Evaluation of a Model Maximizing the Quality Value of Selected Software Components in a Library

Koffi Kouakou Ive Arsene[1], Samassi Adama[2], Kouamé Appoh[3]
Ecole Doctorale Polytechnique, Institut National Polytechnique (INP-HB)
Yamoussoukro, Côte d'Ivoire

***Abstract*—Reusable software components are selected from libraries by developers and integrated into existing software systems to improve their quality. In this article, we evaluate a mathematical model based on an approach of optimization of the selection of the software components according to their quality. This is a linear programming model with constraints. It takes into account the quality characteristics of the components based on standard ISO / IEC 9126, the financial cost and the adaptation time. The experience with the ILOG Cplex Studio optimization tool gave satisfactory results.**

***Keywords*—Software component quality; reuse; reusable components; reusability; mathematical model; simulation; validation; maintenance effort**

I. Introduction

The development of modern and complex software systems involves to the use of reusable software components. These components selected and integrated into the systems must first be evaluated and tested. This has the consequence of strengthening user confidence by integrating reusable software components into their software systems. The choice of the selection of these components depends first on the needs and functional requirements of customers and users. Thus in the thesis of YAHLALI Mebarka, the researcher argues that quality is not often the essential of the development process[1].Yet users are often confronted with the vastness of available offer in the various software libraries but also the multiplicity of software that can insure same services. In addition to the functional properties, it is also necessary and important to know how software studied render their services. In our selection process, the non-functional requirements related to the quality of the software component are taken into account. The addition of the evaluation of the quality criteria of the software components to their functional properties in the selection process so conditions the selection of the most appropriate components, better adapted and at reduced cost. For economic reasons, studies have shown that since 2006 software development based on software components exceeds 40% of total software systems developed [2], [3]. Beyond the economic reasons, other researchers have noticed technological and scientific interests [4], [3], [5]. In [6], the researchers affirm that the developers give a particular interest to technologies related to software components and especially when these intervene in the development of complex and large-scale applications. The aim is to improve productivity and speed up the time of marketing for the products developed. In [7], the authors proposed an optimization approach of software components selection based on their quality. The developed model, score based, evaluates the quality value of the selected software components. This score is calculated from the quality characteristics with values associated in relation to quality services rendered, financial cost and predicted adaptation efforts. This allows us to select the component best suited to the functional needs and quality needs on demand. To determine the quality attributes and the factors that affect the selection and reuse of reusable components, we formulate the following research question: Can the financial cost and maintenance effort affect the selection and reuse of the selected software components? In order to respond to the concerns raised, we propose, in this article an automatic method for selecting reusable software components. This method makes it possible to maximize the selection process by taking into account the quality characteristics of the component, the financial cost and the maintenance effort on the one hand and moreover, assess the quality of the reusable software components selected according to the indicators and quality needs desired by users or companies. This research work is summarized in these following points:

- Identification of problems relating to the selection and reuse of software components.
- Identification of methods from the literature for solving problems related to the reuse of software components.
- Proposal for a new model for automatic selection of reusable software components. This model establishes a link between the financial cost, the maintenance effort and the quality indicators of the software component defined by the ISO / IEC 9126 standard. It is based on linear programming by constraints [7]. It also takes into account the selection of components in a large repository with many characteristics requirements.
- Evaluation of the proposed method. This evaluation consists first of all in defining the decision variables of our model. These variables relate to the financial cost of the component, the adaptation effort and the possibility of choosing the given component from a set of software components. Then we determined the constraints related to the defined variables. At last, we calculated the quality value of the selected components with an optimization tool. This quality depends on the values of the quality characteristics of the software component and on other factors which impact the selection intervening in our model.

- Comparison of results obtained with previous work by researchers.

This paper is organized as follows. Section 2 presents previous work. In Section 3, we propose our mathematical model and its constraints. We have in section 4 the results of our simulations. In section 5, we compared our results with the methods and models proposed in [2] [8] and [9]. We conclude in Section 6.

II. State of the Art

Several works have been carried out for the selection of software components taking into account the functional and non-functional properties.

Some authors have based on surveys to identify important attributes that influence the reusability of selected components [10] [11] [12]. They then validated their work by empirical manner.

In [11], the research concerns a survey of 22 cases next to experts and practitioners in equitable manner to understand how the choice of software components from various sources is effected for industrial practice. The objective of this study is to understand how to adapt research solutions to the needs of the industry. This allows to facilitate industry decision-making regarding the reuse of ready to use software components. The results showed that solutions of expert are deterministic and based on optimization approaches. While those of practitioners are non-deterministic and seem better suited to decision-making for component selection in the industry.

In [9], the researchers developed a metric between the *time-effort* and the reusability of the software component. They then validated their work from a survey next to experts. The results showed that the time taken to understand the software component has a correlation with the value of reusability. They concluded that if the time taken to understand the software component is too much, the lower the value of reusability. Therefore this software component will be used less.

Other authors have developed optimization models based on quality criteria and attributes defined by user [8] [2] [13]. These models have been sometimes simulated and validated.

In [2], the research carried on the process of selecting software components taking into account the attributes defining the dependencies between the quality criteria. The method used is based on matrix calculations of vectors and eigenvalues. The goal is to help developers to understand component details based on quality criteria. This allows to facilitate decision making in the selection process.

In [13], the authors developed a model to automate the selection of software components. They defined a common format for the functional and non-functional properties of these components. They then calculated their *satisfaction index*. The objective is to evaluate the level of matching and conformity of the candidate components to be selected and those of the library Version 3.1 of substitute tool was used to measure the degree of satisfaction with a distribution weight. Those works have made it possible to automate the selection of components with the aim of saving time.

In [8], the researchers developed a so-called reliability metric model that evaluates the quality of software components based on quality attributes of diverse dimensions. This model is based on linear programming by constraint. It allows to select relevant components relating to the solutions of the problems posed. Using computer experimentation with the Cplex solver version 12.2, those authors have found solutions to the problems posed. It is about selecting software components in libraries of large size with large number of requirements in a short time.

In [14], the authors developed an optimization algorithm for selection in large functionality models with multiple objectives. This algorithm called IVEA-II. It allows to do automatic researches to balance different objectives and improve existing methods.

In other works again, the researchers validated their model by the combination of led surveys and optimization algorithm.

Research in [15] has focused on an approach of identification high-quality software components from several sets. The authors modeled the problem of research and identification of reusable software components trough multi-objective optimization. They defined three objective functions. The first objectives correspond to two metrics making it possible to simultaneously maximize cohesion based on the frequency of use of the components and the cohesion linked to the semantic relationship of these components. The last objective is a relation which makes it possible to minimize the coupling on the change about the changes historics from the evolutions of the underlying software system. By successively applying the different metrics above and the NSGA-III search algorithm which allows the grouping of different software components, the authors obtained better quality for different software components measured in terms of reusability characteristics. The results made it possible to identify highly cohesive and least coupled software components.

In [16], researchers worked on the efficient recovery of components from a large repository. They have shown that this technique is a new challenge in the process of selecting reusable components. This retrieval model is based on two (2) steps. The first is to select components that correspond the functional requirements. The second is an algorithm that allows to weight the quality requirements of software components and recommend to developers the highly weighted. The objective of those works is to involve the users in the choice of the quality attributes of the components during the construction of the project but also to organize into a hierarchy the quality attributes according to its needs.

In the next section, we present our model. This model is an optimization approach to select of software components according to their quality, based on linear programming with constraints. It makes it possible to establish an extension, a generalization of the model for evaluating the quality of the factors which influence the selection and reuse of the software components treated in [8]. We associated the maintenance effort with the model he developed.

III. Presentation of the Developed Model

Our job is to select a component and evaluate its quality. The mathematical model that we have developed makes it possible to maximize the value of the quality of the component according to the characteristics, the financial cost and the adaptation time of the component. Our model has two steps. The first is leaned on the prediction of the maintenance effort defined in [7].According to the literature review, it is rare to find a component that can satisfy perfectly the requirements of users. In other words, the selected components may not fully the quality and service requirements expressed by the customer (see Fig. 1).

Fig. 1 designates N components, some features of which make the services perfect but others can do them partially. We will estimate the time required to improve the functionalities which partially render service at the level of defective components. We then applied the cosmic methods developed in [17], [18] and [19]. This allows us to determine the estimate of the maintenance and adaptation effort in accordance with equations (1) and (2).

$$functional\ size\ of\ each\ procces\ (i) =$$

$$\sum_j^p\ functional\ size\ of\ process(i,j) \quad (1)$$

$$\forall\ i \in Sc\ et\ 1 \leq j \leq P \quad (2)$$

Where

Sc : set of the available components

Component (i) denotes the i [th] component of the set Sc

Process (i, j) denotes the process j of the i [th] component

Then, when we apply the estimate of the adaptation effort developed according to [17]. We obtain equation (3)

$$Estimated\ Development\ effort =$$

$$Component\ Size * Unit\ Cost \pm Predictive\ interva \quad (3)$$

This phase makes it possible to determine the adaptation time interval of each component to be predicted. This method then evaluates a financial cost and an adaptation time.

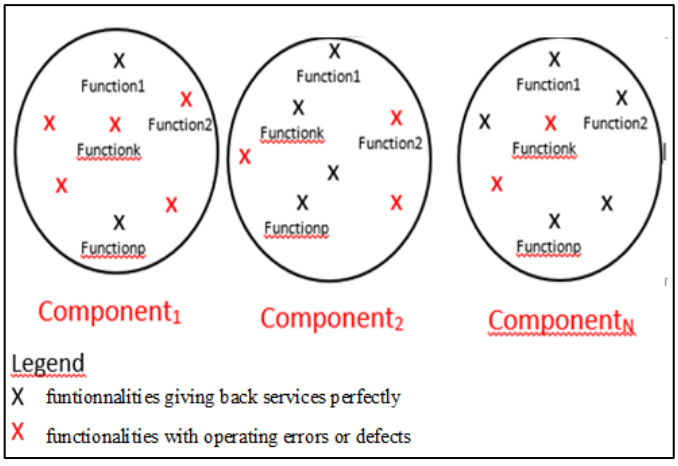

Fig. 1. Set of Selected Components with Some Malfunctions.

The second step allows us to define the objective function. This model is a score that calculates and evaluates the quality of software components based on quality characteristics, time predicted in equation (3), and financial cost. The following model (4) represents the objective function.

$$S_i = \sum_{h \epsilon A} w_h\ q_{hi} x_i - [ac_i + (1-a)t_i]x_i$$

$$and\ \forall\ i \in Sc \quad (4)$$

Where

A: set of software quality characteristics;

Sc: set of available component;

$q_h i$: the standard level of the quality attribute ;

h∈ A for component *i* ;

C_i: Standardized cost of maintenance of the component I;

t_i: Standardized adaptation and maintenance time of the component i;

α : Coefficient of adaptation

For the resolution of this model, we define the following constraints and decision variables.

Decision Variables

- A Boolean variable x_i for selection of the component i corresponding to 1 if the component is selected, otherwise 0;
- An real variable C_i designating the cost of the selected component i;
- An real variable t_i designating the effort of adaptation of the selected component i.

Constraints

$$0 \leq a \leq 1$$

$$t_i \in [\![t_{min}; T_{max}]\!]$$

$$t_i = \frac{t_{i_{rel}}}{T_{max}}\ and\ 0 \leq t_i \leq 1$$

$$T_{max} = 15\ days = 1.296 * 10^3 s$$

$$C_i = \frac{C_{i_{rel}}}{C_{max}}\ et\ 0 \leq C_i \leq 1$$

$$Qmax = 5$$

$$\sum_{h \epsilon A} w_h = 1 \quad (5)$$

In [7], we defined a metric which evaluates the quality of the characteristics of the given software component. It designates the ability of the component to fulfill the criterion linked to functionality. This metric was associated with an ordinal variable of modalities taking the values in the set B.

$$B=\{Bad, Insufficient, Average, Good, Excellent\} \quad (6)$$

These modalities that we defined in (6), are associated respectively with the numerical values: 1; 2; 3; 4 and 5.

This allows us to determine the maximum value Qmax = 5;

To optimize the cost and maintenance time parameters, we maximize the objective function. By considering equation (4) with constraints, we obtain the following equation (7)

After presenting our model we move to the validation and simulation phase.

$$\begin{cases} max(\sum_{h\in A} W_h q_{hi} x_i - [aC_i + (1-a)t_i]x_i)\ \forall i \in Sc \\ 0 \le a \le 1 \\ t_i = \frac{t_{i\_rel}}{T_{max}}\ and\ 0 \le t_{i\_rel} \le T_{max} \\ C_i = \frac{C_{i\_rel}}{C_{max}}\ and\ 0 \le t_{i\_rel} \le C_{max} \\ q_{hi} = \frac{q_{hi\_rel}}{Q_{max}}\ and\ 0 \le q_{hi_{rel}} \le Q_{max} \\ x = 1\ if\ component\ selected\ else\ x = 0 \end{cases} \quad (7)$$

Where

A: set of software quality characteristics;

Sc: set of available component;

$q_h i$: the standard level of the quality attribute;

h∈ A for component *i*;

C_i: Standardized cost of maintenance of the component i

C_{i_rel}: relative cost generated by component i;

C_{max}: maximum cost achieved by one of the selected components;

t_i: Standardized adaptation and maintenance time of the component i;

t_{i_rel} : Relative time, generated by component i;

T_{max}: is the maximum time achieved by one of the selected components;

α : Coefficient of adaptation.

IV. Validation Phase

Current software systems have become increasingly complex and large size [20], [21]**.**They can integrate a large number of software components during their construction. To benefit from a quality system, the best components of library must be selected. Indeed in [21], the authors notice the need for the management and use of quality software in order to avoid computer weakness in software systems that have become complex and large size. They developed a framework to assess the quality of software and to support decision-making in the engineering of large-scale critical systems.

Also those components must adapt to the needs and quality requirements of users. Our aim is to determine quality attributes that affect the reuse of software components. Our selection approach is described by the SlectCompo algorithm (See the algorithm Fig. 2).

A. Presentation of the Algorithm

Our algorithm allows to select from a set of available components (Cd), the optimized and selected component (Cos). The different steps of this algorithm are described in [7]. In order to show the practical utility of our model, we performed an experiment on a set of components with non-functional requirements.

SelectCompo Algorithm
1. *Input*: Set of available components (Cd)
2. *Output*: Optimized component and selected (Cos)
3. Begin
4. *While* (needs and requirements expressed in Cd) do
5. *For* i= 1 to Component (Cd) do
6. *Select* (the component Ci)
7. Put in the list of selected components (Cs)
8. **Endfo**r
9. **EndWhile**
10. *If* ((conditionsCaracterisks Filled) and (cost and relative time in intervals required) **then**
11. **For** i= 1 to Component(Cs) **do**
12. *evaluate* (thequality value of the selected components)
13. **End if**
13. **If** (SatisfactionQuality) **then**
14. *Optimize* (the factors of cost and time of adaptation)
15. *Select* (the component(Cos))
16. *else* choose another component in the set Cs
17. **End if**
18.**End**

Fig. 2. Pseudo Code of SelectCompo.

B. Definitions of Characteristic Weights

Depending on the needs of the user, we define the importance of each characteristic in relation to the others. This allows us to determine the different weights of features defining quality criteria such as reliability, usability, security and maintainability using the Hierarchical Process Analysis (AHP) method. First we considered the binary comparison matrix of the different characteristics of the expert-defined component in Table I[1].

TABLE I. Binary Comparison Matrix of the Features of the Software Components

Binary Comparison Matrix	reliability	usability	security	Maintainability
reliability	**1**	**3**	**2**	**0,25**
usability	**0,3333**	**1**	**0,3333**	**1**
security	**0,5**	**3**	**1**	**0,2**
maintainability	**4**	**1**	**5**	**1**

[1]https://www.uqtr.ca/~gelinare/Logistique/ahp.doc

Applying the AHP method, we obtain the weights of the different features which are quality attributes of the software component in Table II as follows:

TABLE II. WEIGHT OF CHARACTERISTICS

weights of	**characteristics**			
characteristics	reliability	usability	security	maintainability
weights	0,2221	0,2221	0,1656	0,4547

C. Selection of the Best Component

According to the AHP method, the sum total of the weight of the characteristics is equal to 1. We admit that we want to make the selection in a set of p components available in a library noted *Cd*. The user then defines his functional requirements and his non-functional requirements which are related to the quality of operation of the software components. The selection of the best component is done in two stages.

In the first part, the candidate components are selected based on the functional requirements. To do this, a matching between the requirements expressed by the client and the functionality of the library components is made.

We obtain a list (Cs) of k selected candidate components and verifying the defined condition (with $k \leq p$)(line 7 of the algorithm).

If we take for example, k = 5 we define the following set:

Cs ={component1, component2, component3, component4, component5}

The next step is to evaluate the quality of the components of the list (Cs) by binary comparison of their characteristics see Table I. For the realization of our experimentation, we considered the software components of the componentSource [2] platform to define the values of the different parameters financial cost and maintenance effort. This software component market has components whose dollar financial costs (Us dollars $) are values in the interval [195.02 ; 3,000.00].Maintenance efforts must be made within a maximum of fifteen (15) days.

Normalized values are obtained by doing the ratio of the relative value and the maximum value. For any available component i of the library, we have:

$$t_i = \frac{t_{i\_rel}}{T_{max}} \text{ et } c_i = \frac{c_{i\_rel}}{c_{max}} \quad (8)$$

Where

t_i: normalized time of the component i ;

t_{i_rel}: Relative time generated by component i;

T_{max} : the maximum time achieved by one of the selected components;

C_i: Normalized cost of the component i

C_{i_rel}: relative cost generated by component i;

Cmax: maximum cost achieved by one of the selected components;

Relative and standardized financial costs then relative and standardized maintenance efforts of the components are grouped into Table III.

TABLE III. RELATIVE STANDARDIZED COST AND MAINTENANCE EFFORTS

	Component1	**Component2**	**Component3**	**Component4**	**Component5**
Relative Costs in $	975,1	343,18	1000	2240,1	2032,15
Relative Maintenance effort within days	12	5	9	10	2
Standardized cost	0,42396	0,14921	0,43478	0,97396	0,88354
Standardized maintenance efforts	0,8	0,33333	0,6	0,6667	0,13333

We search to maximize the quality values of the components by our model (see equation (8)). To do the experimentation, we will use an optimization tool.

D. Utilization a Solver

We move on to the practical phase of evaluating the model with an optimization tool. This type of software uses the combination of modeling capabilities and the power of Cplex.

Optimizers to quickly and easily solve important and difficult problems for users. This step gives two (2) possibilities of solution for the selection of software components.

- Using the CPLEX studio 12.8.0 solver from IBM and its OPL language, the solution proposed is: *X=[0 , 0 , 0 , 0 , 1]; C =[0.42396 , 0.14921, 0.43478, 0.97396, 0.88354]; T=[0.8, 0.33333, 0.6, 0.6667, 0.13333]*. This means that component 5 is the best result and retained. It is therefore the best choice among this set of *Cs* elements. This means that component 5 is the best result and kept. So this is the best choice among these elements of the set Cs. We retain then *Cos = {component5}*.
- Otherwise, we take back the selection in the list (Cs).

V. INTERPRETATION AND DISCUSSION

In the first part, we interpret our model and the results of our experimentation. Then in the second part we compare our results to the works of other authors cited above in this document.

A. Interpretation

We want our model to assess the quality of software components in a large repository and with a large number of requirements. It takes into account the quality characteristics

[2]https://www.componentsource.com/fr/

of ISO / IEC 9126, on the one hand and on the other hand factors of financial cost and maintenance effort.

In the case where the parameter α is zero (α=0), the term containing the financial cost is canceled. The equation (4) becomes:

$$S_i = \sum_{h \epsilon A} w_h \, q_{hi} x_i - [(1-a)t_i]x_i$$

$$and \; \forall \; i \in Sc \qquad (9)$$

We will be in the presence of a library of open source software components. Then the selection will be done in a set of open source components.

In the opposite case, if α is 1 (α = 1), we have some ready-to-use components, that is the "cost" components. The equation (4) becomes:

$$S_i = \sum_{h \epsilon A} w_h \, q_{hi} x_i - [ac_i]x_i$$

$$and \; \forall \; i \in Sc \qquad (10)$$

In this case, the selection depends only on quality characteristics and cost. This corresponds to the model developed in [8].

In short, our developed approach corresponds to a generalization of the selection of components in a large repository with many requirements taking into account the quality characteristics, cost factors and the maintenance effort. We present the results of our simulations with the graphical representations Fig. 3 and Fig. 4. Those different forms of graphs will allow us to better interpret and explain our results. Then we will make a comparative study of those results with previous works.

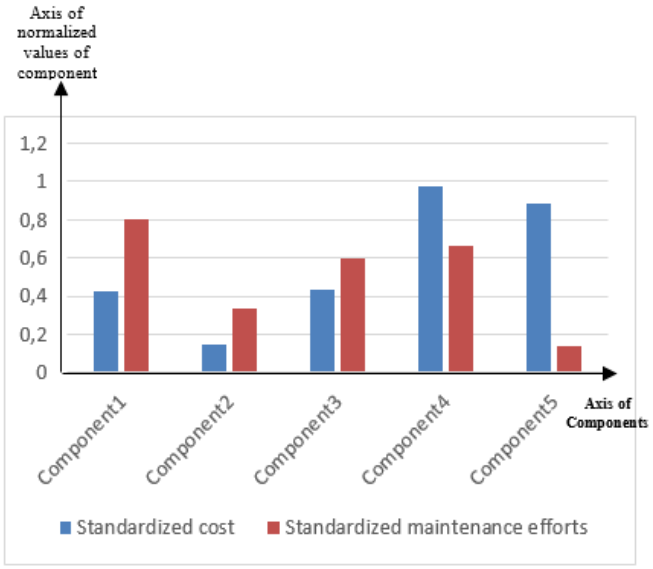

Fig. 3. Representation of Standardized Costs and Standardized Maintenance Efforts.

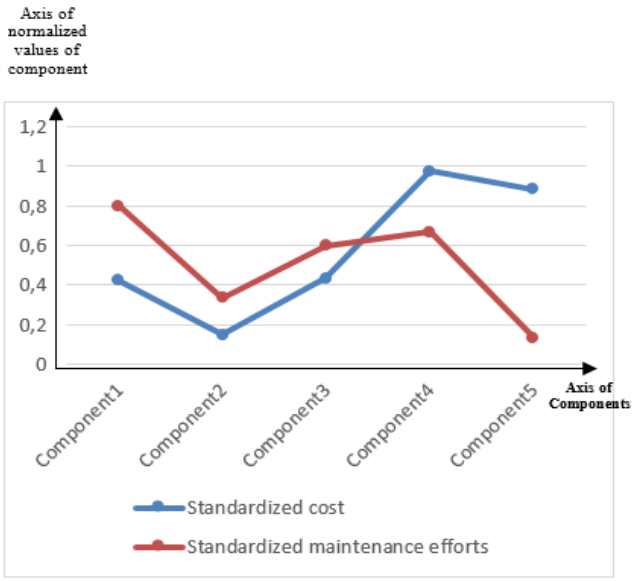

Fig. 4. Representation of Standardized Costs and Standardized Maintenance Efforts.

Our results show that the software component admitting a high maintenance effort, is not selected whatever the financial cost. Our study also argues that the maintenance effort must be of low value. So the components selected are those admitting a low maintenance effort like software component 5.

B. Discussion

In this section, we compare our results with the work of the authors [2], [8] and [9]. In [2], the authors realized the selection of software components based on quality criteria. The method used is the analytical network process. This study does not take into account the particular qualities of the features of the component. The result of the experimentation is based on an overall judgment of the quality of the components by binary comparison between them. This technique uses manual calculations, very complex and very tedious. This results in huge waste of time to select the relevant components.

The effectiveness of works in [8] lies in the consideration of quality characteristics of ISO / IEC 9126 and the factor defining the financial cost in the selection process. Using the binary comparison technique of quality attribute, the authors added important elements in decision making to select software components. This technique made it possible to assess the functional and non-functional qualities of software components. Also, experimenting with the Cplex solver to automatically solve the developed model accelerates the results. Their works are saving of time a greatly compared to the work of the authors of [2].

Our work shows two important results. On the one hand, we obtain a saving time of less than a minute to automatically select the components with the same quality values as in [8]. On the other hand we have shown that the choice of software component selection depends on the maintenance effort and the financial cost.

So our results are in concordance with the work in [9]. Therefore, the maintenance effort parameter that we have associated in our model is an important factor and impacts decision-making for the selection of relevant software components in a library or in a component market. In addition, our results are in agreement with those of [9] where the authors have shown that if the time and effort required to understand a component increases, the reusability decreases.

VI. Conclusion and Perspectives

We have developed a model that allows to optimize the selection of software components based on their quality. Our approach corresponds to the generalization of the selection of components in a large repository with many requirements taking into account the quality characteristics, cost factors and the maintenance effort. In this paper we simulated our model. Our results show that the reusability of components depends on several factors. There are the quality characteristics, the financial cost and the maintenance effort. The lower the maintenance effort, the more the component is reused.

Our model shows that the maintenance effort is an important factor that influences decision-making in the process of selecting software components from component markets.

In the future work, we would like to develop a model that will allow us to assess the dependency ratio of the different factors having a correlation in the selection process of the software components of the libraries. We will also work on improving our selection process and our algorithm which support it to increase the quality of the selected components.

References

[1] Yahlali Mebarka "Assemblage d'une application à base de composants : approche de calcul de qualité d'une composition", these,Université des sciences et de la technologie d'Oran Mohamed Boudiaf, 2015.

[2] Shah Nazir, Sajid Anwar, Sher Afzal Khan,Sara Shahzad, Muhammad Ali, Rohul Amin, Muhammad Nawaz, Pavlos Lazaridis, John Cosmas, Software component selection based on quality criteria using analytic network process,. Abstract and Applied Analysis Volume 2014.

[3] A. Rawashdeh and B. Matalkah, "A new software quality model for evaluating COTS components," Journal of Computer Science, vol. 2, pp. 373–381, 2006.

[4] Pfarr, T. and J.E. Reis,. "The integration of COTS/GOTS within NASA's HST command and control system", 2002.

[5] José P. Miguel , David Mauricio and Glen Rodríguez " A Review of Software Quality Models for the Evaluation of Software Products"International Journal of Software Engineering & Applications (IJSEA), Vol.5, No.6, November 2014.

[6] Shabnam Gholamshahi Seyed Mohammad Hossein Hasheminejad, « Software component identification and selection: A research review », 31 October 2018, https://doi.org/10.1002/spe.2656.

[7] Koffi Kouakou Ive Arsene, Samassi Adama, Kimou KouadioKonan Marcellin Brou" Proposal of Automatic Methods for the Reuse of Software Components in a Library ", International Journal of Advanced Computer Science and Applications 10(2) · January 2019, DOI: 10.14569/IJACSA.2019.0100208.

[8] Pande, CJ Garcia, D Pant, "Optimal Component Selection for Component Based Software Development using Pliability Metric", ACM SIGSOFT Software Engineering Notes, January 2013.

[9] Neha Sadana, Surender Dhaiya, Manjot Singh Ahuja, "A Metric for Assessing Reusability of Software Components", International Journal of Computer Application, R S. Publication, Vol. 1, Issue 4, pp. 98-108, Feb. 2014.

[10] N. Sadana, S. Dhaiya, M.S. Ahuja, ''An empirical review on factors affecting reusability of programs in software engineering,'' International Jorunal of Computing and Corporate Research, vol.3, July 2013.

[11] Petersen K., Badampudi D., Shah S., Wnuk K., Gorschek T., Papatheocharous E., Axelsson J., Sentilles S., Crnkovic I. and Cicchetti A.: Choosing Component Origins for Software Intensive Systems: In-house, COTS, OSS or Outsourcing?--A Case Survey. IEEE Transactions on Software Engineering. Vol. PP, No. 99, March, pp.1-1, 2017.

[12] N. Sadana, S. Dhaiya, MS Ahuja, "A Metric for Assessing Reusability of Software Components", International Journal of Computer Application, Issue 4, Volume 1, February 2014.

[13] Bart George, R. Fleurquin, S. Sadou, H. Sahraoui « Un mécanisme de sélection de composants logiciels », juillet 2010.

[14] Lian, X., Zhang, L., Jiang, J. and Goss, W., An approach for optimized feature selection in large-scale software product lines. Journal of System s and Software., pp. 636- 651, 2017.

[15] Amit Rathee, Jitender Kumar Chhabra, "A multi-objective search based approach to identify reusable software components",Journal of Computer Languages 52, p.26–43 ,3 January 2019https://doi.org/10.1016/j.cola.2019.01.006.

[16] Sumit Sharma, Upinder Kaur, Pawanpreet Kaur « Component Recommender System Based on Collaborative Approach in Incremental Development », World Journal of Technology, Engineering and Research, Volume 2, Issue 1, 2017, p 51-59.

[17] Cosmic : mesure de la taille fonctionnelle avec la méthode, https://info.uqam.ca/midi-confs/2017-02-22-cosmic.pdf.

[18] Sylvie Trudel, mesure de la taille fonctionnelle avec la méthode cosmic (iso 19761): recherches récentes et applications industrielles » , conférence du latece 2012.

[19] Christof Ebert and Hassan Soubra, « functional size estimation technologies for software maintenance »IEEE software technology, November/December, 2014.

[20] Rakesh Garg, R. K. Sharma, Kapil Sharma, Ramesh Kumar, **«** Identification, selection and evaluation of COTS selection criteria using Fuzzy Set Theory », International Journal of Computing and ICT Research, Vol. 9, Issue 2, December 2015, P 21-36.

[21] Gabriella Carrozza, Roberto Pietrantuono, Stefano Russo, "A Software Quality Framework for Large-Scale Mission-Critical Systems Engineering", Information and Software Technology,2018, doi: 10.1016/j.infsof.2018.05.009.