarity function helps in the ranking of analogues in terms of most similar and least similar [6,18]. Usually, this method uses the project with the lowest mean rank as analogue for the target project and in case there are more than one project ranked as first or second, ACE chooses the most accurate project to be used as analogue for the target project. Finally, ACE depends highly on a function point (FP) metric to estimate and adjust the effort value for the target project [19].

Effort = (Effort $_{Analogue}$ / FP $_{Analogue}$) x FP$_{Target}$ (4)[19]

In case two analogues are being used for the effort estimation, ACE makes use of the following formula for size adjustment [20].

Effort$_{Target}$=[(Effort$_{Analogue1}$/FP$_{Analogue1}$)+(Effort$_{Analogue2}$/FP$_{Analogue2}$)]x(FP$_{Target}$/2) (5)[20]

Assuming a target project with a size measure of 320 function points, a source analogue being identified with 350 function points and the effort required to complete the source analogue was 1200 person-hour, the effort prediction for the target project will be 1097 person-hour [19] using the extrapolation formula (4).

3. Critical Analysis

This section proposes a concise discussion and analysis explaining the need for a new software tool. ANGEL, ESTOR and ACE are the three most commonly used analogy-based tools but nevertheless despite the numerous advantages they provide; each of them has a degree of deficiency.

First and foremost, ACE is known to provide a lower degree of accuracy and also a lack of details is a significant contributor to ACE analysis conservatism [21]. ESTOR and ANGEL are said to use the same principles in order to provide a list of most similar analogues but though, the content of the list is not always accurate.

Moreover, ANGEL is computationally expensive as compared to the other methods since it saves and compute similarity for all cases. It is intolerant to noise and also gives little usable information regarding structure of data [6]. As stated by Walkerden and Jeffery [18], ESTOR is said to be best utilized as an expert support system instead of an expert system since it cannot generate adjustment heuristics unlike the human.

ESTOR also requires additional domain knowledge in order to succeed to accurately estimate projects from very different environments (for example, embedded military systems) but yet its predictions are considered to be more accurate than ANGEL.

The scenario below has been used to check the accuracy in results by ANGEL.

"*A project manager is given the task to estimate software effort based on the dataset given in the table below. In order to be able to proceed with this particular task, he is allowed to use any of the existing software tools. He chose ANGEL for effort prediction of the target project. How reliable is the list of similar cases provided by ANGEL during the effort estimation process?*"

I. TABLE1. Dataset

Project Category	Web Technology
Organization Size	23
Number of Application tiers	3
Middleware used	SOAP
Programming Language	JAVA
Backend Technology	Oracle
Development Tool	EESS
Development Process	RAD
Development Locations	Developer
Number of organization locations	6

Using ANGEL_PLUS 2.0 [22], the above dataset is fed into the software to acquire a list of best matching analogues. Figure 1shows an ordered list of analogues gained from the software tool after the ranking process along with their similarity value which shows the closeness to the target project. The shorter the distance between the analogue and the target project, the more they are said to be similar to each other.

Project Name	Distance between analogue and target project	Effort
Web Project B	0.447	30
Web Project A	0.632	60.9
Project CRM D	0.707	56.09
Web Project C	0.724	57.52
Web Project E	0.774	50
Web Project D	0.724	57.52
Project CRM C	0.774	89.74
Project ERP A	0.774	49.87
Project ERP E	0.790	157.86
Project ERP D	0.790	331.00
Project CRM B	0.790	34.90

Figure 1. Most similar cases generated by ANGEL

It can be noticed in Figure 1 that Web Project D has a similarity distance of 0.724 and Web Project E has a similarity distance of 0.774. According to the ranking process, basically the Web Project D is closer to the target project but however, ANGEL_PLUS 2.0 considers Web Project E as being closer. This implies that the result gained from ANGEL_PLUS 2.0 is not so accurate and consequently any decision made according to the predictions of ANGEL_PLUS 2.0

might run high risk of impacting on the company's profitability.

In the event of finding a solution to the above mentioned issues, this paper proposes EffortEst as a novel software tool to reduce the risk of inaccurate predictions.

EffortEst makes use of the Case-Based Learning algorithm (CBL1) [23] which in turn, outcompetes ANGEL for its numerous advantages. EffortEst will enable its end-users to view details about any project in the list of analogues. Additionally, effort may be estimated in accordance with the best matching case and the end user is allowed to base his predictions on any of these projects in the list of analogues in order to view the result. In short, EffortEst is designed to better meet the software development requirements.

4. System Architecture

This section comprises of an interaction model as well as an architecture diagram of EffortEst. Additionally, a concise description of the main algorithm is given along with an explanation of how EffortEst proceeds through the four steps of cased-based reasoning to accomplish its goal of effort estimation.

4.1 Interaction Model

Figure2 is an interaction model which is basically showing an interaction between EffortEst and its end user describing the inputs, flow of processes and outputs.

The EffortEst System consists of the step by step processes to perform estimation. Similar cases are retrieved; new cases are solved and tested.The system also has an integrated learning part which considers any tested solved case as being an analogue for solving future cases. This learning section is responsible for storing of the new case in the database. The database is accessible only to authorized users andonly certain modifications can be made by the administrator. However, the latter cannot modify any details about historical cases since this might lead to inaccurate predictions, thus making the system unreliable.

The Project Manager is the system end-user who is considered as an authorized user having a unique login and password. Ultimately, he can request system to generate report of the prediction made.

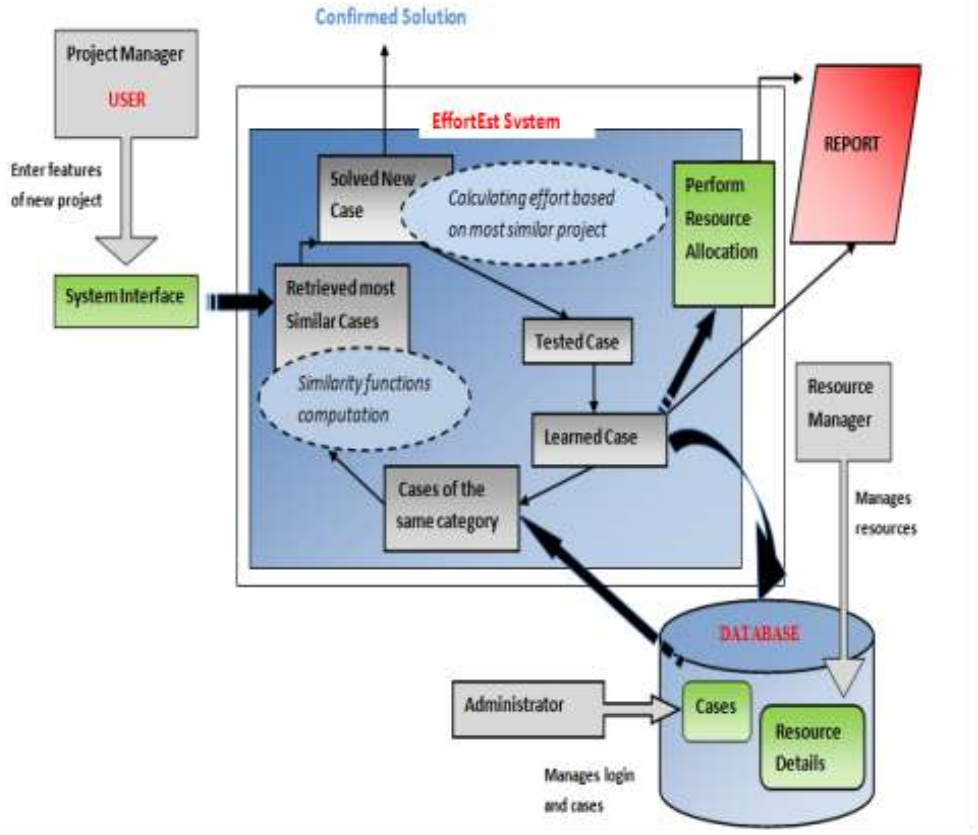

Figure 2. Interaction Model

4.2 Architectural Diagram

Figure 3 shows an architectural diagram which is giving clear description of the different layers of the EffortEst system.

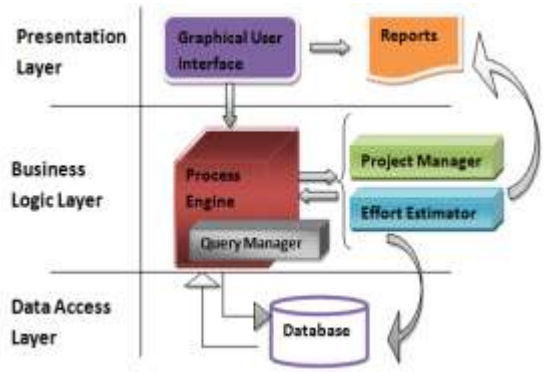

Figure 3. Architecture Diagram

4.2.1 Presentation Layer

The Presentation Layer is a simple representation of the interface and output of the EffortEst system. At this level, the access rights of users are also dealt with and usually the output of the system is in form of reports.

4.2.2 Business Logic Layer

The Business Logic Layer consists of the Project Manager Tool, the Effort EstimatorTooland finally the Process Engine which also includes a Query Manager Tool.

Process Engine: The task of the Process Engine is to intake the target project attributes from the graphical user interface and then queries the database accordingly.

Query Manager Tool: Query Manager Tool gets a list of historical cases from the database and stores it for future use by the Project Manager.

Project Manager Tool & Effort Estimator Tool: Project Manager Tool makes use of PMEE's algorithm to compare the historical project attributes with that of the target project and then provides the effort estimator tool with a list of most similar cases. The user needs to input a set ofrequired details about the target project in the system.First and foremost, the system reads the user inputsand then proceeds with the comparison of project through the use of IF-ELSE condition in PMEE's algorithm and also byapplying the concept of similarity computation so asto retrieve a list of historical projects.

The Effort Estimator then considers the best matching case from the list in order to compute effort for the target project. Additionally, Effort Estimator retains the target project as a new experience to solve future cases and also it is able to generate reports upon request of end-users.

Following the four steps of case based reasoning concept, PMEE & Effort Estimator Tool make use of EffortEst Algorithm to be able to proceed with the task of effort estimation.

PMEE's Algorithm

1. Read user inputs (target project attributes).
2. Retrieve all historical projects from the database and stores them as a list of cases.
3. Make use of IF-ELSE conditions to find best matching analogue

```
// Loop through list of cases
// Check for cases similar to target cases
While (criteria ==true){
IF Case is similar THEN
//apply rules on case attributes
If (condition == true){
                    Assign importance weight
                    Assign feature weight
                    Apply similarity functions
}
If (new_condition == true){
                    Assign importance weight
                    :
                    :
}
END IF
} //END LOOP
Save similarity value for all cases in a list.
```

4. Display an ordered list of most similar cases

EffortEst Algorithm

- ***Step 1 (Retrieval of similar cases)***

According to the statement of Gray and MacDonell [24], the accuracy of analogy-based depends on finding similar projects. Consequently, as defined in the PMEE's Algorithm, using the method of Shepperd and Schofield [3], similarity between projects are measured by comparing the different attributes of each case in accordance to a set of rules. Moreover, similarity computation in this study is subdivided into importance value which is gained

while measuring similarity at the object level and feature value which is acquired at the feature level.

The importance value which is decided over a Global Similarity measure has been discussed by R. Bergmann [6]. The global similarity measure combines local similarity measure as well as takes care of the different importance of attributes. Bergmann made use of the six-point scale similarity and brand extensions concept [25] to determine feature importance of a car with respect to car diagnosis.
The six-point scale similarity and brand is an approach to distinguish between "very important" and "less important" aspects where a value of 6 is assigned to features with high importance and a value of 1 is used for the features with low importance. The value considered by the six-point scale is 6 since only six most important factors are being used. However, just like the six-point scale, a ten-point scale or even four-point scale [25] may be used.

Bergmann made use of the six-point scale in his car diagnosis system in order to distinguish between the high importance and low importance features of a car [26]. Likewise, since EffortEst identifies six of its attributes as most important, the six-point scale is chosen to categorize project attributes in terms of "very important" and "less important".
In accordance to the argument of Shepperd and Scoffield [6, 27], feature value which ranges from 0 to 1, is dependent on the idea of Euclidean distance. In addition, feature value is usually used to determine the local similarity percentage between each attribute of each case.

Feature Value = Difference between value of current project and value of analogue project / the largest of the two values (6)

- ***Step 2 – Reuse of historical data***

Research made by Li et al. [28], reported that the feature value helps in providing a fair prediction when used with similarity functions. Thus, using the similarity function, EffortEst ranks the analogues from most similar to least similar in the process of case retrieval and matching. An appropriate similarity function formula adapted from Bergmann's formula is shown below:

Similarity Functions = (1/20) * (∑(feature weight*importance weight)) (7)

Referring to research about Memory-based reasoning [29] and CBL1 Algorithms [23], Case-based reasoning considers a maximum of 10 most similar cases to perform estimation. Since EffortEst is depending on the concept of case-based reasoning, it is also considering a maximum of 10 most similar cases when computing the similarity functions unlike Bergmann who took into consideration 20 cases since it is said that the more sample cases used, the more accurate will be the result of predictions.
Besides, after ranking the analogues from most similar to least similar, EffortEst has derived the following formula to perform estimation by adapting to the effort of the most similar analogue.

Effort Estimation= [(MaxPosSim + (MaxPosSim - SFV)) / MaxPosSim] x Effort value (Best Case) (8)

Where,
MaxPosSim, is the maximum similarity between the two cases,
SFV, is the similarity function value for the best case.

- *Step 3 & 4 – Revise and Retain case*

After the process of effort value adaptation, EffortEst automatically stores the current project along with the result gained as adapted effort value for a new experience to solve future cases.

4.2.3 Data Access Layer

The Data Access Layer provides simplified access to data stored in the database, thus making the system to successfully response to the user queries.

5. System Prototype

The implementation of the EffortEst software tool involves the use of a number of technologies. JAVA programming language has been used since it allows the creation of modular programs and reusable code and also because it is platform- independent. When it comes to data storage, MySQL database has been used for the numerous advantages it has: It is open source software that is easy to use. It is secure in terms of access rights, scalable, fast and supports several development interfaces. Moreover, MySQL database can run on many operating systems including Windows, Linux and many more.As development environment, Eclipse IDE was used with NetBeans IDE since they both provide framework for desktop application developers and moreover, a large number of features such as update facilities are also provided. Both Eclipse and Netbeans provide for a rich set of APIs. Netbeans

even provide for drag and drop facilities which ease the task of designing user-friendly interfaces.

6. Evaluation

This section presents an evaluation of EffortEst against the existing software estimation tools described in Section 2.

6.1 Feature comparison of software estimation tools

Table 2 shows a feature comparison of the estimation tools. Considering Table 2, it can be seen that even if all the four systems are based on the idea of estimation by analogy, EffortEst makes use of a combination of attributes completely different from the other three systems. Unlike the other three existing tools, EffortEst makes use of a combination of the case-based reasoning and rule-based reasoning approach. Furthermore, EffortEst makes use of the Similarity functions approach as in ACE instead of Mean Magnitude Relative Error (MMRE) which is used by ESTOR and ANGEL. Likewise, the table shows the different attributes used by the different software tools.

II TABLE 2. FEATURE COMPARISON OF SOFTWARE ESTIMATION TOOLS

Comparative Factors	*ANGEL*	*ESTOR*	*ACE*	*EffortEst*
Use of Estimation by Analogy	✓	✓	✓	✓
Involves Euclidean Distance	✓	✓	✓	✓
Ranking	✗	✓	✗	✓
Function Points Analysis	✓	✓	✓	✗
Expert judgement	✗	✓	✗	✗
Estimation based on most similar case	✓	✓	✓	✓
Use of Case-based reasoning	✗	✗	✗	✓
Mean Absolute Error	✓	✓	✗	✗
Involves Rule based	✗	✗	✗	✓
Programming Language	Visual Basic	Any high-level language	Pascal Programming/ C++/ Networking	Java
Development Environment	VB.net	depends	depends	Eclipse NetBeans Xampp
Database	Microsoft Access	MySQL/ Microsoft Access	Oracle SQL Developer	MySQL
Interfaces	Visual Basic	Any high-level language	Pascal Programming/C++ Network Programming	User-friendly Consistent Simple with help provision

6.2 Evaluation of Results

Under this section, an evaluation of the existing systems is done against EffortEst using the dataset given in table 1 in section 3 so as to be able to determine how accurate the prediction of EffortEst is as compared to the other systems.

As discussed in the previous sections, EffortEst makes use of similarity functions to retrieve and rank the analogues. Figure 4 shows a list of most similar cases retrieved by EffortEst based on the dataset in Table 1.

ANGEL and ESTOR form part of the case-based reasoning approach and thus, case retrieval and matching is one of the fundamental tasks they carry.

Project Name	Effort
Web Project B	30
Web Project A	60.9
Project CRM D	56.09
Web Project D	57.52
Web Project C	57.52
Web Project E	50
Project ERP A	49.87
Project ERP B	100.00
Project ERP E	157.86
Project CRM C	89.74
Project CRM A	20.4

Figure 4. List of Similar Projects generated by EffortEst

According to the scenario provided in section 3, the predicted effort for the target project is considered to be 30 as ANGEL considers the effort of the closest analogue as the effort for the target project. Even if the best matching case is not so closely similar to the target case, ANGEL will still consider the prediction for the target case to be same as that of best matching case. Consequently, this shows a risk of inaccurate result and as also the predicted result by ANGEL is not so closely related to that of EffortEst.

Coming to ESTOR, it is also said to depend highly on the MMRE formula used by ANGEL and thus, the risk of inaccurate prediction here, is not less. Though, based on the case based reasoning approach, ESTOR makes use of formula (3) which takes intoconsideration the three best analogues to perform effort estimation and as a result, the predicted effort by ESTOR according to Table 3 would be as follows:

Effort= [(3*30)+(2*60.9)+(1*56.09)] /6 = 44.65

ESTOR is said to perform better than any of the three existing tools and since the predicted results by ESTOR and EffortEst are very much similar, EffortEst can be considered as being relatively better than that of ACE and ANGEL.

Compared to ESTOR and ANGEL, ACE software makes use of function points for case retrieval and matching. Using an online function point calculator along with the ACE Software, the result gained as estimated effort for the target project is as such:

FP for best analogue =48.23
FP for target project =130.9
Effort for Analogue = 30

Therefore, Effort = (30/48.23) * 130.9 =81.42

As mentioned in section 3, among all the three software tools, ACE is considered as being the most inaccurate one and if the results provided by ACE is analyzed, it can be determined that it is indeed very far from the results provided by any of the other three software tools.

Table 3 summarizes the different results gained from the different systems with respect to the dataset in Table 1.

III TABLE 3. RESULTS OF ANALYSIS

Software Tools	*Estimated Effort*
ANGEL PLUS 2.02	30
ACE	81.42
ESTOR	44.65
EffortEst	45.48

7. Conclusion

Software Cost Estimation has always been a topic of discussion for project managers as failure to estimate effort can drastically impact the budget and schedule. The paper analyses three software cost estimation tools namely ANGEL, ESTOR and ACE and proposes an enhanced software effort estimation method. Asystem prototype named EffortEst has been implemented and evaluated based on the enhanced method. The system provides the near-best estimation of software project effort with limited user intervention. As EffortEst provides a result closest to that of ESTOR which is considered to be the best software tool with the most accurate result, it can be said that EffortEst is reliable enough in terms of accuracy. Furthermore, it is said that the best estimates can be calculated by taking into consideration the maximum number of analogues. EffortEst makes use of a number of analogues as

compared to other tools.

References

[1] J. Li, R. Conradi, " Issues of Implementing the Analogy-based Effort Estimation in a Large IT Company", Norweigian University of Science and Technology, Trondheim, Norway, 2007.

[2] L. Angelis, I. Stamelos, "A simulation tool for efficient analogy based cost estimation. Empirical Software Engineering", 7(4), 2002.

[3] M. Shepperd, C. Schofield,"Estimating software project effort using analogies",Software Engineering, IEEE Transactions on, 23(11):736–743, 0098-5589 ,1997.

[4] E. Mendes, N. Mosley, S. Counsell, "A replicated assessment of the use of adaptation rules to improve web cost estimation", In International Symposium on Empirical Software Engineering (ISESE'03), pages 100–109, 2003.

[5] E. Mendes, B. Kitchenham, "Further comparison of cross-company and within-company effort estimation models for web applications", In 10th IEEE International Symposium on Software Metrics (METRICS'04), pages 348–357, 2004.

[6] J. Keung, "Software Development Cost Estimation Using Analogy: A Review", Software Engineering Conference,ASWEC'09. Australian. IEEE, 2009.

[7] T. Mukhopadhyay, S. Vincinanza, M. J. Pietula: Estimating the feasibility of a case-based reasoning model for software effort estimation. MIS Quarterly, 1992.

[8] V. Khatibi, D. N. A Jawawi, "Software Cost estimation methods", Journal of Emerging Trends in Computing and Information Sciences, Vol. 2 :1, 2011.

[9] A. Aamodt,, E. Plaza, "Case-based reasoning: Foundational issues, methodological variations, and system approaches", Artificial Intelligence Communications, 7:39–59, 1994.

[10] L. Wu, "The Comparison of the Software Cost Estimating Methods", University of Calgary, 1997.

[11] H.Y. A Abu Tair, "Predicting the cost estimation of software projects using case-based reasoning", ICIT 2013 The 6th International Conference on Information Technology. Supported by Research Center of College of Computer and Information Sciences, King Saud University, 2013.

[12] R. Bisio, F. Malabocchia, "Cost estimation of software projects through case base reasoning", In 1st Intl. Conf. on Case-Based Reasoning: Research and Development, 1995.

[13] E. Stensrud "Alternative approaches to effort prediction of ERP projects", Information and Software technology conference 43.413-423. Supported by Stensrud Consulting, Austliveien 30, N-0752, Norway, 2001.

[14] D. Singh,K. S.Dhindsa, "Estimation By Analogy In The Perspective Educational Web Hypermedia Applications", International Journal of Information Technology and Knowledge Management. Volume 4, No. 1, pp. 305-313, 2011.

[15] R. M. Sonar, "Chapter 4: Intro to Intelligent Systems", Shailesh J Mehta School of Management, Indian Institute of Technology Bombay, Powai Mumbai-400 076.

[16] T. Mukhopadhyay, S. Vincinanza, M. J. Pietula., "Estimating the feasibility of a case-based reasoning model for software effort estimation",MIS Quarterly, 16:155, 1992.

[17] E. Mendes, N. Mosley, S. Counsell, "Web effort estimation",ICWE'07 Proceedings of the 7th international conference on Web engineering. Pages 90-104, 2007.

[18] F. Walkerden, R. Jeffery, "An empirical study of analogy-basd software effort estimation",Empirical Software Engineering, 4:135–158, 1999.

[19] R. Jeffery, M. Ruhe, I. Wieczorek, "A Comparative Study of Cost Modelling Techniques using Public Domain multi-organisational and company-specific Data", 2000.

[20] E. Mendes, N. Mosley, L. Watson, "A Comparison of Case-Based Reasoning Approaches to Web Hypermedia Project Cost estimation", The University of Auckland and MxM Technology, New Zealand ,2002.

[21] N.J. Wang, A. Mahesli, S.J. Patel, "Examining ACE Analysis Reliability Estimates using fault injection", ISCA'07. San Diego, California, USA, 2007.

[22] Empirical Software Engineering Research Group at Bournemouth University

[23] D. W. Aha, "Case Based Learning Algorithms", The Turing Institute 36 Glasgow G1 2AD Scotland, 2011.

[24] H.K.N Leung, " Estimation Maintenance Effort By Analogy", Empirical Software Engineering, 7, 157-175. Department of Computing, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong, 2002.

[25] L.E. Hem, N.M. Iversen,K. Gronhaug,"Decomposed similarity and brand extensions: The Role of Brand Personality,

Self-image Congruity, and Intercategory Relatedness", Foundation of Research in Economics and Business Administration & Norwegian School of Economics and Business Administration, 2001.

[26] I. Watson, Case-Based Reasoning 1. University of Auckland.

[27] A. R Gray, S. G MacDonell, "A Comparison of Techniques for Developing Predictive Models of Software Metrics", Information and Software Technology 39: 425-437, 1997.

[28] J. Li, G. Ruhe, A. Al-emran, M. Richter, "A flexible method for software effort estimation by analogy", Empirical Software Engineering, 12:65-106, DOI 10.1007/s10664-006-7552-4, 2007.

[29] C. W. Stanfill, "Memory-Based Reasoning Applied to English Pronunciation", AAAI-87 Proceedings,1987.

Author Profile

Nagowah Soulakshmee Devi, MSc, is a lecturer at the Computer Science and Engineering Department of the University of Mauritius. She has a BSc (Hons) in Computer Science with First Class Honours and an MSc Computer Science with specialization in Software Engineering. She has over ten years of teaching experience. Her research interests are directed towards Mobile Computing, Context-Awareness, Internet of Things, Software Engineering, Knowledge Engineering and Big Data. She has published a number of research papers in renowned international conferences and journals and has supervised projects in these areas.She has also worked on a research project entitled "A Secure Data Access Model for the Mauritian Healthcare Service" funded by the Mauritius Research Council. She has been awarded *Best Paper Award in the second place* by IEEE Software for the paper "An Automatic and Intelligent Workflow Design" for the IEEE co-sponsored ACSEAC – African Conference on Software Engineering and Applied Computing, 2011**.**

Rumjaun Shaheema S.B. has successfully graduated with a BSc(Hons) Information Systems from the University of Mauritius.She is a software engineerand began her career at Accenture TechnologiesMauritius. She works as a Siebel Expert in thedevelopment of software applications. Accenture is aleading global professional services company, providing a broad range of services and solutions in strategy, consulting, digital, technology and operations.

Gutteea Kheshan A. has successfully graduated with a BSc(Hons) Information Systems from the University of Mauritius. He is a software engineer. He started his career with ADfinance and works as technical expert in development of ADbaking software. ADfinance is a leading provider of innovative IT solutions to the micro finance industry in Africa.

Nagowah Leckraj, MSc, is a lecturer at the Computer Science and Engineering Department of the University of Mauritius. He has a BSc (Hons) in Computer Science with First Class Honours and an MSc in Computer Science with specialization in Software Engineering. He has about 10 years teaching experience. His research interests are mainly in Software Engineering, Automated Testing, Mobile Technologies, Internet of Things and Analytics. He has published more than 15 research papers in international conferences and journals, and has supervised projects in these areas. He is also a scientific committee member for a number of international conferences.