Estimation metrics in software projects

IOP Conf. Series: Journal of Physics: Conf. Series **1126** (2018) 012050 doi:10.1088/1742-6596/1126/1/012050

Estimation metrics in software projects

M P Rojas Puentes[1], M F Mora Méndez[1], L F Bohórquez Chacón[2] and S M Romero[2]
[1] Grupo de Investigación GIDIS, Universidad Francisco de Paula Santander, San José de Cúcuta, Colombia
[2] Grupo de Investigación GISOFT, Universidad de Santander, San José de Cúcuta Colombia

E-mail: fbohorquez@cucuta.udes.edu.co, sromero@campus.udes.edu.co

Abstract. A software product that implements a hybrid metric was developed. It articulates relevant characteristics of three estimation metrics in Communication and Information Technology development projects: (i) history points, (ii) use case points, (iii) function points; as a planning strategy to control problems in software projects that tend to last longer than expected and generate higher costs. The Software product provides tools for making estimates based on expert judgments, planning poker and analogies with other projects that apply to agile projects. A descriptive statistical method was used in the first phase and an analytical method in the second phase. The analysis process is accomplished through the development of a Web application, used in the planning process of software projects in the Systems Engineering program of the Francisco de Paula Santander University. The article describes the characteristics implemented in each metric and its articulation process for the development of estimates. EstimaSoft is obtained as a result, as a support tool in the estimation of projects, in the identification of development trends, through the documentation of lessons learned.

1. Introduction

Currently, there are issues regarding software development. Projects tend to last longer than expected and generate higher costs than anticipated. According to studies carried out by [1], only 29% of projects are successful in terms of scope, budget and timeline; in 52% of cases, there is no information regarding their success or failure, and 19% of them failed. This result, 9 years after the report [2], enables the analysis of the fact that project success percentage is still a factor of study. The failure of an IT project is explained by engineers paying more attention to operational issues than to planning, despite knowing its impact on project control, resource allocation and task reprogramming.

Nowadays, planning is supported by effort estimation based on expert judgments [3]. Issues arise when the team is developing a system which is different from the ones it usually works with, or when the technical and environmental factors that can affect the product are not taken into consideration, reason for which project duration planning with the use of estimation metrics is an efficient alternative. This type of estimation uses mathematical equations to predict project behaviors, equations that consider factors that intervene in the quality of the software, the size of the product, the experience of the developing team, The Stability of software requirements and the programming environment among others. These algorithmic models require specific input and use certain factors and variables to obtain estimates; they are objective, detectable and deliver efficient results.

Some estimation metric references are Enterprise Architect [4], SystemStar & Cost [5], and CostModeler [6] Estimation Metrics, among others. The project is based on software projects estimation methods, categorized into three types: (i) expert judgement, (ii) algorithmic models and (iii) analogy. The Function Points - FP and Use Case Points (UCP), are examples of the first two categories. The third type corresponds to an experts' judgment variant, which, in addition to the opinions of the estimators, clearly characterize the project and seek previous projects with a high degree of similarity, to estimate development times.

2. Methodology

In the diagnosis development, open interviews and documentary reviews were performed, as well as assessments of previous experiences in project management within the Computer Systems Department of the Academic Institution, and technological media used for its operation (Table 1). Time estimation metrics were given in detail with all main components, turning into a reality in a software tool on the web.

The Population studied was University Francisco de Paula Santander, 1186 Professional Graduates of Systems Engineering and the Sample was sixty nine (69), Calculated with a 95% level of confidence and 5% margin of error. The measuring instrument was a 10 open question survey, applied by the form in lines with Google Drive tools.

Table 1. Operation chart.

Variable	Component	Description	Indicator
Type of methodology	Applied methodology in Software business.	Stages present in software development and used by the business; Owned or in the market.	Using methodologies.
	Estimation methods applied in Software business	Tools and techniques of estimate of effort and costs employed in business with the need to develop software.	Description of the time and cost estimation procedure applied in the business.
Planning process	Planning in procedures.	Features of the stages executed in the effort estimation and cost structures offered in software projects for scope achievement.	Description of the planning procedure.
	Estimation metrics.	Models for calculating the work effort and software product costs.	Description of the time and cost estimation procedure known to the professional. Metric used in the methods known to the professional Metrics known to the professional.

The characterization of techniques in those metrics used the most, was determined from the bibliographic review from [7-10] mentioned in the metrics use, including topics such as Scrum Flexibility, Computer Project Development and Management, practical guide for COCOMO 2.0 [11].

3. Professional methodological approach diagnosis results

94% of the Systems Engineering professionals apply project management methodologies; being PMBOK the most commonly used, with 54%, followed by Prince (20%), CMMI (14%) and SCRUM (12%). Those professionals who do not use project management methodology, do their planning focused on decisions made by a technology manager who is responsible for establishing activities and resources needed for the project.

Among the categories analyzed based on software projects planning process responses, the following categories were established: (i) Planning process subject to call requirements; the Project schedule is subject to the call, and necessary developments are planned; these developments are based on the previous call system, (ii) Planning subject to outsourcing processes, (iii) Planning processes subject to the requirement analysis, (iv) Planning processes subject to previously specified times and costs; planning processes included in an Information Technologies – PETI strategic plan , with periodic monitoring.

Regarding knowledge about methods for time and cost estimation, 68 out of 69 professionals responded; 32% is aware of the existence of estimation metrics, 31% know expert judgment, 25% know analogy, 12% know about the bottom-up and top-down method. Most people know at least two methods. Of this segment, 84% uses at least one algorithmic estimation metric. The remaining 16% state several reasons for not using estimation metrics; support by experienced people; normally, technology managers who make value judgments compare with previous similar projects, due to ignorance and time savings, which should be used to provide immediate solutions.

Among responses analysed based on time and cost estimation procedures, the following categories were established: (i) Expert judgment based on specification of functional and non-functional requirements, (ii) Lessons learned in similar projects, (iii) Case method use for time estimation, (iii) No procedure; subject to time and cost established by the company manager.

Of people using estimation metrics, 32% use “use case points”, followed by a 25% using “function point metrics”, “story points” with 17% and COCOMO I/II with 12%.

4. Estimation metrics characterization and adoption of a high level structure for estimasoft software modelling

Based on the interpretation of the operation of the 3 estimation metrics object of study, a common structure is defined for the modeling in the EstimaSoft software. The estimation metrics, function points, use case points and history points have a common structure, as shown in Figure 1. This structure has been defined in three (3) consecutive stages: Input definition, Estimates (Unadjusted estimates, Adjusted estimating factors, Adjusted estimates), Work effort.

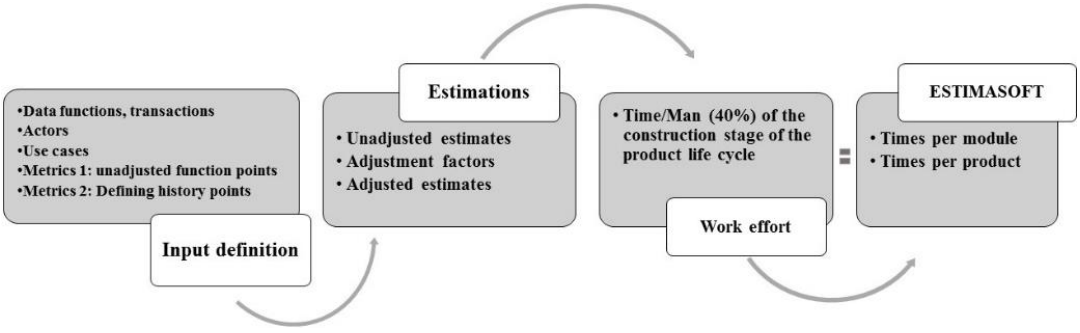

Figure 1. Structure applied in the modeling of EstimaSoft.

4.1. Input definitions

Input elements are identified in each software development metric; or fundamental inputs during the conception phase in the development process life cycle [12], as represented in Figure 2, which, regardless of the paradigm, permit a better understanding among all stakeholders regarding the scope of the project, reflecting the System's functionalities, and vary from one interaction to another in the conception phase. Function points [7] require internal logical files (ILFs) and external interface files (EIFs); both are software database tables, except that EIFs are referenced, but not managed by some transaction within the software boundary.

The complexity of a data function is obtained by counting the data element type (DETs), user abstracted data fields and record element type (RETs); data groups. From the previous count the ILF and EIF are obtained, whose number determines the high, medium, and low complexity, according to

the ranges established by the metrics [7]. External inputs (EI), external outputs (EO) and external inquiry (EQ), establish the complexity of transactional functions, from the number of DETs; unique information fields and file type referenced (FTRs); file types to which a transaction refers, known as transactional functions. The first ones are identified by the user and enter or exit the transaction, the latter are maintained and/or referenced by the transaction. The process includes an EI, EO, EQ count in each high, medium, low complexity.

In such a way, values will be assigned for EI - Low, EI - Medium, EI - High; as well as EO, EQ, ILF, and EIF, to form the Table 2 with the corresponding quantities, according to the counting process.

Table 2. The number of input elements according to complexity.

Type	low	Medium	High
EI	Value - 1	Value - 2	Value - 3
EO	Value - 1	Value - 2	Value - 3
EQ	Value - 1	Value - 2	Value - 3
ILF	Value - 1	Value - 2	Value - 3
EIF	Value - 1	Value - 2	Value - 3

The use case point metric comprises two inputs; the AC system actors and the UC use cases. The use cases are assigned a complexity based on transactions or on the number of analysis classes, while the system actors are assigned complexity according to their interaction with interfaces or other systems, the complexity in both is classified as simple, medium or difficult, according to complexity a weight according to model [8] is assigned.

Regarding the story points, information on the persistent layer of the project and the functionality of the project to be estimated is entered. In the persistence layer, unadjusted function points are used and in this first stage the count is obtained in each of the high, medium, low complexity for the ILF, EIF, EI, EO, EQ elements, explained above, and the project functionality is set by defining the user stories (US), indicating a description and a high, medium, or low complexity. The schematic in Figure 2. represents a comparison of the inputs for each metric.

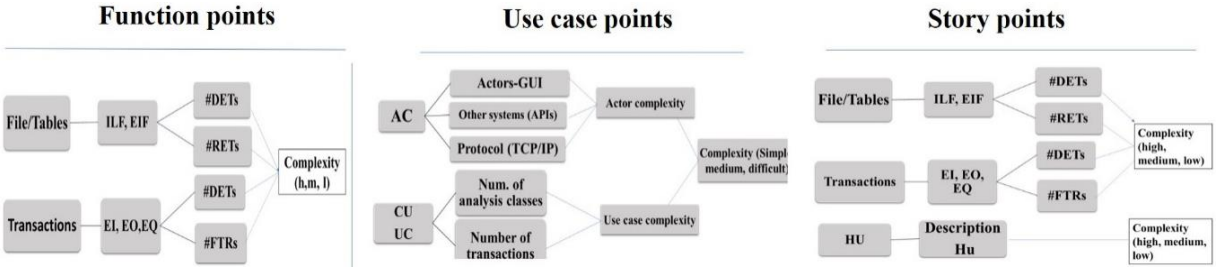

Figure 2. Input elements.

4.2. Estimations

4.2.1. Function points estimation. Function points are estimated based on input elements which are either functional and non-functional requirements. However, the first part of the estimate assumes only the functional requirements to establish the unadjusted function points – PFSA (Equation 1). PFSAs are calculated by adding the number of each one of the elements by the weight established according to complexity [8].

$$PFSA = \sum_{n=1}^{\#elements} (number * weight) \quad (1)$$

Considering that software in the development stage is sensitive to non-functional requirements, the second part of the estimate adjusts the function points, according to the technical adjustment factors –

FTA weight (Equation 2), by the degree of influence perceived in the real environment for product development; this is a scale between zero (0) and five (5) according to criteria [13]. Equation (2) is implemented.

$$FTA = 0.65 + 0.01 * \sum_{n=1}^{14\ factors}(weight * degree\ of\ influence) \quad (2)$$

This estimation stage ends once the adjusted function points are obtained and calculated as the product of the PFSA by FTA. $PFA = PFSA * FTA$.

4.2.2. Use case point estimation. The Unadjusted Use Case Points (UUCP) [9] (Equation 3), are obtained from the sum of the weights according to the complexity defined for each of the actors (UAW) and the sum of the weights according to the complexity of each of the use cases (UUCW) identified in the inputs.

$$UUCP = \sum_{n=1}^{\#\ ac}(weight\ actors) + \sum_{n=1}^{\#uc}(weight\ by\ Use\ Case) \quad (3)$$

The metric establishes thirteen (13) technical complexity factors (TCF) (Equation 4), and eight (8) environmental complexity factors (ECF) (Equation 5). The sum of the products, between weights and estimated degree of influence for each factor, on a scale of zero to five, determine the complexity associated with technical and environmental factors.

$$TCF = 0.6 + [\ 0.1 * \sum_{n=1}^{\#TCF}(weight * degree\ of\ influence)\ ] \quad (4)$$

$$ECF = 1.4 + [\ -0.03 * \sum_{n=1}^{\#\ ECF}(weight * degree\ of\ influence)\ ] \quad (5)$$

The adjusted use case points (UCP) are a result of the estimation stage, as well as the product of the adjusted use case points (UUCP), the technical complexity factor (TCF) and environment complexity factor (ECF), as shown in Equation 6.

$$UCP = UUCP * TCF * ECF \quad (6)$$

4.2.3. Estimate by points of history. The story point estimation was adopted by reviewing lesson learned measurements from other projects and planning poker adjustment [11]. Lessons learned cannot always be applied to estimation in the absence of these measurements, previously performed, between and after the development of similar products. The estimated duration of the project versus the actual duration, the characteristics of the project, and the development team, are a benchmark in project estimation and provide arguments that justify decisions when using adjustments by poker planning. Among the characteristics of the software are technical and environmental factors, times, methodologies, among others, that determine the similarity in the complexity of the development. The experience of the development team is a key factor in this assessment (Figure 3).

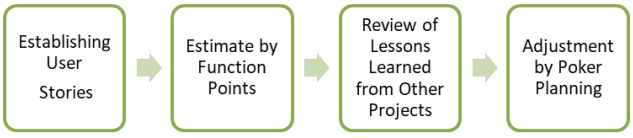

Figure 3. History points estimation route.

Poker planning is based on the previous function points estimation, the list of user stories, lessons learned from other projects and a deck of cards [14]. A deck of cards listed was designed for the EstimaSoft case. The value used in the chart to represent complexity does not have an absolute value. Its value is a function of its scale position, this deck designed in the software is under the fundamentals of [14]. Finally, the software at this stage obtains adjusted history points once the

verification of lessons learned is used together with other projects and the poker planning deck, work done through expert judgement.

4.3. Work effort

4.3.1. Function points efforts. Once the adjusted function points (AFP) are obtained, the work effort is calculated; the software development at this stage was based on the average size of the source lines of code (LOC) and the man-hours per FP invested by the FP development team, within a development environment. For this purpose, tables with reference values for the measurement were established considering the approaches of [15]. This is how the equation is established (Equations 7 and 8).

$$H_{function\ points} = Loc_{xy} \tag{7}$$

Where *x* corresponds to the language and *y* to the environment.

$$effort = PFA * H_{function\ points} \tag{8}$$

4.3.2. Use case points work effort (UCP). The effort, is estimated, thinking only about the development according to the use case functionalities. Assigning a number to person hours for the development of a use case. The software calculates the sum of the environmental factors E1 to E6 that are below a score of 3 and suggests a person hours (CF) value between 20 and 28 hours, leaving the CF decision to expert judgement. Effort (E) represents 40% of the software lifecycle construction stage (Equation 9).

$$E = UCP * CF \tag{9}$$

These values are not absolute but may vary according to the characteristics of the organization and the project.

4.3.3. Hard work by history points. Once the total number of adjusted history points has been estimated, the speed of the work team is determined, indicating the number of stories to be worked on during the week and the number of effort hours to be spent weekly; values that the software enables by means of simple rules of three, to find out the estimations of the project such as the number of weeks, the total estimated hours of the project, the number of iterations of the project and the number of history points per iteration, using the measure that an iteration contemplates a number of weeks of development, value given by the work team (Figure 4).

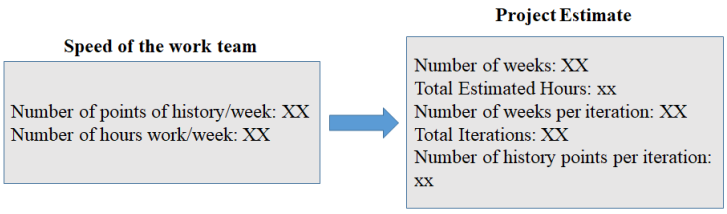

Figure 4. From project speed to project estimation.

5. Conclusions

Estimation in the early software development stage is a challenge for IT professionals, a critical process within the triad of scope, time and costs, as a consequence of working without techniques that would generate a work effort according to the functional and non-functional requirements which need to be satisfied and according to the requirements delivered by the client.

EstimaSoft includes the modular programming concept; providing tools for teams to estimate the project by programming modules or conceiving it as a whole. The possibility of using several metrics

in the same context and the development of a hybrid metric enables the comparison of estimates from transactional and data functions, from actors and use case classification, thus providing a preliminary basis for later adjustments of the history point metric. This grants us a highly reliable estimate in the early stages of software development and guarantees minimum risk in extra costs for the development team and fair prices for the customer.

Estimation methods are included, such as analogies with other projects and expert judgment; accompanied by a preliminary estimate provided by a metric, which generates reference estimates that are highly correlated to the actual work effort. EstimaSoft maintains a lesson learned database from the projects calculated in the tool with generated estimates of the applied metrics or the adjusted and actual estimates. This base becomes the input for the development of Analogies with other projects. In addition, it allows adjustments to be made according to the experience of the work team in similar projects, encouraging the use of expert judgment. Projects stored in the database do not handle sensitive data that may violate project privacy.

The lessons learned are considered a good practice in project management, giving the academic world an added value, such as providing information on trends in the software development market. Documenting the lessons learned is an auxiliary tool to strengthen the training skills of systems engineers and related areas; to know the quantity and characteristics of developments in the area of biotechnology, astrophysics, administration, finance, among others, as part of the purpose of the project.

Estimasoft allows research groups, students, professionals and project managers within the information and communication technologies area, to establish communication with project teams, pursuing their inclusion and their cooperation with the projects. But feedback is not only obtained this way, the user who believes the tool can answer a set of satisfaction questions related to the tool and include suggestions, can obtain constant improvement of the software and generate a greater future impact.

References

[1] Wojewoda S and Hastie S 2015 *Standish group 2015 chaos report – Q&A whit Jennifer Lynch* Consulted on: https://www.infoq.com/articles/standish-chaos-2015#theCommentsSection
[2] Avellanet J 2006 *Six rules for great it project success* Consulted on: https://www.projectsmart.co.uk/six-rules-for-great-it-project-success.php
[3] Jorgense M 2005 Practical guidelines for expert-judgment-based software effort estimation *IEEE Software* **22** 57-63
[4] Sparx Systems 2017 *Building models* Consulted on: http://www.sparxsystems.com/resources/user-guides/modeling/building-models.pdf
[5] Boehm B W and Valerdi R 2008 Achievements and challenges in cocomo-based software resource estimation *IEEE Software* **25** 74-83
[6] Axelrad V, Granik Y, Boksha V V, and Rollins J G 1994 Cost and yield estimation in a virtual IC factory *Microelectronics Manufacturability* **2334** 102-109
[7] Albrecht A J and Gaffney J E 1983 Software Function, source lines of code, and development effort prediction: A software science validation *IEEE Trans. Software Eng.* **9** 639-647
[8] Karner G 1993 Resource estimation for objectory projects *Objective Systems SF AB* **17** 1-9
[9] Clemmons R K 2006 Project estimation with use case points *CrossTalk, The Journal of Defense Software Engineering* **19** 18-22
[10] Sliger M 2012 Agile estimation techniques *PMI global Congress 2012* (Vancouver: Project Management Institute)
[11] Mora Mendez M F 2018 *Aplicativo web que implementa las métricas de estimación para proyectos de software* (Cúcuta: Universidad Francisco de Paula Santander)
[12] Ramos D, Noriega E, Laínez J R and Durango A 2017 *Curso de ingeniería de software* (España: IT Campus Academy)
[13] Dreger J B 1989 *Function point analysis* (United States of America: Prentice-Hall)
[14] Cohn M 2005 *Agile estimating and planning* (United States of America: Prentice Hall)
[15] Jones C 2007 *Estimating software costs: Bringing realism to estimating* (New York: McGraw-Hill)