Università di Pisa

Dipartimento di Informatica

Technical Report : TR-96-22

Software Cost Estimation: an experimental study of model performances

M. Boraso, C. Montangero, H. Sedehi

June 06, 1996

ADDR: Corso Italia 40,56125 Pisa,Italy. TEL: +39-50-887111. FAX: +39-50-887226

Software Cost Estimation: an experimental study of model performances.

M. Boraso, C. Montangero
Dipartimento di Informatica, Università di Pisa, Italy

H. Sedehi
HELP SpA, Roma, Italy

Abstract

The accurate prediction of software development costs may have a large economic impact. As a consequence, considerable research attention is now directed to understand better the software development process. The objective of this paper is to provide an experimental evaluation of the applicability, universality and accuracy of some algorithmic software cost estimating models (COCOMO, TUCOMO, PUTNAM, COPMO, ESSE and Function Points). Data on nine Italian MIS (Management Information Systems) projects were collected and used to evaluate the performance of the models. The evaluation of the estimates was based on the Mean Magnitude Relative Error ($\overline{\text{MRE}}$) and Prediction at level 25% (PRED(25%)) criteria. Results indicated that the models provided interesting performances, better ˝if recalibrated with local data.

***Index Terms*: Software cost models, Function Points, Metrics-performance measures, Effort estimates.**

1. Introduction

The accurate prediction of software development costs may have a large economic impact: in fact, some 60% of large projects significantly overrun their estimates (with an error percentage that can vary from 100% to 200%) and 15% of the software projects are never completed due to the gross misestimation of development effort [16, 18]. Software estimation is difficult for many reasons including [1, 3, 11]: *highly complex process*, i.e. many variables with a high interaction between them;

premature estimate requests, i.e. difficulty in estimating what we don't understand (and we presume to know); *need to be calibrate*, i.e. all estimation techniques require recalibration to local environments; *dynamic evolution environment*, i.e. software projects " live " in not static ˝conditions.

Moreover, project schedules and costs are considered to be important quality factors according to Capers Jones [13] in his attempt to provide a precise definition of quality: "software quality means being on time, within budget, and meeting user needs". As a consequence, considerable research attention is now directed at gaining a better understanding of the software development process as well as constructing and evaluating software cost estimation ˝tools.

The objective of this paper is to provide an experimental evaluation of the applicability, universality and accuracy of some algorithmic software cost estimating models when used in environments that are different from those in which they were developed.

The experiments are made comparing the ex-post estimates, provided by cost models, with the actual costs collected in a project database containing data on nine Italian MIS (Management Information System) projects.

Section 2 discusses the selection of software cost estimating models and gives a brief description of these. Section 3 describes the research design, the data collection procedure, the project database, and the methodology used to evaluate the estimating models. Section 4 reports and interprets the results obtained applying the models. Finally, Section 5 provides ˝some summary conclusions.

2. Background

According to Boehm, there are three main methodology categories to approach the problem of software cost estimation [5]:

a) Expert consulting, which consists in consulting one or more experts in order to obtain a final evaluation through "expert consensus" techniques such ˝as, for example, the "Delphi Technique";

b) Estimation by analogy, which requires a new project cost estimation be made by analogy with past projects sharing similar characteristics;

c) Algorithmic modeling, which requires to use one or more procedural techniques to obtain a cost estimation as a function of a certain number of variables, strictly related to software costs, and called "cost drivers".

Algorithmic models have the advantage of giving an objective estimation, because they are not affected by human factors such as desire of creating a good impression, ˝or a particular estimators' mood.

Conte, Dunsmore and Shen [9] suggest a possible classification of the algorithmic models, grouping them into categories, called methodologies, depending on the method used in deriving the model. The most important methodologies are:

STATISTICALLY-BASED: regression analysis is often used to determine the relationship between development effort[1] and other parameters such as project size.

THEORETICALLY BASED: They are based on theories and mathematical laws that the software development is assumed to follow.

COMPOSITE: Incorporate a combination of analytic equations, statistical data and expert judgement.

We have selected a group of the most popular estimation models from the literature, for each algorithmic methodology, as shown in Table I. The interested reader is referred to the references provided in Table I for detailed explanations: here, we only comment our choice of ˝the parameters used in the experiment.

TABLE I
MODEL CATEGORIES AND LITERATURE REFERENCES

ALGORITHMIC METHODOLOGY	NAME	MODEL EQUATION
Statistically-Based	FPPROD [12]	$\hat{E} = FP / p$
	FPREGR [2, 4]	$\hat{E} = a + b\ FP$
Theoretically-Based	PUTNAM [9, 10, 19]	$\hat{E} = 1 / T^4\ (S / c)^3$
Composite	COCINT [5]	$\hat{E} = d_i\ S^{e_i}\ m(X)$
	TUCOMO [17]	$\hat{E} = f_i\ S^{g_i}$
	COPMO [9]	$\hat{E} = u_i\ S + v_i\ \overline{P}^{\,d}$
	ESSE [8, 20]	proprietary

[1] Following Boehm [5], we measure the effort in man months ˝which correspond to 152 man-hours.

Legend:

$\hat{E}$	≡ "number of estimated man months (MM)"
FP	≡ "number of Function Points"
p	≡ "Productivity index expressed in FP/MM"
a, b	≡ "constants"
c	≡ "technology factor that reflects the effect on productivity of numerous cost drivers. c can be expressed by the following formula: $c = c^1 \Pi$, where Π is the product of the cost drivers"
u_i, v_i	≡ "technology factor that are believed to affect ˝productivity"
T	≡ "total project development time measured in months"
S	≡ "Thousands of Delivered Source Instructions (KDSI). Boehm [5] defines DSI as program instructions created by project personnel that are delivered as part of the final product. They exclude comments, unmodified utility software and include job control language format statements and data declarations"
d_i, e_i	≡ "constants which change with the development mode (Organic, Semidetached or Embedded) and level (Basic, Intermediate and Detailed)"
m(X)	≡ "composite multiplier depending on 15 cost drivers which take care of the product, computer, personnel and project attributes"
f_i, g_i	≡ "constants which change with the development mode ˝(Organic or Embedded)"
$\overline{P}$	≡ "average full-time equivalent team size"

2.1 FPPROD (Function Points - Productivity)

In this paper we have used a productivity index equal to ˝17.48 FP/MM [12] obtained from a database of 112 MIS projects developed in approximately 25 Australian organisations over the period from 1983 to 1987, using primarily COBOL and PL/1 as implementation languages, ˝with code generators in some companies.

2.2 FPREGR (Function Points - Linear Regression)

This statistically model is based on the correlation ˝between effort and Function Points. In this paper we have set parameters *a* and *b* to 3.96 and 0.06, obtained by B. Kitchenham and K. Känsälä [15] applying linear regression techniques to a database containing 40 MIS ˝projects developed by 9 Finnish organisations.

2.3 PUTNAM

The PUTNAM estimating model is based on the hypothesis that manpower utilisation during software development follows the Rayleigh curve. The calibration of the model by the constant c^1 (see legend of Table I) has been made using the same calibrating database of the COCOMO model and applying least squares regression to these projects. We have also assumed that Π is the product of the cost drivers considered by Boehm, due to the lack of a complete ˝formalization of the model.

2.4 COCINT (Intermediate - COCOMO)

The COnstructive COst MOdel (COCOMO) is the most comprehensive and thoroughly documented model in the literature.

The performance analysis of the COCOMO model will be made on the Intermediate level, which differs from the Basic level by the absence of the cost drivers (m(X)= 1), and from the Detailed level by that the project is divided into four phases: Product Design, ˝Detailed Design, Coding & Unit Test, and Integration & Test.

2.5 TUCOMO

TUCOMO (TUnisian COst MOdel) is based on the Basic-COCOMO model, recalibrated using a project database containing 47 Tunisian MIS projects [17]. Because of the recalibration the constants f_i and g_i have been set to 2.74, 0.76 for the Organic Mode and 6.85, 0.66 ˝for the Embedded Mode.

2.6 ESSE

ESSE (Expert System for Software cost Estimation) is a proprietary tool developed by HELP SpA. Since it is a proprietary model, details are not available. However it's an expert system that combines the following software cost estimation knowledge [8, 20]:

a) Consolidated algorithmic models: ESSE is conceptually based on a Function Point-like metric used as front-end to the COCOMO model.

b) Experimental heuristics, i.e. the universally accepted rules set up by the software cost estimation experts which compensate the conceptual lack of ˝algorithmic models.

c) Applicative heuristics, i.e. all those personalised rules, recalibrated to local environment, whose validity is limited to a restricted environment and which are necessary to detect all the exceptions which may seriously alter the estimation ˝process.

2.7 COPMO

In order to suggest a model that supports Brooks' law (as the manager of the colossal OS/360 project, Brooks proposed a law based on his experience [7]: "*adding manpower to a late software project makes it later* ") COPMO (COoperative Programming MOdel) includes metrics concerning project size and staff size.

The determination of the coefficients b_i and c_i is based on various software attributes that are believed to affect productivity, for example the 15 cost drivers of the COCOMO model as suggested in [9]. The constant d has been assumed to be 1.5, close to the interaction complexity order between $\overline{P}$ persons ($O(\overline{P}^2)$).

3. Research Method

3.1 Research design

The key issue we investigate is the accuracy of algorithmic software cost models. This study focuses on comparing actual efforts with the ex-post estimates ˝obtained from the seven models. Our attempt to estimate effort proceeds in two steps. First we convert the total Function Point size to a size in KDSI (thousands of Delivered Source Instructions) for an implementation in a specific language by using an expansion factor for that language (C. Jones has published a conversion table from Function Points to lines of code in selected language [13]). This is done for those models that require KDSI as input size (COCOMO, TUCOMO, COPMO and PUTNAM). Function Points have been chosen as software size metrics because they are attractive for sizing systems written in fourth-generation languages since other measures (such as lines of code) are difficult to use directly [21] and because Function Point can be calculated in the early stages of development. The second step makes effort estimates applying the seven cost models to each project using the Function Points to KDSI conversion as front-end together with other parameters (such as composite multiplier, team size, etc.), when requested.

3.2 Data collection procedure

The data on the projects have been collected through a questionnaire[2] designed to capture both background information (e.g., software environment, hardware environment, personnel attributes, etc.) and project

[2] Readers interested to have the questionnaire are referred to ˝[6].

numbers (e.g., man months requested to develop the software application, Schedule time, number of persons, etc.).

The questions are "project oriented" and not "estimation model oriented", so that the collected information can be applied to all estimation models.

The structure of the questionnaire can be described as the ˝following:

i. *Generality* (e.g., application typology, short description, start date ˝of the project, end date of the project, etc.);

ii. *Conventions* (DSI counting conventions, man month definitions, software ˝life cycle, etc.);

iii. *Project information* (work group organisation, project structure, priori cost estimation approach, priori cost estimates, actual costs, technical characteristics, environmental characteristics, etc.).

For each project, there was an in-person meeting with the project manager who filled out the questionnaire. The one exception to this procedure was organisation "δ" whose data collection procedure was done by mail and telephone. This labour intensive approach has ˝permitted:

— to discuss each of the questions to ensure that they were understood and that each manager answered consistently;

— to impress upon the managers the importance of their ˝participation in this work;

— to ensure the truthfulness of the collected data.

3.3 Project database

Table II shows summary data for the projects involved in the study that reflect a wide range of data-processing or MIS (Management Information System) ˝applications, such as banking and finance, insurance, telecommunication and motor way traffic. For all data the mean project size is 1255.53 Function Points. The average productivity is 18 Function Points/man month, very close to 17.5 reported in the study conducted by D.R. Jeffery and G. Law [12] in a similar environment. The implementation languages used were primary 4GL (TELON/COBOL and S-COBOL) and COBOL, with Hardware environment based on

mainframe, except for the project ε_1, which has been developed in a PC-based environment using DBIII as implementation language.

TABLE II
PROJECT BACKGROUND

Prog.	Beginning year	Development mode (3)	Target hardware	Language	Π	ACTUALS E	T	$\overline{P}$	FP	KDSI	FP/MM
α1	1987	Semidetached	IBM 3090+PC	S-COBOL/COBOL+SQL	0,45	187,00	22,00	8,50	5187,83	N/A	27,74
α2 (1)	1990	Semidetached	IBM 3090+PC	S-COBOL/COBOL+SQL	0,65	161,00	41,00	3,93	2829,64	N/A	17,58
α3 (2)	1992	Semidetached	IBM 3090+PC	Gupta/SQL Windows + S-COBOL/COBOL	0,57	46,00	18,00	2,56	991,82	N/A	21,56
β1	1990	Semidetached	IBM 3090	COBOL	0,41	100,50	21,87	4,60	606,98	65,7	6,04
β2	1992	Organic	IBM 3090	COBOL	0,23	12,40	6,67	1,86	241,08	14,4	19,44
χ1	1992	Semidetached	Mainframe	COBOL	0,31	29,50	11,10	2,66	515,06	59,7	17,46
δ1	1992	Semidetached	Mainframe	TELON/COBOL	0,57	19,80	7,50	2,64	350,00	N/A	17,68
δ2	1991	Semidetached	Mainframe	TELON/COBOL	0,86	21,30	7,40	2,88	411,84	N/A	19,34
ε1	1992	Organic	PC	DBIII	0,40	11,00	6,50	1,69	165,48	28,8	15,04
Media					0,49	65,39	15,78	3,48	1255,53		17,99
Std. Dev.					0,20	72,32	12,23	2,24	1801,92		6,14

(1) Is inclusive of the times for the contract of the implementation phases.
(2) Is inclusive of the feasibility study and installation & acceptance test phases.
(3) Only for COCINT, TUCOMO and ESSE.

Legend:

Π = composite multiplier depending on the 15 cost drivers of the COCOMO model.
E = actual Effort measured in man months
T = total develpment time measured in months
$\overline{P}$ = average full-time equivalent team size
FP = size measured in Function Points
FP/MM = productivity measured in Function Points/man month
N/A = not available

3.4 Evaluation criteria

To evaluate the various estimation models we have used two criteria: accuracy and accuracy of the recalibrated model.

The accuracy of a project estimate can be measured by the ˝Magnitude Relative Error (MRE):

$$MRE = \left|RE\right| = \left|\frac{\hat{E} - E}{E}\right|$$

where $\hat{E}$ is the estimated effort and E the actual effort. For a set of projects the accuracy can be evaluated using the average MRE (denoted with $\overline{MRE}$). In addition, a second measure can be used to examine the cumulative frequency of MRE for a specific error level. This measure is called PRED(r) or Prediction at level r. If k is the number of projects in a set of n projects whose MRE ≤ r, then PRED(r) = k/n. According to Conte, Dunsmore and Shen [9], a value of PRED(25%) ≥ 75% and $\overline{MRE}$ ≤ 25% are considered desirable for effort models.

The accuracy of a method is important because it affects ˝project scheduling and staffing decisions. However, an inaccurate model may be able to be recalibrated in a particular environment and offer a better performance. In fact the criteria that define accuracy do ˝not take care of the following aspects:

— Difference between the calibrating and the validating ˝environment;

— Arbitrary omission of one or more of the most important cost drivers in the software cost estimation model;

— Misinterpretation of the cost drivers;

So, we define the accuracy of the recalibrated model to be the Mean Magnitude Relative Error of the Recalibrated Model (denoted with $\overline{MRE}_{RM}$), assuming the recalibration set to be the validating set of projects (α_1, ..., ε_1). The recalibrated model is based on the equation:

$$\hat{\hat{E}} = a + b\,\hat{E}$$

where $\hat{\hat{E}}$ is the effort estimated by the recalibrated model, $\hat{E}$ is ˝the effort estimated by the model, *a* and *b* are two constants that minimise the square relative error sum ˝for a set of n projects:

$$S = \sum_{i=1}^{n}\left(\frac{a + b\,\hat{E}_i - E_i}{E_i}\right)$$

Consequently, MRE_{RM} is defined by the following formula:

$$MRE_{RM} = \left|RE_{RM}\right| = \left|\frac{\hat{\hat{E}} - E}{E}\right|$$

For a set of projects the MRE_{RM} of a model can be evaluated using the average MRE_{RM} (denoted by $\overline{MRE}_{RM}$).

Analogously to the PRED(r), we define the prediction at ˝level *r* as:

$$PRED_{RM}(r) = H/n$$

where $H = \#\{i \mid MRE_{RM_i} \le r,\ i \in \{1, .., n\}\}$.

4. Results and Interpretations

In this section we present the results of our effort analysis from nine Italian MIS projects. Table III shows the performances of the cost models for each project analysed in terms of the Relative Error (RE) and Prediction at level 25% (PRED(25%)).

From this table we observe that the model that offers the best performances is ESSE ($\overline{MRE}$ = 13.35%, PRED(25%)= 88.89%) and the worst is PUTNAM ($\overline{MRE}$ = 220.84%, PRED(25%)= 11.11%).

TABLE III
MODEL PERFORMANCES

Project	α1		α2		α3		β1		β2		χ1		δ1		δ2		ε1			
Actual Effort	187,00		161,00		46,00		100,50		12,40		29,50		19,80		21,30		11,00			
	Ê	RE %	Ê	RE %	Ê	RE %	Ê	RE %	Ê	RE %	Ê	RE %	Ê	RE %	Ê	RE %	Ê	RE %	$\overline{MRE}$	PRED(25%)
COCINT	201,20	7,59	147,50	-8,39	39,90	-13,26	128,90	28,26	22,20	79,03	81,80	177,29	12,50	-36,87	22,40	5,16	9,20	-16,36	41,36	55,56
TUCOMO	120,40	-35,61	80,70	-49,88	40,40	-12,17	97,90	-2,59	29,40	137,10	87,90	197,97	20,30	2,53	22,60	6,10	10,60	-3,64	49,73	55,56
ESSE	242,40	29,63	179,12	11,25	44,87	-2,47	106,06	5,53	14,39	16,04	36,50	23,73	24,28	22,63	22,80	7,04	10,80	-1,82	13,35	88,89
COPMO	113,22	-39,46	97,20	-39,63	36,10	-21,52	77,00	-23,38	29,60	138,71	62,50	111,86	17,00	-14,14	20,50	-3,76	8,80	-20,00	45,83	55,56
PUTNAM	507,47	171,37	152,50	-5,28	62,50	35,87	272,20	170,85	106,20	756,45	187,30	534,92	34,10	72,22	31,70	48,83	32,10	191,82	220,84	11,11
FPREGR	315,91	68,94	174,11	8,14	63,60	38,25	40,46	-59,75	18,45	48,82	34,93	18,40	25,00	26,28	28,72	34,84	13,91	26,43	36,65	22,22
FPPROD	296,79	58,71	161,90	0,56	56,70	23,26	34,70	-65,47	13,80	11,29	29,50	0,00	20,00	1,01	23,60	10,80	9,50	-13,64	20,53	77,78

Analysing in a more detail the data reported in Table III, we notice that, for a given model, some projects are grossly misestimated respect to the others. For example from Figure 2 (a), displaying graphically the RE for each cost model, we can see that there is a project that is abundantly overestimated by COCINT. This project is χ_1 and can be individuate using Figure 2 (b) which displays the RE of the estimates produced by each cost model for each project.

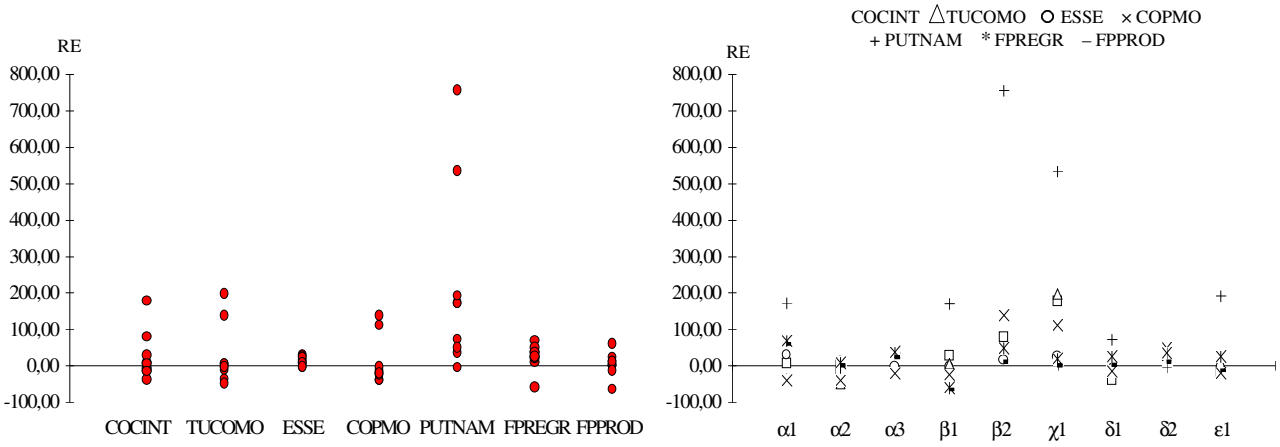

Fig. 1. Analysis of effort estimate error. (a) Software Cost Estimation Models versus Relative Error distribution. (b) Projects versus Relative Error distribution.

So visual inspection of Figure 1 and intuition tell us that β_2 and χ_1 projects can be eliminated from consideration. Figure 2 is an integration of Table III showing graphically the relationship $\hat{E}$ (effort estimate) versus E (actual effort) for each model, and indicating β_2 and χ_1 projects.

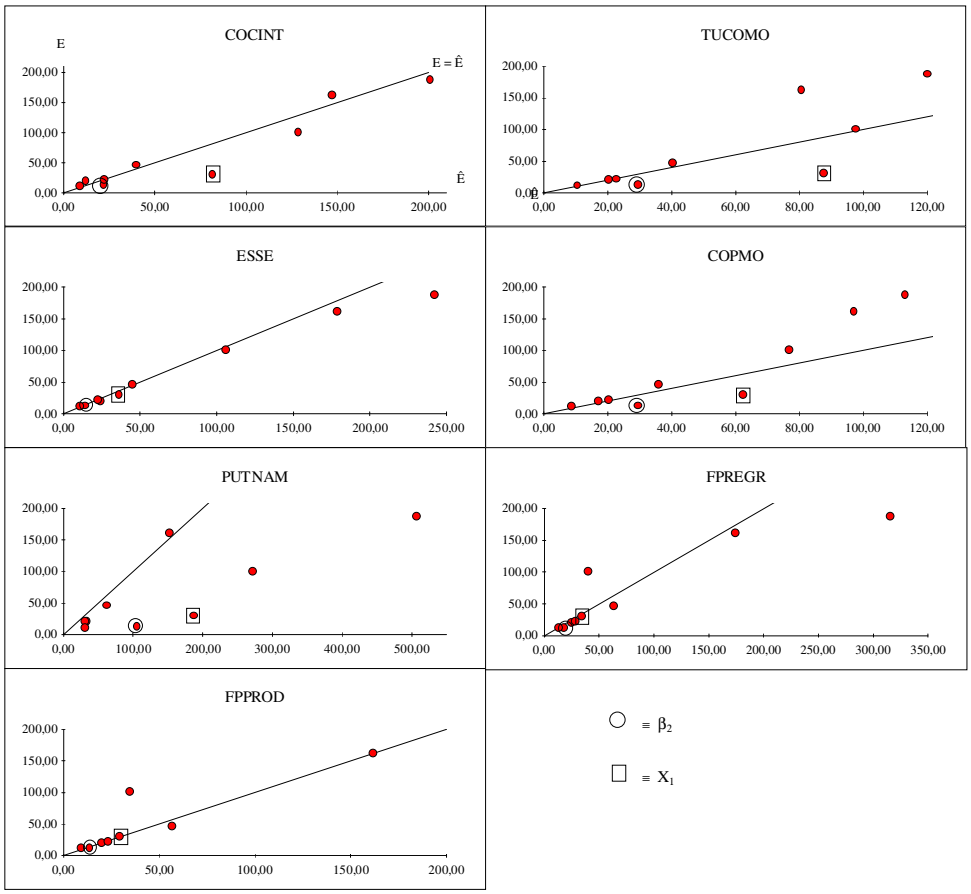

Fig. 2. Effort Estimates ($\hat{E}$) versus Actuals (E).

Table IV reports the new model performances excluding projects β_2 and χ_1, which have been abundantly misestimated by the majority of the models (COCINT, TUCOMO, COPMO and PUTNAM). The $\overline{MRE}_{RM}$ and $PRED_{RM}(25\%)$ columns indicate the performances of the recalibrated models as already discussed in the evaluation criteria section.

TABLE IV
MODEL PERFORMANCES EXCLUDING β_2 AND χ_1 PROJECTS

	α1		α2		α3		β1		δ1		δ2		ε1					
	RE %	RE_{RM} %	RE %	RE_{RM} %	RE %	RE_{RM} %	RE %	RE_{RM} %	RE %	RE_{RM} %	RE %	RE_{RM} %	RE %	RE_{RM} %	$\overline{MRE}$	$\overline{MRE}_{RM}$	PRED(25%)	$PRED_{RM}(25\%)$
COCINT	7,59	-1,96	-8,39	-15,88	-13,26	-14,21	28,26	18,25	-36,87	-24,13	5,16	11,99	-16,36	9,82	16,56	13,75	71,43	100,00
TUCOMO	-35,61	-21,76	-49,88	-39,81	-12,17	1,67	-2,59	17,72	2,53	9,9	6,10	15,6	-3,64	-11,89	16,07	16,91	71,43	85,71
ESSE	29,63	11,58	11,25	-3,99	-2,47	-13,38	5,53	-8,32	22,63	12,39	7,04	-1,46	-1,82	-2,43	11,48	7,65	85,71	100,00
COPMO	-39,46	-17	-39,63	-17,47	-21,52	3,51	-23,38	4,16	-14,14	5,89	-3,76	21,36	-20,00	-13,41	23,13	11,83	71,43	100,00
PUTNAM	171,37	14,32	-5,28	-59,36	35,87	-39,55	170,85	14,88	72,22	-19,5	48,83	-29,88	191,82	37,3	99,46	30,68	14,29	42,86
FPREGR	68,94	21,64	8,14	-21,66	38,25	2,48	-59,75	-69,55	26,28	-1,23	34,84	4,29	26,43	5,69	37,52	18,08	14,29	85,71
FPPROD	58,71	21,68	0,56	-21,64	23,26	2,4	-65,47	-69,58	1,01	-1,39	10,80	4,37	-13,64	5,76	24,78	18,12	71,43	85,71

Figure 3 reports graphically the global performances, based on the $\overline{MRE}$, PRED(25%) and $\overline{MRE}_{RM}$, $PRED_{RM}(25\%)$ from which we can elicitate a cost model classification, although the project database is small in order to assure what researchers call a robust ˝result.

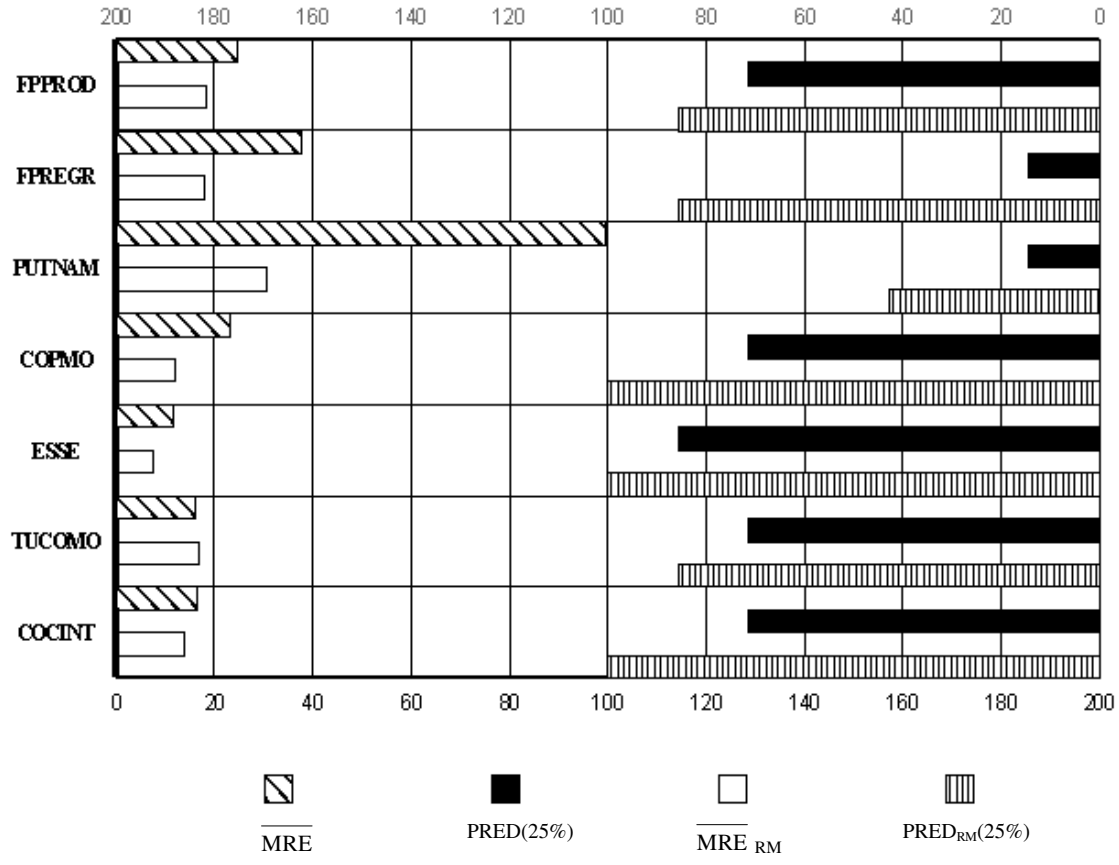

Fig. 3. Model performances diagram in terms of $\overline{MRE}$, PRED(25%) and $\overline{MRE}_{RM}$, $PRED_{RM}$ (25%).

5. Conclusions

We now analyse and comment the performance of the various models, obviously excluding project β_2 and χ_1. Of course it is important to bear in mind that the results stem from a sample of nine projects[3], so that the conclusions reported here need to be verified with more ″experiments.

COCINT does well via the $\overline{MRE}$ indicator (equal to 16,56%) and quite well by the PRED(25%) (equal to 71,43%). However the results seems to indicate that the model is recalibration sensitive ($\overline{MRE}_{RM}$= 13,75% and $PRED_{RM}(25\%)$= 100%).

TUCOMO performs quite well, bearing in mind that it's a simple recalibration of the Basic-COCOMO model.

ESSE provides very good estimates ($\overline{MRE}$= 11,48% and PRED(25%)= 85,71%) validating the approach that uses the software cost estimation heuristics proper to experts and consequently improving the performances of the COCOMO cost model. The results even indicate the recalibration sensitivity of the model.

COPMO performances are quite similar to those offered by COCINT, validating the relationship between effort, size and manpower, and consequently ″Brooks' law.

PUTNAM offers insufficient performances in the $\overline{MRE}$ and PRED(25%) tests. A possible explanation for this (according to [14]) is that the model was originally developed using data from US defense-related projects, including real-time systems. Productivity for these is typically less than that in the business data-processing systems developed by α,..., ε. Anyway, the performances can be improved by a recalibration, as indicated by the $\overline{MRE}_{RM}$ and $PRED_{RM}(25\%)$.

FPPREGR does not do well in the $\overline{MRE}$ and PRED(25%) tests. A possible explanation lies in the differences between the environment in which the 40 MIS Finnish projects were developed and the

[3] We would like to underline that we faced a lot of difficulty to take into account projects data acquisition. So we preferred to examine less projects with highly accurate information analysis rather ″than a lot of projects with a low level study.

α,..., ε environment. However the model has a high degree of recalibration sensitive, as indicated by the $\overline{\text{MRE}}_{\text{RM}}$ and $\text{PRED}_{\text{RM}}(25\%)$.

FPPROD, the model based on Function Points and productivity, produces good performances (including β_1 and χ_1 projects) due to the productivity index measured by Jeffery [12] in a similar MIS environment and to the proved high correlation between effort and Function Points. Obviously this model is recalibration sensitive.

References

[1] T.K. Abdel-Hamid, "Adapting, Correcting and Perfecting Software Estimates: A Maintenance Metaphor," *IEEE Computer*, pp. 20–29, March 1993.

[2] A.J. Albrecht and J.E. Gaffney, "Software Function, Source Lines of Code, and Development Effort Prediction: A Software Science Validation," *IEEE Trans. Software Eng.*, vol. SE-9, pp. 639–648, Nov. 1983.

[3] J.W. Bailey and V. R. Basili, "A meta-model for Software development resource expenditures," in *Proc. 5th Int. Conf. Software Eng.*, pp. 107–116, 1981.

[4] C. A. Behrens, "Measuring the Productivity of Computer Systems Development Activities with Function Points," *IEEE Trans. Software Eng.*, vol. SE-9, pp. 648–652, Nov. 1983.

[5] B.W. Boehm, *Software Engineering Economics*. Englewood Cliffs, NJ: Prentice-Hall, 1981.

[6] M. Boraso, *Analisi e Verifica Sperimentale delle Metodologie per la Stima del Costo del Software*. Tesi di Laurea, Corso di Laurea in Scienze dell'Informazione, Università di Pisa, Ott. 1994 (in Italian).

[7] F. P. Brooks, "The Mythical Man-Month," *Datamation*, pp. 44–52, Dec. 1974.

[8] A. Chojwa, *Un Sistema Esperto per la Stima dei Costi del Software*. Tesi di Laurea, Corso di Laurea in Scienze dell'Informazione, Università di ˝Pisa, Nov. 1991 (in Italian).

[9] S. D. Conte, H.E Dunsmore and V.Y. Shen, *Software Engineering Metrics and Models*. Melno Park, CA: Benjamin Cummings, 1986.

[10] T. DeMarco, *Controlling Software Projects*. New York, Yourdon Press, 1982.

[11] R. E. Fairley, "Recent Advances in Software Estimation Techniques," in *Proc. IEEE 14th Int. Conf. Software Eng.*, pp. 382–391, 1992.

[12] D.R Jeffery and G. Low, "Calibrating Estimation Tools for Software Development," *Software Engineering Journal*, pp. 215–221, July 1990.

[13] C. Jones, *Applied Software Measurement: Assuring Productivity and Quality*. McGraw-Hill, 1991

[14] C. F. Kemerer, "An Empirical Validation of Software Cost Estimation Models," *Commun. ACM*, vol. 30, pp. 416–429, May 1987.

[15] B. Kitchenham and K. Känsälä, "Inter-item Correlations among Function Points," in *Proc. IEEE 15th Int. Conf. Software Eng.*, pp. 477–480, 1993.

[16] A. L. Lederer and J. Prasad, "Nine Management Guideline for Better Cost Estimating," *Commun. ACM*, vol. 35, pp. 51–59, Feb. 1992.

[17] R. Marouane and A. Mili, "Economics of Software project management in Tunisia: Basic Tucomo," *Information and Software Technology*, vol. 31, no. 5, pp. 251–257, June 1989.

[18] T. Mukhopadhyay and S. Kekre, "Software Effort Models for Early Estimation of Process Control Applications," *IEEE Trans. Software Eng.*, vol. 18, pp. 915–923, Oct. 1993.

[19] L.H. Putnam, "A General Empirical Solution to the Macro Software Sizing and Estimating Problem," *IEEE Trans. Software Eng.*, vol. SE-4, pp. 345–361, July 1978.

[20] H. Sedehi, U. Biader and S. Capezzone, "An Expert System for Software Cost Estimation: ESSE," *EXPERSYS-91*, Technology & Transfer Series, 1991

[21] J. Verner and G. Tate, "Estimating Size and Effort in Fourth-Generation Development," *IEEE Software*, pp. 15–22, July 1988.