Journal of Computing and Information Technology - CIT 21, 2013, 1, 23–34
doi:10.2498/cit.1002133

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, COCOMO, 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`).

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

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 I 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:

- Frequent request for change by customers.
- 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 previous 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

$$\text{Absolute error} = (\text{Epred} - \text{Eactual})$$

2. Relative error:

   $$\text{Percentage error} = (\text{Epred} - \text{Eactual})/\text{Eactual}$$

   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 intermediation between requirements and cost. It presents rules for extracting function point, an input parameter for existing cost models, from goal and scenario based requirements and thus, links requirements with function point.

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

Kemerer 1987 paper [17] is a popular paper with case studies for evaluating cost estimation in different software projects. The paper evaluated the prediction accuracy of four popular cost estimation 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 metrics. 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] included 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 citations, that includes an extensive survey of cost estimation papers and approaches.

Briand et al paper 1999 [20] represents also another 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: case-based reasoning and collaborative filtering. 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 sample or training dataset and similarity measure. Based on the attribute type (e.g. nominal, ordinal, 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 software business domains.

Challenges in cost estimation for distributed software projects were discussed in [15]. An observational phase to first identify the root causes of cost estimation errors is introduced, then a proposal of a semi-automated solution is presented: 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 common factors in some of those datasets, nonetheless, there are more metrics and attributes than those drivers while in addition, those same drivers are used in either different units or terminologies across all cost estimation datasets. In the first step, and based on the different classifications we proposed earlier for the dataset attributes, we will present all attributes from the different datasets according to those classifications. The following formula shows the general equation for effort estimation in COCOMO models.

$$PM_{estimated} = A \times (size)^{SF} \times \prod_{i} EM_i$$

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.

DATA	Database size.	TOOL	Use of software tools
CPLX	Product complexity.	APEX	Application experience.
TIME	Execution time constraint.	ACAP	Analyst capability.
STOR	Main storage constraint.	PCAP	Programmer capability.
RUSE	Required reusability.	PLEX	Platform experience.
DOCU	Documentation match to life-cycle needs	LTEX	Language and tools experience.
PVOL	Platform volatility.	PCON	Personnel continuity.
SCED	Scheduling factor.	SITE	Multisite development.
RELY	Required reliability.		

Table 2. COCOMO II 17 drivers.

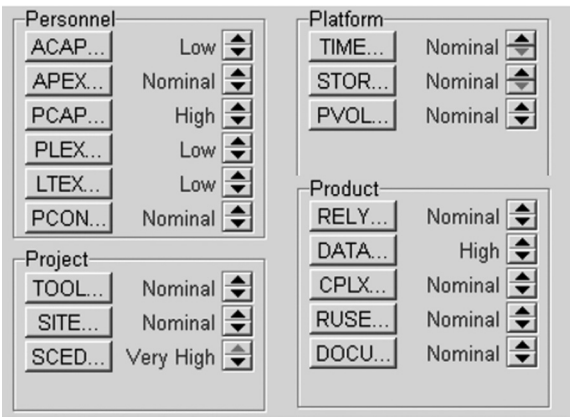

Figure 1. COCOMO II 17 drivers (classified).

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.

Very Low	Low	Nominal	High	Very High	Extra High
0.75	0.88	1.00	1.15	1.4	–
–	0.94	1.00			–
0.70	0.85	1.00	1.15	1.30	1.65

Table 3. Sample of nominal categorization of COCOMO drivers.

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.

Dataset	Subjective (ordinal) attributes	Nominal or Textual attributes	Numerical attributes
1	4	6	5
2	4	8	5
3	0	1	3
4	15	7	2
5	0	1	8
6	16	6	5
7	0	4	6
8	2	3	3
9	3	3	3
10	3	2	7
11	15	0	2
12	15	0	2
13	0	0	8
14	0	1	18
15	24	0	33
16	8	2	12

Table 4. Subjective and objective attributes.

Dataset	Numerical attributes	Atomic	Compound
1	5	5	0
2	5	5	0
3	3	3	0
4	2	2	0
5	8	8	0
6	5	4	1
7	6	4	2
8	3	2	1
9	3	2	1
10	7	4	3
11	2	2	0
12	2	2	0
13	8	5	3
14	18	5	13
15	33	30	3
16	12	10	2

Table 5. Atomic vs. compound attributes.

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.

Dataset	People	Process	Product	Projects
1	2	4	4	5
2	2	4	5	6
3	0	0	1	3
4	7	2	3	12
5	0	0	5	2
6	6	6	3	12
7	0	0	1	12
8	0	1	3	4
9	1	0	4	4
10	2	1	4	5
11	4	2	3	8
12	4	2	3	8
13	0	0	7	1
14	0	1	12	6
15	50	0	0	7
16	20	0	0	2

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).

	Best feature subset	MMRE	Rem
1	ID, effort, DataFile, DataEn, DataOut, Lang. ToolExp, AppExp. TeamSize	**0.201**	**6**
2	ID, effort, DataFile, DataEn, DataOut, Lang. ToolExp, AppExp. DBMS, AppType	0.383	7
3	Files, Effort	0.4	1
4	Name, mode, stor, Vexp, Kloc, effort	0.515	18
5	KLOC, SCRN, FROM, ManMonth	0.392	4
6	T15, Duration, Size, Effort	0.543	23
7	Effort, First estimate	0.355	8
8	Lang. RawFP, Effort	0.649	5
9	LnEff, DivEff	**0.049**	**7**
10	Effort, PointsNonAdjust, and language	0.434	9
11	None	VLN (Very Large Number)	17
12	None	VLN	17
13	Output, inquiry, RAWFP, AdjFP, Effort	0.63	3
14	Resource, N_Effort, Effort	**0.113**	**16**
15	Degree, Grad_Cou, HW, Works, Assembly, Fortran, PHP, TCL, Proc. Scale, Corr.	1.013	50
16	TechCou., HWExp., SWExp., ABS	5.34	11

Table 7. Effort best features' predictors: 1 analogy.

	Best feature subset	MMRE	Rem
1	Effort, DataFile, DataOut, Lang. Tools, AppExp. TeamSize, AppType	0.32	7
2	ID, effort, DataFile, DataEn, DataOut, Lang. ToolExp, AppExp. DBMS, UFP, AppType	0.41	
3	Files, Effort, Changes	0.625	0
4	Rely, Modp, LOC, effort	0.788	20
5	KLOC, ManMonth	0.396	6
6	T05, Duration, Size, Effort	0.443	23
7	Effort, First estimate	0.299	8
8	KSLOC, Effort	0.592	6
9	LnEff, DivEff	**0.081**	**7**
10	Effort, PointsNonAdjust, and language	0.368	9
11	None	VLN	17
12	None	VLN	17
13	Input, Output, File, AdjFP, Effort	0.622	3
14	N_Effort, Effort	**0.114**	**17**
15	Grad_Cou, Tech. Conf., C#, Corr.	0.863	56
16	TechCou., HWExp., SW, HW, Dom_Exp., ABS	7.656	9

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$\leq$.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:

Effort (in requirement stage) =A* 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

No	Effort Equation using M5P algorithm
1	0.0035 * ID + 0.2293 * DataOut + 13.3927 * Lang+ 3.5157 * Tools1 -3.0584* Tools2+ 15.4772 * TeamSize +2.1568 * DBMS + 0.1809
2	-0.0957 * ObjType = RQ,PJ +7.6203 * ObjType =PJ+ 0.3878 * FunctPercent1+ 0.5104 FunctPercent2+ 0.9286 * FunctPercent3+ 0.2671 * FunctPercent4+ 0.8334 * FunctPercent5- 2.5366 * FunctPercent6+ 3.8828 * FunctPercent7+ 0.4613 * IntComplx+ 6.965 * DataFile+ 0.2158 * UFP+ 0.179 * Lang+ 0.9547
3	1.606 * CHANGES + 62.6137 [18/68.582%]
4	393.1026 * mode=embedded + 361.0229 * data=vl,xh,h + 1703.6997 * time=xh + 3.9232 * equivphyskloc - 45.7682
5	666.8711 * ID=E2,L3,M1 + 45.7956
6	-557.1529 * Har + 3760.0497 * T08 + 748.9552 * T09 + 135.8155 * Duration + 11.44 * Size - 19479.6209
7	-2.8608 * Project + 0.7392 * Actualduration + 0.5063 * AdjustedFP + 0.8333 * Firstestimate + 55.7193
8	-16.9701 * @attributeID + 58.7094 * @attribute_Hardware + 0.3857 * @attribute_AdjFP - 167.3211
9	-0.1601 * FP + 11230.6991 * lneff - 92810.5527
10	-366.8084 * TeamExp + 353.1945 * ManagerExp + 181.7017 * Length + 5.2616 * Transactions + 5.932 * Entities + 4.6713 * PointsAdjust + 110.3042 Envergure + 4315.4628 * Language=2,1 - 7107.6825
11	393.1026 * mode=embedded + 361.0229 * data=vl,xh,h + 1703.6997 * time=xh + 3.9232 * equivphyskloc - 45.7682
12	520.4254 * TIME=Very_High,High + 365.9082 * VIRT=Low,High + 168.2373 * VEXP=Nominal,High + 7.1874 * LOC - 961.138
13	0.1026 * Output + 0.6237 * Inquiry - 23.8873 * FPAdj + 0.0255 * AdjFP + 19.6942
14	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
15	-0.0043 * Comp_Sc_Undergrad_Courses - 0.002 * Comp_Sc_Grad_Courses -0.0023* Soft_Engr_Grad_Courses+0.0018* Tech_Grad_Courses+0.0042 * Hardware_Conferences + 0.0042 * MIS_Workshops- 0.0144 * Proj_Mgmt_Conferences + 0.0084 * Soft_Engr_Conferences+0.0025*Database_Experience - 0.0025 * Domain_Exp + 0.4153
16	ABS((TotalEstimates-TotalActual)/TotalActual) = + 1.7901

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 [16] 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.

Table 9 includes several equations calculated by the M5P prediction algorithm where each equation includes one or more cost parameters described earlier or exists in cost estimation datasets with numbers or factors.

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

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

[1] SHAHID IQBAL, M. NAEEM AHMED KHAN, Yet another Set of Requirement Metrics for Software Projects. *International Journal of Software Engineering and Its Applications*, 6(1) (2012). http://www.sersc.org/journals/IJSEIA/vol6_no1_2012/2.pdf

[2] B. W. BOEHM, *Software Engineering Economics*. Prentice Hall, New Jersey, 1981.

[3] B. W. BOEHM, *Software Cost Estimation with COCOMO II*. Prentice Hall, New Jersey, 2000.

[4] http://dec.bmth.ac.uk/ESERG/ANGEL/ Accessed Aug 2012.

[5] T. MENZIES, D. PORT, Z. CHEN, J. HIHN, S. STUKES, Validation Methods for Calibrating Software Effort Models. *Proceedings of the ICSE*, (2005), http://menzies.us/pdf/04coconut.pdf. 587-595

[6] Y. F. LI, M. XIE, T. N. GOH, A Study of Mutual Information Based Feature Selection for Case Based Reasoning in Software Cost Estimation. *Expert Systems with Applications*, 36(3) (2009), 5921–5931. http://www.sciencedirect.com/science/article/pii/S0957417408004429

[7] M. AZZEH, D. NEAGU, P. COWLING, Improving Analogy Software Effort Estimation using Fuzzy Feature Subset Selection Algorithm. *Proceedings of the 4th International Workshop on Predictor Models in Software Engineering*, (2008), 71–78. http://scim.brad.ac.uk/staff/pdf/picowlin/ICSE2008.pdf

[8] Z. KHALIFELUA, F. GHAREHCHOPOGH, Comparison and evaluation of data mining techniques with algorithmic models in software cost estimation. *Procedia Technology*, 1 (2012), 65–71.

[9] N. MITTAS, L. ANGELIS, Visual comparison of software cost estimation models by regression error characteristic analysis. *Journal of Systems and Software*, 83(4) (2010), 621–637.

[10] Y-S. SEO, D-H. BAE, R. JEFFERY, AREION: Software effort estimation based on multiple regressions with adaptive recursive data partitioning. *Information and Software Technology*, In press (2013) http://www.sciencedirect.com/science/article/pii/S0950584913000803

[11] S. CHOI, S. PARK, V. SUGUMARAN, A rule-based approach for estimating software development cost using function point and goal and scenario based requirements. *Expert Systems with Applications*, 39 (2012), 406–418.

[12] T. BENALA, S. DEHURI, R. MALL, Computational Intelligence in Software Cost Estimation: An Emerging Paradigm. *ACM SIGSOFT Software Engineering Notes*, 37(3) (2012), 1–7.

[13] J. Z. LI, G. RUHE, A. AL-EMRAN, M. M. RITCHER, A Flexible Method for Effort Estimation by Analogy. *Empirical Software Engineering*, 12(1) (2007), 65–106.

[14] T. MENZIES, Z. CHEN, J. HIHN, K. LUM, Selecting Best Practices for Effort Estimation. *IEEE Trans. Softw. Eng.*, 32, 11 (November 2006), 883–895. http://menzies.us/pdf/06coseekmo.pdf

[15] N. RAMASUBBU, R. BALAN, Overcoming the Challenges in Cost Estimation for Distributed Software Projects. *Proceedings of the International Conference on Software Engineering ICSE*, (2012) Zurich, Switzerland.

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

[17] CH. F. KEMERER, An empirical validation of software cost estimation models. *Communications of the ACM*, 30(5) (May 1987), 416–429.

[18] M. JØRGENSEN, M. J. SHEPPERD, A Systematic Review of Software Development Cost Estimation Studies. *IEEE Trans. Software Eng.*, 33(1) (2007), 33–53.

[19] B. BOEHM, CH. ABTS, S. CHULANI, Software development cost estimation approaches. *A survey, Annals of Software Engineering*, 10 (2000), 177–205.

[20] L. C. BRIAND, KH. EL EMAM, D. SURMANN, I. WIECZOREK, K. MAXWELL, An Assessment and Comparison of Common Software Cost Estimation Modeling Techniques. *ICSE*, (1999), 313–322.

[21] S. D. CONTE, H. E. DUNSMORE, V. Y. SHEN, *Software engineering metrics and models*. Benjamin-Cummings Publishing, 1986.

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.