A Regression Model with Mamdani Fuzzy Inference System for Early Software Effort Estimation Based on Use Case Diagrams

Ali Bou Nassif and Luiz Fernando Capretz
Department of ECE, University of Western Ontario
London, Ontario, Canada
{abounass, lcapretz}@uwo.ca

Danny Ho
NFA Estimation Inc.
Richmond Hill, Ontario, Canada
danny@nfa-estimation.com

Abstract—Effective software effort estimation is one of the biggest challenges in software engineering. One of these challenges occurs when it is required to estimate software effort in the early stages of the software life cycle, as software requirements in this stage are usually incomplete. As Unified Modeling Language (UML) model became more prominent in software requirements and design processes, software estimators became more interested to use UML model and especially the use case diagrams to estimate software. In this paper, a novel regression model based on the use case point method is created for software effort estimation. Moreover, a Mamdani Fuzzy Inference System (FIS) approach is applied on this model to enhance the accuracy of estimation. Results show that an overall improvement of 10% can be achieved after applying the proposed approach.

Keywords—Mamdani FIS, regression model, software effort estimation, use case diagrams, Use Case Points

I. INTRODUCTION

Soft computing techniques have been widely used in many domains and especially in software engineering. These techniques include neural networks, fuzzy logic and neuro-fuzzy. Software estimation is crucial in any IT project. Project managers conduct software estimation to derive the cost and effort of the software in order to allocate resources efficiently. Historically, software estimators used to estimate software effort based on the source lines of code (SLOC) of a project. Programming languages became more sophisticated and besides, they can now handle more software functionalities in the same number of SLOC than older or procedural languages used to do. This has led estimators to look for new methods to estimate software effort. The use case point (UCP) model is one of these new methods and it has widely been used in the last decade [1]. In the UCP, software effort estimation is conducted based on the use case diagrams. The use case diagram is a UML diagram that represents the functional requirements of a system. The UCP model has been criticized because it assumes that the relationship between software size and effort is linear. In this paper, a new regression model is introduced to tackle the limitations of the UCP model and has enhanced the software effort estimation by 6%. The proposed model has crisp values in the productivity factor (see table V). A Mamdani fuzzy logic [2] approach has been used to soften these crisp values (see table VI). An additional 4% improvement of effort estimation was obtained after using the fuzzy logic approach.

A. Fuzzy Logic

The fuzzy logic concept was proposed by Zadeh in 1965 [3]. In the classical or Boolean logic, a member can either belong or not belong to a set. On the contrary, in fuzzy logic, a member can belong to a set to a certain degree. This is represented by the grade of the membership of an element within a set which falls in the interval [0,1]. The representation of a fuzzy set \( A \) can be described as follows:

\[
F_{\mu}(x) = \mu_A(x) : \mathbb{R} \rightarrow [0,1],
\]

where \( A \) is the fuzzy set, \( x \) is an element in the set \( A \), \( \mu_A \) is the degree of membership of \( x \) and \( \mathbb{R} \) is the universe of discourse. The higher the grade of membership \( x \) has in \( A \), the more \( x \) belongs to \( A \). Fuzzy logic can handle problems incurred by incomplete and imprecise data. It is used to map an input in the universe of discourse to an output. This is represented by membership functions. A membership function is a curve that defines the mapping between input and output. Membership functions include Gaussian Bell, Singleton, Trapezoidal and Triangular. The knowledge base is represented in fuzzy logic by if-then rules. The fuzzy logic system consists of three parts. These include Fuzzification, Fuzzy Rule Application and Defuzzification. In the first stage (Fuzzification), membership functions are applied to inputs. In the second stage, the Fuzzy Rule Application is applied to build associations and inferences among members in different groups. The final step in the fuzzy system is Defuzzification. This is used to defuzzify the system in order to provide an output and to make a reasonable decision. In this paper, Mamdani fuzzy logic is applied to calibrate the values of the productivity factor depicted in Table V.

B. Use Case Points

The use case point model was first described by Gustav Karner in 1993 [4]. This model is used for software cost estimation based on the use case diagrams. First, the software size is calculated according to the number of the actors and use cases in a use case diagram multiplied by their complexity weights. The complexity weights of use cases and actors are presented in tables I and II respectively.
The complexity of a use case is determined by the number of its transactions as shown in the use case description or the sequence diagram of each use case. Let $\text{TC}$ be the technical complexity of the project while $\text{EF}$ contributes to the team requirements of the software. $\text{TF}$ contributes to the technical factors (TF) and the environmental factors (EF). $\text{TF}$ and $\text{EF}$ represent the non-functional factors (UNS) and environmental factors (ENS) respectively. The technical and environmental points (UCP) is calculated. $\text{UCP}$ is achieved by multiplying the Unadjusted Use Case Weight (UUCW) and Unadjusted Actor Weight (UAW). $\text{UUCW}$ can be defined as follows:

$$\text{UUCW} = \sum_{i=1}^{3} n_i \cdot W_i,$$

where $n_i$ is the number of items of variety $i$ of the use cases and $W_i$ is the complexity weight of the corresponding use case. Similarly, $\text{UAW}$ is represented as follows:

$$\text{UAW} = \sum_{j=1}^{3} m_j \cdot C_j,$$

where $m_j$ is the number of items of variety $j$ of the actors and $C_j$ is the complexity weight of the corresponding actor. Consequently, $\text{UUCP}$ can be defined as follows:

$$\text{UUCP} = \text{UUCW} + \text{UAW}.$$

After calculating the $\text{UUCP}$, the Adjusted Use Case Points (UCP) is calculated. $\text{UCP}$ is achieved by multiplying $\text{UUCP}$ by the Technical Factors (TF) and the Environmental Factors (EF). TF and EF represent the non-functional requirements of the software. TF contributes to the complexity of the project while EF contributes to the team efficiency and productivity. The technical and environmental factors are depicted in tables III and IV respectively. The technical factor is detailed as follows:

$$\text{TF} = 0.6 + 0.01 \sum_{i=1}^{13} T_i \cdot W_i,$$

where $T_i$ is a factor that takes values between 0 and 5. The value “0” indicates that the factor is unrelated while the value “5” indicates that the factor is indispensable. The value “3” specifies that the technical factor is not very important, nor irrelevant (average). For instance, if all of the factors have a value of “3”, the technical factor (TF) will be 1. $W_i$ represents the weight of technical factors (Table III.)

The environmental factor (EF) can be described as follows:

$$\text{EF} = 1.4 - 0.03 \sum_{i=1}^{13} E_i \cdot W_i.$$
\[
MMRE = \frac{1}{N} \sum_{i=1}^{N} MRE_i.
\] (10)

On the other hand, PRED(x) is the percentage of projects for which the estimate falls within x% of the actual value. For example, PRED(25) gives the percentage of software projects that were estimated with MMRE less than 0.25. The estimation accuracy is proportional to PRED(x) and inversely proportional to MMRE.

II. PROBLEM DEFINITION

The original use case point model proposed a method to estimate software based on the size in use case points (see Equation 8). However, this method has some limitations, which includes:

1. The method proposed by the original use case point model assumes that the relationship between the software effort and size is linear. However, this assumption is not true in real life. For instance, if the effort required in building a software project of size 200 UCP is 4,000 person-hours, the effort needed to build the same project type of size 400 UCP would be more than 8,000 person-hours. This is because the larger the project is, the larger the team required to build this project [6]. When the number of the team members increases, the number of the communication paths among this team will dramatically increase, and consequently, this requires more effort for the team communication and project management. The number of communication paths is calculated as [7]:

\[
\text{Communication Paths} = \frac{N(N-1)}{2}
\] (11)

where N is the number of the team members.

2. The original use case point model assumes that the adjusted use case point (UCP) size can increase the unadjusted use case point (UUCP) size by 30% (see Equation 7). The UCP incorporates the non-functional requirements which are represented by the technical and environmental factors of the software. However, IBM states that the non-functional requirements of a software project might cost more than 50% of the total effort [8]. This means that the non-functional requirements may virtually increase the unadjusted size by 100%.

The proposed regression model tackles these shortcomings. However, the values of the productivity factor in the proposed model (see Table V) are crisp. Mamdani fuzzy logic has been used to calibrate the productivity factor values.

III. RELATED WORK

Some work has been done to estimate software effort using soft computing techniques [9], [10], [11], [12], [13], [14] and [15]. Other work was done to enhance the effort estimation of the use case point model. Other research was conducted to build regression models based on Function Points.

Sparks et al. [16] reported that the effort to develop one UCP should be between 15 and 30 person-hours. However, the authors did not detail how this range should be applied.

Schneider et al. [17] mentioned that although the environmental factors (EF) are included in the adjusted use case points (UCP), EF should also be considered while calculating the effort. The authors suggested counting the number of factor ratings of E1-E6 in Table IV (Environmental Factors) that are below 3 and the number of factor ratings of E7-E8 that are above 3. If the total is 1 or 2, then 20 person-hours per UCP should be used. If the total is 3 or 4, then 28 person-hours per UCP should be used. If the total is 5 or more, then the project team should be reconstructed so that the numbers fall at least below 5. A value of 5 indicates that the project is at significant risk of failure with this team. The main limitation of this method is that the effort required to develop one UCP is either 20 or 28 person-hours.

Jiang et al. [18] and Xia et al. [19] built regression models based on function points using ISBSG data. The main concern of these models is that they ignore the influence of the non-functional requirements on estimation.

None of the existing work deal with creating regression models based on the UCP model.

IV. RESEARCH METHODOLOGY AND IMPLEMENTATION

In this section, two main approaches are described. First, a regression model is established to calculate software effort estimation. Secondly, a fuzzy logic approach is used to attune the values of the productivity factor in the proposed model. The proposed model is an extension to the model proposed in [20].

A. Regression model

In general, the main attributes that affect software effort are software size, project complexity and team productivity. Effort is proportional to project size and project complexity and inversely proportional to the productivity of the team. Software effort can be represented as follows [21]:

\[
\text{Effort} = \frac{\text{Complexity}}{\text{Productivity}} \times \text{Size}.
\] (12)

where Complexity is the complexity factor of a project and Productivity is the productivity factor of the team that is developing the project. As mentioned in Section II, the relationship between software effort and size is non-linear. The first step of the proposed model is to discover the non-linear relationship between software size and software effort. For this purpose, regression analysis was applied on several projects that had similar project complexity and team productivity. Regression analysis assumes that data should be normally distributed [22]. The algorithm used to generate linear regression is depicted in Figure 1.
After testing the original data, data were not normally distributed. However, data were normally distributed after normalization. A linear regression formula was generated between Log Effort and Log Size. With the help of the statistics software Minitab version 16, a regression equation was generated as follows:

\[
\text{Effort} = 8.16 \times \text{Size}^{1.17}. \tag{13}
\]

where \( \text{Size} \) is the software size in UCP and \( \text{Effort} \) is the software effort in person-hours. For instance, Equation 13 shows the non-linear relationship between Effort and Size and ignores the Complexity and Productivity factors. The main equation of software effort is expressed in Equation 14.

To measure the accuracy of the regression equation (see Equation 13), we measured the value of the coefficient of determination \( R^2 \). \( R^2 \) is the percentage of variation in Effort explained by the variable Size. The value \( R^2 \) reported for the regression model in Equation 13 is 0.972. Approximately 97% of the variation in Effort can be explained by the variable Size. This shows a strong relation between Size and Effort.

In the original use case point model, effort is a function of UCP and \( \text{Effort} \) is the software effort in person-hours. For instance, Equation 13 shows the non-linear relationship between Effort and Size and ignores the Complexity and Productivity factors. The main equation of software effort is expressed in Equation 14.

To measure the accuracy of the regression equation (see Equation 13), we measured the value of the coefficient of determination \( R^2 \). \( R^2 \) is the percentage of variation in Effort explained by the variable Size. The value \( R^2 \) reported for the regression model in Equation 13 is 0.972. Approximately 97% of the variation in Effort can be explained by the variable Size. This shows a strong relation between Size and Effort.

In the original use case point model, effort is a function of UCP and \( \text{Effort} \) is the software effort in person-hours. For instance, Equation 13 shows the non-linear relationship between Effort and Size and ignores the Complexity and Productivity factors. The main equation of software effort is expressed in Equation 14.

To measure the accuracy of the regression equation (see Equation 13), we measured the value of the coefficient of determination \( R^2 \). \( R^2 \) is the percentage of variation in Effort explained by the variable Size. The value \( R^2 \) reported for the regression model in Equation 13 is 0.972. Approximately 97% of the variation in Effort can be explained by the variable Size. This shows a strong relation between Size and Effort.

In the original use case point model, effort is a function of UCP and \( \text{Effort} \) is the software effort in person-hours. For instance, Equation 13 shows the non-linear relationship between Effort and Size and ignores the Complexity and Productivity factors. The main equation of software effort is expressed in Equation 14.

To measure the accuracy of the regression equation (see Equation 13), we measured the value of the coefficient of determination \( R^2 \). \( R^2 \) is the percentage of variation in Effort explained by the variable Size. The value \( R^2 \) reported for the regression model in Equation 13 is 0.972. Approximately 97% of the variation in Effort can be explained by the variable Size. This shows a strong relation between Size and Effort.

In the original use case point model, effort is a function of UCP and \( \text{Effort} \) is the software effort in person-hours. For instance, Equation 13 shows the non-linear relationship between Effort and Size and ignores the Complexity and Productivity factors. The main equation of software effort is expressed in Equation 14.

To measure the accuracy of the regression equation (see Equation 13), we measured the value of the coefficient of determination \( R^2 \). \( R^2 \) is the percentage of variation in Effort explained by the variable Size. The value \( R^2 \) reported for the regression model in Equation 13 is 0.972. Approximately 97% of the variation in Effort can be explained by the variable Size. This shows a strong relation between Size and Effort.

In the original use case point model, effort is a function of UCP and \( \text{Effort} \) is the software effort in person-hours. For instance, Equation 13 shows the non-linear relationship between Effort and Size and ignores the Complexity and Productivity factors. The main equation of software effort is expressed in Equation 14.

To measure the accuracy of the regression equation (see Equation 13), we measured the value of the coefficient of determination \( R^2 \). \( R^2 \) is the percentage of variation in Effort explained by the variable Size. The value \( R^2 \) reported for the regression model in Equation 13 is 0.972. Approximately 97% of the variation in Effort can be explained by the variable Size. This shows a strong relation between Size and Effort.

In the original use case point model, effort is a function of UCP and \( \text{Effort} \) is the software effort in person-hours. For instance, Equation 13 shows the non-linear relationship between Effort and Size and ignores the Complexity and Productivity factors. The main equation of software effort is expressed in Equation 14.

To measure the accuracy of the regression equation (see Equation 13), we measured the value of the coefficient of determination \( R^2 \). \( R^2 \) is the percentage of variation in Effort explained by the variable Size. The value \( R^2 \) reported for the regression model in Equation 13 is 0.972. Approximately 97% of the variation in Effort can be explained by the variable Size. This shows a strong relation between Size and Effort.

In the original use case point model, effort is a function of UCP and \( \text{Effort} \) is the software effort in person-hours. For instance, Equation 13 shows the non-linear relationship between Effort and Size and ignores the Complexity and Productivity factors. The main equation of software effort is expressed in Equation 14.

To measure the accuracy of the regression equation (see Equation 13), we measured the value of the coefficient of determination \( R^2 \). \( R^2 \) is the percentage of variation in Effort explained by the variable Size. The value \( R^2 \) reported for the regression model in Equation 13 is 0.972. Approximately 97% of the variation in Effort can be explained by the variable Size. This shows a strong relation between Size and Effort.

In the original use case point model, effort is a function of UCP and \( \text{Effort} \) is the software effort in person-hours. For instance, Equation 13 shows the non-linear relationship between Effort and Size and ignores the Complexity and Productivity factors. The main equation of software effort is expressed in Equation 14.

To measure the accuracy of the regression equation (see Equation 13), we measured the value of the coefficient of determination \( R^2 \). \( R^2 \) is the percentage of variation in Effort explained by the variable Size. The value \( R^2 \) reported for the regression model in Equation 13 is 0.972. Approximately 97% of the variation in Effort can be explained by the variable Size. This shows a strong relation between Size and Effort.

In the original use case point model, effort is a function of UCP and \( \text{Effort} \) is the software effort in person-hours. For instance, Equation 13 shows the non-linear relationship between Effort and Size and ignores the Complexity and Productivity factors. The main equation of software effort is expressed in Equation 14.

To measure the accuracy of the regression equation (see Equation 13), we measured the value of the coefficient of determination \( R^2 \). \( R^2 \) is the percentage of variation in Effort explained by the variable Size. The value \( R^2 \) reported for the regression model in Equation 13 is 0.972. Approximately 97% of the variation in Effort can be explained by the variable Size. This shows a strong relation between Size and Effort.

In the original use case point model, effort is a function of UCP and \( \text{Effort} \) is the software effort in person-hours. For instance, Equation 13 shows the non-linear relationship between Effort and Size and ignores the Complexity and Productivity factors. The main equation of software effort is expressed in Equation 14.

To measure the accuracy of the regression equation (see Equation 13), we measured the value of the coefficient of determination \( R^2 \). \( R^2 \) is the percentage of variation in Effort explained by the variable Size. The value \( R^2 \) reported for the regression model in Equation 13 is 0.972. Approximately 97% of the variation in Effort can be explained by the variable Size. This shows a strong relation between Size and Effort.

In the original use case point model, effort is a function of UCP and \( \text{Effort} \) is the software effort in person-hours. For instance, Equation 13 shows the non-linear relationship between Effort and Size and ignores the Complexity and Productivity factors. The main equation of software effort is expressed in Equation 14.

To measure the accuracy of the regression equation (see Equation 13), we measured the value of the coefficient of determination \( R^2 \). \( R^2 \) is the percentage of variation in Effort explained by the variable Size. The value \( R^2 \) reported for the regression model in Equation 13 is 0.972. Approximately 97% of the variation in Effort can be explained by the variable Size. This shows a strong relation between Size and Effort.

In the original use case point model, effort is a function of UCP and \( \text{Effort} \) is the software effort in person-hours. For instance, Equation 13 shows the non-linear relationship between Effort and Size and ignores the Complexity and Productivity factors. The main equation of software effort is expressed in Equation 14.

To measure the accuracy of the regression equation (see Equation 13), we measured the value of the coefficient of determination \( R^2 \). \( R^2 \) is the percentage of variation in Effort explained by the variable Size. The value \( R^2 \) reported for the regression model in Equation 13 is 0.972. Approximately 97% of the variation in Effort can be explained by the variable Size. This shows a strong relation between Size and Effort.

In the original use case point model, effort is a function of UCP and \( \text{Effort} \) is the software effort in person-hours. For instance, Equation 13 shows the non-linear relationship between Effort and Size and ignores the Complexity and Productivity factors. The main equation of software effort is expressed in Equation 14.

To measure the accuracy of the regression equation (see Equation 13), we measured the value of the coefficient of determination \( R^2 \). \( R^2 \) is the percentage of variation in Effort explained by the variable Size. The value \( R^2 \) reported for the regression model in Equation 13 is 0.972. Approximately 97% of the variation in Effort can be explained by the variable Size. This shows a strong relation between Size and Effort.
The main drawback of the productivity factor shown in Table V is that the values are crisp and there is no gradation in the productivity factor values as the value of \((\sum_{i=1}^{n} E_i \times W_i)\) increases. For instance, if the value of \((\sum_{i=1}^{n} E_i \times W_i)\) is 10, the productivity factor is 0.7, however, if the value of \((\sum_{i=1}^{n} E_i \times W_i)\) is 11, the value of the productivity factor is 1. To tackle this drawback, a fuzzy logic approach has been used.

B. Fuzzy Logic Approach

A fuzzy logic approach is applied on the proposed regression model to adjust the values of the productivity factor. In the proposed approach, the fuzzy system type is Mamdani [2], the input membership of the fuzzy logic system used is Trapezoidal where the output membership is Triangular. Figures 2 and 3 show the input and the output memberships respectively.

There are four fuzzy rules in the proposed approach. These include:

1- If \((\sum_{i=1}^{n} E_i \times W_i)\) is less than 0, then productivity factor = 0.4.
2- If \((\sum_{i=1}^{n} E_i \times W_i)\) is between 0 and 10, then productivity factor = 0.7.
3- If \((\sum_{i=1}^{n} E_i \times W_i)\) is between 10 and 20, then productivity factor = 1.
4- If \((\sum_{i=1}^{n} E_i \times W_i)\) is greater than 20, then productivity factor = 1.3.

The centroid method is used for Defuzzification and calculates the center of gravity of a surface.

After applying the fuzzy logic approach, the productivity factor has a specific value for each value of \((\sum_{i=1}^{n} E_i \times W_i)\). Table VI shows some samples of the new values of the productivity factors. The labels IN, PO and PN correspond to \((\sum_{i=1}^{n} E_i \times W_i)\), old productivity factor and new productivity factor respectively.

As seen in Table VI, the values of the new productivity factor (PN) are not as crisp as the values of the old productivity factor (PO). This leads to better estimation values. For instance, a complete list of the productivity factor values can be obtained using the proposed fuzzy logic system. An example is shown in Figure 4.

V. EVALUATION

The evaluation of the proposed model was performed using 24 projects that were not included among the projects used in the regression analysis. The actual efforts of these projects vary between 1,000 and 5,000 person-hours. Software estimation was conducted using the original use case model (Karner’s model), Schneider’s model and the proposed model.

Table VII shows the evaluation results. The columns Kar, Sch, Pro_B and Pro_A represent Karner, Schneider, the proposed model before applying fuzzy logic and the proposed model after applying the fuzzy logic approach respectively. MMRE, PRED (25), PRED (35), PRED (50), PRED (75) and PRED (100) were used as evaluation criteria.

In Table VIII, the results show that the proposed model before applying the fuzzy logic approach improves the MMRE of Karner’s model by 6% and Schneider’s method by 2%. After applying the Mamdani fuzzy logic approach, 10% improvement over Karner’s model was obtained and 6% over Schneider’s model. This means that an additional 4% improvement was obtained after applying the fuzzy logic approach. Moreover, the results also show substantial improvements in PRED(x).
TABLE VII. COMPARISON BETWEEN PROPOSED AND OLD MODELS

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Kar (%)</th>
<th>Sch (%)</th>
<th>Pro_B (%)</th>
<th>Pro_A (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MMRE</td>
<td>34</td>
<td>30</td>
<td>28</td>
<td>24</td>
</tr>
<tr>
<td>PRED (25)</td>
<td>37.5</td>
<td>62.5</td>
<td>54</td>
<td>54</td>
</tr>
<tr>
<td>PRED (35)</td>
<td>50</td>
<td>66.6</td>
<td>75</td>
<td>75</td>
</tr>
<tr>
<td>PRED (50)</td>
<td>87.5</td>
<td>83.3</td>
<td>91.6</td>
<td>95.8</td>
</tr>
<tr>
<td>PRED (75)</td>
<td>95.8</td>
<td>91.7</td>
<td>95.8</td>
<td>95.8</td>
</tr>
<tr>
<td>PRED (100)</td>
<td>100</td>
<td>95.8</td>
<td>95.8</td>
<td>100</td>
</tr>
</tbody>
</table>

TABLE VIII. IMPROVEMENT OF THE PROPOSED APPROACH

<table>
<thead>
<tr>
<th>Criteria</th>
<th>IProBK (%)</th>
<th>IProBS (%)</th>
<th>IProAK (%)</th>
<th>IProAS (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MMRE</td>
<td>6</td>
<td>2</td>
<td>10</td>
<td>6</td>
</tr>
<tr>
<td>PRED (25)</td>
<td>16.5</td>
<td>-8.5</td>
<td>16.5</td>
<td>-8.5</td>
</tr>
<tr>
<td>PRED (35)</td>
<td>25</td>
<td>8.4</td>
<td>25</td>
<td>8.4</td>
</tr>
<tr>
<td>PRED (50)</td>
<td>4.1</td>
<td>8.3</td>
<td>8.3</td>
<td>12.5</td>
</tr>
<tr>
<td>PRED (75)</td>
<td>0</td>
<td>4.2</td>
<td>0</td>
<td>4.2</td>
</tr>
<tr>
<td>PRED (100)</td>
<td>-4.2</td>
<td>0</td>
<td>0</td>
<td>4.2</td>
</tr>
</tbody>
</table>

For instance, the columns IProBK, IProBS, IProAK and IProAS represent the improvement of the proposed model before fuzzy logic over Karner’s model, improvement of the proposed model before fuzzy logic over Schneider’s model, improvement of the proposed model after fuzzy logic over Karner’s model, and improvement of the proposed model after fuzzy logic over Schneider’s model respectively.

VI. CONCLUSION AND FUTURE WORK

The use case point model is used by many practitioners to conduct software estimation in the early stages of the software life cycle. The main advantage of this model is that it is simple and can easily be automated. However, the original model ignores the non-linear relationship between software effort and size, and lacks accuracy in the effort estimation because it does not take into consideration the influence of team productivity while estimating software effort. A new regression model has been proposed to tackle the drawbacks of the original model. In order to calibrate the values of the productivity factor in the proposed model, a fuzzy logic approach has been used. Results show that by applying the Mamdani FIS approach, an overall improvement of 10% can be obtained in software effort estimation.

As a comparison between applying Mamdani FIS and Sugeno FIS [25] on the proposed regression model, results prove that Sugeno FIS can surpass Mamdani FIS by 1% when using MMRE as an evaluation criterion. However, Mamdani FIS yields slightly better results when using PRED. We conclude that there is no major difference when comparing Mamdani FIS with Sugeno FIS.

Future work will focus on two main concerns. First, the proposed model should be tested with projects of larger sizes (greater than 5,000 person-hours) when data are available. Secondly, the productivity factors proposed in Table V should be calibrated using neuro-fuzzy as neuro-fuzzy gives better results than fuzzy logic.

REFERENCES