Software components selection: An optimized selection criterion for component-based software engineering (CBSE)

ahmad nabot

Zarqa University

November 30, 2022

Abstract

Component-based software engineering (CBSE) is becoming the frequently used approach for software development concerning time and cost constraints. While most people’s lifestyle depends on the use of software applications, the development cost and time of software applications are becoming more difficult to achieve. Also, high-quality and competent applications that fit users’ needs became essential. Therefore, most software development organizations are reusing commercial-off-the-shelf (COTS) to reduce development costs and time. However, most organizations and developers face problems selecting components that fit customer needs to integrate with the target system. So, decisions regarding software component selection are hard to consider regarding the entire quality of the software system. This study aims to investigate the most important matters to software industry practitioners and experts involved in component selection. First, the author asked the practitioners to select the most important quality criteria for an online bookstore from a list by providing their subjective judgment and evaluation grades; after that, utilizing the evidential reasoning (ER) approach for software component selection problems due to its ability to analyze decisions with multilevel evaluations and information uncertainty. Moreover, the main features of the ER approach, such as weight normalization, probability assessment, dealing with uncertainty, and utility intervals, offer several advantages for COTS selection problems, including cost and time minimizing, improved software reliability, effectiveness, and efficiency. This study concentrated on assessing the quality criteria for an online bookstore using the ER approach. Analysis results are provided depending on the computational steps of this approach. Finally, the findings show the rank of the four components according to their weights, evaluation grades, and belief degrees for the selection.
RESEARCH ARTICLE

Software components selection: An optimized selection criterion for component-based software engineering (CBSE)

Ahmad Nabot *

1Department of Software Engineering, Zarqa University, Zarqa, Jordan

Correspondence
*Ahmad nabot, Zarqa, Jordan. Email: anabot@zu.edu.jo

Present Address
Zarqa, Jordan

Abstract

Component-based software engineering (CBSE) is becoming the frequently used approach for software development concerning time and cost constraints. While most people’s lifestyle depends on the use of software applications, the development cost and time of software applications are becoming more difficult to achieve. Also, high-quality and competent applications that fit users’ needs became essential. Therefore, most software development organizations are reusing commercial-off-the-shelf (COTS) to reduce development costs and time. However, most organizations and developers face problems selecting components that fit customer needs to integrate with the target system. So, decisions regarding software component selection are hard to consider regarding the entire quality of the software system. This study aims to investigate the most important matters to software industry practitioners and experts involved in component selection. First, the author asked the practitioners to select the most important quality criteria for an online bookstore from a list by providing their subjective judgment and evaluation grades; after that, utilizing the evidential reasoning (ER) approach for software component selection problems due to its ability to analyze decisions with multilevel evaluations and information uncertainty. Moreover, the main features of the ER approach, such as weight normalization, probability assessment, dealing with uncertainty, and utility intervals, offer several advantages for COTS selection problems, including cost and time minimizing, improved software reliability, effectiveness, and efficiency. This study concentrated on assessing the quality criteria for an online bookstore using the ER approach. Analysis results are provided depending on the computational steps of this approach. Finally, the findings show the rank of the four components according to their weights, evaluation grades, and belief degrees for the selection.

KEYWORDS:
Keywords: Multi-criteria decision making (MCDM); Assessment; Evidential reasoning; Software component; Criteria.
1 | INTRODUCTION

Recently with the tremendous spread and use of software applications, most software-developing organizations are reusing software components from commercial-off-the-shelf (COTS), in-house developed components, and open-source components. Reusability increases the reliability of the reused components due to their usage history, which impacts process risk reduction. Thus, reducing the development and maintenance costs as well as reducing time and effort is required for building software to reach the markets within the expected time. Component-based software engineering (CBSE) is an approach for complex software development based on the reuse of existing software components from different environments to provide the functional and non-functional requirements of the targeted system. Therefore, the selection process of software components that fulfill the end-users needs is challenging. Thus, the quality criteria of the components are taken into account because they increase the difficulty of selecting the proper components for integration. In addition, when the available number of components that comply with the functional and non-functional requirements in the repository is large, the selection process will be complex, which might conflict with the reuse objectives such as minimizing price and maximizing quality. However, developers face many challenges when selecting the right components, such as multi-criteria decision-making or multi-objective optimization. Moreover, optimizing software requirements and development costs are essential for component selection. Eventually, the evaluation process of COTS when selecting them, especially in a complex system, is complicated due to the lack of information, uncertainty, evolving COTS, and changing requirements. Thus, systems developed using existing COTS should consider all stages of software development, such as requirement analysis, software design, integration, and maintenance. Over the past years, many researchers conducted research to solve the component selection problem. These research studies used different decision-making and integrated artificial intelligence (AI) techniques (i.e., AHP, ANP, CV, PROMETHE, and clustering-based) to handle incomplete or uncertain information. Furthermore, these techniques have many shortages, such as the number of software requirements, kind of software requirements, uncertain information, and many more problems. Thus, making the selection decision crucial requires further investigation. The contribution of this study is to deploy a method for selecting software components that fit the user’s needs in terms of use and quality. Therefore, this study utilized the evidential reasoning (ER) approach to achieve this aim. ER is a multi-criteria decision-making (MCDM) for analyzing problems under uncertainties or incomplete information by representing them in a multilevel or multi-criteria evaluation decision matrix. Also, the ER approach is developed based on Dempster-Shafer’s (D-S) theory of evidence to deal with uncertainties such as missing or incomplete information using belief structure. D-S theory of evidence provides a method for dealing with an incomplete decision matrix, ensuring that all values in the decision matrix are qualitative. The hierarchy contains sets of criteria distributed on different levels. Each criterion is evaluated using a set of evaluation grades with varying degrees of belief. In addition, ER approach has more features than other techniques that optimize decisions regarding component selection problems. An online bookstore software system contains four components, each containing the same quality criteria with different belief degrees, evaluation grades, and weights assessed using the ER approach. The complete steps of the assessment process, implementations, and final decision results are provided. The paper is structured as follows: section 2 explains the research method. Section 3 presents a background of the related work. Section 4 presents a description of the problem of software component selection. Section 5 presents the methodology of the ER approach. Section 6 presents the used real-life case study to explain the ER approach. Section 7 presents the results of the used case study. Section 8 presents the discussion of the RQs and results. Finally, Section 9 concludes the study.

2 | RESEARCH METHOD

This study followed Kitchenham and Charters (2007) guidelines, which consists of three stages. The first stage is establishing a strategy for conducting the review and a framework for conducting the research. Stage two entails doing the review, which entails searching, choosing the papers, extracting the data, and synthesizing it. In the third stage, researchers report their findings and decide how the review will be shared with the public. Consequently, this paper was developed as a result of such a period.

---

1 This is an example for title footnote.
2 Abbreviations: CBSE, component-based software engineering; COTS, component off the shelf; ER, Evidential reasoning; D-S, Dempster-Shafer theory; IDS, intelligent decision system
2.1 | Review planning

The primary objective of the review is to provide a synopsis of the field and to analyze the elements of the approach, such as artifacts, search criteria, artifact representations, evaluation methods, and results, that are used in the experimentation.

2.2 | Research question

The review of this study aims to answer the research questions presented in Table 1. These questions aim to cover all aspects of software component selection and selection methods/techniques.

**TABLE 1** Research Questions

<table>
<thead>
<tr>
<th>RQ. No.</th>
<th>Research Question</th>
</tr>
</thead>
<tbody>
<tr>
<td>RQ1</td>
<td>What are the most important criteria for COTS selection?</td>
</tr>
<tr>
<td>RQ2</td>
<td>Does ER approach assess criteria and sub-criteria of software components?</td>
</tr>
<tr>
<td>RQ3</td>
<td>What are the constraints of the selection methods?</td>
</tr>
<tr>
<td>RQ4</td>
<td>Do current selection methods deal with large-size components and incomplete information to generate reliable results?</td>
</tr>
<tr>
<td>RQ5</td>
<td>Does ER approach cost and time effective?</td>
</tr>
</tbody>
</table>

2.3 | Search Strategy

The search terms in the search strategy are derived from the study research questions and the related studies. Figure 1 shows the search strategy terms.

```
("Software") AND ("Component") AND ("Selection" OR "Select") AND ("Method" OR "Technique" OR "Multi-Objective" OR "Fuzzy Logic" OR "Artificial Intelligence" OR "MCDM" OR "Functional Requirements" OR "Non-Functional Requirements" OR "Uncertainty" OR "Incomplete Information" OR "Constraints" OR "Evaluation" OR "Software Engineering" OR "Component-Based Software Engineering")
```

**FIGURE 1** Search Strategy Terms

2.4 | Related studies selection criteria

To select all related studies from different sources, a set of inclusion and exclusion criteria was identified, as shown in Table 2. Paper selection started by applying the search terms on the primary sources and search engines to narrow the required search time. After that, read the title and abstract for the final selection decision. Primary sources and search engines are online repositories selected based on their research impact in publishing software engineering publications. These sources are as follows: IEEE Xplore, Wiley online library, ACM Digital library, Springer, Science Direct, Elsevier, and Web of Science. Figure 2 shows the number of related studies to software component selection for 1999-2022.
### TABLE 2 Inclusion and Exclusion Criteria

<table>
<thead>
<tr>
<th>Inclusion Criteria</th>
<th>Exclusion Criteria</th>
</tr>
</thead>
<tbody>
<tr>
<td>All papers that use software component selection methods/techniques</td>
<td>Out of Scope papers;</td>
</tr>
<tr>
<td>Multi-criteria decision-making methods</td>
<td>Papers were not written in English;</td>
</tr>
</tbody>
</table>

**FIGURE 2** Number of Published Papers in the area of Software Component selection

### 3 | BACKGROUND

#### 3.1 | Software component selection methods/approaches

In component-based software engineering (CBSE), software component selection methods/approaches are critical. This is because selecting the components is essential to developing high-quality, cost-effective, and efficient software systems. The component selection consists of several activities that enhance usability, such as qualification to ensure the functional and non-functional requirements, adaptation to adapt a specific component depending on a set of rules, and composition to integrate the selected component with other components to build up the software. However, component selection can be made through specialized sourcing engines (SSE) or component repositories, making such a process challenging for software developers. Therefore, researchers proposed different selection methods/techniques for solving such problems. For instance, Various methods and techniques were proposed covering different issues to reduce the gap between the actual stated requirements to enable decision-makers to make the right decision as both OTSO and PORE have problems of inability to deal with incomplete information and criteria weighting problems. Kunda and Brooks (1999) proposed the social-technical approach to COTS evaluation (STACE) for software component selection and assessment based on socio-technical or non-technical factors, including costs, business issues, and reliability. Grau, Gemma, et al. (2004) developed DesCOTS software that combines different software component selection and evaluation tools. Maxville, Armarego & Lam (2004) used an AI technique to automate the process of
component assessment for selection and evaluation. In addition, they used C 4.5 and neural network classifiers as an alternative to the aggregation process used in traditional selection methods to increase the accuracy level of the selection process. Carvallo, Franch, and Quer (2006) presented an approach to extend the ISO/IEC 9126-1 catalog for managing software requirements at the selection stage of COTS. Bhuta, Jesal, et al. (2007) proposed an attribute-driven framework for COTS and connectors selection based on interoperability and glue code during the software design phase. Haghpanah, Nima, et al. (2007) used genetic and greedy algorithms to solve the problem of COTS selection by reducing selection cost and time. Gashi and Popov (2007) applied Bayesian assessment methods to select COTS based on probability distributions. Neubauer and Stummer (2007) introduced a multi-objective approach that addresses the challenges decision-makers face in selecting COTS. This approach can be integrated with any COTS selection approach to support the selection decision. Cortellessa, Vittorio, et al. (2007) proposed a framework for COTS selection in the requirement phase based on DEER optimization tool to minimize development costs. However, the adopted framework has problems, such as uncertainty in selecting the right component and requirement priorities conflict. Vijayalakshmi, K., Ramaraj, N., and Amuthakannan, R. (2008) proposed an automated approach based on a Genetic algorithm (GA) for component selection that deals with functional and non-functional requirements. Neubauer, T., Pichler, J., and Stummer, C. (2008) evaluated two approaches with the Atana approach that allows decision-makers to make decisions semi-automatically. In addition, Atana provides a precise selection for the business best needs to optimize their investments. Jadhav, Anil, and Rajendra Sonar (2009) compared the analytic hierarchical process (AHP), weighted scoring method (WSM), and hybrid knowledge-based system (HKBS) approaches for component selection. The authors found that HKBS is better than AHP and WSM in terms of efficiency, flexibility, knowledge reuse, and consistency. Şerban, Camelia, Andreea Vescan, and Horia F. Pop (2009) proposed a new algorithm based on metrics and fuzzy clustering for component selection. The proposed approach has some constraints in dealing with dependencies. Kwong, Chun Kit, et al. (2010) utilized a genetic algorithm (GA) to select the optimal software component by developing a financial system for small and medium size (SMEs) enterprises. However, GA has trouble dealing with fuzziness caused by subjective judgments and information complexity. Mancilla, Astudillo, and Visconti (2010) combined COSTUME and Azimut+ to address functional and non-functional requirements and classify components according to their non-functional requirements for selection, where the efficiency of such a technique is not proved. Ibrahim, Elamy, Far, and Eberlein (2011) proposed uncertainty handling in COTS selection (UnHOS) using the analytic hierarchy process (AHP) and Bayesian belief network (BBN) for selecting and evaluating COTS. In addition, many studies use these different techniques to make the right decision while selecting software components where the scalability of such a method is limited to a specific number of criteria. Jha, P. C., Shivani Bali, and U. Dinesh Kumar (2011) proposed a fuzzy multi-objective approach for selecting software components to build a reliable and less execution time software system. Rafsanjani, Marjan Kuchaki, and Noushin Rakshan (2011) proposed an approach based on the 0/1 knapsack algorithm to reduce development costs and increase cohesion between the components. Nazir, Shah, et al. (2012) proposed a method using OTSO and fuzzy logic to select software components, which is not validated using a real-life case study. Thus, the proposed method is considered invalid and not reliable. Gupta, Pankaj, Mukesh Kumar Mehlawat, and Shilpi Verma (2012) used multiple approaches, including quality model, AHP, and fuzzy mathematical programming (FMP), to develop a fuzzy multi-objective optimization model for components selection. This approach has the problem of not considering unreliable input information. Pandey, Jeevendra, Christopher J. Garcia, and Durgesh Pant (2013) proposed a method based on integer programming to maximize liability to select software components. This method did not consider all quality attributes and their weights which are considered shortcomings for such a method. Tomar, Pradeep, and Nasib Singh Gill (2013) proposed an algorithm using best-fit and first-fit strategies for component selection by selecting components using a simple selection problem (SCSP) and criteria component selection problem (CCSP). Nevertheless, the proposed algorithm is ineffective in terms of cost and time. Faridi, M. Shakeel, et al. (2013) proposed the idealized recommendation Off-The-Shelf (IROTS) approach based on the ISO/IEC 25010 model. The proposed approach does not consider the weights of the requirements, which causes conflict. Numerical illustrations are provided to demonstrate the model developed. Mittal and Bhatia (2013) proposed a framework for reusability as the main selection criteria for selecting and evaluating the chosen components using AHP. However, if the number of criteria is large, the process will be costly and time-consuming. Faundes et al. (2013) proposed a method for selecting components based on fuzzy decision-making systems to compare and evaluate COTS and their impact on IT organizations. Kaur and Singh (2014) proposed PROMETHE as a method for component evaluation and selection, which considered some quality attributes for COTS selection, such as reliability, Integrability, performance, cost, and maintainability. However, the proposed method in this study did not consider uncertain information, leading to unreliable and invalid results. Nazir et al. (2014) proposed an analytic network process (ANP) as a selection method for components that considered effectiveness, efficiency, satisfaction, safety, and usability as the criteria for the selection process. Compared to the ER approach, the ANP method features are insufficient to
provide the decision-maker with strong viewpoints for the final selection decision. Khan et al. (2014). Proposed component-based software engineering for software reusability method for component selection which considers reducing development costs and time. Kaur and Tomar (2015) proposed the multi-objective optimization model (MOO) using pre-emptive goal programming for software component selection. The MOO method optimizes the conflicting criteria of component selection by selecting components depending on achieving the highest priority goals first. However, this technique does not provide precise results in case of the number of components is large. Konys, Agnieszka (2015) used two ontologies to support the process of components selection based on ontology web language (OWL) and information tools ontology. The approach is timely and increases the cost and complexity of the selection process. Nazir, Shah, et al. (2015) proposed a fuzzy logic model to handle security requirements to evaluate COTS security based on ISO/IEC 18028-2. The proposed approach has limitations in dealing with other requirements and does not consider large-size components. Kaur and Tomar (2016) presented a validation of clustering-based algorithms for component selection depending on fuzzy-c, subtractive clustering, hybrid XOR-based clustering, and fuzzy relation-based clustering. This method has some clustering shortcomings, such as required identification for the number of clusters in advance, cluster radius, selecting the correct distance cluster, and overlapping clusters. Sekar and Sethuraman (2017) Proposed a method for component selection in web engineering based on fuzzy ranking and rough sets to provide extra functionality in web applications. Tian, Wang, Jiang, and Chen (2017) Proposed a method for component selection based on clustering and information entropy weighting, which industry experts can use to enable them to select the appropriate components depending on the selection results and artificial experience choice of the optimal set of components. Gupta, Pankaj, Mukesh Kumar Mehlawat, and Divya Mahajan (2018) proposed data envelopment analysis (DEA) using a multi-objective optimization model for selecting software components. This approach has some limitations, such as the inability to deal with information uncertainty, increased execution time, and reduced reliability. Farshidi, Siamak, et al. (2018) proposed a decision support system (DSS) based on Moscow and the quality models ISO/IEC 25010 and ISO/IEC 9126 to solve the problem of software component selection. However, this approach is limited to a specific kind and number of requirements. Verma, Shilpi, Mukesh Kumar Mehlawat, and Divya Mahajan (2018) introduced a non-linear multi-objective optimization model and supported their model by utilizing a technique for order preference by similarity to ideal solution (TOPSIS) for software components selection. The proposed approach did not consider the requirements priorities of the selected components, which led to requirements conflict. Kaur and Tomar (2018) Proposed an architecture of four tiers for component selection using clustering. The architecture tiers are the component requirements and selection tier, query and decision tier, application logic tier, and component cluster tier. Each cluster can have only one component to be evaluated, which takes a long time to evaluate many components. Rodas-Silva, Jorge, et al. (2019) introduces a prototype component-based recommender system (RESDEC) to select the right component depending on the selected features. The introduced approach does not deal with information uncertainty and requirements priorities. Padhy et al. (2019) Introduced component selection criteria by drawing a reusability matrix for all the classes of components depending on reusability features. Gusev, Alexander, Dmitry Ilin, and Evgeny Nikulchev (2020) introduced a swarm intelligence approach by constructing a mathematical model using the artificial bee colony algorithm to select a set of software components. The presented approach deals only with functional requirements and does not deal with information uncertainty. Garg and Rakesh (2020) proposed a fuzzy set theory and the modified distance-based approach (FMDBA) to assess decision-makers to select the right components. The proposed approach cannot deal with large-size components. Mehlawat, Gupta, and Mahajan (2020) proposed a multi-period multi-objective optimization framework that combines software component selection, evaluation, and integration of components and vendors, which is constrained to a specific type of components and parameters. Chatzipetrou, Papatheocharous, Wnuk, Borg, Alégroth, and Gorschek (2020) focused on the most important factors for industry practitioners to select the suitable component from different component sourcing options. The study started with a descriptive survey to discover the most critical factors for the practitioners. Then, the collected data were analyzed using techniques such as hundred dollars (100$), compositional data analysis (CoDA), non-parametric test, and biplots. Bibi, Rana, and Naseer (2021) Proposed a hybrid approach that depends on the techniques of natural language processing and domain knowledge. This allows users to inquire about the required component according to its criteria using such methods. Jabbarpour, Saghiri, and Sookhak (2021) Proposed a framework of four phases depending on artificial intelligence (AI) techniques to solve the problem of component selection. The framework starts by analyzing the available components, extracting criteria, weighting, and finally formulating them into knapsacks to select the suitable component. The proposed framework in this study has political, social, and financial issues because it has troubles with the utilized AI techniques. Kalantari, Samira, et al. (2021) Optimized software component selection via multi-objective by maximizing the fuzzy-Intra coupling density (Fuzzy_ICD) and functionality as objective functions by taking into considerations budget, delivery time, and reliability as constraints of such selection techniques. Banga, Anjali, and Pradeep Kumar Bhatia (2022) proposed a long-short term memory approach integrated...
with neural network mechanisms such as PSO and MVO to optimize the selection problem. The proposed approach optimized results accuracy and time efficiency\textsuperscript{21}. As it has problems in dealing with uncertain information and subjective judgments.

Mehta, Tandon, and Sharma (2022) implemented the fuzzy analytic hierarchy process (FAHP) technique to calculate criteria weights. Then, software components are ranked to select the best alternative using a complex proportional assessment of alternatives with grey relations (COPRAS-G)\textsuperscript{22}.

3.2 | The Evidential Reasoning Approach (ER)

CBSE’s approach focuses on integrating various components to build the required software system that fits the needs of software stakeholders. The main problem of this approach lies in satisfying the set of requirements of the final software system. The selection problem of a set of components to build a software system depends mainly on the targeted requirements that should be provided to the stakeholders to fit their needs. While the selection criteria of the right set of components to fit the stakeholder needs may depend on different criteria. Therefore, the Evidential reasoning approach (ER) is considered a superior method in dealing with uncertain assessments and decision-making\textsuperscript{23}. Thus, this study utilized the ER approach for selecting and weighing the criteria for ranking and selecting the appropriate component. As mentioned before, ER is an MCDM method that consists of a multilevel assessment hierarchy for assessing software components based on different levels. Also, it uses an extended decision matrix to describe software component criteria and sub-criteria using belief structure. Moreover, it has been utilized for solving different problems in different research studies. For instance, Zhang and Deng (2018), and Dong et al. (2019), utilized the ER approach to analyze fault diagnosis issues in an uncertain environment\textsuperscript{24,25}. Akhoundi et al. (2018), used the ER approach for wastewater reuse alternatives assessment\textsuperscript{26}. Ng and Law (2020) used ER to analyze affection words in social networking to explore user preferences\textsuperscript{27}. Tian et al. (2020) integrated the ER approach with probabilistic linguistics to solve multi-criteria decision-making problems by considering the decision-makers’ psychological preferences\textsuperscript{28}.

4 | PROBLEM DESCRIPTION

Selecting software components depending on subjective judgments to differentiate between alternatives based on quality attributes. Therefore, using belief structure, ER uses an extended decision matrix to describe each criterion and its alternative sub-criterion depending on evaluation grades. For instance, the result of the evaluation grades of the quality of the software component could be described as follows:

\[ H = \{ H_1, H_2, H_3, H_4, H_5 \} = \{ \text{Worst, Poor, Average, Good, Excellent} \} \]  \hspace{1cm} (1)

The evaluation grades capture the different types of uncertainties, such as obscurity in subjective judgments, and provide precise degrees of belief structure\textsuperscript{29}. A software component is the general term and is hard to evaluate directly. Thus, it needs to be composed into lower-level concepts such as criteria and sub-criteria. If the sub-component is still abstract to evaluate directly, it should be divided into more detailed concepts\textsuperscript{30}. For instance, let’s consider an online bookshop store. The sub-component "order item" may be measured by usability, performance, reliability, security, maintainability, portability, and flexibility. These criteria can be aggregated to be directly assessed and referred to as sub-criteria of the system, which takes place at the lowest level in the hierarchy. For instance, the criteria usability (y) can be measured by (e_1, e_2, \ldots, e_n) Where (e_i) denotes learnability, (e_2) for ease of use, and (e_3) for satisfaction. However, assessment criteria consist of a multilevel structure to assess the criteria in the higher-level structure through the lower-level sub-criteria\textsuperscript{31} as shown in Figure 3.

5 | METHODOLOGY

The motivation for using the ER approach is the different way of developing decision support systems (DSS) than other MCDM approaches. In addition, it deals with the problems of having qualitative and quantitative information with subjectivity and uncertainty\textsuperscript{22}. ER approach was developed based on several sciences disciplines, including statistical analysis, artificial intelligence, and information technology\textsuperscript{32,33,34,35,36,37,38}. Moreover, the difference between such an approach and other MCDM approaches is the employment of the ER algorithm to aggregate the belief degrees based on the (D-S) theory\textsuperscript{39}. This framework is flexible in terms of describing an MCDM problem and preventing information loss by converting two-dimensional values into
one-dimensional values through the modeling process. Suppose we have L basic criteria at the lower level of the hierarchy $A_i$ ($i = 1, \ldots, L$) associated with the general component concept $y$, $K$ alternatives $O_j$ ($j = 1, \ldots, K$) and $N$ for evaluation grades $H_n$ ($n = 1, \ldots, N$) for each criterion where $S\{A_i(O_j)\}$ is given as follows:

$$ S \left( A_i, O_j \right) = \left\{ \left( H_n, \beta_n, i, O_j \right) \right\}, n = (1, \ldots, N), i = (1, \ldots, L), \text{and } j = (1, \ldots, K) $$

Where $(\beta_n, i(O_j))$ represents the degree of belief of the alternative $O_j$, which is assessed by the $n$th grade of the $i$th criterion. Each criterion might have its evaluation grades, which could differ from other criteria in the hierarchy. ER algorithm can be described by transforming the belief degrees into masses where $m_{(n,i)}$ and $m_{(H,i)}$ are calculated as follows:

$$ m_{n,i} = \omega_i \beta_{n,i} $$

$$ m_{H,i} = 1 - \sum_{n=1}^{N} m_{n,i} = 1 - \omega_i \sum_{n=1}^{N} \beta_{n,i} $$

Suppose the weight of the $i$th criterion $m_{(n,i)}$ is given by $\omega = (\omega_1, \omega_2, \omega_3, \ldots, \omega_i)$. So, the probability mass represents the $n$th evaluation grade $H_n$ of the $i$th criterion. The residual probability mass $m_{(H,i)}$ unassigned to any individual grade after assessing the $i$th criterion.

$$ m_{H,i} = \bar{m}_{H,i} + \tilde{m}_{H,i} $$

for $i = 1, \ldots, L$ and $\sum_{i=1}^{L} \omega_i = 1$. Assigns the evaluation grades $H = \{ H_1, H_2, \ldots, H_N \}$ to the probability masses and $L$ criteria are aggregated to generate the combined belief degree for each evaluation grade $H_n$. The unassigned evaluation grades $H_n$ can be calculated as follows:

$$ \bar{m}_{H,i} = 1 - \omega_i \quad \text{and} \quad \tilde{m}_{H,i} = \omega_i (1 - \sum_{n=1}^{N} \beta_{n,i}) $$

Where $\bar{m}_{H,i}$ is used for calculating the relative importance of the $i$th criterion and $\tilde{m}_{H,i}$ is used for the incomplete information of the $i$th criterion. Therefore, $m_{H(i,L)} = \bar{m}_{H(i,L)} + \tilde{m}_{H(i,L)}$, $n = (1, \ldots, N)$ are the combined probability assignments by aggregating all original probability masses using the aggregation of the following ER algorithm.

**FIGURE 3** Evaluation Hierarchy for software components
REAL-LIFE CASE STUDY

When the assessments of \(O_j\) to \(H_n\), and the upper bound can be computed by \((\beta_n + \beta_H)\). Also, in case there are uncertainties such as missing or incomplete information, they can be characterized by the maximum, minimum, and average score of \(S(A^*)\) as follows:

\[
\begin{align*}
 u_{\text{max}}(O_j) &= \sum_{n=1}^{N} \beta_n(O_j) u(H_n) + (\beta_N(O_j) + \beta_H(O) u(H_N)) \\
 u_{\text{min}}(O_j) &= \sum_{n=2}^{N} \beta_n(O_j) u(H_n) \\
 u_{\text{avg}}(O_j) &= \frac{u_{\text{max}}(O_j) + u_{\text{min}}(O_j)}{2}, \text{ where } u(H_n + 1) \geq u(H_N)
\end{align*}
\]

When the assessments of \(S(A_i(O_j))\) in the belief decision matrix are complete, the result of \(\beta_H(O_j) = 0\) and \(u(S(O_j)) = u_{\text{max}}(O_j) = u_{\text{min}}(O_j) = u_{\text{avg}}(O_j)\). These mathematical equations are used for assessment characterization rather than criteria aggregation.

The key difference between ER framework and other MCDM techniques is that ER transforms various evaluation information to assess functional and non-functional criteria. In addition, information uncertainties of the assessed criteria can be treated based on Dempster-Shafer’s \((D - S)\) theory of evidence.

6 | REAL-LIFE CASE STUDY

In this section, the ER approach is applied to analyze the quality attributes of online bookstore software to select the best component depending on field practitioners and experts. Four components were selected, as shown in Figure 3, including Component 1, Component 2, Component 3, and Component 4, that fit the customer’s needs. Nevertheless, ER approach can be used for
selecting and evaluating many components. In this study, only qualitative software attributes are considered for illustrating the ER algorithm, while quantitative attributes will be included in a later study. To demonstrate and validate the proposed method, a group of practitioners and experts from different software companies and academia was established according to their knowledge in the field of software engineering. As the data are important for COTS selection and evaluation and the insufficient data about COTS selection, field experts’ opinion approach was opted to collect the relevant data to the COTS selection and evaluation problem. Data were collected using a questionnaire due to its exploratory nature. The questionnaire was used to collect primary data, such as the estimated weights of COTS criteria. The distributed questionnaire has three parts: First part contains a cover letter explaining the study’s purpose. The second part consists of demographic data for the practitioners and experts, including organization name, expertise field, length of experience, qualification, and role. The third part of the questionnaire consists of weights assigned to the selection criteria/quality criteria by rating them using five evaluation grades (worst, poor, average, good, and excellent) and providing a rank for evaluation grade from 0 to 1. The weight of the selection criteria using practitioners’ and experts’ opinions are estimated and presented in Table 4.

Main software quality attributes at higher levels, such as usability, performance, reliability, security, maintainability, probability, and flexibility, are considered general attributes and difficult to evaluate directly. Therefore, lower-level attributes will be used for assessing the higher-level attributes. For instance, the quality attributes of component 1 might be assessed through components’ learnability, ease of use, and satisfaction. The attribute hierarchy for the four software components is shown in Figure 3, where $\omega_i$, $\omega_{ij}$, and $\omega_{ijk}$ represent the relative weights of all attributes.

Thus, degrees of belief, evaluation grades, and attributes defined for these sub-criteria can be represented as follows:

- Learnability = (Average, 0.4), (Good, 0.5)
- Ease of use = (Good, 1.0)
- Satisfaction = (Good, 0.4), (Excellent, 0.6)

The assessment can be complete in the case of the value of the belief degree of the sub-criteria (=1) and incomplete in the case of the belief degree value (<1). For instance, the above assessment of learnability $0.4 + 0.5 = 0.9 < 1$ is incomplete, while the assessment of ease of use and satisfaction is complete. Only grades with degrees of belief > 0 are listed in the distributions.

The subjective assessment for usability sub-criteria, including their degrees of belief, is shown in Table 3, where W, P, A, G, and E denote Worst, Poor, Average, Good, and Excellent, respectively. In addition, the number in the brackets denotes a degree of belief to which an attribute is evaluated. For example, G (0.5) means "good to a degree of 0.5 (50%)." ER framework generates the overall usability assessment by aggregating it into several sub-criteria, as shown in Figure 3. In addition, the ER framework provides a way of dealing with the aggregation problem.

**TABLE 3** Subjective judgments for evaluating software system usability

<table>
<thead>
<tr>
<th>Degree of belief ($\beta$)</th>
<th>Evaluation grade</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Worst</td>
<td>Poor</td>
</tr>
<tr>
<td>Learnability</td>
<td>0.4</td>
<td>0.5</td>
</tr>
<tr>
<td>Ease of Use</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Satisfaction</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The assessment of how the original judgments, as in Table 3, could be aggregated to assess the quality of the usability criteria of components to select the component with the highest usability quality. Therefore, the relevant importance of the other three components should be assigned to generate a precise assessment. There are different methods used for weight assessment. In this study, ER approach will be applied to deal with the assessment problem.
TABLE 4 Quality attributes’ degrees of belief distribution on each component

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Sub-criteria</th>
<th>Component 1</th>
<th>Component 2</th>
<th>Component 3</th>
<th>Component 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Usability (ω₁)</td>
<td>Learnability (ω₁₁)</td>
<td>A (0.4), G (0.5)</td>
<td>A (1.0)</td>
<td>G (0.5), E (0.5)</td>
<td>E (1.0)</td>
</tr>
<tr>
<td></td>
<td>Ease of Use (ω₁₂)</td>
<td>G (1.0)</td>
<td>A (0.3), E (0.7)</td>
<td>P (0.4), A (0.6)</td>
<td>G (0.5), E (0.5)</td>
</tr>
<tr>
<td></td>
<td>Satisfaction (ω₁₃)</td>
<td>G (0.4), E (0.6)</td>
<td>A (0.8)</td>
<td>P (0.4), A (0.5)</td>
<td>E (1.0)</td>
</tr>
<tr>
<td>Performance (ω₂)</td>
<td>Time Efficiency (ω₂₁)</td>
<td>G (0.5), E (0.5)</td>
<td>P (1.0)</td>
<td>A (0.3), G (0.6)</td>
<td>P (0.7), A (0.3)</td>
</tr>
<tr>
<td></td>
<td>Speed Efficiency (ω₂₂)</td>
<td>A (0.5), G (0.5)</td>
<td>G (0.9), E (0.1)</td>
<td>P (0.6), A (0.4)</td>
<td>G (1.0)</td>
</tr>
<tr>
<td>Reliability (ω₃)</td>
<td>Consistency (ω₃₁)</td>
<td>G (0.3), E (0.7)</td>
<td>A (0.8), G (0.2)</td>
<td>G (0.5), E (0.4)</td>
<td>A (0.6), G (0.4)</td>
</tr>
<tr>
<td></td>
<td>Robustness (ω₃₂)</td>
<td>G (0.6), E (0.3)</td>
<td>A (0.3), G (0.4)</td>
<td>E (0.9)</td>
<td>G (0.5), E (0.5)</td>
</tr>
<tr>
<td></td>
<td>Accuracy (ω₃₃)</td>
<td>A (0.5), G (0.4)</td>
<td>P (0.3), A (0.6)</td>
<td>G (1.0)</td>
<td>G (0.5), E (0.5)</td>
</tr>
<tr>
<td>Security (ω₄)</td>
<td>Confidentiality (ω₄₁)</td>
<td>G (1.0)</td>
<td>E (0.7)</td>
<td>A (0.3), G (0.7)</td>
<td>A (0.2), G (0.8)</td>
</tr>
<tr>
<td></td>
<td>Integrity (ω₄₂)</td>
<td>A (0.5), G (0.5)</td>
<td>A (0.7), G (0.3)</td>
<td>E (1.0)</td>
<td>P (0.2), A (0.8)</td>
</tr>
<tr>
<td>Maintainability (ω₅)</td>
<td>Serviceability (ω₅₁)</td>
<td>G (0.4), E (0.6)</td>
<td>G (0.9)</td>
<td>A (0.3), G (0.7)</td>
<td>E (1.0)</td>
</tr>
<tr>
<td></td>
<td>Testability (ω₅₂)</td>
<td>E (0.9)</td>
<td>G (0.6), E (0.4)</td>
<td>A (0.2), G (0.8)</td>
<td>G (0.9)</td>
</tr>
<tr>
<td></td>
<td>Modifiability (ω₅₃)</td>
<td>G (0.8), E (0.2)</td>
<td>P (0.4), G (0.6)</td>
<td>W (0.4), P (0.6)</td>
<td>A (0.4), G (0.6)</td>
</tr>
<tr>
<td>Portability (ω₆)</td>
<td>Modularity (ω₆₁)</td>
<td>G (0.6), E (0.4)</td>
<td>A (0.2), G (0.8)</td>
<td>G (0.5), E (0.5)</td>
<td>G (1.0)</td>
</tr>
<tr>
<td></td>
<td>Independence (ω₆₂)</td>
<td>W (0.6), P (0.4)</td>
<td>G (0.5), E (0.5)</td>
<td>A (0.3), G (0.7)</td>
<td>P (0.7), A (0.2)</td>
</tr>
<tr>
<td></td>
<td>Descriptiveness (ω₆₃)</td>
<td>A (0.3), G (0.6)</td>
<td>G (0.6), E (0.4)</td>
<td>A (0.2), G (0.8)</td>
<td>W (0.3), A (0.7)</td>
</tr>
<tr>
<td>Flexibility (ω₇)</td>
<td>Generality (ω₇₁)</td>
<td>A (0.2), G (0.6)</td>
<td>P (0.5), A (0.5)</td>
<td>G (0.4), E (0.5)</td>
<td>E (1.0)</td>
</tr>
<tr>
<td></td>
<td>Expandability (ω₇₂)</td>
<td>A (0.4), G (0.6)</td>
<td>E (0.9)</td>
<td>G (0.7), E (0.3)</td>
<td>G (0.1)</td>
</tr>
</tbody>
</table>

The subjective assessments for the qualitative attributes are concisely elaborated in Table 4 using their degrees of belief on each component in the hierarchy. This sub-criteria can be evaluated by professional people in the field of software development depending on their importance to the software.

7 | RESULTS

To aggregate assessments for components’ quality attributes, hypothetical weights of the ER are used for this purpose. The aggregated sub-criteria weights are considered to generate the assessment for all criteria, as shown in Figure 3. From equation (3), we have the belief degrees for usability sub-criteria on component 1 as follows:

\[
\beta_{1,1} = 0, \quad \beta_{1,2} = 0, \quad \beta_{1,3} = 0, \quad \beta_{4,1} = 0.4, \quad \beta_{4,2} = 0.5, \quad \beta_{5,1} = 0, \\
\beta_{1,2} = 0, \quad \beta_{2,2} = 0, \quad \beta_{3,1} = 0, \quad \beta_{4,2} = 0, \quad \beta_{5,2} = 0, \\
\beta_{1,3} = 0, \quad \beta_{2,3} = 0, \quad \beta_{3,3} = 0, \quad \beta_{4,3} = 0.4, \quad \beta_{5,3} = 0.6
\]

Then, To calculate the basic probability masses \(m, n, i\) when the seven main quality attributes are of equal importance \((ω₁) = (ω₂) = (ω₃) = (ω₄) = (ω₅) = (ω₆) = (ω₇) = 1/7\), we use equations (3) and (4) as follows:

\[
m_{1,1} = 0, \quad m_{1,2} = 0, \quad m_{1,3} = 0.4/3, \quad m_{4,1} = 0.5/3, \quad m_{5,1} = 0, \quad ̄m_{H,1} = 0.858, \quad ̃m_{H,1} = 0.1/3, \\
m_{1,2} = 0, \quad m_{2,2} = 0, \quad m_{3,1} = 0, \quad m_{4,2} = 1/3, \quad m_{5,2} = 0, \quad ̄m_{H,2} = 0.858, \quad ̃m_{H,2} = 0, \\
m_{1,3} = 0, \quad m_{2,3} = 0, \quad m_{3,3} = 0, \quad m_{4,3} = 0.4/3, \quad m_{5,3} = 0.6/3, \quad ̄m_{H,3} = 0.858, \quad ̃m_{H,3} = 0
\]
After that, the combined probability masses are calculated using the recursive equations (6a-6e). Thus, suppose \( m_{n,1} = m_{n,1} \) for \( n = 1, \ldots, 5 \). The results are calculated using the intelligent decision system (IDS) software developed to implement the ER approach and generate the necessary calculations graphically for component selection, as shown in Figure 4.

![Component assessment graphs](image-url)

**FIGURE 4** Components’ upper-level criteria assessment

Figure 4 shows the assessment results generated by the IDS software, which illustrates the aggregated assessments for the upper-level criteria for the 4 components depending on their evaluation grades and belief degrees. Such assessment takes the weight of each criterion and sub-criteria into consideration and ranks them first. After that, the differences between the four components can be identified and used to rank them to make the selection process easier.

For instance, the aggregated assessments for the upper-level criteria are generated for component 1 as follows:

\[
S(U_sability) = S(learnability \oplus easeofuse \oplus satisfaction) = (average, 0.03), (good, 0.73), (Excellent, 0.23)
\]

\[
S(Performance) = (average, 0.36), (good, 0.55), (Excellent, 0.09)
\]

\[
S(Reiability) = (average, 0.2), (good, 0.52), (Excellent, 0.2)
\]

\[
S(Maintainability) = (good, 0.38), (Excellent, 0.58)
\]

\[
S(Probability) = (worst, 0.27), (Poor, 0.18), (average, 0.13), (good, 0.34), (Excellent, 0.04)
\]

\[
S(Flexibility) = (average, 0.34), (good, 0.63)
\]
As shown in Figure 3 and Table 4, the assessment problem for the four components based on the criteria and sub-criteria assessment information arises. Therefore, the relative weights of criteria at a specific level are associated with the same upper-level criteria and defined by $\omega_i$, $\omega_j$, and $\omega_{ijk}$ for the criteria at other levels in the hierarchy. Thus, to demonstrate the ER algorithm without generality loss, all criteria are considered of equal importance to ensure the assessment reliability, where $\omega_i$ is the relative weight of the $ith$ criterion ($A_i$) with $0 \leq A_i \leq 1$. Suppose we have $L$ basic criteria at the lower level of the hierarchy $A_i$ ($i = 1, \ldots, L$) associated with the general component concept $Y$. Weights are given by $\omega = \omega_1, \omega_2, \ldots, \omega_L$ as follows:

$$\omega_1 = \omega_2 = \omega_3 = \omega_4 = \omega_5 = \omega_6 = \omega_7 = 0.142$$

$$\omega_{11} = \omega_{12} = \omega_{13} = 0.3333$$

$$\omega_{21} = \omega_{22} = 0.5; \omega_{31} = \omega_{32} = 0.5; \omega_{41} = \omega_{42} = 0.5$$

$$\omega_{51} = 1.0; \omega_{61} = 1.0; \omega_{71} = 1.0$$

After generating the aggregated assessment for the criteria of component 1, the final assessment for component 1 is generated as follows:

$S(\text{Component1}) = (\text{Worst, 0.037}), (\text{Poor, 0.0249}), (\text{Average, 0.209}), (\text{Good, 0.567}), (\text{Excellent, 0.135}), (H, 0.0258)$

The degree of incompleteness $H$ in the evaluation of component 1 is 0.0258 due to incomplete assessment for the sub-criteria for component 1 as shown in Table 2; the incompleteness degree is 10% which is reduced due to the large number of complete assessments in other sub-criteria.

Figure 5 shows the final assessment of the four components for partial ranking and selection. There are differences between the four components regarding the weights, evaluation grades, and belief degrees in each component’s criteria and sub-criteria.
For instance, the average assessments for the four components are 0.684, 0.701, 0.702, and 0.681, respectively. Thus, the four components could be ranked according to the assessment results of large and smaller assessment degrees where component 3 is preferred to component 2 and component 1 is preferred to component 4. However, the four components can be precisely ranked by estimating their five evaluation grades’ utilities. Therefore, normalization for the utility grades is required in this case. Suppose the utility value for the worst grade is 0 and the best grade is 1 as follows:

\[ u(H_1) = u(\text{poor}) = 0, u(H_2) = u(\text{excellent}) = 1 \]

Depending on the probability method, the utilities of the grades can be estimated by offering two hypothetical options to the decision-maker to select the right component. The first option offers a component with an average grade assessment. In contrast, the second option offers one component with a poor grade assessment with a probability of \(1 - p\) and another with an excellent grade assessment with a probability of \(p\). The probability \(p(0 \leq p \leq 1)\) is regulated until the decision-maker cannot distinguish between both options. Thus, suppose the decision-maker is tardy for both options when considering the value of \(p = 0.55\); the utility for the first option is calculated by \(u(H_3) = u(\text{average}) = (1 - p) \times u(\text{poor}) + p \times u(\text{excellent}) = 0.45 \times 0 + 0.55 \times 1 = 0.55\).

Similarly, the utilities of the worst and good grades might be estimated by supposing that \(u(H_2) = u(\text{worst}) = 0.35\) and \(u(H_4) = u(\text{good}) = 0.85\). Then, belief degrees for component 1 are given as follows:

\[ \beta_1 = 0.037, \beta_2 = 0.0249, \beta_3 = 0.209, \beta_4 = 0.567, \beta_5 = 0.135, \beta_H = 0.0258, \]

where \(\beta_H\) denotes the degree of incompleteness.

Since \(\beta_H \neq 0\), component 1 assessment is not complete and should be characterized by the utility interval \([u_{\text{min}}(\text{component 1}), u_{\text{max}}(\text{component 1})]\) depending on equations 10, 11 and 12 which are implemented graphically as shown in Figure 6 using IDS software to show the final ranking results for the four components using their utility intervals.

**Figure 6** Components’ final ranking using utility intervals

Figure 4 shows the min and max values for the four components to provide more precise assessment results to select the component with the highest-ranking quality criteria. Thus, the four components’ assessment results indicate the ranking for the four
components as follows:

Component 3 > Component 2 > Component 1 > Component 4

Where > denotes "better than" to indicate the best component to be selected. This ranking is generated based on the identical weights for all criteria in the hierarchy, as shown in Table 3.

8 | DISCUSSION

This study aimed to resolve the problem of COTS selection for online bookstore. The problem of components evaluation and selection for the used case study is modeled using multi-criteria decision-making (MCDM). Alternative COTS are evaluated and ranked based on criteria and sub-criteria weights using belief structure and an extended decision matrix to show their evaluation grades. Moreover, applying the main feature of the ER approach, such as information uncertainty, weight normalization, probability assessment, and utility intervals, helps decision-makers to select the right set of COTS. As shown in the results section of this study, criteria, and sub-criteria of the quality attributes for the online bookstore were aggregated for assessment to reduce the complexity of the selection process. As shown in section 7, field practitioners define belief degrees and evaluation grades for the sub-criteria depending on their subjective assessment and criteria importance for such systems. Then, sub-criteria weights are set to ensure the reliability of the assessment, which answers RQ1. In addition, uncertain information is assessed to calculate the incompleteness of all sub-criteria for each component for partial ranking and selection. After that, the basic and combined probability masses were calculated using an intelligent decision system (IDS) based on a set of mathematical equations. The results are based on the belief degrees and evaluation grades for upper-level criteria. Partially, COTS are ranked based on the generated assessment results, which answer RQ2. Finally, utility intervals are applied for a precise assessment after weight normalization to enable decision makers to make the final selection decision. Applying the features of the ER approach improves COTS selection based on its features, which lowers development costs and delivery time. In addition, it deals with large-size COTS, which increases the developed software’s reliability, functionality, and reusability. Many researchers proposed the selection methods, such as AHP, CV, PROMETHE, fuzzy methods, and multi-objective optimization models. Most of these methods have problems and limitations in the selection process, as stated in section 3. While the utilized approach in this study assesses functional and non-functional criteria and deals with information uncertainty using the D-S theory of evidence. In addition, ER uses an extended decision matrix to aggregate all components criteria, which reduces the selection process complexity, which answers RQ3, RQ4, and RQ5 of this study. The proposed approach had not been employed earlier for COTS selection and can be used as a benchmark method for the rapid solution of such component selection problems. Compared with other proposed approaches for solving the problem of COTS selection, ER approach is better in weight assigning, computational efficiency, problem-solving efficiency, reusability, and consistency and presentation of COTS selection results.

9 | CONCLUSION

Selecting the best software component that fits customer needs reduces the cost and time of software development and improves software quality. These factors of component-based software development have a direct impact on investment in the software development industry. However, the success of the selection process of software components based on its quality criteria requires a reliable method. In this study, the ER approach is applied to solve the problem of selecting a software component based on its quality criteria. Also, it is used for both functional and non-functional quality criteria. ER approach is one of the MCDM methods for decision-making. It has an extended decision matrix and belief structure for the criteria of each component and its alternatives after aggregating these criteria. Nevertheless, the ER framework establishes a non-linear relationship between the aggregated and essential measures in the hierarchy. Incomplete information can be handled to increase the quality of the data analysis. On the one hand, ER approach helps improve subjective judgments. It provides precise degrees of belief structure, improving consistency and reliability in the analysis process and enhancing decisions. On the other hand, decision matrices are used to lead to transparent and high-quality choices. The assessment of software components could be done depending on the criteria and original evaluation information related to the sub-criteria of these components. The criteria weights play an essential role in the reliability indication of the assessments. The case study dealt with software component selection problems for an online bookstore to show that the ER approach can solve such issues. The proposed approach in this study significantly
contributes to the body of knowledge in assisting decision-makers in the software COTS evaluation and selection process. Thus, selecting the right COTS that fits customer needs improves software reliability and effectiveness, minimizes development cost and delivery time, and maximizes software quality.

10 | ACKNOWLEDGMENT

This research is funded by the Deanship of Research and Graduate Studies at Zarqa University /Jordan.

References


70. Xu L, Yang JB. Introduction to multi-criteria decision making and the evidential reasoning approach. 2001.


82. Tzeng GH, Huang JJ. Multiple attribute decision making: methods and applications. CRC press . 2011.

How to cite this article: Nabot A., (2022), Software components selection: An optimized selection criterion for component-based software engineering (CBSE), Q.J.R. Meteorol. Soc., 2017;00:1–6.