A Linear Programming Model to Optimize the Production Process in Software Development

Israel D. Herrera Granda, Jorge A. Castro Querembas, Christopher S. Estrella Grijalva, Roberto M. Álvarez Quishpe, Leandro L. Lorente Leyva and Carlos A. Machado Orges
Facultad de Ingeniería en Ciencias Aplicadas
Universidad Técnica del Norte
Ibarra, Ecuador
idherrera@utn.edu.ec, jacastroq@utn.edu.ec, csestrellag@utn.edu.ec,
rmalvarezq@utn.edu.ec, lllorente@utn.edu.ec, camachado@utn.edu.ec

Abstract

In the present work a Linear Programming model to optimize the costs and times incurred in a software development project considering the lines of source code as a metric of the effort made by the group of available programmers was proposed, for the modeling of the problem a limit for the execution budget was assumed, a productivity rate that reflects the real percentage of a programmer's contribution to the development of the project. The acquisition of the data for the model was done through the system called SISCOST, and using surveys and statistics on the salary of the programmers as well as applications such as COCOMO II and SLOCCount to estimate the total Source lines of code of a system. Finally, the optimal amounts of participation of the different types of programmers in software development were determined and the possible implications of implementing the model are analyzed.

Keywords
Optimization, Production, Costs, SLOC, Software Development, Linear Programming

1. Introduction
For the estimation of costs in software development projects, models based on quantifiable metrics such as the Source Lines of Code (SLOC) have been developed (Sommerville and Galipienso, 2005; Rosenberg, 1997). One of these models is the COCOMO 81 that was developed by Barry Boehm in 1981 justified by a study applied to 63 specific domain software development projects. Using the cascade process model and ranging from 2,000 to 100,000 SLOC (Baik et al., 2002; Clark et al., 1998).

In the year 1997, COCOMO II was launched, this incorporated improvements in the estimates of his previous model called COCOMO 81. Taking into account development technologies in the current technological conjuncture, the main metric of this model is the lines of code (SLOC) taken from a quantitative point of view. However, for the use of this metric, the programming language in which it is being developed should be considered, because writing a certain instruction varies from one language to another. It is also important do not considering lines such as comments or documentation. The objective that COCOMO proposes is to measure the amount of intellectual work put into the software development independently of other resources not related to it. So the logical source statement has been chosen as the standard source code line (Sommerville and Galipienso, 2005). This logical source is defined by the list provided by the Software Engineering Institute (SEI) (O'Regan, 2015), which has resulted that the SLOC becoming a feasible metric that can be quantified in existing software (Rosenberg, 1997). With which, an estimate of its total number could be made in a new software development project based on use cases and function points, which would also allow to calculate the cost and time that the development of a particular software would entail (Remón and Thomas, 2010).

It is important to mention that not all the software development projects are the same or they have the same peculiarities. So the COCOMO II application proposed by the University of Southern California takes into account the programming language, five scale factors, and several multipliers of effort depending on whether the development

is a post architecture or an initial software design (Boehm et al., 1995). This application also provides three estimates that correspond to an optimist, one more possible and one pessimistic, for this reason this model is quite complete and even today it is studied and considered in the curriculum of several universities.

Regardless of the metric that is used for the proper development of software, one of the fundamental stages is the specification of software requirements (ERS). If this step is carried out with thoroughness you get to have a complete description of the required system behavior, allowing have a clear vision of how a software development project should be carried out. This implies defining functional and non-functional points that require greater attention, and necessary resources. The lifting of these requirements accompanied by the point of view of an expert or analyst should always be considered in the development of any software development project.

The programmers or developers can be heterogeneously qualified to carry out the tasks of software construction. For that reason, there is a differentiation and a scale that qualifies according to their abilities as a programmer. The abilities considered are usually the period of experience, the expertise with a specific technology, the cohesion that may come to be part of a team, and the real performance that the programmer can perform under high pressure levels. These factors determine the productivity, quality and degree of innovation of the programmers.

Taking these aspects into account, a Senior Programmer is considered an expert who has high programming skills as well as excellent interpersonal skills and a broad track record. Semisenior is the programmer who has considerable experience but still needs to improve in aspects specific to their development. Junior is called to the programmer who begin in the construction of the software, being vulnerable to the pressure of the working environment and still has much to learn about the actual development of software.

It is fundamental to consider the differentiation of programmers for estimating the costs of software development projects. In the technology industry it is common for a Senior Programmer to have a higher remuneration than a Semisenior programmer, who in turn receives a higher remuneration than a Junior programmer. That implies that the cost for the development of each programmer involved in its construction will fluctuate and change. So it is necessary to consider the type of programmer to obtain a more accurate cost estimate, as shown in the Table 1. It is also true that all developers must be paid according to their effort and skills as in any other industry.

Table 1. Qualitative indicators of the types of programmers.

Programmer	Quality	Productivity	Innovation
Senior	high	High	high
Semisenior	half	Half	little
Junior	low/half	low/half	little or null

2. Materials and methods

The present work is based on a mixed research method, since it is considered a quantitative and qualitative perspective in the estimation of costs and time required for software development (Elliott, 1990). It is necessary to have a real perspective of the situation and the environment in which a developer performs, which would lead to an accurate estimation and then propose models that can optimize the costs and development time. That is, converting these models into means or instruments to decide the most feasible route for software development, taking into account the available resources that are available in very specific cases and using as a metric the SLOC.

Linear programming (LP) is an optimization tool that allows to maximize or minimize a linear function called objective function which will be measure to restrictions expressed by a system of inequalities (Hillier, 2012). In the present work a LP model to optimize the development of software will be proposed. To create the LP model some details have been included, and this is contrasted with models and situations in the development of the software. It is also necessary to establish the differences between software development some years ago and in the present, as well as to understand the situation of the programmers, their differences and the real economic situation that vary depending on the specific place where these they work.

The COCOMO II software makes an estimate of costs based in the amount of SLOC that software has or calculating the lines it would have if it has not yet been developed using function points. This model is relatively old, its first version appeared in the year 1981, and its most recent version was conceived in the year 1996 and published in the

year 2000. Thus, 37 years have passed since the publication of its first version and 21 years since the conception of improvements for the model, that is why it is not cataloged as a frequently updated model.

The above implies that in the years in which COCOMO II had not been updated, many improvements in the field of software development have been occurring. Due to different aspects such the more agile access to information with which today is counted (Cadavid et al., 2013). In addition, to a much wider and easier to obtain feedback, and the emergence of new agile development frameworks and methodologies (Bajarlía et al., 2008), it is possible that software construction times and the intellectual effort that they present at present are much smaller. That is, what could take months now may perhaps be done in weeks.

However, COCOMO II continues to be an attractive method that in the present work allows us to estimate the time a programmer would take to produce an SLOC, the time it takes to think, understand and write an SLOC. In addition, COCOMO II has as an important characteristic to consider the specific programming language for each case, since in a software development it is important to understand that the language can have an effect not only on the difficulty and syntax of writing of the code. Can also define the quality of the programmers that are available for that development, since a programmer can usually be considered Senior, Semisenior or Junior for their expertise in a certain technology. Which would imply that the programmer is mostly or less remunerated by the language and their abilities to develop the code. This factor implies a greater or minor cost for each one of the SLOC that is produced according to the type of programmer.

Commonly in a Software Development Project there is a limited number of Senior Programmers, Semiseniors and Juniors with their differences in abilities, who will have to face the task of producing a software. Taking into account their occupations specifically related to the development of the software. In addition, programmers must comply with other tasks assigned by the organization in which they work. Due to this, the actual performance of a programmer in their hours of software development could be affected, so that their real performance rate it could vary between its maximum capacity and a possible decrease. This factor is relevant to take the choice and conformation of the most feasible development team for a given software development project.

To determine a real productivity rate, it would be necessary to know by means of a previous follow-up how many SLOC a programmer produces under normal conditions and what is their actual performance in the realization of software development projects. For this reason, it is necessary to establish a point of comparison, determine the production metric and mathematically establish said measure. In the present work a rate is used that can vary from 100% if the developer develops the maximum of his time in the development or a smaller percentage if it is not like that.

Productivity is defined as the ability of the programmer to produce a SLOC. It has been established that the rate of productivity is equal to the efficiency multiplied by the fulfillment of the plan. By efficiency, the relationship between the time that is actually used and time available is defined; and the compliance with the plan has been defined as the relationship between what has been produced and what has been planned, that is, when the produced quantity is less than the quantity planned, the lower the compliance with the plan (Rodríguez, 2002; Hidrobo and Rueda, 2011). With these considerations we obtain equations 1 and 2.

$$\text{Productivity Rate} = \text{Efficiency} * \text{Compliance with the Plan} \tag{1}$$

$$\text{Productivity Rate} = \frac{\text{Real time (hours)}}{\text{Available time (hours)}} * \frac{\text{Units produced (SLOCs)}}{\text{Planned units (SLOCs)}} \tag{2}$$

The remuneration received by programmers is a factor that varies depending on the conditions of the local software industry. For this reason, it is necessary to know the level of competition and demand in the software market. It should also take into account elements and practices that are specific to each location, and even the exchange rate and currency with which the programmers are remunerated. The above factors cause that the programmers receive different salaries even if they perform in a similar way (Fuentes, 2016). Next, they show reference values of salaries received by programmers in Ecuador and Argentina, taking as sources registered values by ministries, agencies and surveys.

Table 2. Salaries of a programmer in Ecuador and Argentina

Country	Type of programmer	Monthly salary (USD)	Source
Ecuador	Senior	1105.85	Ministry of Labor of Ecuador (Ministerio del trabajo, 2018)
Ecuador	Semisenior	414.38	
Ecuador	Junior	411.28	
Argentina	Senior	1929.44	Chamber of the Software Industry (LA NACION, 2017)
Argentina	Semisenior	1421.98	
Argentina	Junior	967.20	

As it was possible to verify, the monthly salaries of a programmer are different in each country, it was also possible to notice the marked difference of the salary of a Semisenior programmer in Latin American countries. This factor makes the calculation of the cost by SLOC varied from a place to another. So, to use this value it is important to understand the characteristics of the locality in which the software development project is being carried out, in order to properly implement a model that seeks to obtain the most feasible combination in the use of the available programmers that optimizes the costs and times. You could even reach a much more detailed level taking into account the salaries that each programmer receives depending on the organization or company in which he works. This would result in a cost per SLOC of each programmer more accurately.

2.1 Approach of the Linear Programming Model

The proposed LP model incorporates cost, time and quantification concepts of the SLOC that are produced by the programmers available for a software development project. Specifically, the skills and fitness of the programmers are quantified through their productivity. A maximum budget is considered for the payment of salaries to the programmers and a limit period for the development of the project that can not be exceeded. In short, the proposed LP model seeks to minimize project costs by determining the optimal combination with which each programmer will participate.

The cost of the software development project is defined as the objective to be minimized in the model in the form of a linear function in which the decision variables will define the number of SLOC that each programmer will contribute to the development. In addition, they will give us the pattern of what in terms of time and costs will correspond to each programmer in the project, so as constants will have the time and cost for each SLOC that a programmer performs. Also takes into account the constant productivity that will be inversely proportional in relation to cost, that means the lower the real performance of each developer, the higher the cost will be.

The restrictions of the model are given by the number of SLOC corresponding to the source code of the development project, in addition to time and cost limit that should not be exceeded. For the restriction of the limit cost, the sum of the cost generated by each programmer should be considered. In the case of the restriction regarding the time limit, several restrictions are considered since the programmers in the development will be simultaneously programming. So that each programmer is assigned its own time limit restriction which will be less than or equal to the time available in which the software development. Even said time could be much less if a certain programmer was not available during the entire duration of the project developmental. The following LP model is proposed to minimize the costs of a software development:

Objective Function

Let the types of programmers $X_i;\ \forall\ \{i = 1, 2, 3\}$

$$\text{Min Z} = \sum_{i=1}^{n} \frac{X_i C_i\ t_i}{k_i}$$

Where:
X_i is the SLOC number written by each type of programmer.
C_i represent the cost of each SLOC depending on the type of programmer.
t_i are the times that each type of programmer takes to write an SLOC.
k_i is the constant or productivity rate of each programmer.
Z is the total cost of the software implementation project

Restrictions
The first restriction indicates that the sum of the number of SLOCs produced by the programmers must be equal to the total number of SLOCs required for the development of the project, represented as "TSLOC".

$\sum_{i=1}^{n} X_i$= TSLOC

The following group of constraints control that the time used by each type of programmer does not exceed the total time available for each type of programmer "T_i"

$X_i t_i \leq T_i$

The following restriction controls that the budget "C" assigned to the Development Project is not exceeded.

$$\sum_{i=1}^{n} X_i c_i \leq C$$

The following set of restrictions indicate the positivity of all the variables and constants.

$X_i, c_i, t_i, k_i \geq 0$

2.2 Data acquisition for the Implementation of the Model

Once the objective function and the restrictions of the LP model have been established, the operation of the model must be evidenced. To obtain data, reliable sources have been used, such as the COCOMO II and SLOCCount applications. The objective is to contrast and analyze the development of a possible real implementation of the model, taking into account that the fundamental thing is the testing of the model.

The application of the LP model requires calculating the TSLOC of the software to be developed. In a real case would be based on the survey and specification of software requirements made by an analyst with the consulting of an expert in similar developments (Hoger and Angel, 2010). Through the function points and use cases and thus get to calculate the amount TSLOC that the system would have.

In the present work it has been decided to analice a previously developed system in order to have an estimate of the SLOC number. The system to be used is a web application programmed with JAVA technology made in 2013 called "SISCOST". This system consists of customized software for process management and billing of a company dedicated to the manufacture of clothing. It is important to note that this system is operating in a local company (Gosling et al., 2014; Andrade, 2013).

For the counting of the SISCOST SLOC, the free software "SLOCCount" was used, that allows to count SLOC by discriminating comment lines and focusing on the logic of some programming languages, among them JAVA. To have a correct interpretation in the SLOC count which are what really require the intellectual effort of the programmer. In addition, this application performs a calculation based on the COCOMO II that shows us a result of the possible cost and time that would require developing for this specific case the SISCOST system. This application is used through an Ubuntu operating system with an architecture of 64 bits, as shown in Figure 1.

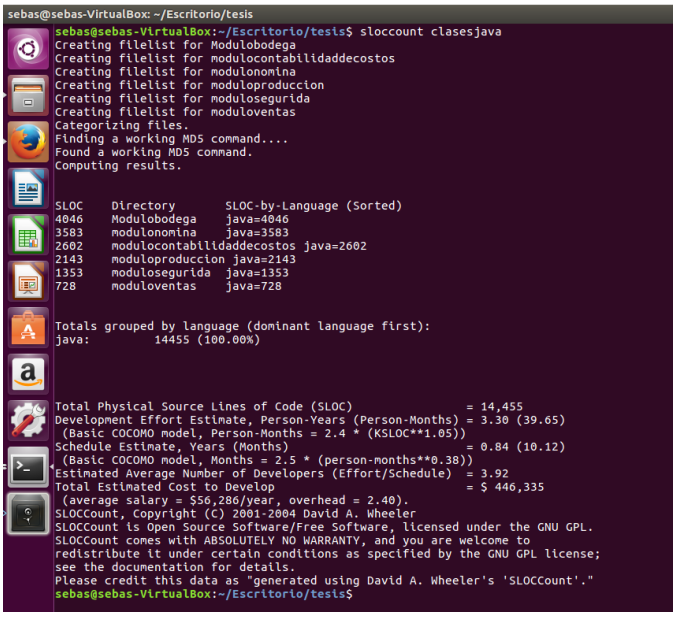

Figure 1. Results obtained in SLOCCount.

Using the SLOCCount application, it has been estimated that the SISCOST system has a total of 14455 SLOC. This data will be used to test the proposed LP model. This application estimates that the development would need to have 3.92 programmers for a time of 10.12 months with a cost of 446,335 USD. For the estimation of these costs, the application the nominal parameters of the COCOMO II model and his default salary value were considered. This cost estimate given by COCOMO II does not represent a conclusive contribution in the analysis of this research, but it is an example of applications that perform the estimations using SLOC.

When implementing the solution of the model it is necessary to define some requirements for data acquisition. It is thus established that the group of programmers that would be available to develop the SISCOST software are three: A Senior, Semisenior and Junior programmer. In the case of the Junior programmer, half of the total project time will be available to meet the development. The maximum number of hours per month for development will be 160 hours, 18 months will be available to complete, and the budget for programmers will be 80000 USD.

To obtain data of the time by SLOC for each programmer, the COCOMO II application was used, which was configured indicating that the development is an initial design. The development language is JAVA and the time of hours per person per month is 160. In addition, the factors were varied of scale per programmer. For the senior programmers the scale factors were calibrated as very high (VHI), for the programmers Semisenior the values were calibrated in nominal value (NOM), and in the case of the Junior programmers the values were calibrated as low (LO). Once the previous process was carried out, the most possible amount of SLOC production of the programmer per month was obtained, with this production being carried out the quotient between 160 hours that corresponds to the time the developer works per month and the production per month. Each result of this quotient it is accepted as time by SLOC of each programmer.

To estimate the cost per SLOC for each programmer, the quotient between the monthly salaries mentioned in Table 2 and the most possible amount of programmer production per month that was previously used was calculated. Each result of this quotient is accepted as the cost per SLOC of each programmer.

Example of estimation of cost and time by SLOC by type of programmer and locality
The SLOC production for a Semisenior programmer in Ecuador according to COCOMO II is 328.5 SLOC per month, its monthly available time is 160 hours, as shown in Figure 2. Next, it is determined that the time by SLOC for said programmer is 0.48 hours. Then, the salary of said programmer is used, which in this case is 414.38 USD, as shown in Table 2. When performing the quotient with the production of SLOC, the cost per SLOC is 1.26 USD.

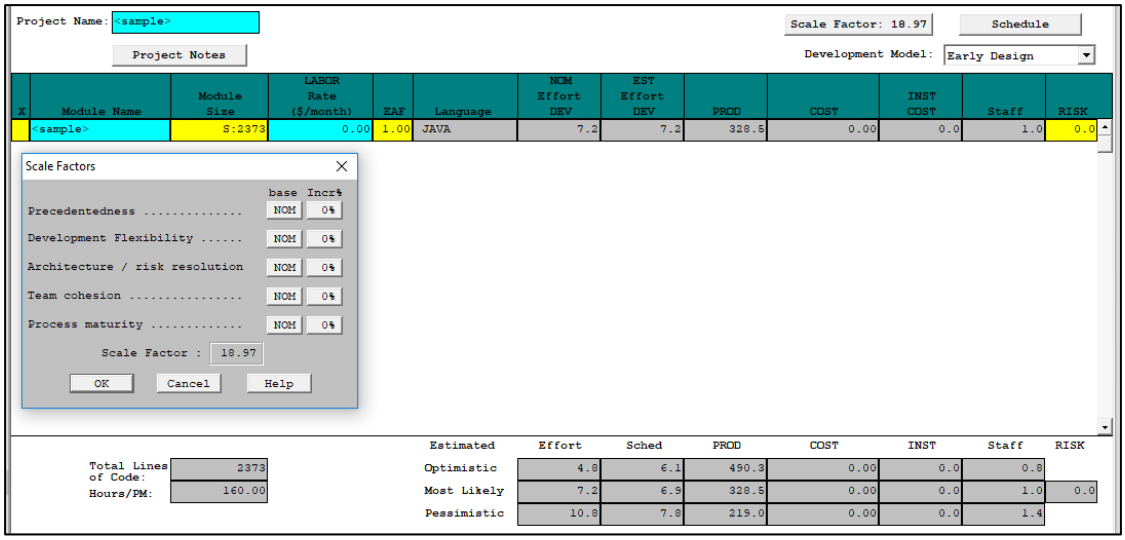

Figure 2. Results obtained in COCOMO II for the Semisenior programmer.

Similarly, the costs and times for each type of programmer are estimated considering the place where this software would be developed. To contrast the proposed model of LP in two different places in which the programmers would develop. To determine the productivity rate for each type of programmer, equation 2 is used for its calculation, as shown in Table 3.

Table 3. Costs, times and productivity for each type of programmer.

Location	Programmer	Cost per SLOC - C_i (USD)	Time per SLOC - t_i (hours)	Productivity rate - k_i
Ecuador	Senior – x1	3.01	0.43	64.31 %
Ecuador	Semisenior - x2	1.26	0.48	77.84%
Ecuador	Junior – x3	1.31	0.51	95.67 %
Argentina	Senior –x1	5.26	0.43	85.36 %
Argentina	Semisenior – x2	4.32	0.48	90.21%
Argentina	Junior – x3	3.10	0.51	40.43 %

3. Results and Discussion

The LP model was implemented using the GAMS software version 23.9.4 proposed in section 2.1, including the characteristics mentioned in section 2.2. The model was executed depending on the location where the software development project is executed, that is, in Ecuador and Argentina.

Once the model was implemented in GAMS, the optimal values were obtained by minimizing the LP model, taking as scalar parameters according to the Ecuadorian and Argentinean context. Through the execution of the LP model under the parameters of Ecuador and Argentina, it could be noted that the total costs of implementation of the project are higher in Argentina than in Ecuador, although in Ecuador, more TSLOC must be done. This is given due to the productivity rate of the programmers in those countries and the costs incurred in the payment of salaries in each country. Thus verifying that the cost of the project must be determined according to the country where the software is developed, the types of programmers that do it, and its productivity rate, as shown in Figure 3.

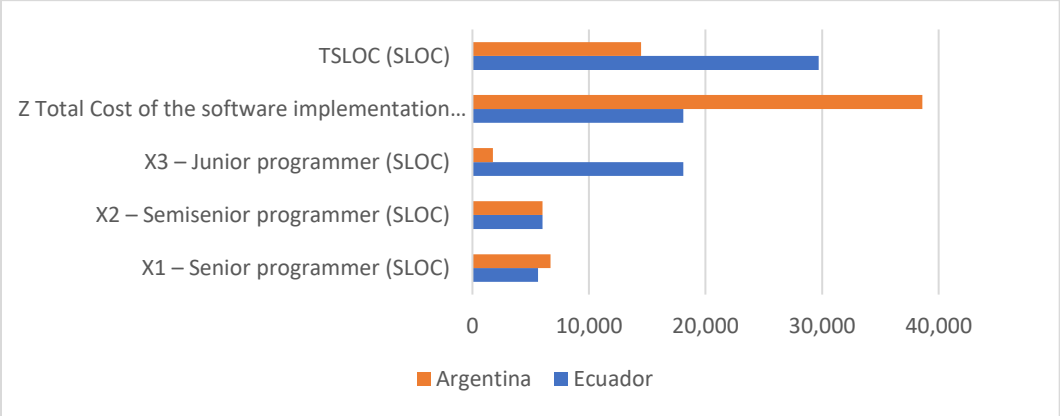

Figure 3. Summary of results obtained

Sensitivity analysis of the model

In the case of Ecuador, after the sensitivity analysis it could be determined that the amount of SLOC produced by the Junior and Senior programmers can increase slightly without damaging the optimality of the model. Also, regarding the restrictions it could be noted that if it is increased slightly the TSLOC of the project, this would affect the feasibility of the model. In addition, if you need to increase the number of hours to the types of programmers these should be from the Semisenior or Junior group. You could also notice a notable reduction in the money allocated to the project through the variation in the amount of SLOC assigned to each type of programmer. This demonstrates the effectiveness of the model, as shown in Figure 4.

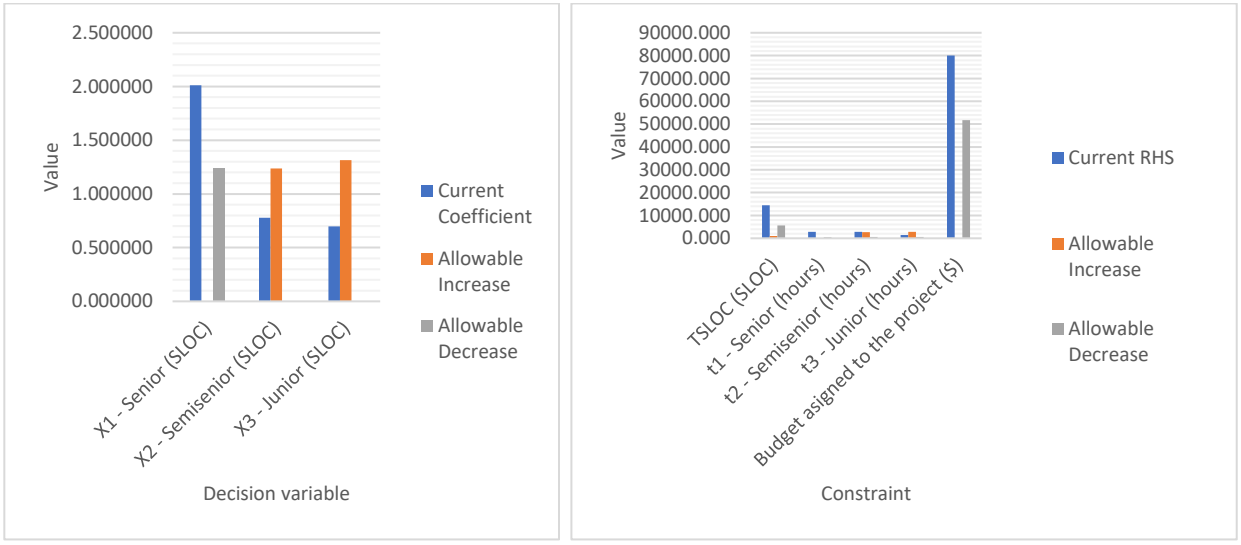

Figure 4. Sensitivity analysis of the Ecuador Model

In the case of Argentina, after the sensitivity analysis it could be determined that the amount of SLOC produced by the Semisenior and Senior programmers can increase slightly without damaging the optimality of the model. Also, regarding the restrictions it could be noted that if it is increased slightly the TSLOC of the project, this would affect the feasibility of the model. In addition, if you need to increase the number of hours to the types of programmers these should be from the Semisenior or Junior group. You could also notice a minor reduction in the money allocated to the project through the variation in the amount of SLOC assigned to each type of programmer, which demonstrates the effectiveness of the model, as shown in Figure 5.

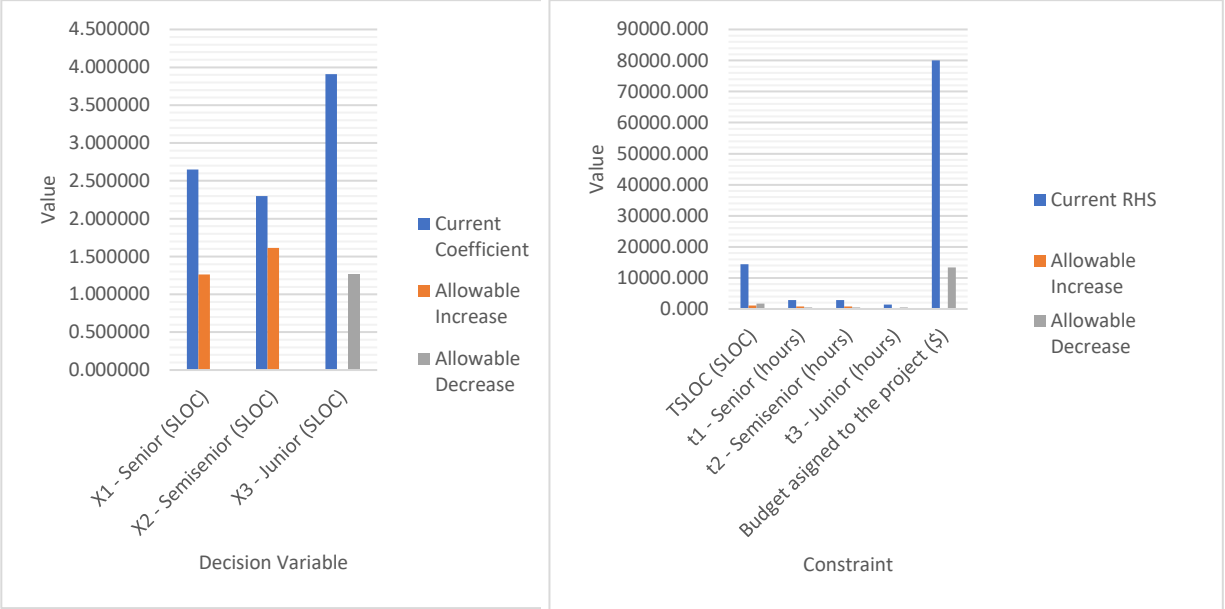

Figure 5. Sensitivity analysis of the Argentina Model

4. Conclusions

The proposed model is able to calculate the optimal quantity of SLOC with which each type of programmer would participate in a software development project. Considering the costs in the country in which the project will be carried out, the capacities and skills of the developers and a productivity rate that describes the percentage of real performance that a type of programmer has in the project. Proving that the cost of the project must be determined according to the country where the software is developed.

In the future, the possible implementation of the proposed model should be analyzed using a more precise methodology for estimating costs and a more exhaustive comparison between different countries and localities. It is also important to consider a specific programming language and its implications, as it would be to include in the model the advice of expert analysts for cost estimation.

References

Andrade Pozo, A. P. (2013). Implementar un sistema de contabilidad de costos para PYMES de confección textil en el cantón Antonio Ante. Aplicativo: Sistema de Contabilidad de Costos, SISCOST.

Baik, J., Boehm, B., & Steece, B. M. (2002). Disaggregating and calibrating the CASE tool variable in COCOMO II. *IEEE Transactions on Software Engineering*, *28*(11), 1009–1022.

Bajarlía, M. V. S., Eterovic, J., & Ierache, J. S. (2008). Elaboración de especificación de requerimientos de seguridad en el desarrollo de sistemas de información basado en la modelización de conocimientos.

Boehm, B., Clark, B., Horowitz, E., Westland, C., Madachy, R., & Selby, R. (1995). Cost models for future software life cycle processes: COCOMO 2.0. *Annals of Software Engineering*, *1*(1), 57–94. https://doi.org/10.1007/BF02249046

Cadavid, A. N., Martínez, J. D. F., & Vélez, J. M. (2013). Revisión de metodologías ágiles para el desarrollo de software. *Prospectiva*, *11*(2), 30–39.

Clark, B., Devnani-Chulani, S., & Boehm, B. (1998). Calibrating the COCOMO II post-architecture model. In *Proceedings of the 20th international conference on Software engineering* (pp. 477–480). IEEE Computer Society.

Fuentes Medina, A. F. (2016). Análisis de los modelos de negocio del software libre y su aplicabilidad a PYMES ecuatorianas con orientación al mercado internacional.

Gosling, J., Joy, B., Steele, G. L., Bracha, G., & Buckley, A. (2014). *The Java language specification*. *Pearson Education.*

Hidrobo, H., & Rueda, R. (2011). Curso Taller de Productividad. Quito: Pontificia Universidad Católica del Ecuador. Retrieved from http://www.nextrategies.com/puce/materiales-productividad/productividad/Manual de

Productividad1.pdf
Hillier, F. S. (2012). *Introduction to operations research*. Tata McGraw-Hill Education.
Hoger, M., & Angel, C. (2010). EJEMPLO ESPECIFICACIÓN DE REQUISITOS SOFTWARE.
LA NACION. (2017). Cuál era el sueldo promedio de un programador en la Argentina en agosto - LA NACION. Retrieved May 15, 2018, from https://www.lanacion.com.ar/2066253-cual-era-el-sueldo-promedio-de-un-programador-en-la-argentina-en-agosto
Ministerio del trabajo. (2018). Ministerio del Trabajo – Ecuador. Retrieved May 15, 2018, from http://www.trabajo.gob.ec/
O'Regan, G. (2015). Software Engineering Institute (SEI). In *Pillars of Computing* (pp. 195–205). Cham: Springer International Publishing. https://doi.org/10.1007/978-3-319-21464-1_30
Remón, C. A., & Thomas, P. J. (2010). Análisis de Estimación de Esfuerzo aplicando Puntos de Caso de Uso.
Rodríguez, C. (2002). Identificación y adecuación de indicadores de gestión para un sistema de la calidad basado en lineamientos iso 9000: 00. Universidad Católica Andrés Bello. Retrieved from http://biblioteca2.ucab.edu.ve/anexos/biblioteca/marc/texto/AAQ2674.pdf
Rosenberg, J. (1997). Some Misconceptions About Lines of Code. *Proceedings Fourth International Software Metrics Symposium*, 137–142. https://doi.org/10.1109/METRIC.1997.637174
Sommerville, I., & Alfonso Galipienso, M. I. (2005). *Ingeniería del software*. Pearson Addison-Wesley.

Biographies

Israel D. Herrera Granda is an Investigator Professor of the Industrial Engineering Career, at the Universidad Técnica del Norte, Ibarra, Ecuador. Holds a Automotive Engineering degree and a Master in operations and logistics (MOL) degree from Escuela Superior Politécnica del Litoral. He has published conference papers and chapters of regional books. Has participated in numerous projects and completed research in several areas. Specialist in Operational Research, Logistics, and Transport reseach.

Jorge A. Castro Querembás is a student of the Computational Systems Engineering Degree at the Universidad Técnica del Norte, Ibarra, Ecuador. He has technical training in computer programming JAVA technology and is a junior programmer prospect. He also has training related to computer networks, databases and free software. He has been a collaborator in several research projects and will specialize his career in information technologies in server's administration.

Christopher S. Estrella Grijalva is a Student of the Computer Systems Degree at the Universidad Técnica del Norte, Ibarra, Ecuador. He has experience in the development of web applications specializing in Front-end, also has expertise in the management of GIT technology and has participated in several local web developments in his city and has collaborated in research projects.

Marcelo R. Alvarez Quishpe is an engineering student in Computational Systems at Universidad Técnica del Norte, Ibarra, Ecuador. He is passionate about computer science since he was a child, he hopes to specialize in computer security, he has collaborated, and he is a member of Ethical Hacking UTN club, and lead to the investigation of metadata and macros.

Leandro L. Lorente Leyva is an Investigator Professor of the Industrial Engineering Career, at the Universidad Técnica del Norte, Ibarra, Ecuador. Holds a Mechanical Engineering degree and a Master of Computer Aided Design and Manufacturing (CAD/CAM) degree from Universidad de Holguín, in Cuba. He has published journal and conference papers. Has participated in numerous projects and completed research in several areas. Specialist in computer-assisted design, planning and manufacturing.

Carlos A. Machado Orges is Industrial Engineer and Master an Industrial Engineering. Computer Science Specialist and Assistant Professor of the Department of Industrial Engineering, Universidad de Holguín, Cuba. Investigator Professor of the Industrial Engineering Career, at the Universidad Técnica del Norte, Ibarra, Ecuador.