Journal Homepage: -www.journalijar.com

INTERNATIONAL JOURNAL OF ADVANCED RESEARCH (IJAR)

Article DOI:10.21474/IJAR01/16258
DOI URL: http://dx.doi.org/10.21474/IJAR01/16258

RESEARCH ARTICLE

VALIDATION OPTIMIZATION ENVIRONMENT FOR IMPROVED SELECTION OF SOFTWARE COMPONENTS: CONCEPTUAL MODELING AND ARCHITECTURE

Koffi Kouakou Ive Arsene[1,2], Brou Konan Marcellin[1,2], Kouamé Appoh[1,2] and Kouamé Abel Assiélou[1,2]

1. Ecole Doctorale Polytechnique de l'INP-HB Yamoussoukro, Côte d'Ivoire.
2. Laboratoire de Recherche en Informatique et Télécommunication (LARIT).

Manuscript Info

Manuscript History
Received: 15 December 2022
Final Accepted: 19 January 2023
Published: February 2023

Key words: -
Qualitative Selection, Software Component, Simulator, Software Environment,Optimization Model

Abstract

Responding to the needs of a user, the selection of software components for building computer applications is a crucial step. In the matter, optimization tools exist to experiment the developed models. However, these tools are not suitable for the user and especially for a non-expert. In this article, we propose to build a software environment validating our optimization model. It is therefore an automatic aid simulator for the choice of software components of websites for the construction of modular systems. By using the method of analysis in particular the unified process of UML, at the end of our conceptual modeling, the diagrams of classes, sequences and activities are obtained. Moreover, a simulator architecture, an operation algorithm and an optimization algorithm capable of improving the choice in the selection of software components among the existing software libraries are built. Our simulator system environment is nearing completion.

Introduction:-

Since the breakthrough of information and communication technology (ICT), software development has strengthened its rise from small industries to larger ones. The uses of software and their demand for customization are clearly increasing. Trivially, the same applies to software development. This is why in such a context, anything that can be taken into account as increase in value is welcome in the development and use of the software. As such, software components have a place of choice for both expert and non-expert developers. For these use cases contributing to the development of the software, we are given to note a lack of sufficient description to be able to guide the developers on the choice of these components. For these use cases contributing to the development of the software, we are given to note a lack of sufficient description to be able to guide the developers on the choice of these software components on the one hand and on the other hand, a random selection of components software in software libraries.Consequently, it follows a degradation of the performances of the system with sometimes conflicts of execution of the software causing slowdowns, blockings and plantings involving problems of operation to computers. This is why we created a mathematical model focused on the qualitative description of software components [6].If the expert developer often manages to understand the mechanism of integration of software components more quickly, this is not at all the case for less or non-expert users.Moreover, if for the expert developers the saving of time constitutes an increase in value, the help of the non-experts in the qualitative choice does not remain about it less.Our research works come in reinforcement those carried out in the field of component-based software engineering (CBSE) [1] [2] [3], also called Component-Based Software Development (CBSD) [4] [5]. They seek to optimize quality on the basis of several criteria specific to the needs of the user and the technical

Corresponding Author:- Koffi Kouakou Ive Arsene
Address:- Ecole Doctorale Polytechnique de l'INP-HB Yamoussoukro, Côte d'Ivoire.

characteristics of the software components to be selected. These criteria will constitute the backbone of the qualitative choice to be made. Our work in this paper focuses on the first part of the development of a software environment for an aid of automatic assistance in software component-based software construction. These are strictly speaking the research activities carried out with the object of the conceptual models and an architecture, all of which are based on the mathematical model of our previous work [6] [7].

More than one definition exists in the literature expressing the concept of software component. The one proposed in the works of [8] and taken up in those of [9] [10] begins by achieving consensus. It emerges that: "a component is a composition unit which specifies its interfaces by contractualization, and which explains its context dependencies. A software component can be deployed independently and is subject to composition by third entities".

This definition refers to the context of development, use and assembly of components, as addressed by our work. Four main sections structure the rest of the paper. The state of the art is the first section. The second is devoted to our general methodology. The third shows our results, in particular the conceptual models in UML, the architecture of our simulator system, the optimization and operating algorithms. The discussion completes this third section. In the last part, we have the conclusion and the perspectives.

State of the art: -

In the field of software engineering, the selection of components that meet customer needs is often problematic. This difficulty is justified by the diversity not only of component supplier sites but also of applications providing the same services. These applications are developed from existing software components by assembling, replacing or updating components in a well-structured software architecture. Another difficulty is that often linked to a suitable selective quality. Research works are being done to value to what extent these software components can be reused optimally. Very varied and different evaluation models and methods are then developed to improve and facilitate the selection process as shown in the work in [11] [12] [13] [14] [15]. This supposes the holding of important tasks before any selection. Thus, research work is empirically manner validated [16] [17] [18] [19]. On the other hand, in other works, authors have used specific tools to evaluate and process the optimization models they have developed [20] [21] [22]. Through their models, they have contributed enormously to the work of improving the selection of software components. In addition, others have developed software to validate their optimization models and illustrate its applicability [23][24][25][26][27]. In [25], it results the development of an optimization method to quantify the reusability of the software component according to quality criteria. This method is a combination of the Fuzzy Analytic Hierarchy Hybrid Process (FAHP). It makes it possible to evaluate the weights of the criteria then to classify the various alternatives on the one hand and on the other hand, to use aquality metric to evaluate the reusability of the software components facilitating their selections. A case study of this hybrid FAHP-Metrics approach is applied to e-commerce website payment gateway components to rank their reusability values. In addition, a web application has been developed automating the process of quantification and selection of relevant information. In [26], the research works are based on an optimization model called multi-objective and integrated optimization. It allows the development and maintenance of a modular system based on software components. It's about an optimization model for multi-objective problems. It takes into account the software maintenance and the decision of the choice of the module supplier or the component to be integrated in a software system. The problem is to know with who to buy the component or to build it internally or externally. Thiswork facilitates the evaluation and selection of software components in large software systems such as Enterprise Resources Planning (ERP). We have for example Odoo, Sage, Microsoft Dynamics, etc. For the validation of the optimization model proposed in their article, a test was carried out on an e-commerce software. In [27], an optimization model called the "credibilist" model with multiple objectives was developed. This model makes it possible to solve the problems of selection of commercial products subjected to many constraints. This approach is a "fuzzy" technique based on a genetic algorithm and the use of an exponential function. It allows to solve the selection of commercial products with uncertain multi-objectives to build a modular software system. Their work made it possible to build a software package for the applicability of the model. This software package is an enterprise resource planning software system for small and medium-sized businesses. In [28], the authors have shown the development of an integrated, nonlinear multi-objective optimization model. It is based on a data envelopment analysis that considers several criteria to assess the suitability of off-the-shelf components or in-house developed components. It helps to rank and select relevant software components by calculating their efficiency scores based on important attributes such as cost, reliability, execution time, delivery time, incompatibility and redundancy. Their objective is to minimize the total cost of the software system to be built. Smart TV software was built as a real case study to illustrate the applicability of their model. These works treat with the optimization of the selection of reusable software components. In spite of

the existence of several optimization tools developed by commercial technologies such as Cplex studio, Lingo, chip v5 on the one hand or on open-source software sites like Comet, Gecode, Choco, etc, on the other hand, application development to validate models does not specifically address the selection of reusable components from software libraries. Moreover, these optimization tools have weaknesses and are therefore less suitable. Indeed, difficulties related to the modeling of constraints, programming, use, understanding of syntax and vocabulary for non-expert users are noted. In addition, the interfaces presented are not user-friendly and not ergonomic for these non-specialist users [29]. As regards the software, applications are built to solve problems in particular domains. Yet, research has shown that the proposed models and the developed applications work well under certain specific conditions. Studies have shown that when the assumptions and conditions vary in higher numbers, these proposed tools no longer provide the expected performance [30]. In the field of software component selection, simulators have not yet been developed to improve the selection process and to serve as an automatic assistance aid for non-specialist users. To solve the difficulties mentioned above, we started building a software environment to validate our research work. This software application will search for an ordered list of relevant software components based on given quality criteria. This involves developing a software environment that will serve as automatic assistance of decision-making that accomplishes the tasks listed below:

- improve the process of selecting reusable components,
- improve the selection of software components by weighting the characteristics according to the type of software to be built,
- present a user-friendly interface for the non-expert user,
- give an ordered list of relevant software components based on quality criteria;
- validate our model.

Methodology:-

The search method used is the one that identifies reusable software components from markets and component sites (ComponentSource, WordPress and Zyro). ComponentSource is a marketplace of approximately 1861 proprietary and reusable software components. As for WordPress and Zyro, they allow you to quickly and efficiently create online websites with reusable components. However, these components do not have the same descriptions. In the description of software components, while componentSource adds the cost of the component, Wordpress presents the overall cost of the site after completion. Furthermore, version release dates and updates are not available on the Zyro site. Based on the work described in the chapter on the state of the artand our mathematical optimization model, the implementation of the targeted environment of simulator requires the creation of our conceptual models and an adequate architecture. The mathematical model explained in our previous works [6] [7], is considered as the general engine of this simulator. In a context of very varied and diversified representation of software components as noted, the description in a common format of these components is more than necessary. This brings to construct our class, sequence and activity diagrams according to the PU/UML unified process method. The architecture of the simulation system highlights its execution environment, and more precisely its internal interfaces on the one hand and on the other hand, the interactions between the user (eg the non-expert developer, or moderately expert, etc) and the whole system. This entire system of simulator comprises two internal modules: an evaluation system and a database, both communicating.

Presentation of the optimization model: -

In previousworks, wedefined an optimization model to determine the quality value of reusable components in a software library [6],[7]

Presentation of the MathematicalOptimization Model

This model allows to maximize the value of the component quality according to the quality characteristics, the financial cost and the adaptation time of the component [7].

$$\begin{cases} max(\sum_{h\in A} W_h q_{hi} x_i - [aC_i + (1-a)t_i]x_i) \ \ \forall i \in Sc \\ 0 \le a \le 1 \\ t_i = \dfrac{t_{i\_rel}}{T_{max}} \ et \ \ 0 \le t_{i\_rel} \le T_{max} \\ C_i = \dfrac{C_{i\_rel}}{C_{max}} \ \ et \ \ 0 \le t_{i\_rel} \le C_{max} \\ q_{hi} = \dfrac{q_{hi\_rel}}{Q_{max}} \ \ et \ \ 0 \le q_{hi_{rel}} \le Q_{max} \\ x = 1 \ seletionné \ \ sinon \ x = 0 \end{cases} \quad (1)$$

Where
A: set of software quality characteristics;
Sc: set of available components;
q_{hi}: 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.

We have the objective function from the following system of equation (1):

$$S_i = \sum_{h \in A} w_h q_{hi} x_i - \left[\alpha c_i + (1-\alpha) t_i \right] x_i \quad \text{and } \forall i \in Sc \tag{2}$$

We have already carried out the experimentation of the model with the optimization tool ILOG Cplex Studio by defining the decision variables and the constraints.

Constraints
The constraints are as follows:

$$\begin{aligned} & 0 \le a \le 1 \\ & t_i \in [\![t_{min} ; T_{max}]\!] t_i = \frac{t_{i_{rel}}}{T_{max}} \text{ and } \quad 0 \le t_i \le 1 \\ & T_{max} = 15 \text{ jours} = 1.296 * 10^3 \text{ in second} \\ & C_i = \frac{C_{i_{rel}}}{C_{max}} \text{ et } \quad 0 \le C_i \le 1 \\ & \sum_{h \in A} w_h = 1 \\ & Q_{max} = 5 \end{aligned} \tag{3}$$

Decision variables
Decision variables are defined as follows:

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

Result and Discussion:-

Following our previous work and reinforcing some of its parts, we propose an optimized software component selection process, based on the developed mathematical optimization model. This process is leaned on our simulator. In terms of results, we have in this article an architecture of this simulator system supported by an algorithm of its operation. Also, we have produced diagrams (eg of activities, sequences and classes) according to the UP/UML(Unified Process/Unified Modeling Language) formalism, highlighting the central role of the expert and non-expert user. By order of interest and in order to promote the development of software applications through assistance, we favor the non-expert user before the expert.

Results:-

In consideration of the weaknesses noted above using the optimization tools, we opted for the development of a simulator capable of evaluating the quality of the components and improving their selections. It is good to note the character of standardization concerning the descriptive presentation of the characteristics of these components towards which our work will have to lead us. As previously noted, the results are presented progressively in what follows.

Database

We constructed the class diagram below on fig 1

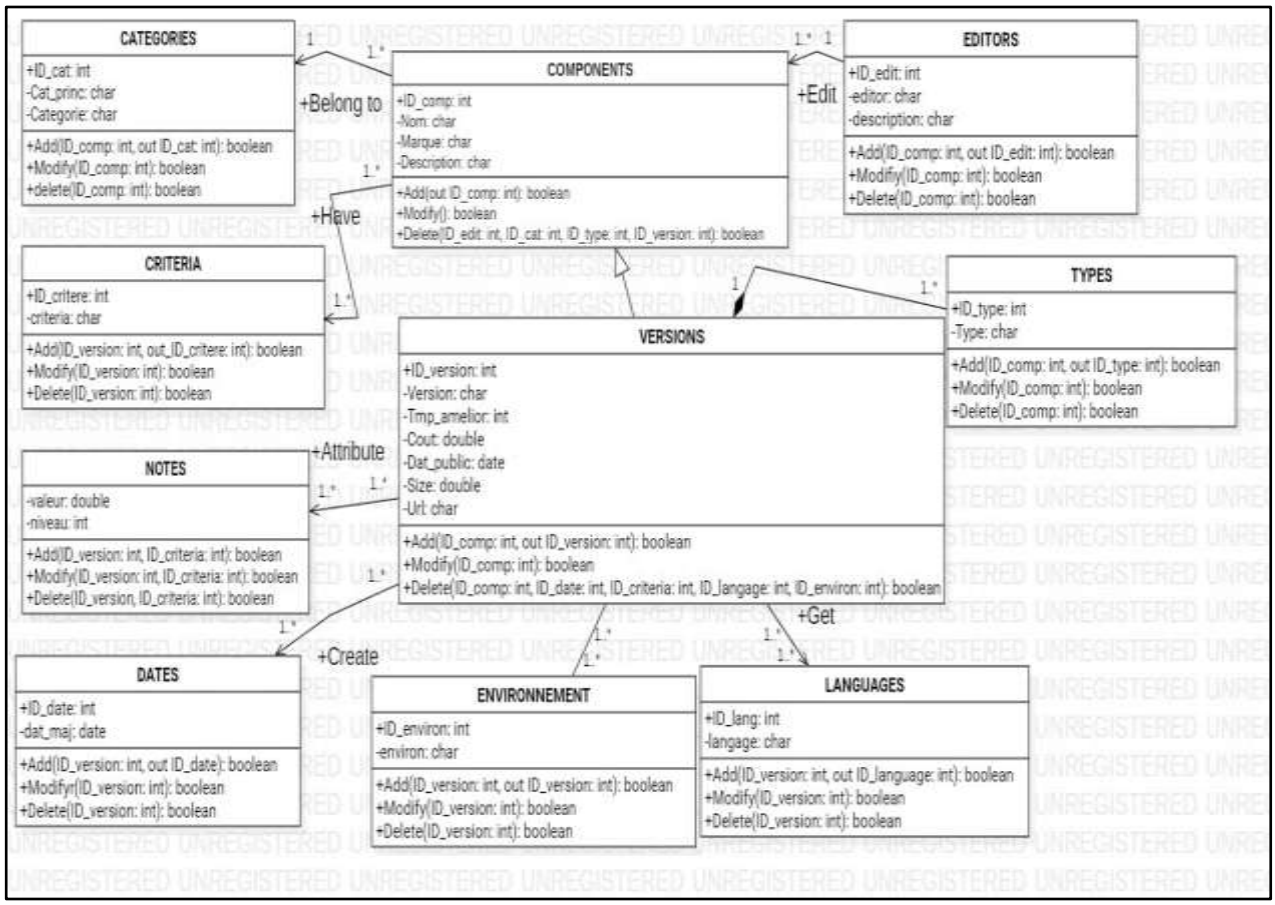
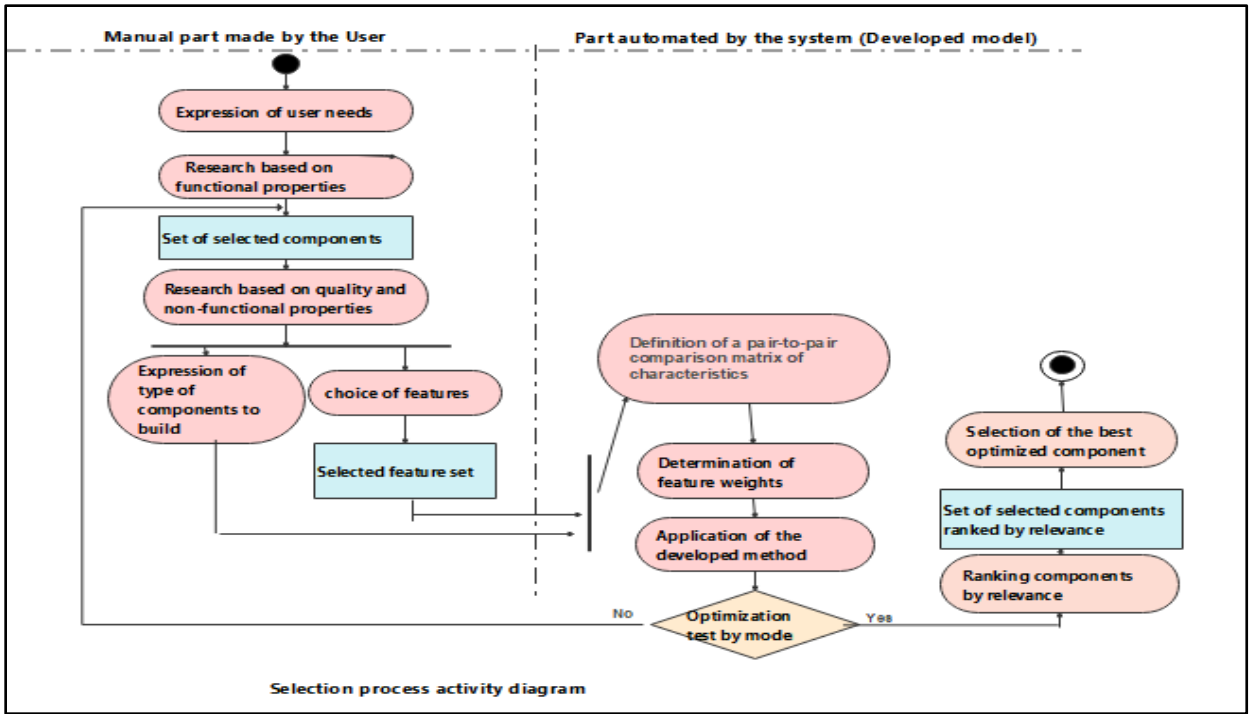
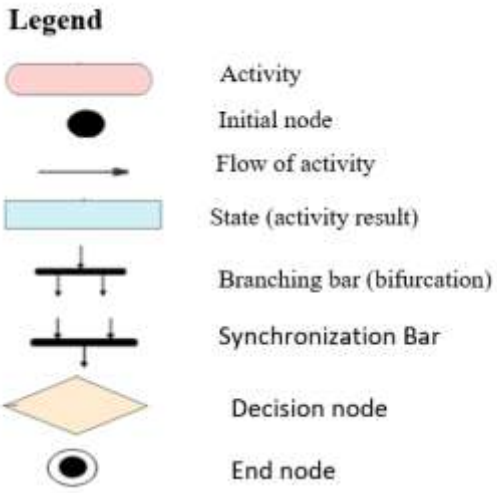


Fig 1:- Class diagram obtained from componentSource and WordPress objects.

Proposal of software component selection process

The component selection process describes the different phases carried out for the evaluation of the characteristics and the selection of an ordered list of relevant software components on the basis of quality criteria. It is represented by the activity diagram below see fig2.





This diagram is divided into 2 main parts, one of which is semi-manual and the other automatic:

1. The first part, semi-manual describes the quality needs expressed by the user for the construction of his software on the one hand and on the other hand, because of his interaction of communication with the interface of the application in the future.
2. The second part, automatic, takes into account the developed optimization model and the Analytical Hierarchy Process (AHP) method. It allows to calculate the scores and takes care of the scheduling of the score list according to the software components by relevance. A database is used for storing data relating to the software components. It is of paramount importance to our entire software system. This module is a simulator, an automatic assistance tool for the selection of components while interacting with the database. For the calculation of the scores, this database provides the general engine of the software system with the values of parameters such as the financial cost and the maintenance effort of the software components.

Functional architecture of the simulator

In this section, we describe the architecture of our simulator, followed by its functioning.

Simulator architecture

The basic architecture recommended here observes that of three-tier applications. Thus, we have defined a database server, a web server or http (HyperTextTransfert Protocol) and a presentation server. These three servers are in perfect projection in the functional architecture presented in figure fig 3. Still via this figure, three modules make up the functional architecture of our simulator system. These are the Human-Machine Interface (HMI), the quality-of-service evaluation module called MEQoS and the database. Let's do a horizontal reading of this figure fig3, starting from the users towards the database. Represented by the red colored frame, this first module made up of the HMI makes it possible to recover on the client station, the requests and needs specified by the user. The second module is the MEQoS framed in green color. This part designates the web server and the application or presentation server. It includes the driving elements, the programs developed in relation to our mathematical optimization model. Part of MEQoS serves as the core engine for our entire Evaluation system. This part is simply called ESE, which stands for Evaluation System Engine. Focused on the scores, this engine is more general, using the mathematical model [6] [7]. The ESE takes into account the AHP method. It makes it possible to determine the weights of the quality characteristics of the components on the one hand and on the other hand, and to calculate the scores in [6]. Clearly, the objective function of the mathematical model of equation (2) made it possible to calculate the scores of the software components of the database according to the indicators of quality, cost and adaptation effort on the one hand and on the other hand, to give a list in order of relevance of the candidate components for the selection. The database management system is the third functional module. It allows the manipulation of our database on components. To be fed, this future database will be associated with a tool for extracting information from websites from their URLs.

The general functional architecture of the simulator is presented in figure fig 3.

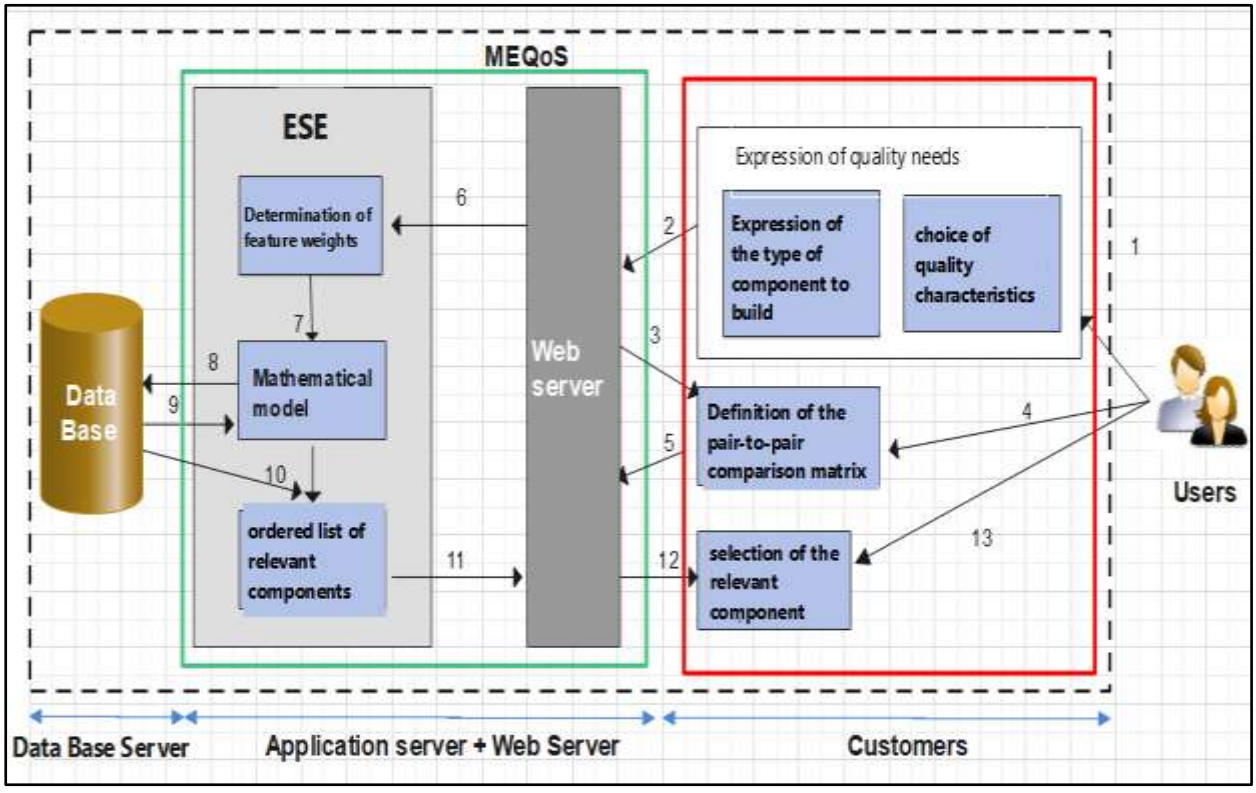


Fig 3: -Synoptic and architectural view of the software simulator.

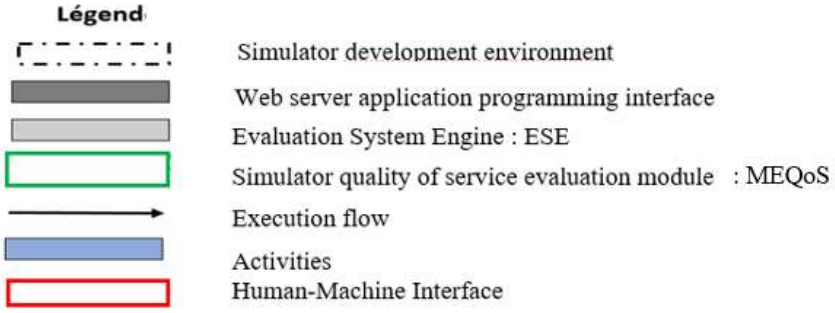


Fig4:- Pseudo code of SearchCompo algorithm.

Functioning of the selection assistance tool

This section presents the algorithm named RearchCompo via figure fig 4 to lean to the functioning of our automatic assistance tool under construction**.**

Presentation of the algorithm and operation

Presentation of the algorithm

The SearchCompo algorithm aims at selecting among a set of available components (Cd), the relevant component in an ordered list named ListComp[ ] of software components.

SearchCompo algorithm

Input: Pair-to-pair comparison matrix ($M_{[\,][\,]}$)
Output: Ordered list of relevant components from the database ListComp[ ])

```
Begin
 While (needs and requirements expressed in Cd) do
     For i from 1 to NbreComponent do
       For j from 1 to NberCharacteristics do
         Fill in the comparison matrix (M[i] [j]))
         Calculate the consistency ratio (Rc)
         If (Rc ≤ 10%) then
          Determine the weight of each feature W[j]
        End if
      else
       Resume comparison matrix (M[i] [j])
Endfor
        Calculate the score of each component in Cd
        Fill the list of components (ListeComp[i])
Endfor
    Give the ordered list of relevant components (ListComp[ ])
EndWhile
End
```

Algorithm and funtioning

The algorithm takes as input the pair-to-pair comparison matrix (M[i] [j]) defined by the user after the specification of the type of software system to be built and the choice of the characteristics in the set of characteristics Ch to be taken into account. The AHP method evaluates the consistency of the judgment of the defined matrix. In accordance with this method, if the judgment ratio called ***Rc*** is less than or equal to 10%, the defined matrix is acceptable and the process of calculating the weights of the different characteristics continues by calculating the weight W[j] of each characteristic. If the ratio Rc is strictly greater than 10%, the pair-to-pair comparison matrix (M[i][j]) is redefined. After determining the weights, the MSE calculates the scores of the available components in the set of available components Cd in the database. Information about the available components is extracted from the software library to fill the database using an extraction tool. The ESE system sorts and orders the calculated scores. The

database is responsible for classifying by order of relevance the software components linked to the calculated scores. The algorithm displays the list of components in the list named ListComp[ ].

Functioning of simulator

The architecture of the simulator will be developed based on the Model-View-Controller (MVC). The design pattern goal of MVC is the flexible separation of data, views, and actions. Therefore, in our project, this structure can be used as part of our simulator. The model manages the data. Its role is to retrieve information from the database, organize it and assemble it so that it can then be processed by the controller. This Controller manages the logic of the code that makes the decisions. It returns those decisions to the view for display. In thirteen (13) steps, we describe the operation of our simulator in the following. Each of the 13 steps is in correspondence with the execution flow of figure fig3 with the same numbering.

Step 1: The user expresses his functional and quality needs according to the type of software to be built for the definition of the judgment matrix.

Step 2: The user opens the home page. It makes a form request to the system from the browser for the construction of the pair-to-pair matrix.

Step 3: The request to the system being made, the web server will generate a form from the user interface. As a reminder, our simulator is composed of a web server, a database server and an application server under a given operating system. The web server, representing the interface between the client station and the application server, receives the requests and files sent via the browser. It searches for the requested page and sends it back to the browser. This part is the module represented in green color on figure fig 3. Then we have the third part which is the database server

Step4: The system returns the software component quality characteristics pair-to-pair comparison matrix form to the user. This matrix makes it possible to determine the weights of the characteristics with the method of the hierarchical analysis process (AHP) developed by Saaty in 1980 [31]. For the pair-by-pair weighting of the alternative criteria, we used the scales of values defined by Saaty (1977) in [32] and by Saaty (1980) in [33], [34]. These have been supplemented and improved in [35]. These scales are used to assign numerical scores to the characteristics of software components in the 9-point binary comparison matrix based on their importance and user-defined preference. This makes it possible to clearly express the needs of the user and to center the selection on the latter. We obtain a new complete scale of values that we use in the rest of our work. After the construction of the pair-to-pair comparison matrix, the mathematical models and the AHP method are used to calculate the maximum eigenvalue λ_{max} of the judgment matrix, the consistency index $IC = \frac{\lambda_{max} - n}{n-1}$ where n is the number of comparisons, the ratio of the judgment $Rc = \frac{IC}{RI}$, $with\ Rc \leq 0,1$ and determination of the weights of the quality characteristics of the software components by the system.

If $Rc > 0,1$ then the matrix is to be reconstructed. The form is returned to the Quality of Service Evaluation Module (MEQoS). This module designates the application server. It includes applications developed in java and python for project management and the Evaluation System Engine.

Step 5: The MEQoS module receives feature data through the web server API and then passes it to the ESE module. This ESE is based on the AHP method and the mathematical optimization model with equation (2)for the purpose of the determination of the weights of the characteristics of the software components.

Step 6: The ESE module receives the elements of the matrix provided by the web server API. It thereafter determines the weight of each of the characteristics due to their quality.

Step 7: The weights of the characteristics being determined; they are retrieved by the mathematical optimization model part of the ESE module. This part calculates the quality score of each of the candidate components to be selected in interaction with the database. The component score values will be stored in a called list S_i and will be associated with the different software components in the database.

Step 8: The simulator engine interacts with the database server to retrieve the costs and values of the maintenance efforts of the software components stored in it.

Step 9: The database server sends back to the ESE module the data requested in the part of the mathematical model. Scores are calculated taking into account all quality indicators. These are the rating q_{hi} , the characteristic h of componenti, the cost C_i of component i and the adaptation effort t_i of component i.

Step 10: The scores calculated by the ESE module containing the mathematical optimization model are ordered and associated with the software components of the database via a connection. So, the user can make the component selection request according to the given quality criteria. The system provides it with a list named ListComp[ ] of relevant software components. These components are associated with the various calculated scores S_i.

Step 11: The ordered list of software components by relevance is passed from the application server to the web server along with the data to be displayed.

Step 12: The web server transmits the received data to the client (user) browser. Thus, from the workstation, the user can retrieve the records in the form of a Tableview in a selectable and displayable list.

Step 13: The user selects the relevant and best suited software component for his software system to be built from his client station.

The functioning of the simulator is summarized by the sequence diagram fig 4 below.

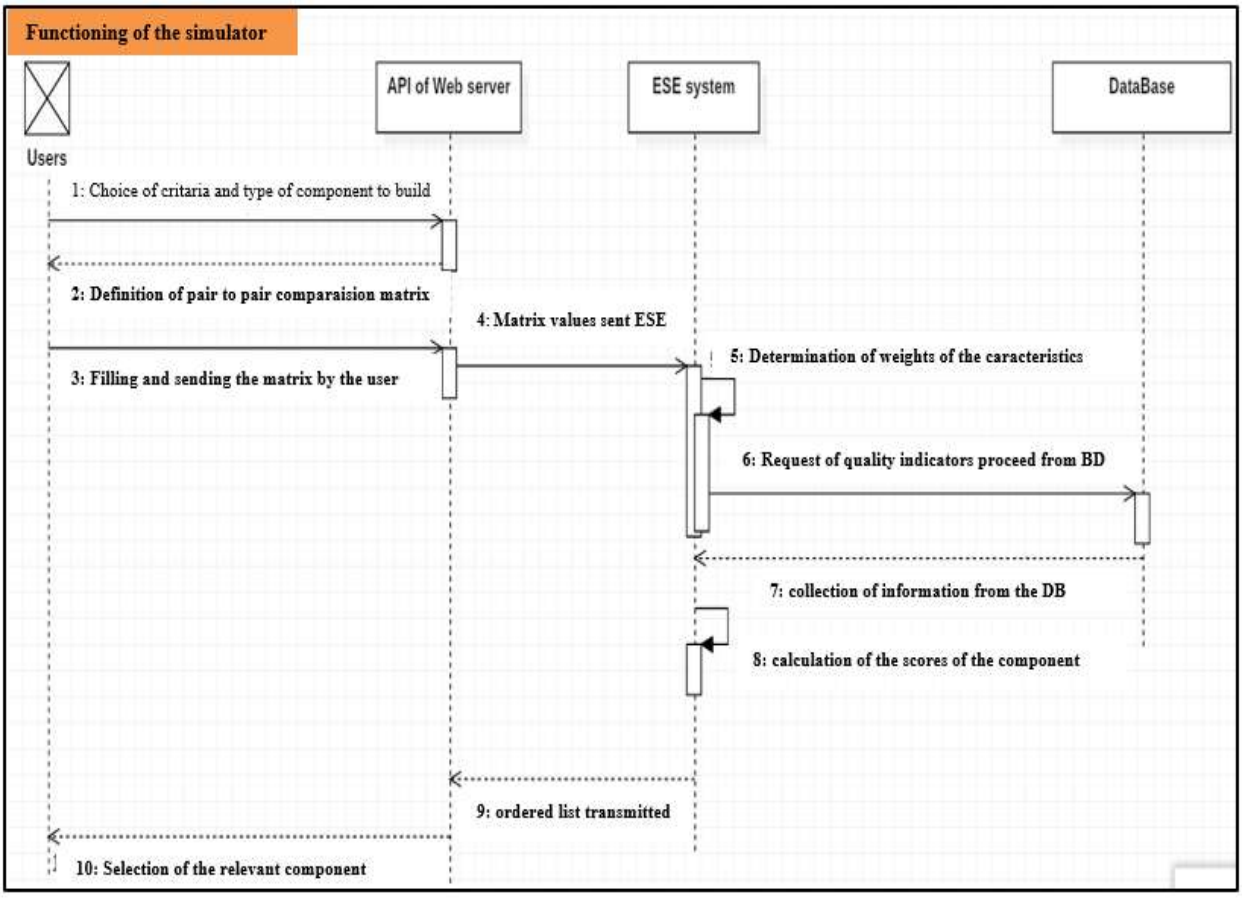


Fig 5:- Functioning of the software simulator.

We will then present the impact of our simulator on the development of component-based software applications and in society**.**

Interest of the software environment and discussion

This software environment tool ultimately makes it possible to virtually reproduce the selection process and compare different alternatives after calculating their score. It also makes it possible to study the factors that impact the quality of the components to be selected. Therefore, the available components will be listed in order of preference. This software will then be used to facilitate and improve the decision-making of developers, companies and non-expert customers in the realization of their projects with quality components. Indeed, it will present two (2) user interfaces. The first is an interface for the administrator who is responsible for updating new versions and deleting obsolete versions. The second is the interface for the non-expert user who can consult and select components according to the expression of his needs. The different users will be able to choose free components and/or proprietary components.

Our application in the long term will be able to answer the question relating to the preferential components, and the types of components easier to lay out more than others. Moreover, we are reassured that the modular structural base of our solution allows its easy extension. Thus, following a test bench over time in the medium and long term (5 years) to our application, as soon as it is completed, we expect to receive feedback of users. This could allow to extend following an investigation, the typologies of use. The field of use could be extended to manufacturers of recognized software with a high reputation in the development of various components. For now, in order of interest and in order to promote the development of software applications by assistance, our focus is on the non-expert user before the expert. The graphic assistance chosen for non-experts will undoubtedly have to experience improvements beyond what we have currently stopped, and this above all due to technological progress**.**

Conclusion And Perspectives:-

Selecting relevant software components that meet the needs of the user is an important task. These components are notoriously diverse, often providing the same services. The choice, especially when it is made randomly or on successive trials, then becomes very complex for the user. In this article, we have designed a software environment validating the optimization model taking into account the quality indicators related to financial cost factors, maintenance effort and component characteristics. Among other results, we have the architecture of our simulator described, the class diagram in the constitution of our database of criteria and indicators targeting the quality of choice, the digrams of activities, sequence and our algorithm of operation in formal algorithm description language (LDFA). The extensibility of our system to be built is based on that of the Java language (Java Fx with the scene Builder graphical tool). The technical choice made on the MySQL DBMS partly consecrates the free nature of our tool. An Apache web server will be used. The objective of our application, in the long term, will be to evaluate the components of the database on the one hand and on the other hand, to facilitate the search for an ordered list of relevant software components according to given quality criteria by the user. Our application system will have a user-friendly and ergonomic human-machine interface environment framework whose development is focused on the model-view-controller architecture. As such, a reinforcement in aid is provided for users (single user, programmers and companies) via a library of free, reliable and downloadable software components. Our simulator system environment is under construction and almost completed for final validation.

References:-

1. ArvinderKaur, Kulvinder Singh Mann, « Component Selection for Component based Software Engineering », International Journal of Computer Applications (0975 – 8887), Volume 2 – No.1, May 2010
2. Ramandeep Kaur, Stuti Arora, P.C. Jha, Sushila Madan, « Fuzzy Multi-criteriaApproach For Component Selection of FaultTolerant Software System under Consensus Recovery Block Scheme » International Conferenceon Advanced Computing Technologies and Applications (ICACTA2015), doi : 10.1016/j.procs.2015.03.169
3. Amit Rathee, Jitender Kumar Chhabra, « A multi-objective searchbasedapproach to identifyreusable software components » Journal of Computer Languages 52 (2019) 26–43, https://doi.org/10.1016/j.cola.2019.01.006
4. Pankaj Gupta, Mukesh Kumar Mehlawat, Divya Mahajan, « Multi-objective optimizationframework for software maintenance, component evaluation and selectioninvolving outsourcing, redundancy and customer to customerrelationship", InformationSciences483(2019)21–52, 2019 https://doi.org/10.1016/j.ins.2019.01.017
5. Jyoti sharmaa, Arvind Kumarb, Kavita, « A Design Based New Reusable Software Process Model For Component BasedDevelopmentEnvironment », International Conference on Computational Modeling and Security, 2016 ; doi : 10.1016/j.procs.2016.05.283
6. Koffi Kouakou Ive Arsene, Samassi Adama, Kimou Kouadio Konan 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) • January2019, doi : 10.14569/IJACSA.2019.0100208
7. Koffi Kouakou Ive Arsene,Samassi Adama, Kouamé Appoh ; « Evaluation of a Model Maximizing the Quality Value of Selected Software Components in a Library », International Journal of Advanced Computer Science and Applications (IJACSA), Volume 11 Issue 5, 2020.doi : 10.14569/IJACSA.2020.0110586
8. Thomas Szyperski, Akira Ono, César Fernández, HideoIwai, Shin-ichi Tate, Kurt Wüthrich, and MasatsuneKainosho," Measurement of 3JC2'P ScalarCouplings in a 17 kDasProteinComplexwith 13C,15N-Labeled DNA Distinguishes the BI and BII Phosphate Conformations of the DNA, Journal of the American Chemical Society. 119 : 9901-9902. DOI : 10.1021/ja972290y, 1997

9. Sebti Mouehli, "Contributions à la vérification de la sureté de l'assemblage et de l'adaptation de composants reutilisables", FEMTO-ST- Franche-Comté Électronique Mécanique, Thermique et Optique - Sciences et Technologies (UMR 6174), these, tel.archives-ouvertes.fr › tel-01015089, 2011
10. Bart George," Un processus de selection de composants logiciels multi-niveaux », these, Génie logiciel [cs.SE]. Université de Bretagne Sud, 2007. https://tel.archives-ouvertes.fr/tel-00512356, 2007
11. Michail D. Papamichail, ThemistoklisDiamantopoulos, Andreas L. Symeonidis, « Measuring the Reusability of Software Components usingStaticAnalysisMetricsandReuse Rate Information », Journal Pre-proof, 2019, DOI : https://doi.org/10.1016/j.jss.2019.110423
12. Amit Rathee, Jitender Kumar Chhabra, "A multi-objective searchbasedapproach to identifyreusable software components", Journal of Computer Languages 52, p.26–43 ,3 January 2019 https://doi.org/10.1016/j.cola.2019.01.006
13. Mebarkayahlali, Abdellah chouarfia, « Software Component Quality Evaluation », Software Component Quality Evaluation, International Conferenceon Advanced Aspects of Software EngineeringICAASE, November, 2-4, 2014
14. A.A. Zaidan, B.B. Zaidan, Ahmed Al-Haiqi, M.L.M. Kiah, Muzammil Hussain et Mohamed Abdulnabi « Evaluation and selection of open-source EMR software packages based on integrated AHP and TOPSIS », journal of BiomedicalInformatics, Volume 53, Pages 390-404, February 2015, https://doi.org/10.1016/j.jbi.2014.11.012
15. Nanou Kakou Jean-Paul, « Conception et réalisation d'un logiciel de formulation alimentaire au laboratoire de nutrition et sécurité alimentaire pour une nutrition saine », Université Nangui Abrogoua, mémoire pour l'obtention du diplôme de master en informatique, 2017
16. Leung KRPH, Leung HKN, « On the efficiency of domain-based COTS productselectionmethod. Journal of Information and Software Technology », 44(12):703–715, 2002
17. Abdul Razak Hamdan and Jamaiah H. Yahaya « The successfactors and barriers of information technologyimplementation in small and medium enterprises : an empiricalstudy in Malaysia », Int. J. Business Information Systems, Vol. 21, No. 4, 2016
18. OyvindHauge, Thomas Osterlie, Carl-Fredrik Sorensen, and MarinelaGerea « An EmpiricalStudy on Selection of Open Source Software - Preliminary Results » NorwegianUniversity of Technology and Science, Pages 42–47, May 2009, https://doi.org/10.1109/FLOSS.2009.5071359
19. PanagiotaChatzipetrou, Emil Alégroth, EfiPapatheocharous, Markus Borg, TonyGorschek, Krzysztof Wnuk, « Component selection in Software Engineering - Whichattributes are the mostimportant in the decisionprocess ?Conference Paper • August 2018, DOI : 10.1109/SEAA.2018.00039
20. Oded Berman et NoushinAchrafi, « OptimizationModels for Reliability of Modular Software Systems », ieee transactions on software engineering, vol. 19, no. 11, november 1993
21. Shilpi Verma and Mukesh Kumar Mehlawat, « Multi-criteriaoptimization model integratedwith AHP for evaluation and selection of COTS components, Optimization, A Journal of MathematicalProgramming and Operations Research, DOI :10.1080/02331934.2017.1316502, 2017
22. C.K. Kwong, L.F. Mu, J.F. Tang, X.G. Luo, « Optimization of software components selection for component-based software system development », Computers &Industrial Engineering, 2010, doi : 10.1016/j.cie.2010.01.003
23. Razafinarivo T. Donnah, RahetlahVolatsaraBaholy, Salgado Paulo, Rakotozandriny J. N, Andriarimalala H. José, RalainindrianaIavo, Artus Florence, Le Mézo Lionel, « Développement d'un logiciel (3C-BIOVIS) pour la détermination de la disponibilité des ressources fourragères par une approche de modélisation et télédétection » Madagascar Conservation and Development, 14 (1) : 19-26., 2019, https://doi.org/10.4314/mcd.v14i1.1
24. Mukesh Kumar Mehlawat, Pankaj Gupta,∗, Divya Mahajan« A multi-period multi objective optimizationframeworkfor softwareenhancement and component evaluation, selection andintegration » Information Sciences 523 (2020) 91–110, https://doi.org/10.1016/j.ins.2020.02.076
25. SimrandeepSinghThapar, HimaliSarangal, « Quantifyingreusability of software components usinghybridfuzzyanalyticalhierarchy process (FAHP)-Metricsapproach » December 2019, https://doi.org/10.1016/j.asoc.2019.105997
26. Pankaj Gupta, MukeshKumarMehlawat, Divya Mahaja, « Multi-objective optimizationframework for software maintenance, component evaluation and selectioninvolving outsourcing, redundancy and customer to customerrelationship » Information Sciences 483 (2019) 21–52, January 2019, https://doi.org/10.1016/j.ins.2019.01.017

27. Jayesh M. Dhodiya and Anita Ravi Tailor « Geneticalgorithmbasedhybridapproach to solve uncertain multi-objective COTS selectionproblem for modular software system » journal of intelligent &Fuzzy System, vol 34, n 4 pp2103-2120, 2018, doi : 10.3233/jifs-162225
28. Pankaj Gupta, Mukesh Kumar Mehlawat, Divya Mahajan, « Data envelopmentanalysisbased multi-objective optimization model for evaluation and selection of software components under optimal redundancy », Springer Science+Business Media, LLC, part of Springer Nature, 2018, https://doi.org/10.1007/s10479-018-2842-y
29. Andrew J. Miller, « Outils et logiciels pour l'optimisation » www.math.u-bordeaux.fr › ~anmiller › cours, November 2009
30. M. Paschali, A. Ampatzoglou, S. Bibi, A. Chatzigeorgiou, I. Stamelos, « Reusability of open source software acrossdomains : A case study », Journal of Systems and Software, 2017. Doi : 10.1016/j.jss.2017.09.009.
31. Michele Bernasconi, Christine Choirat, RaffaelloSer, « the analytichierarchy process and the theory of measuremen », management science vol. 56, no. 4, april 2010, pp. 699-7, 2009 doi :10.2307/27784145
32. Koffi DodjiNoumonvi, « Application de la modélisation spatiale multifactorielle pour l'évaluation de la dynamique et la vulnérabilité des écosystèmes forestiers face aux changements globaux : cas de la forêt de Maà¢mora », mémoireonline, https://www.memoireonline.com/04/17/9891/m_Application-de-la-modelisation-spatiale-multifactorielle-pour-l-evaluation-de-la-dynamique-et-la-v26.html2015
33. Renaud Caillet, « Analyse multicritère : étude et comparaison des méthodes existantes en vue d'une application en analyse de cycle de vie », série scientifique, 2003
34. Gaëlle Guesdon, « évaluation des impacts environnementaux (ei), 5e. Méthodes et outils Aide multicritère à la décision-Comparaison de Saaty », Faculté des sciences et de génie Université Laval, 2011
35. Thomas l. Saaty, « decisionmakingwith the analytichierarchy process », katzgraduateschool of business, university of pittsburgh, int. j. services sciences, vol. 1, no. 1, 2008
36. Kouakou Ive Arsene Koffi, Konan Marcellin Brou, Souleymane Oumtanaga, developpement de methodes automatiques pour la réutilisation des composants logiciels, https://hal.archives-ouvertes.fr/hal-01482559, 2017
37. Leila Benkahoul, Youcef Kehila, SihemeAliouche, « Prise en compte de la sensibilité paysagère dans la sélection des sites d'enfouissement de déchets ménagers par systèmes d'information géographique et analyse multi critère », Déchets Sciences et Techniques - N°73, Mai 2017, https://hal.inria.fr/hal-03159897v1.