Implementing Effort Estimation Tool as a Cloud Enabled Service

E. Sudheer Kumar
Assistant Professor, Dept of CSE,
Sree Vidyanikethan Engineering College,
A.Rangampeta, Tirupati, A.P.

V. Jyothsna
Assistant Professor, Dept of IT,
Sree Vidyanikethan Engineering College,
A.Rangampeta, Tirupati, A.P.

ABSTRACT
Software effort estimation is one of the most critical and complex, but an inevitable activity that takes place during the early stages of SDLC. Software size estimate is one of the most popular inputs for software effort prediction models. Providing a good size predictors and the results were encouraging.

CHECK POINT, M
SME
challenged
dynamics
expertise
and
too.

Overestimation
detrimental
effects
most
likely
realistic
understanding
insight
basic
predictor.

Software engineering cost (and schedule) models and estimation techniques are used for a number of purposes. These include: Budgeting, Tradeoff and risk analysis, Project planning and control, Software improvement investment analysis.

Fig 1: Types of Software Estimation Techniques

In the rest of the paper, we use the term “size proxy” for effort prediction rather than the term “size metric” to emphasize that size is measured in a way merely to predict effort and not something else, e.g., understandability. It is essential to note here that the term “proxy” is used to reflect that the “size” is being used as a proxy for the ‘effort’ and not the other way around.

1. INTRODUCTION
Software effort prediction are the basis for project bidding, budgeting and planning that takes place during the early stages of development life cycle. Delivering the software on time and within budget is a critical concern for many software organizations. Underestimating software cost can have unfavorable effects on the quality of the delivered software and may affect the company’s business reputation and competitiveness. Overestimation of software cost is detrimental too. It can result in missed opportunities to fund in other projects and loss of project tenders [1].

Summarizing several classes of software cost estimation models and techniques as shown in Fig.1 parametric models, expertise-based techniques, learning-oriented techniques, dynamics-based models, regression-based models, and composite-Bayesian techniques for integrating expertise-based and regression-based models. Experience to date indicates that neural-net and dynamics-based techniques are less mature than the other classes of techniques, but that all classes of techniques are challenged by the rapid pace of change in software technology.

The primary conclusion is that no single technique is best for all situations, and that a careful comparison of the results of several approaches is most likely to produce realistic estimates.

Keywords
The development of size estimates the mean effort measured computed by the tool. The proposed tool with the help of SDMetrics tool generates a conceptual framework for the development of size estimation. A major reason for using UML conceptual models is that UML is so popular and widely accepted in software industry. With UML it is possible to model an entire software project. The conceptual framework is a way to provide a framework to model the software project. The framework allows to model the software project in a more structured way. The conceptual framework is used to model the software project in a more structured way.

2.1 Existing System

In the existing system, the size estimation is done using the traditional metrics. The traditional metrics are based on the size of the source code. The size of the source code is measured in terms of lines of code. The lines of code are measured using different metrics such as Function Point (FP), Object Point (OP), Feature Point (FP), and so on. The size estimation is done using the traditional metrics by counting the number of lines of code. However, the traditional metrics do not provide an accurate estimation of the size of the software project. The size estimation based on the traditional metrics is not reliable. The size estimation based on the traditional metrics is not reliable because the traditional metrics do not consider the complexity of the software project. The traditional metrics do not consider the complexity of the software project. The traditional metrics do not consider the complexity of the software project.

2.2 Proposed System

In the proposed system, the size estimation is done using the conceptual framework. The conceptual framework is used to model the software project in a more structured way. The conceptual framework is used to model the software project in a more structured way. The conceptual framework is used to model the software project in a more structured way.

Fig. 2.- System Overview

The conceptual framework is used to model the software project in a more structured way. The conceptual framework is used to model the software project in a more structured way. The conceptual framework is used to model the software project in a more structured way. The conceptual framework is used to model the software project in a more structured way. The conceptual framework is used to model the software project in a more structured way.

2.3 Related Work

The related work on software estimation is presented in Chapter 3. The related work on software estimation is presented in Chapter 3. The related work on software estimation is presented in Chapter 3. The related work on software estimation is presented in Chapter 3. The related work on software estimation is presented in Chapter 3.
variance) represents the standard deviation that accounts for the uncertainty in prediction. It is important to mention here that the higher the standard deviation of estimate, the lower the confidence in the estimate; i.e., the higher the uncertainty in the computed estimate.

### 3. ARCHITECTURE DESIGN

The main purpose of this design is to develop an Effort Estimation Tool which generates a probabilistic size proxy to predict effort required to develop a project in the early stages of software development life cycle. This tool can be used in real time scenarios for prediction activities, when they have conceptual diagrams for a future projects. The tool architecture can be used to convert artifact in to metrics extraction, training the size proxy, predict effort and produce a probability density function value. With this architecture the uncertainty issue will be solved satisfactorily. The extraction process inside the architecture can be done by using SDMetrics tool, selecting the suitable predictors can be done with the help of Pearson correlation, providing training to develop the size proxy by using linear regression technique and also provide results for different confidence intervals.

---

**Fig 3:** A Conceptual Framework for Probabilistic Size Proxy

**Fig 4:** Proposed Architecture for Probabilistic Size Proxy
Step1: Metric value extraction: This component extracts the metric values from the current project for which the effort needs to be estimated. This component takes Project’s conceptual models i.e. the UML artifacts along with any other project specific information like the technical factors affecting the project.

Step2: Selection of metric and their values from past projects: This component selects the size proxy constituents i.e. the predictors for effort, based on the metrics whose values are available for the current project whose effort needs to be estimated and also based on the metrics for which we have information available for sufficient number of past projects in the database.

Step3: Training of the size proxy: After selecting the predictors for effort estimation, the predictors along with their values are provided to the training component. The effort values for past completed projects are also provided to the training component from the database. The training component can use one of the different training techniques like regression, neural networks, genetic programming to train the function for mean (μ) and the standard deviation (σ) of the size proxy.

Step4: Effort estimation: The trained functions for mean (μ) and standard deviation (σ) are provided to the effort estimation component. This component obtains the metric values for the current project from the component one and uses them in the functions for mean (μ) and standard deviation (σ) to provide the effort estimate as a PDF.

3.1 Metrics Table Data

Whenever user input xmi file through gui, the file will forwarded to SDMetrics tool. The tool analyzes the file; calculate metrics and then the results will be generated in the specified format as shown in below Table 3.1.

The output file from SDMetrics tool has individual classes of a project and its metrics values. Now we have to calculate overall values for each and every predictor and store it in to another table as shown in below Table 3.2.

Past projects data has been generated using simulation method. Each and every dependent and independent variables value has been generated and with the help of this data estimated μ can be calculated by using linear regression technique. In general, multiple regression procedures will estimate a linear equation of the form as shown in equation (1):

\[ Y = a + b_1 \times X_1 + b_2 \times X_2 + \ldots + b_p \times X_p \]  

(1)

Table: 3.1 SDMetrics Output File Format

<table>
<thead>
<tr>
<th>Name</th>
<th>Num Attr</th>
<th>Num Ops</th>
<th>NumPub Ops</th>
<th>.........</th>
<th>MsgSelf</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class 1</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>Class 2</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>Class n</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>4</td>
<td></td>
</tr>
</tbody>
</table>

3.2 Estimated Mean Data

Calculating mean equation with the help of linear regression, and the dataset structure is tabulated in Table 3.2 and formula is:

Mean Estimated Effort (\( \mu_{\text{effort}} \)) = \( b_0 + \sum_{i=1}^{n} (b_i \times A_i) \)  

(2)

Where,

- \( m \) = number of predictors (independent variables)
- \( b_i \) = Beta coefficient for variable ‘i’.
- \( A_i \) = Independent/predictor variable ‘i’

3.3 Standard Deviation Data

Each perturbed set/cluster number consists of number of projects as one set and also with their corresponding values. The data represented here has different independent variables and standard deviation of error estimate variable which is used for calculating the linear regression equation of standard deviation error along with mean error as of like in formulae (3).

Mean error formula

\[ \mu^i_{\text{error}} = \frac{1}{N} \sum_{j=1}^{N} (\mu^j_{\text{error}}) \]  

(3)

Where,

- \( \mu^i \) = mean error estimate for perturbed set ‘i’.
- \( N \) = Number of projects in each perturbed set.
- \( \mu_{\text{error}} \) = Estimated effort for project ‘i’ in perturbed set ‘i’.
- \( \sigma (i) \) = Standard deviation of errors (μ) for the ‘N’ projects in perturbed set ‘i’.

3.4 Standard Deviation of Error Estimate

\[ \sigma (i) = \sqrt{\frac{1}{N} \sum_{j=1}^{N} (\mu^j_{\text{error}} - (E^j - \mu^j_{\text{effort}}))^2} \]  

(4)

Where

- \( r (i) \) = standard deviation of error in estimation for perturbation set ‘i’
- \( N \) = total number of projects in each perturbed set
- \( \mu_{\text{effort}} \) = mean error in estimation for perturbation set ‘i’
- \( E_{\text{effort}} \) = effort, for perturbed project ‘j’ in perturbed set ‘i’
- \( \mu_{\text{effort}} \) = estimated mean effort for project ‘j’ in perturbed set ‘i’

3.5 Cloud Installation and Application Hosting

The product features automated deployment of cloud stack and provided a convenient environment through portal for infrastructure request, application hosting, view usage and billing information and administrator users to manage their entire cloud environment and security administrators to monitor the security violations.

Insert the Meghdoot DVD into the drive, restart your computer and boot from DVD by editing the BIOS setup. The screen appears with three options:

- Start Meghdoot (BOSS CLOUD) Live
- Install Meghdoot (BOSS CLOUD)-Graphical
- Install Meghdoot (BOSS CLOUD)-Text Mode

You can proceed with the default installation by clicking “Install Meghdoot (BOSS CLOUD) - Graphical” or “Install Meghdoot (BOSS CLOUD)-Text Mode”.

Table: 3.2 Input dataset structure for training mean effort

<table>
<thead>
<tr>
<th>P. No</th>
<th>Dependent Variable (Effort)</th>
<th>Independent Variable (A1)</th>
<th>IV (A2)</th>
<th>.........</th>
<th>IV (Am)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>240</td>
<td>20</td>
<td>18</td>
<td>14</td>
<td>15</td>
</tr>
<tr>
<td>2</td>
<td>350</td>
<td>30</td>
<td>14</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>N</td>
<td>....</td>
<td>....</td>
<td>....</td>
<td>....</td>
<td>....</td>
</tr>
</tbody>
</table>
4. EXPERIMENTAL SETUP

4.1 SDMetrics Tool

The Fig 4.1 is the first and foremost step which is going to happen with the help of user. The user will connect to tool through web graphical user interface and give XMI file of a class diagram for a project. The SDMetrics tool will extract the metrics form that class diagram and stored in the form of table format as shown above fig 4.1.

4.2 Correlation Table

Pearson Correlation Coefficient is used, inorder to find out (linear) relationship between the two variables x and y. Based on the past data available in database, it retrieves data from database and finding pearson correlation between effort variable and all remaining variables. Table 4.1 shows correlation table:

Table: 4.1 Correlation table based on Past Data

<table>
<thead>
<tr>
<th></th>
<th>NC</th>
<th>NA</th>
<th>NP</th>
<th>IFImpl</th>
<th>NumDesc</th>
<th>NumAnc</th>
<th>DIT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Effort</td>
<td>0.98</td>
<td>0.99</td>
<td>0.99</td>
<td>0.98</td>
<td>0.28</td>
<td>0.97</td>
<td>0.98</td>
</tr>
</tbody>
</table>

Effort (Man-Hours)

4.3 Size Metric

The size metric is one of the main inputs used to find effort estimation. There it needs to form a metric, so we are considering multiple linear regression as our training technique. It is a combination of mean, error and standard deviation form. Mean will calculate based on the past data, where as it will consider effort variable independent variable and all other remaining variables as dependent variables, so that we will get an intercept value and coefficients value to form a equation. Standard deviation will also calculate based on the past data, where as it will consider standard deviation error.

First of all we need to calculate error value by considering difference between actual effort values and estimated mean effort values of past projects. We need to perform standard deviation error with the help of error value for all past projects. Consider standard deviation error value as independent variable and remaining variables as dependent variable to get intercept value and coefficients values. With the help of mean and standard deviation error values, the size proxy equation is formed.

\[ E_{\text{estimated}} = N [(311.39 + 0.584 * \text{NC} + 0.317 * \text{NA} + 0.054 * \text{NPM} + 0.015 * \text{IFImpl} + 0.070 * \text{NumDesc} + -0.02 * \text{NumAnc} + 0.052 * \text{DIT})], (-13.08 + 0.259 * \text{NC} + -0.53 * \text{NA} + 0.46 * \text{NPM} + -0.007 * \text{IFImpl} + 0.071 * \text{NumDesc} + 0.49 * \text{NumAnc} + -0.48 * \text{DIT})] \]

4.4 PDF Value

Here we need to substitute the variables value from the user table database and it shows the values of mean and standard deviation error. Now we need to calculate Gaussian probability density function value by adding +/- of 1.96*SDE value to mean, so that we can get probabilistic value instead of certain value.

The Gaussian Probability Density Function value is:

\[ E_{\text{estimated}} = \text{N} [403.35, 10.99] \]

Effort Estimate for the confidence interval of 95% is:

\[ E [381.80, 424.90] \]

5. CONCLUSION AND FUTURE WORK

A tool based approach for effort estimation was proposed, which can be used early in software development process. It is flexible to the amount of information available during estimation and does account for uncertainty by providing estimate as a Gaussian PDF. The Architecture was validated by creating instantiating size proxies consisting of different number of predictors and validating these size proxies using two different data sets. User has to request for Effort Estimation Tool in private cloud, then cloud will check all credentials of user and then it allows user to access the tool. Now user has to send his XMI file through the GUI to tool. The tool is hosted by a vendor or service provider in the cloud and made available to customers over a network (typically the internet) having benefits like high adoption, lower initial costs, painless upgrades and seamless integration, so that user can access from anywhere with the help of browser and does not require any additional resources.
5.1 Limitations and Future Work

Following are some of the major limitations of our work along with proposed future works:

- A major limitation of our work is the shortage of data. Shortage of data is a typical problem faced by research community in software engineering. Dataset one had data on 20 projects with simulation / trial and error. Thus validating the proxy using more data and preferable from industry projects is a definite future work.
- The architecture may also be extended by adding steps to handle imprecision especially when using attributes having categorical values or inputs from expert. Fuzzy logic is a good choice to model imprecision in such cases.
- Another possible future work is to use the intermediate size proxy by providing the non-singleton size measure to effort estimation models such as the one proposed by zeeshan.

We used regression analysis to train our size proxy. Other training algorithms and techniques like ANN can also be used and the results can be compared.

6. REFERENCES


