

# THE COST ESTIMATOR DOCU: AN EMPIRICAL AND THEORETICAL STUDY

Juan José Cuadrado Gallego, Antonio de Amescua Seco, Javier García Guzmán, Emilio Ernica Lafuente,  
Maria Isabel Sanchez Segura

Departamento de Informática. Universidad Carlos III de Madrid  
Avda. Universidad, 30, Leganes 28911, Madrid, Spain  
(+34) 91 624 91 06  
{jcg,amescua,jgarciag,eernica,misanche}@inf.uc3m.es

**Keywords:** COCOMO II, Function Points, cost estimation models, effort, cost drivers, metrics, software documentation

## 1. INTRODUCTION

In addition to the fine-tuning equations and research in new and better calibration methods, one of the main factors in the improvement of parametric models dealing with effort estimation, is the continuous revision of the cost estimators. This revision will imply not only adding or removing cost estimators to reflect changes in future technology and the methods used to produce software, but also thorough knowledge of those selected.

These conclusions have already been tested in the more commonly-used, modern effort estimation models. For example, while the COCOMO II model eliminated some of the cost estimators used in COCOMO 81, others were introduced. One of these new cost drivers is the documentation needs during the life-cycle (DOCU).

From the initial stages of software engineering, one of the most important practices to be carried out during the software development is good documentation generation. Since then, this has become more and more important in the overall process of software production of any company, especially for those that have or are trying to achieve higher maturity levels. So for those organisations with a maturity level higher than CMM level 2, or those that have to comply with the ISO 9000-3 standard, the elaboration and revision of all the components included in the project documentation need an appreciable effort from the development teams.

This means that the effort estimation models will adjust this effort driver, to the most accurate precision, in order to obtain correct estimates, which will be used generically and in local environments.

Keeping this in mind, the work carried out was focused in two parallel and complementary ways. On the one hand, an experiment by a set of teams, using the same standard and working simultaneously in the development of the same software product, was carried out. During this development project, real values of the estimated measures were obtained, as well as the actual effort dedicated to each one of the projects needed to generate the documentation.

On the other hand, we carried out a theoretical study with special attention to those factors which could have influence on the effort dedicated to the documentation. In this way, two projects with the same number of generated documents could have a different percentage of the total effort dedicated to documentation, depending on the quality applied or the accuracy of the standards. These factors, as well as many others, were analysed.

Finally, several questions relating to the Multiplicative Cost Driver DOCU in the COCOMO II Cost Estimation Questionnaire were studied. In order to simplify the selection of

the most accurate range for each project, a set of questions, keeping in mind the most important factors that influence the effort dedicated to the documentation previously deduced, were added to those already present in the proposed questionnaire.

## **2. DOCUMENTATION COST DRIVER IN MODERN ESTIMATION MODELS**

One of the aspects that has gained significant importance in the software development projects, is documentation. At first, this aspect was only considered in a few estimation models, such as SLICE [Kustanowitz, 77] o NAVAIR [Buck, 71]. The importance of this factor has progressively increased. This means that the resources, and consequently the effort dedicated to building the documentation, have increased in the total effort. Thus, NASA recognises that documentation requires at least 11% of the project effort [NASA, 95].

Nowadays, the most common modern, non-proprietary estimation models are experiencing opportune modifications with the aim of adequately reflecting the role of documentation in the software process. For example, in 1990, NASA [NASA, 90] used a lineal model to carry out documentation estimations ( $\text{Pages} = 120 + 0.026 \text{ LOC}$ ). Now the focus is toward an exponential model ( $\text{Pages} = 34,7(\text{KSLOC})^{0.93}$ ) [NASA, 96].

In the same way, the estimation model studied, COCOMO II (Post-Architecture) [Boehm, 95] presents a range for the effort conductor DOCU (Documentation) of 1.52, COCOMOII.1998 [Chulani, 98]. This means that the variation in the effort due to documentation is greater, for example, than that due to experience in programming language and development tools or the use of CASE tools, and is similar to that required for product reliability. This is a notable difference from the previous version of the model, COCOMO 81 [Boehm, 81], which did not take into consideration the documentation variable.

Among the system size adjustment factors, Albretch's function points model [IFPUG, 98] does not include any related to documentation. However, the MKII model [UKSMA, 99] includes an adjustment factor, C-19 Documentation, which permits the adjustment of the system size according to the number of documents to be produced, irrespective of their importance, characteristics, etc. The system size could expand up to 2.5% of its initial size.

Proprietary models, being proprietary, do not make their equations and, in many cases, their calculation methods, public. However, like the previous models, they consider different cost conductors for effort determination and, one of the new variables considered by some of these models, is documentation.

In this way, SEER-SEM [SEER-SEM, 96] introduces software documentation as variables for effort calculation through the requisite for the specification parameter within the set of product development requirements.

PRICE-S [Lockheed-Martin, 97] introduces documentation in the model through customer specifications and reliability requirements categories. The Platform (PLTFM) parameter provides the testing levels and documentation needed.

The SLIM model [Putnam, 92] or ESTIMACS [Jensen, 83], also adjust their results, bearing in mind the documentation variable, although they carry out effort and calendar estimations with a distinct mathematical base using personnel distribution during the production process as a basic variable,

## **3. RESEARCH OBJECTIVES**

The following were established as objectives for this research:

- To establish, experimentally, the proportion of project effort that can be attributed to documentation elaboration.
- To verify, experimentally, the validity degree the COCOMO II estimation model has for estimating the necessary effort to develop the documentation related to the total effort of a software development project.
- To make the selection of an appropriate value for the cost driver DOCU in each software project easier, redefining the questionnaire proposed in COCOMO II. In this way, inexperienced people who have to estimate software projects, could calibrate the effort related to this factor (DOCU).

#### **4. EXPERIMENT DESCRIPTION**

For the purpose of verifying the hypothesis established under Research Objectives, an experiment was developed and carried out in the Software Systems course taught in the 5th year studies in Computer Science at the Carlos III University, Madrid.

Software Systems is a highly practical course in which the students are grouped in teams of 5 or 6 members. Each team is responsible for developing a software project (a business type that can, as an option, run in Internet), from its inception to its execution. Each team was assigned a tutor responsible for determining the initial system specifications to be developed and for revising the different instalments that the development team elaborates during the project.

During the development of the project, the teams followed the European Space Agency (ESA) standard for software, that recommends the execution 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.

The development environment, in which the software system was coded, was Visual Basic 5.0.

With the expressed aim of analysing 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.

## 5. ANALYSIS OF EXPERIMENT RESULTS AND CONCLUSIONS

### Previous Considerations

Before studying the results obtained, it is necessary to consider several factors related to the documentation.

For this study, all the documents, specified by the ESA standards and generated in each one of the phases considered for this software project, were defined as relevant to project documentation. These documents are shown in table 1.

PROPOSAL	SQAP
	SCMP
USER REQUIREMENTS	SPMP
	URD
	SVVP
SOFTWARE REQUIREMENTS	SRD
	PFI
ARCHITECTURE DESIGN	IFZ
	ADD
DETAILED DESIGN	DDD
	PPF
	SUM
	SVR
FINAL BALANCE	AUD
	STD
	PHD
TRACKING AND OVERSIGHT	PROJECT REPORTS

Table 1: Effort distribution related to documentation elaboration

It is important to emphasise that the preparation of the proposal document was not considered in the proposal phase. Although this elaboration is complex and of great importance to a project, it was thought that the project started with the informal extraction of system requirements.

Likewise, during the planning stage of the different projects, a complete GANTT chart was elaborated with the aid of a project planning tool (Microsoft Project) and, subsequently, the software project plan (SPMP) in accordance with the ESA standards. When assessing documentation, only SPMP was considered because it contained a summary of the GANTT chart information.

It was determined that the document measurement unit is the number of characters (not including blank spaces).

This measurement unit was adopted because others seemed less convenient:

- 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.

The number of characters does not cause the problems previously mentioned and, as the projects are carried out on a very tight schedule, the authors do 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 [ICS, 00], 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 6 characters. Therefore, within the scope of this study, an image is equivalent to 1200 characters.

**Objective 1. To establish, experimentally, the proportion of project effort that can be assigned to the documentation elaboration**

From projects studied, the effort dedicated to documentation elaboration fluctuated between a minimum value of 12.34% and a maximum value of 34.67% of the total development effort. The average 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%. It may be that the effort put into the documentation is very important not only because of the quantity generated, but also, because of the quality.

As for the distribution of the percentage of effort that went into the documentation during the different phases of the project, there are variations among the different projects, as shown in table 2.

Phase	Minimum Effort	Maximum Effort	Normal Effort
PROPOSAL	19,2 %	52,2 %	19 %
USER REQUIREMENTS	13,6 %	68,2 %	35 %
SOFTWARE REQUIREMENTS	12,8 %	63,9 %	60 %
ARCHITECTURE DESIGN	18,5 %	63,5 %	60 %
DETAILED DESIGN	7 %	23,5 %	20 %
FINAL BALANCE	25,3 %	50 %	30 %

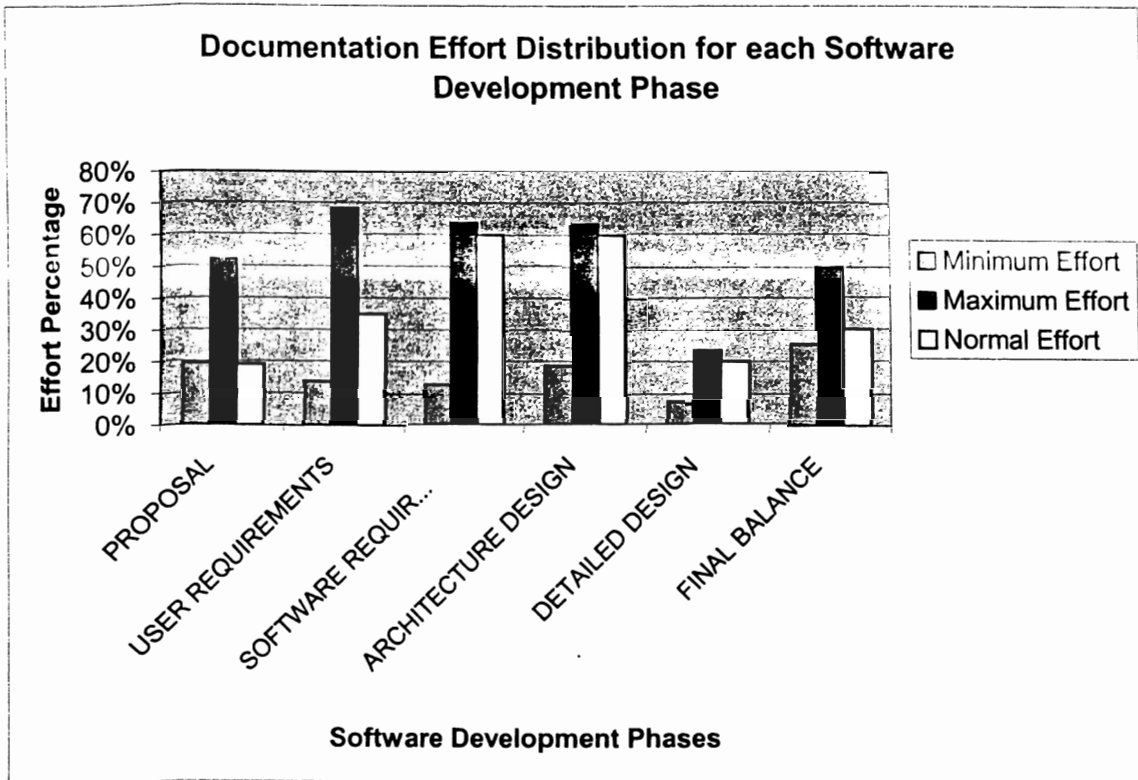


Table 2: Documentation Effort Distribution for each Software Development Phase

The results in table 2 show: the minimum effort by phase put into the set of projects studied in elaborating the documentation, the maximum effort and the normal effort put in. As we can see, there are not only wide variations from one project to another but also important similarities. In any case, maximum effort goes to user requirements, software requirements and architecture design, phases in which documentation has greater importance. Less effort is put into detailed design because it includes work carried out in the previous phases and the proposal only includes preliminary documentation.

**Objective 2. To verify experimentally the validity degree that the COCOMO II estimation model used to estimate the documentation size of a software development project.**

If we take a generic parametric nominal non linear equation for effort estimation, of the type of utilized by COCOMO II (Post-Architecture), is taken:

$$e_n = a(s)^b \quad (1)$$

where  $e_n$  is the nominal development effort in hours,  $s$  the product size in thousands of lines of code, and  $a$  y  $b$  adjustment constants.

An adjustment factor for software documentation, called  $d$ , is introduced. The equation obtained is:

$$e_d = a(s)^b d \quad (2)$$

where  $e_d$  is the development effort in hours, bearing in mind the effort dedicated to software documentation.

From both equations, the value of  $d$  coefficient can be calculated.

In the analysis carried out, both  $e_n$  and  $e_d$  are known for each project, the effort dedicated exclusively to documentation elaboration is:

$$e_{\bar{d}} = e_d p \quad (3)$$

where  $p$  is the percentage of the total effort dedicated only to software documentation. Bearing this in mind and also considering that:

$$e_d = e_n + e_{\bar{d}} \quad (4)$$

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 every project, the coefficient  $d$ , which could have been used in each, can be determined from this point.

The above-mentioned coefficient  $d$  has, for the projects studied, a value that fluctuates between 1.53 for the biggest project and 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. If it is understood that the documentation variable takes the nominal value (1) for a project in which the documentation process takes 11% of the total software development effort as in some studies [SEL, 95], then the documentation cost coefficient  $d$  would fluctuate between 1.01 for the smallest and 1.31 for the largest. Therefore, the effort adjustment factor of a software project, as a result of intensive documentation production, is the following:

$e_d = e_n d$  (5), where  $d=[1.01, 1.31]$ ,  $e_d$  is the adjusted effort in hours and  $e_n$  is the development effort, without taking documentation generation effort into consideration.

If we consider the calibration of the 1998 COCOMO II model [Chulani, 98], it is observed that, for the documentation variable (DOCU), some values for the nominal, high and very high ranges are given (1.00, 1.11 and 1.23) which are within the range of those calculated in the present projects. Values from nominal to very high have been taken, since, bearing in mind the project characteristics described, the documentation generated in this experiment is at least nominal, if not high, in COCOMO II, or very high.

As a conclusion of this aspect, it can be seen that the COCOMO II DOCU coefficient calibrated in 1998 can be of universal use and therefore, the only aspect that could be improved might be the way to assign its value for each software project.

**Objective 3. To make the selection of an appropriate value for the cost driver DOCU in each software project easier, redefining the questionnaire proposed in COCOMO II.**

Once the proportion of the total effort dedicated to the documentation (Objective 1) has been determined, and the coefficients utilized by COCOMO II have been validated again, (Objective 2), we move on to Objective 3 (which is theoretical in nature), that aims to facilitate the person responsible for carrying out estimations using COCOMO II, the choice of the range for the cost driver DOCU in which each project is found. At present the table used is:

	Very Low	Low	Nominal	High	Very High	Productivity Range
<b>DOCU</b>	Many life-cycle needs uncovered	Some life-cycle needs uncovered	Right-sized to life-cycle needs	Excessive for life-cycle needs	Very excessive for life-cycle needs	1.23/0.81=
	0.81	0.91	1.0	1.11	1.23	1.52

Table 3. Guide table for COCOMO II.1998 cost driver DOCU

Where, for example, if there is no documentation to be generated in some phases of the life-cycle, the range “*under*” is selected. For an experienced manager, this question can be very clear, with an easy answer, but less experienced managers may need some other complementary information. We propose an additional questionnaire that permits the accurate selection, the correct range of DOCU for each project, as well as, the consideration of some aspects of the software documentation that influence the effort dedicated to carry it out (i.e., consistency and complexity).

In order to determine the cost driver DOCU a set of questions that can be used as intermediate step for the DOCU value assignment for each project, has been defined. These questions are related to three factors that directly influence the effort needed to elaborate the software development project documentation. These factors are:

- Type of documentation.
- Documentation complexity.
- Standards use and traceability.

#### Type of documentation

The type of documentation prepared during a software development project influences the effort of a software project. In this way, the higher the number of different types of documents, the greater the effort related to documentation is. For this reason, the following guide has been defined to help select the numeric value related to this factor:

DOCU	Very Low	Low	Nominal	High	Very High
<b>TYPE</b>	Only the basic development documentation (user requirements document, software requirements, code documents y user manual)	More refined technical development documentation which includes functional analysis and low level design.	Documentation related to software project management (project plan description, estimation document, oversight reports and final balance) in addition to the previous values	Documentation related to quality assurance: quality plans, test plans and quality registers, in addition to the previous values.	Documentation related to audits, configuration management plans and other documentation, in addition to the previous values.
	0	2	4	6	8

Table 4. Type of documentation factor guide.

## Documentation Complexity

The factor denominated documentation complexity is related to the difficulty that the project team members find in completing the predicted content of each of the documents needed to develop a software project. In assigning a numeric value to this complexity, the following table can be used as a guide.

DOCU	Very Low	Low	Nominal	High	Very High
<b>COMPLEXITY</b>	The documentation prepared using only exits generated by CASE tools without any type of modification.	The documentation is prepared modifying (textual comments) the exits generated by CASE tools.	The contents of the documentation is, in most cases, prepared by modifying the models obtained during software development.	The contents of the documentation is, in most cases, a text written specifically for this purpose.	The contents of the documentation is, in most cases, new models and text written specifically for this purpose.
	0	2	4	6	8

Table 5. Complexity factor guide.

## Standards use and traceability

The factor relative to standards use and traceability is the use of a predefined structure for each software project document section. This structure must be common in all organization projects. It is also important that the contents of the different sections be coherent and consistent between them. In assigning a numeric value to this, the following table can be used as a guide:

DOCU	Very Low	Low	Nominal	High	Very High
<b>STANDARDS USE AND TRACEABILITY</b>	No standards are used to prepare the software project documentation. Documents are unrelated.	Only some technical development documents comply with a standard. There is a high-level relation among them.	All the technical development documents follow a pre-defined standard and their sections are consistent with each other.	All the technical and managerial documents follow a pre-defined standard and their sections are consistent with each other permitting the control of the status of the software project.	There are documents related to quality assurance which comply with a standard and which consistently make explicit reference to the other documents.
	0	2	4	6	8

Table 6. Standards use and traceability factor guide.

Once the project effort estimation manager has established a value for each one of the previous factors, a definitive value for cost driver DOCU must be assigned. For this purpose, the following equation is proposed:

$$r = \sum_{i=1}^3 a_i x_i \quad (6);$$

where  $a_i$  is the relative importance of the factor  $i$ , for each organization in order to determine the effort associated to documentation elaboration, and  $x_i$  is the value that the expert has assigned to this factor for the project being estimated.

At the moment, all  $a_i$  coefficients have one as value because, although from a qualitative point of view, we consider that there are differences in the relative importance of the different factors, we do not have quantitative evidence yet.

By applying the value obtained for  $r$  to the following table, the definitive value for the cost driver DOCU is obtained.

DOCU	Very Low	Low	Nominal	High	Very High
$r$	0 - 4	6 - 8	10 - 14	16 - 18	20 - 24
	0.81	0.91	1.0	1.11	1.23

Table 7. Modified guide for COCOMO II.1998 cost driver DOCU

## 5. FUTURE RESEARCH

While carrying out this research, different important factors have appeared that must be considered. Some of these aspects are:

- To refine the value assignation procedure for the cost driver DOCU using other factors that can influence the effort dedicated to the documentation elaboration.
- To determine, in a quantitative way, the relative importance of the different factors considered to assign a value to the cost driver DOCU for the different projects of an organization.

## 6. REFERENCES

[Boehm, 81] Boehm, B.W., Software Engineering Economics, Prentice Hall, Upper Saddle River, NJ, 1981.

[Boehm, 95] Boehm, B., Clark B., Horowitz E., Westland C., Madachy R., Selby R., "Cost Models for Future Software Life Cycle Processes: COCOMO 2.0," Annals of Software Engineering Special Volume on Software Process and Product Measurement, J.D. Arthur and S.M. Henry, Eds., J.C. Baltzer AG, Science Publishers, Amsterdam, The Netherlands, Vol. 1, pp. 45 - 60, 1995.

[Buck, 71] Buck, F. "A cost by function model for avionics computer system", NADC-SD-7088 Vol.1 Naval Air Development Center, 1971

[Chulani, 98] Chulani, S., Boehm, B., Steece, B., "Calibrating Software Cost Models Using Bayesian Analysis", Technical Report. USC-CSE-98-508, June 1998

[ICS, 00] Author's Resorce IEEE Software. IEEE Computer Society <http://www.computer.org/software/author.htm>

[IFPUG, 98] International Function Point Users Group (IFPUG). "function Point Counting Practices Manual, Release 4.1". IFPUG. Ohio. 1998

[Jensen, 83] Jensen R. W., "A Comparison of the Jensen and COCOMO Schedule and Cost Estimation Models", Proceedings of the International Society of Parametric Analysts, pp. 96-106, April 1983.

[Kustanowitz, 77] Kustanowitz, A.L., "System Life Cycle Estimation (SLICE): A new approach to estimating resources for application program development. Proceedings of the IEEE COMSAC, 1977.

[Lockheed - Martin, 97] PRICE-S Reference Manual, Version 3.0, Mt Laurel, NJ, Lockheed - Martin, 1997

[NASA, 90] Manager's Handbook for Software Development. Revision 1. Software Engineering Laboratory Series. SEL-84-101

[NASA, 95] Software Measurement Guide Book. Revision 1. Software Engineering Laboratory Series. NASA-GB-001-94, June, 1995

[NASA, 96] Software Process Improvement Book. Revision 1. Software Engineering Laboratory Series. NASA-GB-001-95, March, 1996

[Putnam, 92] Putnam L.H. and Myers W., Measures for Excellence, Yourdon Press Computing Series, 1992.

[SEER-SEM, 96] SEER-SEM User's Manual Release 4.5, El Segundo, CA, G.A. SEER Technologies: December, 1998

[UKSMA, 99] United Kingdom Software Metrics Association (UKSMA). MK II Function Point Analysis. Counting Practices Manual Version 1.3.1

[Wolverton, 74] Wolverton, R.W., "The Cost of Developing Large-scale Software" IEEE Transactions on Computers, June 1974, pp. 282-303.