Evaluation of Cost Estimation Metrics: Towards a Unified Terminology

Izzat M. Alsmadi and Maryam S. Nuser
Department of Computer Information Systems, Yarmouk University, Jordan

Cost overrun of software projects is major cause of their failures. In order to facilitate accurate software cost estimation, there are several metrics, tools and datasets. In this paper, we evaluate and compare different metrics and datasets in terms of similarities and differences of involved software attributes. These metrics forecast project cost estimations based on different software attributes. Some of these metrics are public and standard while others are only employed in a particular metric tool/dataset.

Sixteen public cost estimation datasets are collected and analyzed. Different perspectives are used to compare and classify those datasets. Tools for feature selection and classification are used to find the most important attributes in cost estimation datasets toward the goal of effort prediction. In order to have better estimation, it is needed to correlate cost estimation from different resources, which requires a unified standard for software cost estimation metric tools and datasets. It is pertinent that a common cost estimation model may not work for each project due to diverse project size, application areas etc. We suggest having a standardized terminology of project attributes used for cost estimation. This would improve cost estimation as multiple metrics could be applied on a project without much additional effort.

Keywords: cost estimation, estimation by analogy, CO-COMO, effort prediction, ArchANGEL

1. Introduction

Several software projects fail due to management problems instead of technical weaknesses [1]. Among others, cost overrun is the most important reason for project failures. As a result, different estimation models emerged to better estimate the cost of software projects. In the presence of such heterogeneous estimation approaches, having a unified standard for software metrics remained a challenge for software industry. Most estimation tools support a variety of metrics. The metric names may be different in these tools due to “commercial” reasons. When the same metric is applied on a software project by multiple tools, one may have different results, mainly because of calculation methodology. Take for example the lines of code metric, there are several related aspects that may make counting such simple metric ambiguous. For example, some metrics may consider “lines with comments” while others may not. The definition of “line” or “statement” itself may also be controversial. As a result some tools count the lines based on the standard definition of line, while other tools take the statement to represent the line of code.

In this paper, a survey of several cost estimation datasets and related software metrics is carried out. The goal is to evaluate these metrics and provide a common standard in terms of software metrics and attributes that are related to cost estimation. Sixteen public cost estimation datasets (mostly from PROMISE website) are analyzed and compared. Each dataset has several attributes related to the software project, product or process. The comparison results are presented in this paper.

2. Software Cost Drivers

Cost drivers are multiplicative factors that determine the effort required to complete your software project. All cost estimation datasets
take into account factors related to the environment, team, development tools, etc. to predict cost of software projects. These parameters were first described in constructive cost models (COCOMO) [2, 3]. These parameters could be classified in different groups. We describe five different ways highlighted in the software engineering literature to classify these attributes.

- Based on the nature of software project these multipliers are distributed into the following: five multipliers related to the developed product (reliability, complexity, documentation, database size, reusability), three drivers related to the computer/environment (execution time constraints, platform volatility, and memory constraints), six drivers related to the project team/personnel (continuity, capability, programmers’ experience, analysts’ experience, language and tool experience), and three are related to the project in general and the development tools (the use of development tools, schedule compression, environment and communication quality).

- COCOMO II divided effort estimation into four models based on project lifecycle stage where cost estimation is required. These 4 models are:
  I. Application composition model: This model is applied for estimation when the software is composed from existing parts.
  II. Early design model: This model is used when only requirements of software project are available and system design is not started.
  III. Reuse model: This model is used when software project is focused on integration of reusable components.
  IV. Post-architecture model: This model is used when system architecture/design of software project is available.

- The third classification is based on the impact of the attribute on the project effort attribute. This classification divides any dataset attributes into three sets. One set of attributes that have positive impact and is correlated with the effort and second set of attributes that have negative impact and have negative correlation with the effort and the third set of attributes that have no impact on the effort. This classification will be the main one that we will use in evaluating and proposing the overall framework of a universal cost estimation attributes’ list.

- The fourth classification is based on the fact that some of these attributes represent atomic attributes that can be measured directly and others represent measurements or metrics that require formulas of more than one atomic attribute.

- The last classification that we discuss is based on the division of the software concerns (e.g. 4 Ps): project, process, product and people. This is an extension of the first version in which COCOMO I and II divide the drivers into (i.e. product, environment, project, personnel).

### 3. Cost Estimation Datasets

In this section we briefly describe different cost estimation datasets. Table 1 shows a summary of the cost estimation datasets that will be analyzed in this study. The majority of these datasets are collected from PROMISE website (http://promisedata.org) and (http://promise.site.uottawa.ca).

<table>
<thead>
<tr>
<th>NO</th>
<th>Dataset</th>
<th>Instances</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>USP_05_1</td>
<td>76</td>
<td>15</td>
</tr>
<tr>
<td>2</td>
<td>USP_05_2</td>
<td>203</td>
<td>17</td>
</tr>
<tr>
<td>3</td>
<td>Telecom</td>
<td>18</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>NASA93</td>
<td>93</td>
<td>24</td>
</tr>
<tr>
<td>5</td>
<td>Miyazaki94</td>
<td>48</td>
<td>9</td>
</tr>
<tr>
<td>6</td>
<td>Maxwell</td>
<td>62</td>
<td>27</td>
</tr>
<tr>
<td>7</td>
<td>Kitchenham</td>
<td>145</td>
<td>10</td>
</tr>
<tr>
<td>8</td>
<td>Kemerer</td>
<td>15</td>
<td>8</td>
</tr>
<tr>
<td>9</td>
<td>Finnish</td>
<td>38</td>
<td>9</td>
</tr>
<tr>
<td>10</td>
<td>Desharnais</td>
<td>81</td>
<td>12</td>
</tr>
<tr>
<td>11</td>
<td>COCOMONASA</td>
<td>60</td>
<td>17</td>
</tr>
<tr>
<td>12</td>
<td>COCOMO81</td>
<td>63</td>
<td>17</td>
</tr>
<tr>
<td>13</td>
<td>Albrecht</td>
<td>24</td>
<td>8</td>
</tr>
<tr>
<td>14</td>
<td>China</td>
<td>499</td>
<td>19</td>
</tr>
<tr>
<td>15</td>
<td>Boetticher</td>
<td>171</td>
<td>57</td>
</tr>
<tr>
<td>16</td>
<td>Humans</td>
<td>122</td>
<td>22</td>
</tr>
</tbody>
</table>

*Table 1. Summary of cost estimation datasets.*

Each one of cost estimation datasets includes several attributes related to the software project, product or process. There are several units
that can be used for cost estimation. Cost of software projects can be estimated in time (e.g. 4 months), money/expenses (e.g. $50K), features (e.g. 345 features) or effort (e.g. person month, day or year). The majority of cost estimation datasets use the last unit of measure for cost estimation because software development is a creative, labor intensive activity; once you know the effort required from technical people, transforming such value into a cost in terms of money is usually straightforward.

3.1. Software Metrics (Related to Cost Estimation)

Datasets usually include a mixture of software attributes and metrics. Attributes such as LOC usually represent properties that can be calculated atomically (i.e. from a single property) in a straightforward count. On the other hand, metrics such as Halstead or McCabe complexity are calculated using formulas and are based on more than one property or attribute. As mentioned earlier, one classification for the dataset attributes is to divide them into atomic attributes and metrics.

In another classification for the attributes or metrics that are included in software cost estimation datasets is to divide those metrics into subjective (ordinal) and objective (numeric) where many of those attributes require user decisions (e.g. level of knowledge, experience, domain complexity, etc) which may vary from one person to another. In fact one of the major complaints on COCOMO II cost estimation models is that while those models are supposed to replace “expert judgment” approach for cost estimation, such models have a large number of attributes, parameters or drivers that are subjective and depend on user knowledge or experience in the domain and in the estimation process.

3.2. Cost Estimation Accuracy

The cost estimation prediction accuracy may vary due to several factors that can affect such prediction. These include:

- Changes in the requirements, or user’s lack of understanding of the requirements
- Project and staff size and inaccuracies related to the variance of team members’ capabilities.
- Overlooked tasks in estimation and lack of an adequate method or guidelines for estimation.
- Insufficient analysis when developing estimates.
- Problems in the software development process such as: lack of coordination between system development, technical services, operations, data administration, and other functions during development.
- Problems related to environment and integrating the new system with existing systems or components.

The majority of research papers on cost estimation almost come up with an agreement on the idea that whatever cost estimation model is used, there is a need for local customization or calibration. This is largely because, for all earlier mentioned reasons, it is almost impossible to come up with two software projects that will have identical attributes in terms of all previously mentioned variations (i.e. in requirements, people, project, process, etc.).

There are several metrics that are used to evaluate the accuracy of this prediction.

1. Absolute error. The calculation for cost estimation accuracy is given as follows

\[
\text{Absolute error} = (\text{E}_{\text{pred}} - \text{E}_{\text{actual}})
\]

2. Relative error:

\[
\text{Percentage error} = \frac{(\text{E}_{\text{pred}} - \text{E}_{\text{actual}})}{\text{E}_{\text{actual}}}
\]

Mean magnitude of relative error (MMRE): This is an indication of the average errors given by the prediction formula or system. It is calculated based on the absolute difference between actual and predicted estimation divided on actual estimation. This is calculated per each instance or project. The mean is expressed as the average amount of error in the estimated level of effort as opposed to the actual level of effort in person, days, or months.
3. Pred(x/100): Prediction at level x or percentage of projects for which estimate is within x% of the actual. The drawback of Pred(x) metric is that it does not provide accuracy indication for estimations that don’t fall within x%.

4. Boxplots. This is a statistical summary plot which is also used to evaluate the accuracy of cost estimation prediction.

3.3. ArchANGEL

ArchANGEL is a recent version of ANGEL cost estimation automated tool that supports collecting, analyzing, and predicting cost or effort in software projects based on analogy [4]. It has several different options and algorithms and is based upon the minimization of Euclidean distance in n-dimensional space. The tool includes several versions for feature subset selection or optimization to find the least good possible set of attributes that can best predict the effort of the project. We will use this tool to compare the best effort predictors across all evaluated datasets.

In this paper, and most cost estimation papers, accuracy estimators are used to evaluate the accuracy of cost estimation prediction. Relative and absolute accuracy indicators have been used. Examples of those include: Mean Squared Error (MSE), Absolute Residuals (AR), Balanced Residual Errors (BRE), Magnitude Relative Error (MRE), Mean Magnitude Relative Error (MMRE) and Prediction with X (PRED(X)). Those are the prediction accuracy metrics described earlier.

The tool includes several options or parameter values for variables that may impact or decide the feature selection process that users may decide. We will evaluate some of those different parameter values or features in the experiment section.

4. Related Work

Boehm is one of the earlier researchers in software engineering who is famous for his cost estimation models COCOMO I and II [2,3]. In fact, several cost estimation datasets (e.g. NASA93, COCOMO81 and COCOMONASA) are using the exact names and definitions of the drivers described in those models [5].

Many research papers that used the datasets for cost estimation or prediction followed a hold-out strategy where the dataset is divided into training and testing where both parts are, usually, selected randomly from the original dataset. Some other papers tried to deal with the fact that in future projects and, in some cases, not all attributes for a particular model are available (i.e. missing values problem). Feature set selection is another major research subject in cost estimation where some papers tried to evaluate the minimum set of attributes that can best predict the effort estimation[6,7].

Data mining techniques (LR, ANN, SVR and K-NN) are also used in software cost estimation. A comparison between the intermediate COCOMO model and data mining models shows an accuracy improvement using data mining techniques and results are presented in [8].

The technique used for cost modeling is another variable in cost estimation papers. Examples of cost modeling techniques used in analogy cost estimation are: Case Based Reasoning (CBR), Stepwise and multiple stepwise regression (SWR), ANOVA, least squares regression, Bayesian networks, Web-COBRA, decision trees, and regression trees (CART).

A visual comparison of software cost estimation models that is based on regression error characteristic analysis is shown in [9]. Depending on some geometrical properties and a generated graph, a simple visual inspection shows a comparison of the investigated models.

Software effort estimation based on Least Squares Regression with adaptive recursive data partitioning is proposed in [10]. An evaluation of the proposed method is done using empirical experiments that showed an improvement of the proposed algorithm over the basic LSR approach. This is because the proposed technique tries to alleviate the effect of data distribution which is a major practical issue in software effort estimation.

A rule-based approach for estimating software development cost in the requirements analysis phase is another approach that is presented in [11].
The proposed method provides the intermedia-
tion between requirements and cost. It presents
rules for extracting function point, an input pa-
rameter for existing cost models, from goal and
scenario based requirements and thus, links re-
quirements with function point.

Computational intelligent techniques are also
used in cost estimation. A review of some
important computational techniques (which in-
clude neural networks, evolutionary computa-
tion, and fuzzy logic) that have been used for
software cost estimation is shown in [12]. Com-
putational intelligent techniques are generally
considered as powerful tools and promising
techniques because a large number of genera-
lization factors are connected with software
projects.

Kemerer 1987 paper [17] is a popular paper with
case studies for evaluating cost estimation in dif-
f erent software projects. The paper evaluated
the prediction accuracy of four popular cost es-
timation prediction models or methods. The
author tried also to evaluate the impact of using
or excluding Lines of Code (LOC) as one of the
popular, yet, disputed size or estimation met-
rics. Results showed that models and even cost
estimation datasets may not always include or
model properly productivity related attributes.

Jørgensen and Shepperd 2007 paper [18] in-
cluded a systematic review on research work
done in this area. Authors tried to review and
cite popular papers in this area and the nature or
the arena of those publications.

Boehm, Abts and Chulani 2000 paper [19] is
another popular paper in terms of number of ci-
tations, that includes an extensive survey of cost
estimation papers and approaches.

Briand et al paper 1999 [20] represents also an-
other form of survey paper to evaluate different
papers and algorithms for cost estimation. The
paper evaluated different approaches in terms of
prediction accuracy or quality using several
factors or metrics for that purpose.

AQUA method of cost estimation prediction
for future software projects was proposed in
[13]. Authors claimed combining elements
from two analogy based estimation methods: cas-
based reasoning and collaborative filter-
ing. The method improves its accuracy through
continuous evaluation of training datasets. Their
method proposed solutions for missing values
and non-quantitative variables. They conducted
a sensitivity analysis between cost estimation
prediction accuracy and the varying of sam-
ple or training dataset and similarity measure.
Based on the attribute type (e.g. nominal, ordi-
nal, interval, etc) they defined local similarity
measures to evaluate similarities between the
different instances.

Menzies has several contributions to the field
of cost estimation [5,14]. The authors proposed
COCONUT cost estimation calibration tool that
implements an incremental holdout study for
COCOMO model. The authors discussed one
popular problem of “fitness” in cost estimation
models where attributes, formulas, drivers, etc
in each dataset are somewhat proprietary and
may not be, largely, applicable to other soft-
ware business domains.

Challenges in cost estimation for distributed
software projects were discussed in [15]. An ob-
servational phase to first identify the root causes
of cost estimation errors is introduced, then a
proposal of a semi-automated solution is pre-
sented: a solution that is based on case-based
reasoning and the learning-oriented approach
that allows project managers, regardless of their
experience levels, to derive better cost estimates
and therefore reduces estimation errors.

5. Goals and Approaches

The initial drive for writing this paper is to
analyze and compare different cost estimation
datasets hoping to compile a standard list of
common metrics across all those datasets. While
COCOMO I and II drivers are one of the com-
mon factors in some of those datasets, nonethe-
less, there are more metrics and attributes than
those drivers while in addition, those same drivers
are used in either different units or terminolo-
gies across all cost estimation datasets. In the
first step, and based on the different classifi-
cations we proposed earlier for the dataset at-
tributes, we will present all attributes from the
different datasets according to those classifica-
tions. The following formula shows the gen-
eral equation for effort estimation in COCOMO
models.

\[ PM_{estimated} = A \times (size)^{SF} \times \prod EM_i \]
where \( PM \) is the effort estimation using parametric model, \( A \) is a constant, \( size \) is the program size (i.e. line of code), \( SF \) is a scale factor, and \( EM \) are effort multipliers.

Besides the project or code size, the equation defines two major sections of variables:

1. Scale Factors. COCOMO model defines 5 scale factors: Precedentedness (PREC), Development Flexibility (FLEX), Architecture/Risk Resolution (RESL), Team Cohesion (TEAM), Process Maturity (PMAT). Cost or effort estimation datasets rarely include data about those 5 subjective attributes.

2. Cost drivers. Those 17 drivers are shown in Figure 1 and described in Table 2.

### Table 2. COCOMO II 17 drivers.

<table>
<thead>
<tr>
<th>Driver</th>
<th>Description</th>
<th>Documentation Match to Lifecycle Needs</th>
<th>Language and Tools Experience</th>
<th>Personnel Continuity</th>
<th>Required Reliability</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATA</td>
<td>Database size</td>
<td>TOOL</td>
<td>Use of software tools</td>
<td>ACAP</td>
<td>APEX</td>
</tr>
<tr>
<td>CPLX</td>
<td>Product complexity</td>
<td>APEX</td>
<td>Application experience</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIME</td>
<td>Execution time constraint</td>
<td>ACAP</td>
<td>Analyst experience</td>
<td></td>
<td></td>
</tr>
<tr>
<td>STOR</td>
<td>Main storage constraint</td>
<td>PCAP</td>
<td>Programmer capability</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RUSE</td>
<td>Required reusability</td>
<td>PLEX</td>
<td>Platform experience</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DOCU</td>
<td>Documentation match to lifecycle needs</td>
<td>LTEx</td>
<td>Language and tools experience</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PVOL</td>
<td>Platform volatility</td>
<td>PCON</td>
<td>Personnel continuity</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SCED</td>
<td>Scheduling factor</td>
<td>SITE</td>
<td>Multisite development</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RELY</td>
<td>Required reliability</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The sample in Table 3 shows the conversion from categories that users used for estimation to nominal values that are used in the estimation model. COCOMO model has some standard numbers for converting categories to numbers which may however not be the same for all drivers or multipliers and has therefore to be checked according to the model manual.

Most of COCOMO I and II drivers are subjective. The amount of subjectivity of each attribute can also vary where some attributes, despite being subjective, can be calculated based on defined criteria (e.g. database size, team experience (if measured by years)). The majority of attributes in NASA datasets (e.g. COCOMO81 and NASA93) are subjective.

While COCOMO model literatures include some hints and equations on how to calculate those drivers, nevertheless, all of those 17 drivers require user or expert subjective selection which can vary from one person to another.

In order to allow data analysis and calculation, all values of attributes are categorized into levels (e.g. very low, low, medium, high, very high) and each category is given a numerical value. Table 3 shows a sample of nominal categorization of COCOMO drivers.

### Table 3. Sample of nominal categorization of COCOMO drivers.

<table>
<thead>
<tr>
<th>Very Low</th>
<th>Low</th>
<th>Nominal</th>
<th>High</th>
<th>Very High</th>
<th>Extra High</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.75</td>
<td>0.88</td>
<td>1.00</td>
<td>1.15</td>
<td>1.4</td>
<td>–</td>
</tr>
<tr>
<td>–</td>
<td>0.94</td>
<td>1.00</td>
<td>1.15</td>
<td>1.30</td>
<td>1.65</td>
</tr>
</tbody>
</table>

The sample in Table 3 shows the conversion from categories that users used for estimation to nominal values that are used in the estimation model. COCOMO model has some standard numbers for converting categories to numbers which may however not be the same for all drivers or multipliers and has therefore to be checked according to the model manual.

Most of COCOMO I and II drivers are subjective. The amount of subjectivity of each attribute can also vary where some attributes, despite being subjective, can be calculated based on defined criteria (e.g. database size, team experience (if measured by years)). The majority of attributes in NASA datasets (e.g. COCOMO81 and NASA93) are subjective.

While we consider all COCOMO drivers in this research as subjective, nonetheless, cost estimation datasets include several examples of attributes or metrics that can be counted much less subjective than those drivers. Examples of such attributes include:

- **Size attributes.** Datasets may include size related attributes. Those can be related to requirements or to code (e.g. LOC or KLOC). We consider those attributes “objective” as they can be counted or measured and are not based on user heuristic judgment.
• Attributes related to time (e.g. duration, start date, etc). In some datasets, those attributes have two forms: predicted and actual. Unlike attributes related to level of knowledge, skills, experience, domain complexity, etc, attributes related to date are calculated straightforward.

• There are some attributes related to requirements that can be counted or measured numerically. Examples of such metrics include: 1. SCRN: number of different input or output screens, 2. FORM: number of different (report) forms, 3. FILE: number of different record formats, 4. ESCRN: total number of data elements in all the screens, 5. EFORM: total number of data elements in all the forms, and 6. EFILE: total number of data elements in all the files.

Table 4 shows data sets’ attributes classification based on: ordinal, nominal and numerical attributes.

The majority of the attributes in NASA datasets are transferred from nominal into ordinal. Ordinal data has an inherent order, i.e ranking, in its possible values. For example “very low, low, medium, high and very high” are ordinal because there is an assumption that those possible values are higher from one to the next. It can be coded as 1, 2, 3, 4, and 5. Nonetheless, there is no assumption of equal spacing. Nominal data has no inherent ranking, only labeling. For example: male, and female. Any numerical coding does not reflect any quantitative meaning.

In Table 4, COCOMO datasets are distinguished by having a large number of ordinal attributes. Nominal or textual attributes are used to distinguish one instance from another, but they are useless for any statistical or data mining algorithm of prediction, correlation, association, etc. The easiest attributes to deal with statistically are the numerical attributes. However, it is not always possible to compute and have such attributes.

Table 5 shows the division of attributes between atomic and compound. Nominal attributes are excluded. Ordinal attributes are also excluded from this table as they are calculated based on special subjective user-decided attributes. The focus is only on dividing the numerical attributes in this table.

The majority of the numeric values are numeric evaluated based on simple count. This can simplify calculating those attributes and make them consistent among the different datasets.

The last perspective that we will classify the dataset attributes upon is the partition of attributes based on the project perspectives: people, process, product and project.
We will assume that all attributes related to effort are project related. However, they can be also categorized under “product”. Tools and language-related attributes are considered part of the process attributes. Results are shown in Table 6.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>People</th>
<th>Process</th>
<th>Product</th>
<th>Projects</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>4</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>7</td>
<td>2</td>
<td>3</td>
<td>12</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>6</td>
<td>3</td>
<td>12</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>12</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>1</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>0</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>10</td>
<td>2</td>
<td>1</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>11</td>
<td>4</td>
<td>2</td>
<td>3</td>
<td>8</td>
</tr>
<tr>
<td>12</td>
<td>4</td>
<td>2</td>
<td>3</td>
<td>8</td>
</tr>
<tr>
<td>13</td>
<td>0</td>
<td>0</td>
<td>7</td>
<td>1</td>
</tr>
<tr>
<td>14</td>
<td>0</td>
<td>1</td>
<td>12</td>
<td>6</td>
</tr>
<tr>
<td>15</td>
<td>50</td>
<td>0</td>
<td>0</td>
<td>7</td>
</tr>
<tr>
<td>16</td>
<td>20</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
</tbody>
</table>

Table 6. 4Ps classification of attributes.

5.1. Feature Selection to Find Important Attributes

One of the problems in cost estimation datasets is the large number of attributes that those datasets contain. In addition, the number of common attributes between the different attributes is limited. We thought of using feature selection as a method to reduce the number of important attributes for the goal of cost estimation.

We used a tool called “ArchANGEL” which is described earlier. The tool can find, through different algorithms, a reduced set of attributes that can best represent the overall set of attributes toward a specific target. In the cost estimation problem, the goal or the target attribute is the effort or actual effort.

Table 7 shows the comparison between the datasets in terms of feature subset optimization. Selection is based on ArchANGEL tool with the following parameter values: (Target feature: effort, holdout strategy: Jack-Knife, Adaptation strategy: simple average, performance indicator: MMRE, selection strategy: forward sequential selection, and number of analogies: 1).

<table>
<thead>
<tr>
<th>Best feature subset</th>
<th>MMRE</th>
<th>Rem</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. ID, effort, DataFile, DataEn, DataOut, Lang, ToolExp, AppExp, TeamSize</td>
<td>0.201</td>
<td>6</td>
</tr>
<tr>
<td>2. ID, effort, DataFile, DataEn, DataOut, Lang, ToolExp, AppExp, DBMS, AppType</td>
<td>0.383</td>
<td>7</td>
</tr>
<tr>
<td>3. Files, Effort</td>
<td>0.4</td>
<td>1</td>
</tr>
<tr>
<td>4. Name, mode, stor, Vexp, Kloc, effort</td>
<td>0.515</td>
<td>18</td>
</tr>
<tr>
<td>5. KLOC, SCRN, FROM, ManMonth</td>
<td>0.392</td>
<td>4</td>
</tr>
<tr>
<td>6. T15, Duration, Size, Effort</td>
<td>0.543</td>
<td>23</td>
</tr>
<tr>
<td>7. Effort, First estimate</td>
<td>0.355</td>
<td>8</td>
</tr>
<tr>
<td>8. Effort, RawFP, RawFP</td>
<td>0.649</td>
<td>5</td>
</tr>
<tr>
<td>9. LnEff, DivEff</td>
<td>0.049</td>
<td>7</td>
</tr>
<tr>
<td>10. Effort, PointsNonAdjust, and language</td>
<td>0.434</td>
<td>9</td>
</tr>
<tr>
<td>11. None (Very Large Number)</td>
<td>VLN</td>
<td>17</td>
</tr>
<tr>
<td>12. None</td>
<td>VLN</td>
<td>17</td>
</tr>
<tr>
<td>13. Output, inquiry, RAWFP, AdjFP, Effort</td>
<td>0.63</td>
<td>3</td>
</tr>
<tr>
<td>14. Resource, N_Effort, Effort</td>
<td>0.113</td>
<td>16</td>
</tr>
<tr>
<td>16. TechCou., HWExp., SWExp., ABS</td>
<td>5.34</td>
<td>11</td>
</tr>
</tbody>
</table>

Table 7. Effort best features’ predictors: 1 analogy.
Evaluation of Cost Estimation Metrics: Towards a Unified Terminology

<table>
<thead>
<tr>
<th>Best feature subset</th>
<th>MMRE</th>
<th>Rem</th>
</tr>
</thead>
<tbody>
<tr>
<td>Effort, DataFile, DataOut, Lang. Tools, AppExp. TeamSize, AppType</td>
<td>0.32</td>
<td>7</td>
</tr>
<tr>
<td>ID, effort, DataFile, DataEn, DataOut, Lang. ToolExp, AppExp. DBMS, UFP, AppType</td>
<td>0.41</td>
<td></td>
</tr>
<tr>
<td>Files, Effort, Changes</td>
<td>0.625</td>
<td>0</td>
</tr>
<tr>
<td>Rely, Modp, LOC, effort</td>
<td>0.788</td>
<td>20</td>
</tr>
<tr>
<td>KLOC, ManMonth</td>
<td>0.396</td>
<td>6</td>
</tr>
<tr>
<td>T05, Duration, Size, Effort</td>
<td>0.443</td>
<td>23</td>
</tr>
<tr>
<td>Effort, First estimate</td>
<td>0.299</td>
<td>8</td>
</tr>
<tr>
<td>KSLOC, Effort</td>
<td>0.592</td>
<td>6</td>
</tr>
<tr>
<td>LnEff, DivEff</td>
<td>0.081</td>
<td>7</td>
</tr>
<tr>
<td>Effort, PointsNonAdjust, and language</td>
<td>0.368</td>
<td>9</td>
</tr>
<tr>
<td>None</td>
<td>VLN</td>
<td>17</td>
</tr>
<tr>
<td>None</td>
<td>VLN</td>
<td>17</td>
</tr>
<tr>
<td>Input, Output, File, AdjFP, Effort</td>
<td>0.622</td>
<td>3</td>
</tr>
<tr>
<td>N_Effort, Effort</td>
<td>0.114</td>
<td>17</td>
</tr>
<tr>
<td>Grad_Cou, Tech. Conf., C#, Corr.</td>
<td>0.863</td>
<td>56</td>
</tr>
<tr>
<td>TechCou., HEXP, SW, HW, Dom_Exp, ABS</td>
<td>7.656</td>
<td>9</td>
</tr>
</tbody>
</table>

Table 8. Effort best features’ predictors: 3 analogy.

Table 8 is the same exact set with 3 analogies. In both tables MMRE is the Mean Magnitude Relative Error and Rem refers to how many attributes were removed in the feature selection algorithm used. MMRE and Rem are known validation metrics in cost estimation in particular and in data mining in general.

If we take MMRE as the reference and according to Conte et al. 1986 [21], consider MMRE ≤ 0.25 as an acceptable level of performance for effort prediction models, three selections fall within this acceptable level.

The main goal of Tables 7 and 8 is to see the main attributes that affect the cost estimation in trying to reduce the number of attributes, especially where some datasets include a large number of attributes. The used algorithm showed the ability of finding a small number of attributes that can represent the rest with a significant value of MMRE which indicates the quality of the prediction. In case of NASA datasets (i.e. 11 and 12), the tool found a very large value of MMRE and it eliminates all dataset attributes (which means that it fails to predict the best attributes’ representatives). For dataset (15), there were no specific attributes for effort. Instead, the correlation attribute is used as the goal or the target attribute. In most cases, percentage of error is less than one and it is higher where three analogies are selected instead of one (i.e. Table 8 vs. Table 7).

5.2. A Decision Support System for Cost Estimation

There are three major methods for performing cost estimation: expert-based, algorithmic-based and analogy-based estimation models. Currently, there is no significant connection between those models, especially between algorithmic and analogy based models (although initial cost estimation models and drivers are built on analogy based). This is a proposal to build a dictionary for cost estimation which can be a helpful tool for software project managers. In such dictionary, projects should be divided into different aspects or perspectives where each one of those will have its own formulas and values for the drivers. For example, a generic equation such as:

\[
\text{Effort (in requirement stage)} = A \times \text{FP} + B,
\]

where A and B can be linear or complex values that are measured using selections based on values found in the cost estimation dictionary. FP stands for Function Points, a popular requirement based cost estimation metric that tries to normalize measuring requirements by scaling them according to several factors related to the complexity of the requirement. In order to calculate those constants, goal based algorithms (e.g. decision tree) can be used using those
### Evaluation of Cost Estimation Metrics: Towards a Unified Terminology

Table 9 includes an extensive amount of information to discuss. Some attributes (Function percentage in dataset 2, tools in dataset 1) affect the effort equation in different levels depending on the attribute’s actual value in the instance, which is why they are repeated in the effort equations.

For each cost estimation attribute, it is important to investigate in details the attributes impact on the “effort attribute”. This can be: positively correlated (high, medium or low), negatively correlated (high, medium or low), or with no impact. In general, in the above equations, if the attribute was missing, it would mean that it does not have a significant impact on the effort. On the other hand, if the attribute is preceded by (minus) or is in the denominator, then it negatively impacts the effort. The significance of

<table>
<thead>
<tr>
<th>No</th>
<th>Effort Equation using M5P algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.0035 * ID + 0.2293 * DataOut + 13.9327 * Lang + 3.5157 * Tools1 - 3.0584 * Tools2 + 15.4772 * TeamSize + 2.1568 * DBMS + 0.1809</td>
</tr>
<tr>
<td>2</td>
<td>-0.0957 * ObjType = RQ, PJ + 7.9820 * ObjType = PJ + 0.3878 * FunctPercent1 + 0.5104 FunctPercent2 + 0.9286 * FunctPercent3 + 0.2671 * FunctPercent4 + 0.8334 * FunctPercent5 + 2.5666 * FunctPercent6 + 3.8828 * FunctPercent7 + 0.4613 * IntComplex + 6.965 * DataFile + 0.2158 * UFP + 0.179 * Lang + 0.5947</td>
</tr>
<tr>
<td>3</td>
<td>1.606 * CHANGES + 62.6137 [18/68.82%]</td>
</tr>
<tr>
<td>4</td>
<td>393.1026 * mode: embedded + 361.0229 * data = vl, xh, h + 1703.6997 * time = vh + 3.9232 * equivphyskloc - 45.7682</td>
</tr>
<tr>
<td>5</td>
<td>666.8711 * ID = E2, L3, M1 + 45.7956</td>
</tr>
<tr>
<td>6</td>
<td>-557.1529 * Har + 3760.0497 * T08 + 748.9552 * T09 + 135.8155 * Duration + 11.44 * Size - 19479.6209</td>
</tr>
<tr>
<td>7</td>
<td>-2.8608 * Project + 0.7392 * Actualduration + 0.5063 * AdjustedFP + 0.8333 * Firstestimate + 55.7193</td>
</tr>
<tr>
<td>8</td>
<td>-16.9701 * @attributeID + 58.7094 * @attribute_Hardware + 0.3857 * @attribute_AdjFP - 167.3211</td>
</tr>
<tr>
<td>9</td>
<td>-0.1601 * FP + 11230.6991 * leff - 92810.5527</td>
</tr>
<tr>
<td>11</td>
<td>393.1026 * mode: embedded + 361.0229 * data = vl, xh, h + 1703.6997 * time = vh + 3.9232 * equivphyskloc - 45.7682</td>
</tr>
<tr>
<td>12</td>
<td>56.0425 * TIME = Very_High, High + 365.9082 * VIRT = Low_High, High</td>
</tr>
<tr>
<td>13</td>
<td>188.6733 * VEXP = Nominal, High + 7.1874 * LOC = 961.138</td>
</tr>
<tr>
<td>14</td>
<td>0.1026 * Output + 0.6237 * Inquiry - 23.8873 * FPAdj + 0.0255 * AdjFP + 19.6942</td>
</tr>
<tr>
<td>15</td>
<td>1.1637 * AFP - 1.4791 * Input - 1.617 * Enquiry - 2.3919 * Interface + 0.1773 Added + 269.8365 * Resource + 17.1556 * Duration + 0.8405 * N_effort - 443.1543</td>
</tr>
<tr>
<td>16</td>
<td>-0.0043 * CompSc_Undergrad_Courses - 0.002 * CompSc_Grad_Courses - 0.0023 * SoftEngr_Grad_Courses + 0.0018 * TechGrad_Courses + 0.0042 * Hardware_Conferences + 0.0042 * MIS_Workshops - 0.0144 * ProjMgmt_Conferences + 0.0084 * SoftEngr_Conferences + 0.0025 * Database_Experience - 0.0025 * Domain_Exp + 0.4153</td>
</tr>
<tr>
<td>17</td>
<td>ABS((TotalEstimates - TotalActual) / TotalActual) = 1.7901</td>
</tr>
</tbody>
</table>

Table 9. Decision tree classification.

Available cost estimation datasets. For example, using a decision tree, we can go back to each one of the previous datasets to find a suitable equation for effort where in each equation on the left we will have only the effort and on the right we will have all the attributes along with a constant for each attribute that is calculated based on the instances from the cost estimation datasets. Table 9 shows the results of applying the M5P decision tree algorithm on all evaluated cost estimation datasets. In order to reduce the clutter in the Table, several less important issues are ignored. 10-fold cross validation is used for the decision tree test options.
the impact can be noticed through the constant number that is multiplied by the attribute.

As mentioned earlier, eventually a cost estimation dictionary should be built based on those effort formulas to help in future cost estimation predictions.

6. Conclusion and Future Work

Finding a one-for-all cost estimation model that can predict with high accuracy software projects in different sizes, stages of development, problem domain and complexity is not practical or feasible. This paper evaluates and compares different metrics and datasets to find a standardized approach for software cost estimation metrics. 16 datasets and their attributes were analyzed in terms of subjective/objective, atomic/compound, 4Ps. By using ArchANGEL, the best features that could represent the whole attribute set for different datasets were found, and a decision support system for cost estimation that other project managers can benefit was proposed.

This paper suggested a unified “language”, terminology, or set of attributes in cost estimation datasets. Even if some datasets have unique and new attributes, common terminology should be used. This can facilitate data-sets and research reusability. We used different models to compare the different cost estimation datasets and their attributes. It is important and necessary to find common terminologies and methods to evaluate in a consistent way the different information and statistics that we can get from the different cost estimation datasets.

References


[16] http://www.opentox.org/dev/documenta-
tion/components/m5p Accessed Aug 2012.


Received: January, 2013
Revised: May, 2013
Accepted: May, 2013

Contact addresses:
Izzat M. Alsmadi
Department of Computer Information Systems
Yarmouk University
Jordan
e-mail: ialsmadi@yu.edu.jo

Maryam S. Nuser
Department of Computer Information Systems
Yarmouk University
Jordan
e-mail: maryamnuser@yahoo.com

IZZAT M. ALSMADI is an Associate Professor in the Department of Computer Information Systems at Yarmouk University in Jordan. He obtained his Ph.D degree in software engineering from NDSU (USA), his second Master in software engineering from NDSU (USA) and his first Master in CIS from the University of Phoenix (USA). He had a B.Sc. degree in telecommunication engineering from Mutah University in Jordan. He has several published books, journals and conference articles largely in software engineering and information retrieval fields.

MARYAM S. NUSER is an Assistant Professor in the Department of Computer Information Systems at Yarmouk University Jordan. She received a B.Sc. degree in Computer Science from Yarmouk University in 1995, Msc degree from the University of Arkansas, USA in 2002, and a PhD degree from the University of Arkansas in 2004 with the same major.

She worked as Head of the CIS Dept. at Yarmouk University during the period 2006-2008. She has several publications in international and local journals, conferences, and books.