KYBERNETIKA — VOLUME 41 (2005), NUMBER 2, PAGES 249 – 264

SOFTWARE COST ESTIMATION WITH FUZZY INPUTS: FUZZY MODELLING AND AGGREGATION OF COST DRIVERS

MIGUEL–ÁNGEL SICILIA, JUAN–J. CUADRADO–GALLEGO, JAVIER CRESPO AND ELENA GARCÍA–BARIOCANAL

Parametric software cost estimation models are well-known and widely used estimation tools, and several fuzzy extensions have been proposed to introduce a explicit handling of imprecision and uncertainty as part of them. Nonetheless, such extensions do not consider two basic facts that affect the inputs of software cost parametric models: cost drivers are often expressed through vague linguistic categories, and in many cases cost drivers are better expressed in terms of aggregations of second-level drivers. In this paper, fuzzy set elicitation techniques are used as a tool to model vague categories expressing cost driver quantities, focusing on two well-known COCOMO cost drivers. The results clearly indicate that such fuzzy set modelling approach affects significantly the estimation outcomes. In addition, the empirical adjustment of the DOCU cost driver as an aggregation of second-level documentation artifact measures is used to illustrate the modelling of flexible aggregation in the context of parametric estimation. Fuzzy set elicitation and aggregation operator modelling combined provide a novel approach to extending fuzzy parametric models for software estimation, which can be used as a complement to existing approaches.

Keywords: software cost estimation, fuzzy set elicitation, aggregation operator design

AMS Subject Classification: 68U35, 68T37, 03B52, 47S40, 28E10

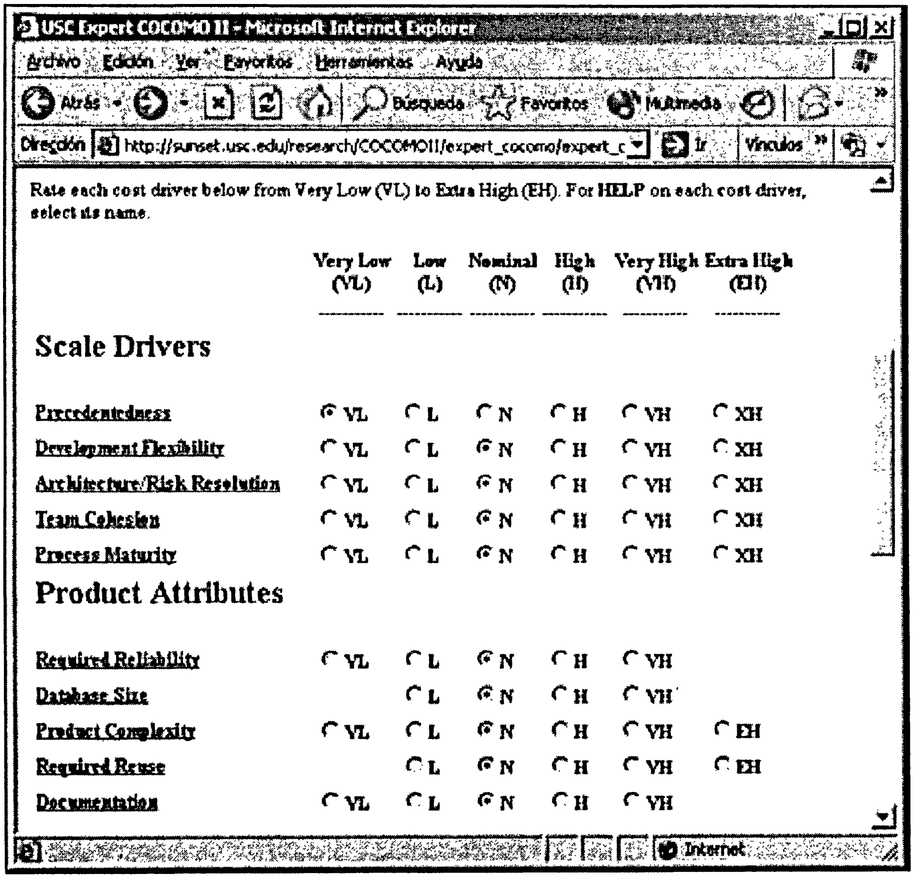
1. INTRODUCTION

Parametric models are one of the principal software cost estimation techniques [3], used both in commercial tools like SLIM[1] and also in open, published models like the different versions of COCOMO [2]. These models essentially use mathematical expressions – obtained usually from conventional curve regression techniques – to derive effort of development estimates from a number of input variables that are often called *cost drivers*. Parametric models are dependent to some extent on the domain of development and vary with the technology used to support the Software Engineering process, so that it is currently accepted that no single model might ever fit every estimation situation [9]. In consequence, the problem of estimation

[1]http://www.qsm.com/

is usually considered as that of obtaining an estimating function useful for a given organization, sector or domain of application.

Existing parametric mathematical models for software estimation have the general form effort $= f(c_1 \ldots c_n)$, where each c_i is a real number, i. e. they simply compute the effort estimated from a number of real inputs that are measures of the cost drivers considered relevant. Even in the case that linguistic labels like "high" or "low" are used in the models, they are simply mapped to concrete, pre-established real numbers that were previously obtained from a process of calibration with empirical data (this is indeed the approach taken in the classical COCOMO model [2]). Nonetheless, cost drivers commonly used as inputs in existing parametric models are intrinsically imprecise, due to diverse reasons [6, 23] that include the fact that in many cases they are expressed in linguistic form. Figure 1 shows an example of how a COCOMO-81 Web tool asks for input in linguistic form.


USC Expert COCOMO II - Microsoft Internet Explorer
Archivo Edición Ver Favoritos Herramientas Ayuda
Atrás Búsqueda Favoritos Multimedia
Dirección http://sunset.usc.edu/research/COCOMOII/expert_cocomo/expert_c
Ir Vínculos
Internet

Rate each cost driver below from Very Low (VL) to Extra High (EH). For HELP on each cost driver, select its name.

	Very Low (VL)	Low (L)	Nominal (N)	High (H)	Very High (VH)	Extra High (EH)
Scale Drivers						
Precedentedness	[✓ VL]	L	N	H	VH	XH
Development Flexibility	VL	L	[✓ N]	H	VH	XH
Architecture/Risk Resolution	VL	L	[✓ N]	H	VH	XH
Team Cohesion	VL	L	[✓ N]	H	VH	XH
Process Maturity	VL	L	[✓ N]	H	VH	XH
Product Attributes						
Required Reliability	VL	L	[✓ N]	H	VH	
Database Size		L	[✓ N]	H	VH	
Product Complexity	VL	L	[✓ N]	H	VH	EH
Required Reuse		L	[✓ N]	H	VH	EH
Documentation	VL	L	[✓ N]	H	VH	

Fig. 1. COCOMO-81 Web tool input form.

Even though COCOMO uses linguistic labels for the expression of cost driver values, they are simple mappings for calibrated, fixed real numbers (often called "rating levels"). In consequence, a naive use of user interfaces like the one depicted in Figure 1 may lead to a misuse of the COCOMO model. It is trivial to find evidence about the fact that different engineers could use different ratings for the same project and cost driver, since most of the cost drivers used are difficult to assess in a precise way, e. g. "required reuse" is a vague requirement that is not connected to any commonly accepted objective measure. Consequently, it can be hypothesized that COCOMO rating levels and other similar estimation models are not necessarily connected to the common understanding of software engineers about the meanings of the linguistic labels used to express the cost driver values (as demonstrated in the study described later in this paper). This leads to an alternative view to rating modelling that begins with the elicitation of the meaning of vague linguistic labels from experts, instead of using calibrated values that are meaningless from the viewpoint of human assessment. Membership function elicitation techniques can be used for that purpose [1].

In addition, many cost drivers considered in estimation techniques are of a highly abstract nature, in the sense that they represent aspects that in turn depend on other factors that are more easily measurable. For example, the required user interface usability can be decomposed in a number of factors [22], and they can be used to build concrete estimation techniques that aggregate those second-level factors considering their characteristics [7]. Classical descriptive frameworks for software quality factors like the one of McCall et al. [15] reflect this compositional scheme in factors and more concrete metrics.

Extending parametric software cost estimation in the two directions just described provides an enhanced estimation framework that can be expressed as in equation (1).

$$\text{effort} = f(c_1, \dots c_m, \widetilde{c}_{m+1}, \cdots \widetilde{c}_n) \quad \text{where } c_i = A_i(c_1^i \dots c_s^i) \tag{1}$$

Expression (1) opens the possibility to combine fuzzy numbers $\widetilde{c}_i$ with crisp real numbers as inputs for the estimation model, and also allows for the expression of some of the cost drivers (those with $s > 1$) to be modelled as the result of a process of aggregation A_i of second-level drivers or factors. Such fuzzy modelling of inputs may be used in combination with fuzzy regression techniques that enable the explicit modelling of input imprecision [6, 13], resulting in a comprehensive account of fuzzy inputs in parametric estimation. The above expression fits into schemes that produce conventional or fuzzy equations, provided that they are able to deal with fuzzy inputs. For example, the *f-regression* method [14] that allows for the representation of fuzzy inputs as *L-type* fuzzy numbers as described by Crespo et al. [6].

Scattered previous research has addressed diverse aspects of fuzziness in software estimation models, including estimation by analogy [12], fuzzy function points [21], generalizing estimation formulas [16], and using fuzzy regression methods to adjust the models [5, 6]. But these approaches neither address the modelling of inputs from empirical data nor consider the empirical adjustment of aggregation schemes from historical databases. Xu et al. [23] make use of clustering and fuzzy rules for determining inputs, but such approach is a replacement for the parametric approach and not an extension, and the experimental account for membership function is not based on direct elicitation.

In this paper, the extensions to parametric estimation expressed in (1) are put into practice in concrete case studies that demonstrate the applicability, benefits and pitfalls of the approach.

The rest of this paper is structured as follows. Section 2 describes the elicitation process of the RELY and AEXP cost drivers. Section 3 provides an empirical method for designing aggregation schemes for second-level cost drivers. Finally, conclusions and future research directions are provided in Section 4.

2. ELICITATION OF FUZZY SOFTWARE ESTIMATION COST DRIVERS

This section describes the result of two elicitation processes for COCOMO cost drivers. RELY and AEXP have been selected as representatives for two different sources of imprecision. While AEXP (which refers to application experience) is intended to reflect "high level human collective knowledge in terms of vague time spans", RELY (required reliability) tries to capture "a required software quality aspect expressed in terms of resemblance to prototypical situations" [6]. In what follows, the design and results of the two case studies are presented and discussed. It should be noted that the results described are not intended to provide a universal account for the ratings, since it is considered that estimation models have only a local or domain-specific validity [9]. Nonetheless, the procedure is valid for local, organization-wide estimation models based in the project baseline of the organization.

2.1. Modelling imprecision in cost driver assessment

The RELY (required reliability) cost driver is defined in COCOMO with the statement: "it reflects the extent that a software product can be expected to perform its intended functions satisfactorily". In addition, indications for estimation of the cost driver are given in the form of examples (presented in Table 1). Consequently, it can be considered that the expert must determine the RELY value by comparison with "prototypical" examples describing prominent or supposedly clear examples. Prototypes of that kind are described as the center of a theory of categorization in existing cognitive psychology studies [19], which considers categories as vague sets organized around prototypical exemplars from which other individuals are considered to lay at a certain "distance". This previous elements suggests the adequacy of studying the cost driver by using such prototypes as the source of value assessment in estimation settings.

Consequently, the first of our attempts for the elicitation of the membership function of RELY labels proceeded by trying to elicit directly such notion of distance, asking experts for an abstract, relative measure of distance. This method was later rejected since it produced a high degree of inconsistency between expert's opinion, both in quantitative terms and in the comments about the usefulness of the method provide by the experts in "thinking aloud" sessions. This raised the need for a well-known and commonly understood measure for reliability. Rate of failure measures were found adequate from that viewpoint [11], as described in what follows.

Table 1. Possible values and examples for the RELY cost driver.

Value	very low	low	nominal	high	very high
Description	slight inconvenience	low, easily recoverable losses	moderate, easily recoverable losses	high financial loss	risk to human life
Example	A demonstration prototype of a voice typewriter. An early feasibility-phase software simulation model.	A long-range planning model. A climate forecasting model.	Management information systems. Inventory control systems.	Banking systems. Electric power distribution systems.	Military command and control systems. Nuclear reactor control systems.

A membership elicitation process [1] was carried out for each of the linguistic values of RELY, in the (decreasing) scale of expected failures per year. Each of the labels was obtained simply by averaging the expert's assessments in the form $\text{RELY}_i = (a_i, b_i, c_i) = \sum_{j=1}^{\#-\text{experts}} \frac{(a_j, b_j, c_j)}{\#-\text{experts}}$, $i = \{VL, L, N, H, VH\}$. Where a_i, b_i, c_i are each one of the aggregated values, a_j, b_j, c_j are the values that for the three variables assigns each one of the experts consulted and *#-experts* is the number of experts consulted.

Ten experts of the same development company took part in the interviews, and the resulting triangular fuzzy numbers are the averaged tolerance intervals given by the experts when confronted to the examples provided in Table 1. Expertise for the study was characterized as a minimum of three years of development experience. The experiment was repeated with a group of twenty experts, with shifts and shape variations that were deemed as acceptable. The resulting triangular functions are provided in Table 2, with the variance for each value put into parenthesis. Variances in the second experiment are systematically lower possibly due to the fact that expert interviews took as input the results of the first phase, which somewhat served as a reference for the assessments.

Table 2. Results of the elicitation experiments for RELY.

Experiment	VL	L	M	H	VH
First	$22.25, 25.75, 29.25$ $(5.3, 6.22, 4.41)$	$13.42, 19.25, 25$ $(2.13, 3.63, 2.52)$	$8.75, 12.71, 16.67$ $(9.8, 6.56, 8.45)$	$0.25, 1.07, 1.75$ $(0.21, 0.32, 0.44)$	$0, 0.63, 1$ $(0, 0.16, 0.13)$
Second	$19.53, 22.43, 28.32$ $(3.2, 2.3, 3.1)$	$11.31, 17.65, 21.2$ $(1.34, 2.3, 2.03)$	$6.32, 8.31, 15.78$ $(4.3, 6.75, 5.89)$	$0.43, 1.3, 1.98$ $(0.12, 0.28, 0.36)$	$0, 0.45, 0.7$ $(0, 0.11, 0.12)$

It should be noted that the procedure used for the obtention of the values should be complemented by the following considerations, which are required to provide the results with a better rationality from the viewpoint of rating assessment:

- The VL and VH functions should be interpreted as "s-shaped" and "z-shaped" functions, respectively, since it can be reasonably considered that values below or above them have full membership in the category, due to the purely quantitative character of the failure-rate measure. This was raised in expert interviews.
- The category of "nominal values" (M) should be considered to cover to some extent the gap to the H and VH requirements for RELY, by increasing the spread of the triangular function, or by considering a trapezoidal function adjusted to cover such gap.
- The gap between the VL and L categories should also be covered by expanding their respective functions.

The overall results for RELY provide a model of the perception of the studied organization regarding required reliability, which can be grossly summarized in that high requirements for reliability are characterized by failure rates per year below two, while more than ten failures per year is considered low or very low reliability. Surprisingly, numbers of failures approaching ten are considered as nominal. In the concrete case study, this was explained by the fact that this number is frequent in the first application year after deployment, as a result of hidden defects. In other cases, the M label should be considered to be shifted to lower numbers of defects, covering the gap between high and low categories.

The AEXP cost driver is dependent on the level of applications experience of the project team developing the software system or subsystem. The ratings are defined in terms of the project team's equivalent level of experience with this type of application. According to COCOMO indications, a "very low" rating should be assigned for application experience of less than two months, a "very high" rating is for a experience of six years or more, and the intermediate labels are approximated by six months, one year and three years. These assumptions can be contrasted by a membership elicitation process.

Figure 2 provides the result of a membership function exemplification process (see [1] for details on this technique). The curves represent the linguistic labels (very-low, low, nominal, high, very-high) from left to right. The procedure for obtaining the functions was that of asking twenty experts to provide "compatibility degrees" with concrete linguistic labels expressed in a [0..100] scale for a number of experience values expressed in months, chosen to cover the domain of 70 months considered. After that, the points obtained were used as input for a standard, straightforward curve regression process, yielding the shapes of the membership functions.

The apparent "anomaly" in nominal is the consequence of a belief that when a developer has reached a certain degree of experience with a given technological context, he/she stops improving his/her level of knowledge, due to the relative degree of self-satisfaction. This is an example of "culture-related" factor that should be isolated or considered as an explicit aspect in more detailed studies.

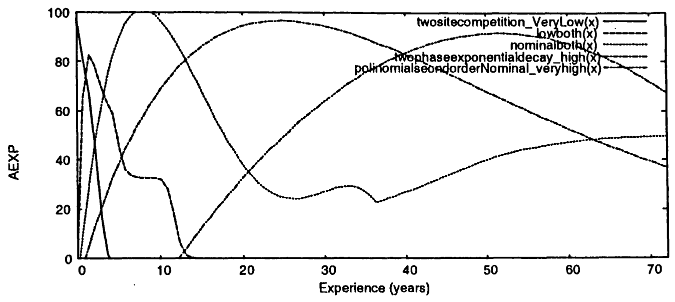


Fig. 2. Fuzzy sets for the AEXP cost driver, expressed in terms of months.

The results of the study should be complemented by the following interpretive considerations:

- The "very low" and "low" labels can be considered as vague generalization of COCOMO sharp levels for AEXP, since they cover these levels and provide fuzzy boundaries to them that can be considered as reasonable generalizations.
- In contrast, the "high" and "very high" categories present divergent shapes, with a high degree of vagueness (i. e. large shape spreads), and a consideration that values significantly below to that of COCOMO actually match the linguistic labels. In addition, they exhibit a degree of decreasing after reaching full membership that is not actually modelled by COCOMO ratings.
- The "nominal" or "medium" category is anomalous in the sense that a high degree of divergence in expert opinions was found. A second round of inquiry evidenced that "medium" as a experience label was difficult to express, and the category could be better expressed through negation of the other categories, e. g. "*not so* high" and the like.

The fact that "very high" resulted in a shape that is not actually covered by the "high" curve comes from the experimental procedure followed, in which the assessment of the labels was done for all the labels at a time, so that "high" and "very high" were perceived as competing categories for classification, as also considered in other recent elicitation studies [18]. It should be noted that elicitation procedures not considering the labels simultaneously could result in some functions "covering" the others.

The two elicitation processes described in this section provide by themselves a sound motivation for further experimental work, since they provide evidence in favor of studies of fuzzy membership functions prior to calibrating ratings from project databases. The following section illustrates also the implications of the use of such ratings in term of variations of estimation outcomes, and introduces second-level cost drivers as a decomposition of higher-level inputs.

2.2. Implications of the experimental account

As the studies described in the previous sub-section point out, the scales used in software parametric estimation models may include varying degrees of imprecision coming from human categorizations. This entails that sensitivity to inputs must be approached from a previous investigation of such degrees of imprecision. As an illustration, let us consider the COCOMO-81 parametric intermediate model, described by the equation $e = 3 \cdot \text{size}^{1.12} \cdot M$, where *size* is a measurement of the system size and it is usually measured in miles of lines of code, and $M = \prod_i c_i$ is the product of the values of the cost drivers considered in COCOMO as adjustment factors.

We can examine "reasonable" variations in input data and the resulting variation in effort to examine the sensitivity of the formula from a fuzzy perspective. Reasonable means in this cases varying a few of the inputs by only an step (above or below) in the rating scale. Since input assessment of cost driver values is often done by humans, reasonable changes provide an account of variations that can be attributed to subjective perception or slightly different beliefs or interpretations.

A reasonable change in RELY can be that of interchanging the "very high" and "high" labels, due to their large degree of overlapping. The situation becomes more complex in the case of AEXP, since categories overlap to larger extent. For example, the "nominal" and "high" categories could be substituted. Table 3 shows the effect of both changes (VH to H for RELY and N to H for AEXP) for different estimated software sizes (eliminating the effect of the rest of the cost drivers).

Table 3. Resulting influence in "reasonable" cost driver substitution.

size	resulting effort variation
10	−50.79038396
100	−725.299949
1000	−9617.067743
10000	−126833.3959

Considering that effort is computed in man-months, the variations have a significative cost impact, and it should be noted that they can not be considered errors but divergent interpretations of the vagueness associated with the description. This clearly points out to the necessity of building parametric models that explicitly deal with such models of the imprecision of each concrete variable and label, in the direction initiated by recent studies [6].

3. ADJUSTING AGGREGATION OPERATORS FOR SECOND-LEVEL COST DRIVERS

The assessment of some input variables used in parametric software estimation may depend on the value of a set of "lower level" or more detailed factors, which somewhat affect such variables. These "second-level" factors may in some cases be heterogeneous, that is, they may reflect very different aspects of the input. For a specific project, each of these factors will require a separate assessment, independent of whether the other factors affect or not the same input variable. In addition, each of these factors will have its own influence on the rating selected for the variable in question and, by extension, on the final estimated values for the project being estimated. If we do not bear this is in mind and we assume that the rating of all the input variables used for the parametric estimation model depends on a single factor, we could be ignoring other factors which, for some variables, might affect its value on a specific project. As a result, we could be erroneously selecting a rating for the current project, which means that the estimations obtained would not be so adequate as could be.

The problem of aggregating second-level factors into first-level drivers thus requires some previous study about the influence of each factor in the overall input. Existing work has used fuzzy measures to model imprecise but known interactions [20], but in other cases, the possible interactions are not known *a priori*, and there is not enough empirical evidence to assess them in a statistically reliable way. Here we deal with the concrete case of the COCOMO DOCU cost driver, which is intended to reflect the amount of documentation generated during the lifecycle of the software. Since documentation artifacts are often prescribed and specified in detail by software development methods, it makes sense to use the different documentation artifacts as second-level factors for DOCU. In what follows, the process of adjusting empirically the aggregation operator of different types of documents is described, resulting in a research method that could be applied to other similar cases of cost driver breakdown.

The following were established as the working objectives for this part of the research:

- To obtain a figure of the proportion of project effort that can be attributed to documentation elaboration, given some empirical evidence.
- To assess the validity of the input rating level selection method proposed by the COCOMO II Post-Architecture estimation model, as compared to the measure obtained as a result of the first objective. In that method, one has to select the correct value of the DOCU first level cost driver, as a broad estimation of the necessary effort to develop the documentation.
- To obtain a concrete aggregation operator design that improves the COCOMO rating selection method.

3.1. Context of the case study

The case study dealt with empirical data obtained from fifty fifth-year Computer Science students in the context of a Software Engineering course. The students were grouped in teams of five to six members, and each team was responsible for developing a software project from its inception to its execution, being the project a common business system. Each team was assigned a tutor responsible for determining the initial system specifications to be developed and for revising the different artifacts that the development team had to elaborate during the project.

During the development of the project, the teams followed the European Space Agency (ESA) standard for software [10], that recommends the elaboration of the following documents: Service proposal document, Software Quality Assurance Plan (SQAP), Software Configuration Management Plan (SCMP), Software Project Management Plan (SPMP), User Requirements Document (URD), Software Verification and Validation Plan (SVVP), Software Requirements Document (SRD), Initial Function Points Document (PFI), User Interface Document (IFAZ), Architecture Design Document (ADD), Detailed Design Document (DDD), Final Function Points Document (PFF), Software Verification Report (SVR), Software Transference Document (STD), Software User Manual (SUM), Audit Document (AUD) y Project History Document (PHD).

Each team member was assigned a role in the project which should not be changed unless one of the members of the group dropped out. The roles recommended were: project manager (only one), analyst, people responsible for configuration, quality, tests, etc. The role of programmer was shared among all of them.

With the explicit aim of analyzing the aspects related to software documentation during the project, the following specific activities were carried out:

- From the beginning, the software system to be developed was defined using a Software Requirements Specification which included the main system functionalities.
- During the project, the software development teams had to fill out an oversight report, thereby assessing the size and the total effort of the activities carried out, and specifically for those activities related to the elaboration of software documentation.

A document on the project history was prepared to reflect the most relevant facts that came about during the project lifecycle.

3.2. Estimating project effort devoted to documentation

The effort dedicated to documentation elaboration ranged between a minimum value of 12.34 % and a maximum value of 34.67 % of the total development effort. The average for the projects studied was 25.47 %. Although an increase in effort is generally observed when the size of the documentation generated grows, in the sample studied there are some instances in which this is not the case. Thus, for example, one of the projects has a documentation size of 534816 characters and the effort that went into its elaboration was 34.06 % of the total. While in another case, the size of the documentation was 606.411 characters and the effort was 28.3 %. This should be due to the fact that the effort put into the documentation generates effort not only because of the quantity generated, but also, because of the quality.

We will consider a generic parametric nominal non linear equation for effort estimation, of the type used by COCOMO II (Post-Architecture): $e_n = a \cdot s^b$, where e_n is the nominal development effort in hours, s is the product size in thousands of lines of code, and a and b are adjustment constants.

To model the influence of documentation-related effort in the global estimates, an adjustment factor d is introduced. The equation obtained then becomes $e_d = a \cdot s^b \cdot d$ where e_d is the development effort in hours. Using both equations the value of d can be calculated.

In the analysis carried out, both e_n and e_d are known for each project, so that the effort dedicated "exclusively" to documentation elaboration is $e'_d = e_d \cdot p$, where p is the percentage of the total effort dedicated only to software documentation. Bearing this in mind and also considering that $e_d = e_n + e'_d$, the sum is the total effort value as the product of $1/(1-p)$ times the nominal effort. As the value for p is known for each project, the coefficient d can be determined from this point.

The above-mentioned coefficient d in the projects studied ranges from 1.53 for the biggest project to 1.13 for the smallest, assuming that the nominal effort of a project in which no documentation is carried out has a value of 1. Considering that the documentation variable takes the nominal value for a project in which the documentation process takes 11 % of the total software development effort as pointed out by existing studies [17], then the coefficient d should range from 1.01 for the smallest and 1.31 for the largest, after adjusting that eleven percent.

Therefore, the effort adjustment factor of a software project, as a result of intensive documentation production, is $e_d = e_n \cdot d$ with d in the range $[1.01, 1.31]$, e_d is the adjusted effort in hours and e_n is the development effort without taking documentation effort into consideration.

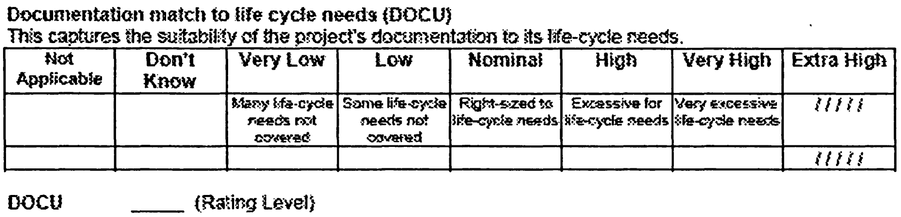
3.3. Assessment of COCOMO rating selection procedure

If the calibration of the 1998 COCOMO II model [4] is considered, it can be observed that for the documentation variable (DOCU), some concrete values for the nominal, high and very high ranges are given, namely (1.00, 1.11, 1.23) which fall within the range of those calculated in our empirical study. According to the table for DOCU rating level selection 3, no other values different except "high" and "very high" should be selected for our case study. This is due to the fact that, according to the context of the projects, the documentation generated in this experiment would be at least high, if not very high.

Taking into account the above comparative results it becomes clear that the COCOMO II Post Architecture rating level selection method does not provide a realistic value selection, since it precludes taking values as 1.01 ("nominal"), which are in our case empirically justified.

3.4. Adjusting the aggregation process for improved results

A rating selection method for DOCU is described in what follows. The method could be used for other similar second-level characterizations, and it makes uses of



Documentation match to life cycle needs (DOCU)
This captures the suitability of the project's documentation to its life-cycle needs.

Not Applicable	Don't Know	Very Low	Low	Nominal	High	Very High	Extra High
		Many life-cycle needs not covered	Some life-cycle needs not covered	Right-sized to life-cycle needs	Excessive for life-cycle needs	Very excessive life-cycle needs	/////
							/////

DOCU _____ (Rating Level)

Fig. 3. DOCU rating level selection table

aggregation operator adjustment as one of its essential characteristics, as will be described in what follows.

The first step in the method is that of gathering and analyzing empirical data. This entails the following processes:

- Select the documentation standard to be followed for the projects inside the organization. In the present case study, the ESA Standard previously mentioned was selected.
- Select a measure for documentation size. In our current study, the document measurement unit is number of characters (not including blank space). This measurement unit was adopted because others seemed less convenient.
- Measure the document size for all of the documents for each project, and the total effort of the project. This should be carried out for several projects in order to establish an average size for each document. In order to normalize this magnitude with the size of the overall project – in the sense that, independent of the standard used, the bigger the project, the bigger the documentation – the size of each document is divided for the total effort of the project.
- Obtain an aggregation scheme from empirical data that fits the selection of input values as described in the previous section.

The rationale for the selection of the measure of documentation in our case study is described in what follows. The number of pages, paragraphs or lines depends directly on the presentation format of the document and, for each project, the documents considered use a different format. The number of sections cannot be used because of the difficulty of assessing the relative importance of each section in a document. Then, the number of characters does not cause the problems previously mentioned and, as the projects were carried out on a very tight schedule, the authors did not take the time to elaborate unnecessarily extensive documents. For this reason, the documents contain only the essential information. The only problem found was in image processing. Images may appear in user interface design documents and in the system architecture design document. In order to solve this problem, the images that contain relevant information were dealt with separately. In accordance with the technical documents publication norms of the IEEE Computer Society a prominent image can be determined as equivalent to 200 words. Since the documentation measurement unit established for this study is the number of characters, and after carrying out a study of the documents in question, it was determined that a word, on average, contains six characters. Therefore, within the scope of this study, an image is equivalent to 1200 characters.

The second step in the method entails the use of the data gathered to obtain a realistic input selection mechanisms. This requires the following:

- For each new project, obtain the relative normalized size between all its documents done and the average size that these documents must have.
- Use the adjusted aggregation mechanism in order to obtain a relative size of the overall document.
- Obtain the rating level for the DOCU cost driver using the rating level selection table proposed by COCOMO but selection the rating level with a quantitative (not qualitative) method.

The quantitative method to select the rating level can be summarized by the following rules:

1. If the relative size of the project documentation lies between the 75 % and the 125 % of the rating level, select "nominal".
2. If it is between the 75 % and the 25 %, select "low", and under the 25 %, select "very low".
3. If the relative size lies between the 125 % and the 175 %, select "high", and over the 175 %, select "very high".

The adjustment of the method relies in modelling the contributions of each documentation factor or artifact through flexible aggregation operators. Beliakov's AOTool[2] was used for the adjustment process. Table 4 summarizes the results of the adjustment of an OWA operator as compared to the arithmetic mean and to the selections as resulting from the original COCOMO-II procedure.

Table 4. Comparison of the results of original COCOMO.

P	d	d(C)	OWA	err-OWA	OW(C)	err-OW(C)	AVG	err-AVG	AVG(C)	err-AV(C)	CO	err-CO
1	107.67	111	137.73	30.06	123	12	188.11	80.44	123	12	111	0
2	83.33	81	70.16	13.17	81	0	85.27	1.94	81	0	111	30
3	91.68	91	73.79	17.89	81	10	98.35	6.67	100	9	111	20
4	95.81	91	91.32	4.48	91	0	106.06	10.25	111	20	111	20
5	148.08	123	160.59	12.5	123	0	270.58	122.5	123	0	123	0
6	95.6	91	66.47	29.13	81	10	73.91	16.29	81	10	111	20
				17.87		5.33		39.68		8.5		15

The data provided in Table 4 has the following meaning for each project P:

[2]http://www3.cm.deakin.edu.au/ gleb/aotool.html

- The empirical values for documentation adjustment factor d and their approximation to the closer COCOMO rating $d(C)$.
- The values provided by the OWA operator generated by AOTool, and their approximation to COCOMO closer rating OWA(C).
- The values provided by the arithmetic mean (AVG), and their approximation to COCOMO closer rating AVG(C).
- The values that should be taken if the original COCOMO rating procedure were used CO.
- Error values for the above mentioned values, expressed as the absolute value of the difference between d and each value.

The OWA was adjusted with AOTool by letting the tool decide the *orness* value. The resulting orness of 0.262916 indicated a sub-additive behavior that could be hypothesized o be a result of some degree of redundancy or correlation between the documentation artifacts considered.

The values provided in Table 4 entail three principal conclusions:

- Observing the experimental values obtained – column d(C) –, it can be interpreted that some of the project would have COCOMO levels with the values "low" and "very low". But COCOMO selection criteria precludes selecting levels other than "high" or "very high", considering the context of the projects under study, which follow a high standard of documentation given the methodology selected. Nonetheless, using OWA aggregation, no levels are precluded, resulting in a more realistic approach.
- From a quantitative viewpoint, the relative errors of the OWA, AVG and original COCOMO are of a different nature. The more relevant fact regarding this quantitative view is that with the OWA aggregator, the errors are at most a level shifted in the COCOMO scale, while for the original COCOMO values, three of them are shifted three levels, and one of them is shifted one. This has a significant impact due to the sensitivity of COCOMO [8].
- The differences between the OWA and the AVG are also significant, specially when considering the absolute error, before adjustment. This is an indicator of "hidden" second-level variable interactions.

The aggregation operator design acts in the method described as a parameter of the overall estimation method that models implicitly the uncertainty about possible relationships between second-level variables. For the sake of comparison, other Choquet-agreggation schemes provided in AOTool were also evaluated as aggregation mechanisms. Nonetheless, they provided worse average error values than the adjusted OWA: 33.06, 42.77 and 31.63 respectively for the additive, symmetric 2-additive and symmetric 3-additive Choquet variants.

4. CONCLUSIONS AND FUTURE RESEARCH DIRECTIONS

Cost drivers in parametric software software cost estimation are often expressed through linguistic assessments and they usually represent high-level concepts for which a single, precise measurement scale is not available. This motivates the use of fuzzy techniques to model estimation inputs and their assessment procedures. The use of common membership function elicitation techniques for RELY and AEXP has been described as an approach to analyze the human linguistic categories used to express cost drivers, resulting in evidence pointing out that existing crisp rating values are not adequate as a model for such expressions. In addition, the empirical adjustment of the aggregation process of second-level cost indicators into first-level cost driver inputs has been described using the DOCU cost driver. An empirically adjusted OWA operator has been demonstrated to be a better model than rating level selection rules and than the arithmetic mean.

Future work should address a more systematic and exhaustive experimental account of cost drivers, and the introduction of models of interaction between second-level cost drivers to obtain more detailed models of cost drivers. It should be noted that the modelling approaches described are not only useful for effort estimation but for general software assessment and benchmarking.

(Received May 15, 2004.)

REFERENCES

[1] T. Bilgiç and T. Türksen: Measurement of Membership Functions: Theoretical and Empirical Work. In: Handbook of Fuzzy Sets and Systems (D. Dubois and H. Prade, eds.), Vol. 1, Chapter 3, Fundamentals of Fuzzy Sets, Kluwer 1999, pp. 195–232.

[2] B. W. Boehm: Software Engineering Economics. Prentice–Hall, Englewood Cliffs, NJ 1981.

[3] B. Boehm, C. Abts, and S. Chulani: Software Development Cost Estimation Approaches – A Survey. Technical Report USC-CSE-2000-505, Center for Software Engineering, University of California 2000.

[4] S. Chulani, B. Clark, B. Boehm, and B. Steece: Calibration approach and results of the COCOMO II post – architecture model. In: Proc. 20th Annual Conference of the International Society of Parametric Analysts (ISPA) and 8th Annual Conference of the Society of Cost Estimating and Analysis (SCEA), 1998.

[5] J. J. Crespo, M. A. Sicilia, and J. J. Cuadrado: On fuzzy regression in software cost estimation models. In: Proc. 2003 ACM–IEEE Internat. Symposium on Empirical Software Engineering (ISESE'03).

[6] J. J. Crespo, M. A. Sicilia, and J. J. Cuadrado: On the use of fuzzy regression in parametric software estimation models: Integrating imprecision in COCOMO cost drivers. WSEAS Trans. on Systems *1* (2004), 3, 96–101.

[7] J. J. Crespo, M. A. Sicilia, and J. J. Cuadrado: On aggregating second-level software estimation cost drivers: A usability cost estimation case study. In: Proc. Information Processing and Management of Uncertainty in Knowledge-based Systems (IPMU04).

[8] J. J. Cuadrado-Gallego, O. Marbán, A. Amescua, L. García, and M. Sánchez: The importance of rating level selection method to obtain accury estimations in parametric mathematical models. J. Cost Analysis and Management, Summer (2004), 14–20.

[9] J. J. Dolado: On the problem of the software cost function. Information & Software Technology *43* (2001), 1, 61–72.

[10] European Space Agency – Board for Software Standardisation and Control. 1991. ESA PSS-05-0 Software Engineering Standards Issue 2.
[11] B. Ghahramani: Software reliability analysis: a systems development model. Computers & Industrial Engrg. *45* (2003), (2), 295–305.
[12] A. Idri, A. Abran, and T. M. Khoshgoftaar: Fuzzy analogy: A new approach for software cost estimation. In: Current Trends in Software Measurement (Dumke and Abran, eds.), Shaker Publ., Aachen 2001, pp. 127–142.
[13] B. Izyumov, E. Kalinina, and M. Wagenknecht: Software tools for regression analysis of fuzzy data. In: Proc. 9th Zittau Fuzzy Colloquium, Zittau 2001, pp. 221–229.
[14] E. Kalinina and M. Wagenknecht: Fuzzy regression analysis and application to a crisp model. In: Proc. 8th Zittau Fuzzy Colloquium, Zittau 2000, pp. 9–18.
[15] J. A. McCall, P. K. Richards, and G. F. Walters: Factors in Software Quality, Vol. 1-3. AD/A 049-014/015/055. Springfield 1977, VA: National Technical Information Service.
[16] P. Musilek, W. Pedrycz, G. Succi, and M. Reformat: Software cost estimation with fuzzy models. Appl. Comput. Rev. *8* (2000), 2, 24–29.
[17] NASA Software Measurement Guide Book. Revision 1. 1995. Sofware Engineering Laboratory Series, NASA-GB-001-94.
[18] D. Palomar and M. A. Sicilia: Web page usability analysis based on vague perceptual concepts. In: Proc. IADIS WWW Conference 2004, to appear.
[19] E. Rosch: Principles of Categorization. Readings in Cognitive Science, Erlbaum 1988, pp. 312–322.
[20] M. A. Sicilia, E. García, and T. Calvo: An enquiry-based method for Choquet integral-based aggregation of interface sability parameters. Kybernetika *39* (2003), 601–614.
[21] O. Souza Lima Jr., P. P. M. Farias, and A. D. Belchior: Fuzzy function point analysis. In: Proc. 4th European Conference on Software Measurement and ICT Control, Heidelberg 2001, pp. 161–172.
[22] M. Van Welie, G. C. van der Veer, and A. Eliëns: Breaking down usability. In: Proc. of Interact'99, pp. 613–620.
[23] Z. Xu, T. M. Taghi, and M. Khoshgoftaar: Identification of fuzzy models of software cost estimation. Fuzzy Sets and Systems *145* (2004), 1, 141–163.

Miguel-Ángel Sicilia and Elena García-Barriocanal, Department of Computer Science, University of Alcalá, Ctra. Barcelona km.33.6 – 28871 Alcalá de Henares. Madrid. Spain.
e-mail: msicilia@uah.es, elena.garciab@uah.es

Juan-J. Cuadrado-Gallego, Department of Computer Science, University of Valladolid, Plaza de Santa Eulalia, 9 – 40005 Segovia. Spain.
e-mail: jjcg@infor.uva.es

Javier Crespo, Department of Computer Science, Complutense University, C/. Juan del Rosal, 8 – 28040 Madrid. Spain.
e-mail: jcrespoy@ucmail.ucm.es