

# Software component selection methods and techniques: a systematic review

Ahmad Nabot

Department of Software Engineering, Faculty of Information Technology, Zarqa University, Zarqa, Jordan

---

## Article Info

### Article history:

Received Nov 9, 2023

Revised Nov 23, 2023

Accepted Jan 6, 2024

### Keywords:

Decision making

Multi-criteria

Selection criteria

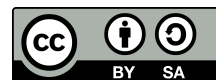
Software component selection

---

## ABSTRACT

Software component selection is critical in software engineering due to its vital role in reducing software development cost and time. This study analyzes software component selection research studies on methodologies, criteria, and multi-criteria decision-making (MCDM) techniques. The key study findings are: first, comprehensive standardized criteria for software component selection are lacking, with ambiguous terminology used in research. Second, current ad hoc selection processes need streamlining to reduce time, cost, and effort. Thus, an integrated approach is required to aid decision-makers. The review suggests developing automated tools or decision support systems combining multiple criteria decision methods to improve selection accuracy and efficiency. Standardized criteria catalogs can also assist software developers in the evaluation. The findings highlight that despite extensive academic research, component selection in practice remains sub-optimal. By informing future research and tool development, this review can benefit practitioners to systematically select the most appropriate software components meeting software requirements.

This is an open access article under the [CC BY-SA](#) license.



---

## Corresponding Author:

Ahmad Nabot

Department of Software Engineering, Faculty of Information Technology, Zarqa University

Zarqa, Jordan

Email: [anabot@zu.edu.jo](mailto:anabot@zu.edu.jo)

---

## 1. INTRODUCTION

Software component selection is a critical activity in software engineering that impacts cost, quality, and development efficiency [1]-[3]. As software reuse and component based development have grown, organizations increasingly rely on commercial-off-the-shelf (COTS) and open-source components [4]. However, selecting inappropriate components can lead to software development delays, integration issues, and higher costs [5]. Therefore, having a systematic selection process is vital. Past literature on software component selection has proposed various approaches, criteria, and decision-making techniques. For example, Kwong *et al.* [6] presented a genetic algorithm optimization model while Mancilla *et al.* [7] combined the COSTIUME and Azimut+ methods. Ibrahim *et al.* [8] introduced an uncertainty handling approach called uncertainty handling in COTS selection (UnHOS). However, as Ayala *et al.* [9] found in interviews with 13 companies, ad hoc selection processes are still common. Existing reviews have also focused on subsets like iterative development [10], criteria [4], or decision methods [11]. This reveals several gaps including the lack of a holistic perspective on selection approaches, ambiguity in criteria terminology, and limited adoption of academic advances by practitioners. To address these limitations, this study conducts a systematic literature review on software component selection covering methodologies, criteria, and multi-criteria decision-making (MCDM) techniques. Therefore, this study contributes to the literature by summarizing the state of research on software component selection, identify frequently used approaches, criteria, and MCDM methods, classifying criteria into functional and non-functional, and finally discussing implications and challenges to guide future research. The rest of the paper is organized as follows. The next section describes the systematic review method, subsequent sections present the results organized by research questions on contribution, approaches, MCDM techniques, and criteria. Finally, the conclusion summarizes findings and discusses limitations and future work.

## 2. RELATED WORK

Software component selection is a widely researched area, but current literature remains fragmented rather than providing an integrated perspective. For example, Ayala *et al.* [9] interviewed 13 companies and found ad hoc processes lacking systematic component selection methods. Only 4 companies had dedicated evaluation processes in place. Colakoglu *et al.* [5] developed a multi-criteria decision model for component selection focused on quality attributes like reliability, compatibility, and functionality. However, practitioner evaluation showed difficulties in accurately estimating criteria weights. Gardas *et al.* [12] tried to address this by designing an analytical hierarchy process (AHP) based decision support system to facilitate choosing open source components by reducing subjective weight bias. While promising, the validation was limited to a case study of an enterprise resource planning (ERP) system. Mehlawat *et al.* [13] applied fuzzy technique for order preference by similarities to ideal solution (TOPSIS) technique for component evaluation but did not address comparative selection across alternatives. Hasselbring [14] emphasized the need for improved selection processes but did not propose specific techniques. Several literature reviews exist but only focus on subsets of the literature. For instance, Kitchenham *et al.* [10] reviewed component selection methods based on iterative development cycles, excluding other paradigms. Mohamed *et al.* [4] studied selection criteria but did not analyze the overall processes. Pai *et al.* [11] examined the applicability of multi-criteria decision methods without broader assessment of their implementation. This fragmentation has led to scattered knowledge without a unified view of effective end-to-end selection approaches, standardized evaluation criteria, integration of qualitative and quantitative data, or dedicated tool support [15]. There are also gaps in understanding long-term costs, dynamic monitoring of deployed components, and comparisons across domains [16]. A systematic review synthesizing findings and limitations could reveal open challenges and future directions. It can also condense recommendations for adoption by practitioners struggling with ad hoc selection processes.

### 2.1. Selection methods/approaches and MCDM techniques

Component selection methods are critical in component-based software engineering (CBSE) as they determine the quality, efficiency, and cost-effectiveness of software systems [17]. Several activities are involved in component selection to improve usability. These include qualification to ensure functional and non-functional requirements are met, adaptation to modify a specific component based on predefined rules, and composition to integrate the selected component with others in building the software [18]. Nevertheless, selecting components can be challenging for software developers, as it often involves using specialized sourcing engines (SSE) and/or component repositories [9]. As a result, researchers have proposed various selection methods/techniques to address these challenges. For example, Kaur and Tomar [19] developed a multi-objective optimization model (MOO) using pre-emptive goal programming for software component selection. The MOO method optimizes conflicting selection criteria by prioritizing goals and selecting components that achieve the highest priority goals first [19]. Several methods and techniques have been proposed to address different issues, such as reducing the gap between actual requirements and enabling decision-makers to make informed decisions, as in the case of CARE [20]. Mancilla *et al.* [7] combined COSTUME and Azimut+ to address functional and non-functional requirements and classify components based on their non-functional requirements for selection. Ibrahim *et al.* [8] proposed a method called UnHOS which uses AHP and bayesian belief network (BBN) to select and evaluate COTS while handling uncertainty. Moreover, several other studies have utilized these techniques to facilitate the decision-making process for software component selection. Mittal [21] introduced a framework prioritizing reusability as the primary selection criterion for evaluating and selecting software components. They employed the AHP in their methodology. Similarly, Faridi *et al.* [22] recommended a technique based on the software quality model ISO/IEC 25010 to determine the suitable components during the selection process. Faundes *et al.* [23] suggested a fuzzy decision-making system-based approach for selecting components that evaluate and compare commercial off-the-shelf (COTS) and their impact on IT organizations. Kaur and Singh [24] introduced PROMETHE, a method for evaluating and selecting components that consider various quality attributes important for selecting COTS components, including reliability, integrability, performance, cost, and maintainability. Nazir *et al.* [25] suggested using the analytic network process (ANP) to select components, considering criteria such as effectiveness, efficiency, satisfaction, safety, and usability. Khan *et al.* [26] proposed a method for component selection based on CBSE to increase software reusability. This approach reduces development costs and time by selecting easily adaptable components to different software systems. Kuar and Tomar [27] demonstrated the effectiveness of clustering-based algorithms for selecting components by validating four approaches: fuzzy-c clustering, subtractive clustering, hybrid XOR-based clustering, and fuzzy relation-based clustering. Sekar and Sethuraman [28] suggested a technique for selecting components in web engineering that employs fuzzy ranking and rough sets to enhance the functionality of web applications. Tian *et al.* [29] suggested a method for selecting components based on clustering and information entropy weighting. Industry experts can use this method to choose suitable components by analyzing the selection results and utilizing artificial experience to make an optimal selection.

Kaur and Tomar [18] proposed a four-tier architecture for component selection that uses clustering. The architecture tiers include the component requirements and selection tier, the query and decision tier, the application logic tier, and the component cluster tier. Padhy *et al.* [30] presented a component selection approach by creating a reusability

matrix that outlines the selection criteria for different classes of components. Mehlawat *et al.* [13] have presented a hybrid framework named TOPSIS, which combines various software component selection and evaluation techniques previously applied in different contexts [31]. Sahu *et al.* [32] proposed a method that consisted of AHP, hesitant fuzzy (HF) sets and technique to test the reliability of software components. This method HF-AHP-TOPSIS tests the weights of software components to assess software engineers in the selection process. Kalantari *et al.* [33] proposed a model called fuzzy-intra coupling density (fuzzy-ICD) based on multi-objective optimization by maximizing the functions of fuzzy-ICD. Also, Mayers classification where utilized to calculate the relationship cohesion and coupling between software components.

However, the proposed method consumes time which increases the cost of software development. Gupta *et al.* [34] proposed an optimization model that uses data envelopment analysis technique (DEA) to evaluate the fitness of the software component based on the input and outputs. The proposed technique helps decision makers to select software components by integrating build and/or buy decisions. However, the proposed model does not deal with uncertainty and the use of redundant components, which increases the execution time. Verma *et al.* [35] proposed a model based on TOPSIS and multi-objective optimization for software component evaluation and selection. The proposed model assists software developers and decision makers to select software components when multiple applications being developed at the same time. Nevertheless, this model does not consider the uncertain information of software component selection. Bali *et al.* [36] presented intuitionistic fuzzy-TOPSIS approach as a solution for COTS assessment and selection. However, the presented approach does not rank the requirements of the component which affect the robustness of the built software.

## 2.2. Evaluation criteria

To choose the appropriate software component, it is necessary to establish a set of evaluation criteria based on both user and system requirements. As a result, selection methods are categorized based on these criteria. For example, some methods assess components based on functional requirements, while others evaluate them based on non-functional requirements. Some methods consider both types of requirements. Also, there are three strategies applied for evaluating software components according to [37]: i) progressive filtering (PF): this method gradually eliminates software components from the repository that do not meet the established evaluation criteria; ii) puzzle assembly (PZ): with this approach, the selected software components must be compatible with the environment of the primary software and integrate seamlessly with other system components [38]; and iii) keystone identification (KS): this method requires that the selected components meet the stakeholder's requirements and are in compliance with the latest technologies [37].

## 3. METHODS

This study applied systematic literature review (SLR) methodology because it incorporates a step-by-step details, explaining the rationale behind the chosen method, and defining the classification approach which makes the review process more reproducible and rigorous. Therefore, SLR used by this study because it is comprehensive, reproducible, reliable, and applicable for software component selection literature. Moreover, the applied method took similar steps as [39], [40]. SLR is a new method to produce an evidence regarding a specific technology to identify the followed direction and grade of research for investigating research questions/hypothesis [40]. SLR differs from ordinary literature review because it uses pre-defined search strings based on the study research questions to identify relevant studies and literature. Therefore, the final results of the SLR are considered precise, consistent, and unbiased in comparison with other methods. SLR has four phases, such as planning phase, selection phase, extraction phase, and reporting phase as shown in Figure 1. The proposed method has already applied the three phases to answer the study research.

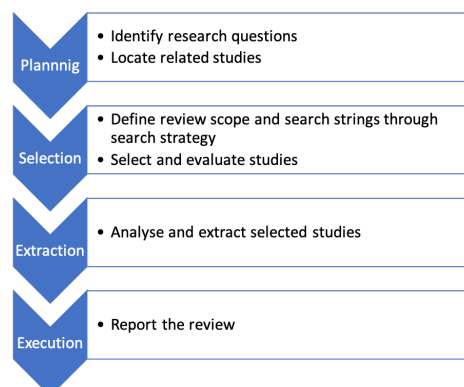


Figure 1. SLR method phases

### 3.1. Eligibility and study characteristics

This section presents inclusion and exclusion criteria used in this study. This review study only considered papers that fulfilled the following criteria as inclusion criteria: i) studies presenting software component selection methodologies and approaches, ii) studies providing criteria for software component selection, and iii) studies employing MCDM techniques to assist decision-makers in selecting the appropriate software component. The exclusion criteria were based on, i) opinion or discussion papers without substantial technical content and ii) papers not written in English. However, due to the scarcity of papers that described the same case, most papers were included in the study, which did not significantly affect the study’s findings.

### 3.2. Information sources and search strategy

In this study several electronic indexed databases by Scopus, and Web of science were searched to synthesis the required information from the published journal articles, conference proceedings, and technical reports. These databases are IEEE Xplore, ACM Digital Library, ScienceDirect, SpringerLink, and Wiley Online library. Additionally, relevant journals in the field were identified, such as Journal of Innovations in Engineering and Technology, Software Quality Journal, Information and Software Technology, Journal of Computer Applications, IEEE Software, and Journal of Computer Science and Information Technologies. We also searched for papers related to COTS in conference proceedings from Science Direct, IEEE, and Springer. The search terms employed for this purpose are illustrated in Figure 2.

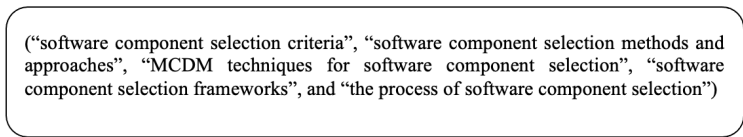


Figure 2. Search strategy terms

### 3.3. Paper selection

The selection process had two stages: i) initial screening based on the studies’ titles/abstracts obtained from the search results, and ii) thorough reading of the main body of the papers for the final selection. Initially, over 100 relevant studies were identified from the search. These papers were examined to narrow the pool to those most pertinent to the research question. After this initial screening, 52 papers were shortlisted for review. a number of the examined papers were eliminated during the second round of paper selection as they did not provide valuable information on software component selection, selection methodologies and approaches, or MCDM techniques for software component selection. Consequently, a total of 24 papers underwent selection for inclusion in the review process as shown in Table 2. Additionally, a cross-verification was conducted on the final list of reviewed papers to ascertain their alignment in addressing the research questions and substantiating the principal objectives of this study.

### 3.4. Data extraction

The study carefully reviewed all the selected papers and extracted the following attributes into a standard template: authors, publication year, title, selection criteria, selection method/approach, and MCDM technique see Table 1. During data extraction, irrelevant information were captured to minimize subjective interpretations. Thus, three papers were found to be completely irrelevant for software component selection.

Table 1. Data extracted from the reviewed studies

Attribute	Description
Author(s)	Author(s) of the study
Publication year	publication year
Title	Title of the study
Methodology/approach used for selection	The study methodology/approach used for software component selection
MCDM technique	The MCDM technique used in the study
Selection criteria	The software component criteria led to component selection

### 3.5. Criteria classification

Functional and non-functional criteria were used to categorize selection criteria based on the software component capabilities they assess. Functional criteria relate to the component’s services, features, and core functionality. While non-functional criteria evaluate non-functional attributes of the component such as reliability, efficiency, usability, and security.

## 4. RESULTS

### 4.1. RQ1: Software component selection contribution to the literature

This study categorized the reviewed literature into three groups: methodologies and approaches for software component selection, MCDM techniques for software component selection, and selection criteria for software components. Table 2 provides an overview of the reviewed literature on software component selection. While Figure 3 illustrates the contribution of the literature in each category.

Table 2. Contribution of the reviewed literature summary

Author(s)	Software component type	Method/approach	Criteria	MCDM technique
Kwong <i>et al.</i> [6]	Financial system	Genetic algorithm (GA)	Yes	Yes
Mancilla <i>et al.</i> [7]	Restaurant administration system	Combination of COSTUME and Azimut+	Yes	Yes
Ibrahim <i>et al.</i> [8]	Airline reservation system	UNHOS	Yes	Yes
Gupta <i>et al.</i> [41]	System prototype	FMP	Yes	Yes
Nazir <i>et al.</i> [42]	Hypothetical case study	OTOSO and fuzzy logic	Yes	Yes
Zhiqiao <i>et al.</i> [43]	Mail server system	Lagrange relaxation decomposition with heuristic	Yes	Yes
Khan and Mahmood [44]	Meeting scheduling system	Cluster-based component selection	Yes	Yes
Gupta <i>et al.</i> [45]	ERP software	FMOP	Yes	Yes
Becker <i>et al.</i> [46]	Different real-world case	Methodical analysis approach	Yes	Yes
Kuar and Singh [24]	General software systems	PROMETHEE	Yes	Yes
Nazir <i>et al.</i> [25]	Different ERP software	ANP	Yes	Yes
Kuar and Tomar [19]	.Net software	Pre-emptive goal programming	Yes	Yes
Vescan [47]	Reservation system	Evolutionary-based algorithm	Yes	Yes
Kaur and Tomar [48]	Set of sorting and searching components	Four-tier architecture technique	Yes	Yes
Kaur and Tomar [27]	Document manager	Fuzzy clustering	Yes	Yes
Vescan and Şerban [49]	Reservation system	Multi-objective optimization approach	Yes	Yes
Verma and Mehlawat [50]	Numerical illustration of software application	Combined MCDM-AHP model	Yes	Yes
Khan <i>et al.</i> [51]	Bounty rescue step game	Multi-agent approach	Yes	Yes
Sekar <i>et al.</i> [52]	Technology gallery	K-means clustering, rank order clustering and similarity-based coefficients	Yes	Yes
Chatzipetrou <i>et al.</i> [53]	Different types of components	CoDA	Yes	Yes
Roopa and Reddy [54]	SMEs software system	Computational algorithm	Yes	Yes
Mehlawat <i>et al.</i> [13]	E-commerce software	TOPSIS	Yes	Yes
Gusev <i>et al.</i> [55]	Digital psychological tools	Mathematical model	Yes	Yes
Garg [56]	E-payment system	FMDBA	Yes	Yes

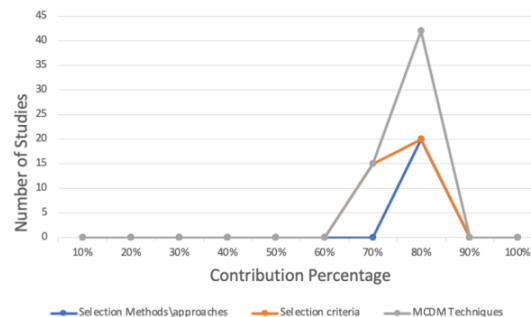


Figure 3. Contribution of the reviewed literature

Table 2 provides a summary for various software components, their associated methods/approaches, evaluation criteria, and the MCDM techniques employed for analysis. Each column in the table represents specific information to the body of knowledge. For instance, authors column represents studies conducted by different authors with their references to easily allow the reader to locate the full details of the respective study. software component type column identifies the type of software system being analyzed in these studies. The method/approach column describes the method/approach being utilised in the analysis of the software component. As shown in this column, different methods/approaches are used to analyse software components. The criteria column indicates whether evaluation criteria were applied to assess the software component. Finally, the MCDM technique column indicates whether MCDM technique is utilised in the analysis or not. The provided data give an evidence of the variety of software component types were examined using different methods/approaches. The study showed the most common techniques being utilised for components analysis. Moreover, the presence of the criteria column indicates that these studies are involved in decision-making process where multiple factors/criteria required. In addition, the MCDM technique column indicates that the authors were interested in assessing and comparing different options based on multiple criteria, which is common in software development and decision-making scenarios. At the end, the table provided a snapshot of a diverse set of studies that addresses different software components related studies and end employing various methods/approaches. It worth's nothing that without a complete context and content of each study, it's challenging to delve into deeper analysis or draw broader conclusions about trends in software development pr decision-making. Figure 3 shows the contribution percentage of the reviewed literature in each category in Table 2. For instance, 80% of the studies contributed to the literature in terms of the selection methods/approaches, 70%

of the studies contributed to the literature in terms of the selection criteria, and more than 80% of the studies contributed to the literature in terms of the utilised MCDM techniques. This indicates that most of the studies depend mainly on one of the MCDM techniques to enable decision makers to take the right decision in the selection process.

#### 4.2. RQ2: Methodologies and approaches for software component selection

Approaches and methods used to select the most appropriate software component can provide clear guidance on the relevant factors and issues to consider during the selection process. They can also serve as a roadmap or guide for the needs of individuals and organizations [57]. As shown in Table 2, 24 of the reviewed studies focused on the methodology of selecting commercial off-the-shelf (COTS) software. Based on the literature review conducted in this study, the following steps are proposed for selecting software components: i) identify the needs of the software component by exploring alternatives and examining vendor offerings to assess quality and features, ii) create a shortlist of potential options to simplify the selection process, iii) determine selection criteria, such as hardware specifications, operating system requirements, and necessary functionalities. iv) evaluate the shortlisted software components using the appropriate technique, v) to make informed decisions, assess and evaluate the selected component quality under specific conditions, and vi) employ MCDM techniques to select the appropriate component that meets individual/organizational needs.

#### 4.3. RQ3: MCDM techniques

The process of selecting software components involves multiple activities as indicated in RQ2, which also indicates the growing need for MCDM techniques to support these activities. As presented in Table 2, many studies proposed various MCDM techniques to aid decision makers in selecting the appropriate software components. Furthermore, several studies utilized these techniques to make informed decisions for software component selection. However, these techniques have limitations, such as constraints imposed by component capabilities and their integration with other components. Furthermore, when component alternatives are available, selecting the appropriate component is a complex task [58].

#### 4.4. RQ4: Selection criteria

The process of selecting a software component involves different activities as shown in RQ2, which requires the developer to consider the selection criteria. These criteria is divided into quality attributes and service attributes including efficiency, reliability, usability, maintainability, and reliability based on the provided services by the software component. Some methods/approaches consider only quality attributes, while others consider both requirements. Moreover, various quality models organize the quality attributes of software components, such as McCall, FURPS, FURPS+, ISO/IEC 9126, and ISO/IEC 25010:2011 models. These models categorize quality attributes into product operation, revision, and transition.

The most common criteria for software component selection in published papers include vendor, software and hardware requirements, and cost. Franch and Carvallo proposed a classification scheme for software criteria, which includes functional criteria, quality attributes, and criteria classification according to the literature. Tables 3-5 represent these categories in detail. However, it should be noted that the selection process may face certain constraints, such as component capabilities and integration with other components, as well as the complexity of selecting the best component when the number of alternatives is large.

Table 3. Software component functional criteria

Criteria	Classification	Description
Service provided by component	Functional	All functionalities/services provided by the software component to the user
Main aim	Functional	Main functionality provided by the software component for which is oriented
Completeness	Functional	The degree of the functional requirements performs well
Authentication	Functional	The capability of the software component to identify its users
Administrative functions	Functional	The capability of the software component to perform routine functions such as reporting

Table 3 presents an explanation of the different software component functional criteria that help evaluate and understand the functionality of a software component. These criteria also help assess how well a software component meets its intended purpose, performs its core functions, and interacts with users. The categorization of these criteria as "Functional" indicates that they focus on the practical capabilities and features of the software component. Moreover, these criteria play a significant role in evaluating and comparing software components to determine their stability for specific application or context. Each row in Table 3 represents a specific criterion, and the columns provide details about its classification and description. For instance, the "Criteria" column represents the specific aspect that is being considered when evaluating a software component. The "Classification" column categorizes the criteria into a certain classification (i.e., functional), which means that all listed criteria are related to functional aspects of the software component's performance and capabilities. Finally, the "Description" column provides a brief explanation of each criterion to help readers understand what the criterion entails. As shown in the Table 4, there are different criteria used for the evaluation process

including services provided by component to indicate the provided functionalities and services. In addition to the main aim of the software component, completeness of the provided functionalities, authentication of the software component users, and administrative functions that deals with software component's capacity to carry out routine administrative tasks.

Table 4. Software component quality attributes

Criteria	Classification	Description
Usability	Non-Functional	Represents code size, independency of other components, number of users, and the ability to handle many users and transactions.
Maintainability	Non-Functional	Represents ease of use, effectiveness, learnability, and memorability of the software component
Reliability	Non-Functional	Component ability to handle user and environmental errors without crashing and support backup and recovery
Security	Non-Functional	The capability of the software component to handle attacks and data encryption
Efficiency	Non-Functional	The capability of the software component to respond to the user in a reasonable time and not consume lots of resources
Portability	Non-Functional	The capability of the software component to run on different models such as CORBA, OLE, .NET, and JDBC
Interoperability	Non-Functional	The capability of the software component to integrate with other software platforms
Reusability	Non-Functional	The capability of the software component to be reused with another software environment

Table 4 presents the required criteria that help assess a component's alignment with intended purposes, core functions, and user interactions. The provided analysis summarises the contents of functional criteria used to evaluate software components. Each row in the table corresponds to a specific evaluation criterion, and the columns provide further details about that criterion. For instance, the "Criteria" column identifies the specific aspect considered when evaluating a software component. The "Classification" column groups the criteria into one category (i.e., functional). Finally, the "Description" column provides concise explanations for each criterion to make it easy to understand what each one involves.

Table 5. Software component literature criteria classification

Criteria	Classification	Description
Tutorial	Vendor	Tutorial availability for the software component
Maintenance and Upgrade	Vendor	Support of the vendor for the component maintenance and upgrade
License and training	Cost	The cost of the software component license and users training
Maintenance and Upgrade cost	Cost	The cost of the software component maintenance and upgrade
Storage and communication protocols	Hardware	The required space on memory, communications protocol to run the software component
Compatibility and source code	Software	The compatibility of the software component and the integration environment, and the availability of the source code

Table 5 presents a set of criteria used to evaluate software components. Each criterion is categorized based on its primary focus, such as vendor-related factors, costs, hardware, and software aspects. The criteria highlights various important facets of software components, ranging from tutorials and support to financial considerations and technical requirements. The "Criteria" column outlines specific aspects that are taken in considerations when evaluating software components. The "Classification" column categorizes each criterion into different classifications that corresponds to relevant characteristics such as vendor, cost, hardware, and cost. Finally, the "Description" column explains each criterion in a concise way.

## 5. DISCUSSION

Based on the established research questions in this study, the study addresses these questions as follows:

RQ1, The study identifies a notable absence of comprehensive criteria catalogs devised for the purpose of software component selection based on standardized and universally acknowledged evaluation measures for software components. Furthermore, scholarly endeavors have presented a multitude of disparate criteria; however, they are plagued by inherent ambiguity and inconsistency in terminologies employed. Additionally, some studies concentrate on quality attributes such as reliability or usability, while others consider functional criteria, vendor-related factors, and cost aspects. This variance in the definition of criteria poses a significant challenge in objectively comparing and selecting software components. Consequently, the presence of comprehensive criteria catalog emerges as a potential remedy to standardize the selection process.

RQ2, emphasizes the necessity for refined selection methodologies within the current industrial landscape. Presently, ad-hoc selection process prevailing in industry settings exhibit inefficiencies, inconsistencies, and susceptibility to sub-optimal choices. Their inherent deficiencies lie in the absence of systematic techniques for the stages of identification, prioritization, assessment, and decision-making. Enhancing these processes necessitates the implementation of streamlined frameworks that encompass comprehensive criteria definition, weighting mechanisms, candidates evaluation protocols, and optimization algorithms, thereby elevating the level of rigor. The integration of automated tools aligned with these selection frameworks stands poised to significantly diminish manual efforts and reduce time consumption. Furthermore, standardized procedures hold the promise of mitigating the occurrence of costly rework resultant from ill-informed choices, thereby underscoring their potential value within the selection paradigm.

RQ3, underscores the importance of integrating MCDM techniques to facilitate informed decision-making in the selection of optimal software components. This process necessitates the evaluation of trade-offs among numerous criteria, mandating the utilization of MCDM techniques. The comprehensive review identified a spectrum of diverse MCDM methodologies, including but not limited to AHP, TOPSIS, and fuzzy logic, which have been employed in software component selection context. Presently, these techniques are predominantly utilized in isolation, primarily aiding in criteria weighting and hierarchical ranking of alternatives. However, the integration of these MCDM methodologies within the selection framework holds promise in systematically guiding the decision-making process. Furthermore, the incorporation of automated tools that implement these techniques stands to augment the expediency of decision-making, facilitating faster and evidence-based selections. This review aimed to offer a holistic perspective on software component selection research, shedding light on various challenges and limitations. Despite decades of academic exploration into systematic methodologies, ad-hoc selection processes persist, indicating industry reluctance to adopt structured approaches. To bolster selection practices, the integration of frameworks, improved tools, and empirical comparisons is necessary. The diverse landscape of criteria terminology, categories, and their significance catalogs to enable consistent evaluation. Established quality models like ISO 25010 serve as valuable starting points. While existing research often delves into criteria weighting and isolated techniques, there's a pressing need for integrating qualitative/quantitative data, considering long-term costs, and dynamically verifying deployed components. Furthermore, the lack of real-world validation for reported techniques hampers practical adoption. Crucially, a few studies delve into organizational and managerial aspects of selection, such as cross-team coordination and vendor evaluations. In essence, this review consolidates scattered literature to reveal ongoing challenges, proposing improvements to steer future research towards comprehensive, standardized, and empirically validated selection methodologies finely attuned to industrial settings.

## 6. CONCLUSION

This study presented a systematic review of software component selection literature focused methodologies, criteria, and MCDM techniques. The findings reveal that while many techniques have been proposed, and ad-hoc selection processes remain common in practice. This demonstration gaps between academic research and industrial adoption. The review also found inconsistencies in criteria terminology, with no standardized comprehensive criteria list. Based on the review, some future research directions are identified. There is a need for integrated selection frameworks that unify criteria, assessments, and decision-making. Developing supporting tools aligned with industrial environments can also facilitate adoption. Comparative empirical studies on different techniques and standardized criteria catalogs can drive consensus. Finally, dynamic verification of deployed components and cost considerations need more focus. The review methodology itself had some limitations. The search process could miss relevant papers despite a broad strategy. Extracted data dependent on the authors' subjective interpretation. Criteria classification schemes also require robust definitions. Nonetheless, the study condenses extensive research on software component selection to highlight key challenges and implications for both academics and practitioners.

## 7. FUTURE WORK

This systematic review study draws several potential future work avenues. First, there is a need to construct a comprehensive software component selection framework that harmonizes criteria definition, assessment mechanisms, and MCDM techniques, encompassing both qualitative and quantitative metrics. This framework should be designed to cater specifically to industry needs, accompanied by supporting tools tailored for criteria weighting, candidate evaluation, and final selections, aiming to ease adoption within practical settings. Moreover, conducting thorough empirical comparisons among various selection approaches using real-world systems can ascertain the most effective techniques. This entails the derivation of a standardized, universally-applicable criteria catalog covering functional, quality, vendor-related, cost, and other pertinent attributes crucial for software component selection. Additionally, exploring aspects such as long-term costs, integration challenges, and post-deployment verification of selected components through field studies will provide valuable insights. Specialized techniques focusing on organizational and managerial perspectives, including vendor evaluations and team coordination during the selection process, require dedicated investigation. Furthermore, it's essential to evaluate these proposed techniques across diverse software domains, such as enterprise systems, embedded systems, and AI applications to ensure broad relevance and applicability. Finally, these future work suggestions seek to standardized and optimized software component selection, enhancing its practical nature. Additionally, these suggestions aspire to bridge the gap between academic research and industrial practices, fostering mutual enhancements for both domains.

## ACKNOWLEDGEMENT

This research is funded (fully/partially) by Zarqa University-Jordan.






## REFERENCES

- [1] A. S. Jadhav and R. M. Sonar, "Evaluating and selecting software packages: a review," *Information and Software Technology*, vol. 51, no. 3, pp. 555–563, 2009, doi: 10.1016/j.infsof.2008.09.003.
- [2] J. Kontio, "Case study in applying a systematic method for COTS selection," *Proceedings - International Conference on Software Engineering*, pp. 201–209, 1995, doi: 10.1109/icse.1996.493416.
- [3] J. Aggarwal and M. Kumar, "Software metrics for reusability of component based software system: a review," *International Arab Journal of Information Technology*, vol. 18, no. 3, pp. 319–325, 2021, doi: 10.34028/iajit/18/3/8.
- [4] A. Mohamed, G. Ruhe, and A. Eberlein, "COTS selection: past, present, and future," in *14th Annual IEEE International Conference and Workshops on the Engineering of Computer-Based Systems (ECBS'07)*, Mar. 2007, pp. 103–114, doi: 10.1109/ECBS.2007.28.
- [5] F. N. Colakoglu, A. Yazici, and A. Mishra, "Software product quality metrics: a systematic mapping study," *IEEE Access*, vol. 9, pp. 44647–44670, 2021, doi: 10.1109/ACCESS.2021.3054730.
- [6] C. K. Kwong, L. F. Mu, J. F. Tang, and X. G. Luo, "Optimization of software components selection for component-based software system development," *Computers and Industrial Engineering*, vol. 58, no. 4, pp. 618–624, 2010, doi: 10.1016/j.cie.2010.01.003.
- [7] F. Mancilla, H. Astudillo, and M. Visconti, "Combining COSTUME and Azimut+ to address functional and non-functional requirements in software component selection," *Proceedings - International Conference of the Chilean Computer Science Society, SCCC*, pp. 102–109, 2010, doi: 10.1109/SCCC.2010.47.
- [8] H. Ibrahim, A. H. Elamy, B. H. Far, and A. Eberlein, "UnHOS: a method for uncertainty handling in commercial off-the-shelf (COTS) selection," *International Journal of Energy, Information and Communications*, vol. 2, no. 3, pp. 21–48, 2011.
- [9] C. Ayala, Ø. Hauge, R. Conradi, X. Franch, and J. Li, "Selection of third party software in off-the-shelf-based software development - an interview study with industrial practitioners," *Journal of Systems and Software*, vol. 84, no. 4, pp. 620–637, 2011, doi: 10.1016/j.jss.2010.10.019.
- [10] B. Kitchenham, O. P. Brereton, D. Budgen, M. Turner, J. Bailey, and S. Linkman, "Systematic literature reviews in software engineering - a systematic literature review," *Information and Software Technology*, vol. 51, no. 1, pp. 7–15, Jan. 2009, doi: 10.1016/j.infsof.2008.09.009.
- [11] P. F. Pai, C. T. Chen, and W. Z. Hung, "Applying linguistic information and intersection concept to improve effectiveness of multi-criteria decision analysis technology," *International Journal of Information Technology and Decision Making*, vol. 13, no. 2, pp. 291–315, 2014, doi: 10.1142/S0219622014500436.
- [12] B. B. Gardas, R. D. Raut, and A. Shrivastav, "Efficient supplier selection - a three-stage multi-criteria decision-making approach," *International Journal of Logistics Systems and Management*, vol. 34, no. 3, pp. 375–394, 2019, doi: 10.1504/IJLSM.2019.103090.
- [13] M. K. Mehlatat, P. Gupta, and D. Mahajan, "A multi-period multi-objective optimization framework for software enhancement and component evaluation, selection and integration," *Information Sciences*, vol. 523, pp. 91–110, 2020, doi: 10.1016/j.ins.2020.02.076.
- [14] W. Hasselbring, "Component-based software engineering," in *In Handbook of Software Engineering and Knowledge Engineering: Volume II: Emerging Technologies*, May 2002, pp. 289–305, doi: 10.1142/9789812389701\_0013.
- [15] P. Vitharana, H. Jain, and F. Zahedi, "Strategy-based design of reusable business components," *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*, vol. 34, no. 4, pp. 460–474, 2004, doi: 10.1109/TSMCC.2004.829258.
- [16] D. M. Fernández et al., "Naming the pain in requirements engineering: contemporary problems, causes, and effects in practice," *Empirical Software Engineering*, vol. 22, no. 5, pp. 2298–2338, 2017, doi: 10.1007/s10664-016-9451-7.
- [17] K. Petersen et al., "Choosing component origins for software intensive systems: in-house, COTS, OSS or outsourcing? - A case survey," *IEEE Transactions on Software Engineering*, vol. 44, no. 3, pp. 237–261, 2018, doi: 10.1109/TSE.2017.2677909.
- [18] J. Kaur and P. Tomar, "Clustering based architecture for software component selection," *International Journal of Modern Education and Computer Science*, vol. 10, no. 8, pp. 33–40, 2018, doi: 10.5815/ijmecs.2018.08.04.
- [19] J. Kaur and P. Tomar, "Multi objective optimization model using preemptive goal programming for software component selection," *International Journal of Information Technology and Computer Science*, vol. 7, no. 9, pp. 31–37, 2015, doi: 10.5815/ijitcs.2015.09.05.
- [20] L. Chung and K. Cooper, "Defining goals in a COTS-aware requirements engineering approach," *Systems Engineering*, vol. 7, no. 1, pp. 61–83, 2004, doi: 10.1002/sys.10055.
- [21] S. Mittal, "Framework for evaluating and ranking the reusability of COTS components based upon analytical hierarchy process," *International Journal of Innovations in Engineering and Technology (IJJET)*, vol. 2, no. 4, pp. 352–360, 2013.
- [22] M. S. Faridi, Z. Javed, M. H. Abid, A. Mudassar, and A. Ngadi, "IROTS: a proposed COTS evaluation a selection methodology for component based software engineering in under-development countries," *2nd International Conference on Advances in Computer Science and Engineering (CSE 2013)*, Atlantis Press, 2013, doi: 10.2991/cse.2013.6.
- [23] M. J. Faundes, H. Astudillo, and B. Hitpass, "Process-based evaluation and comparison of OTS software alternatives," *2013 Federated Conference on Computer Science and Information Systems, FedCSIS 2013*, 2013, pp. 1093–1100.
- [24] K. Kaur and H. Singh, "Quantifying COTS component selection using multi criteria decision analysis," *Global Journal of Computer Science and Technology*, 2014.
- [25] S. Nazir et al., "Software component selection based on quality criteria using the analytic network process," in *Abstract and Applied Analysis*, 2014, vol. 2014, doi: 10.1155/2014/535970.
- [26] A. Khan, K. Khan, M. Amir, and M. N. A. Khan, "A component-based framework for software reusability," *International Journal of Software Engineering and its Applications*, vol. 8, no. 10, pp. 13–24, 2014, doi: 10.14257/ijseia.2014.8.10.0.
- [27] J. Kaur and P. Tomar, "A software component selection technique based on fuzzy clustering," in *2016 1st India International Conference on Information Processing (IICIP)*, Aug. 2016, pp. 1–5, doi: 10.1109/IICIP.2016.7975350.
- [28] K. R. Sekar and J. Sethuraman, "Optimal component selection for rich internet applications in web engineering," in *2017 International Conference on Networks & Advances in Computational Technologies (NetACT)*, Jul. 2017, pp. 420–425, doi: 10.1109/NETACT.2017.8076808.
- [29] S. Tian, H. Wang, Y. Jiang, and X. Chen, "A new software component selection method based on clustering and information entropy weighting," *Computer & Digital Engineering*, no. 12, 2017.
- [30] N. Padhy, R. Panigrahi, and S. C. Satapathy, "Identifying the reusable components from component-based system: proposed metrics and model," in *Advances in Intelligent Systems and Computing*, vol. 863, 2019, pp. 89–99.
- [31] R. Ghlala, Z. Kodia, and L. Ben Said, "Using MCDM and FaaS in automating the eligibility of business rules in the decision-making process," *International Arab Journal of Information Technology*, vol. 20, no. 2, pp. 224–233, 2023, doi: 10.34028/iajit/20/2/9.
- [32] K. Sahu, F. A. Alzahrani, R. K. Srivastava, and R. Kumar, "Evaluating the impact of prediction techniques: software reliability perspective," *Computers, Materials & Continua*, vol. 67, no. 2, pp. 1471–1488, 2021, doi: 10.32604/cmc.2021.014868.
- [33] S. Kalantari, H. Motameni, E. Akbari, and M. Rabbani, "Optimal components selection based on fuzzy-intra coupling density for component-based software systems under build-or-buy scheme," *Complex & Intelligent Systems*, vol. 7, no. 6, pp. 3111–3134, Dec. 2021, doi: 10.1007/s40747-021-00449-z.

- [34] P. Gupta, M. K. Mehlawat, and D. Mahajan, "Data envelopment analysis based multi-objective optimization model for evaluation and selection of software components under optimal redundancy," *Annals of Operations Research*, vol. 312, no. 1, pp. 193–216, May 2022, doi: 10.1007/s10479-018-2842-y.
- [35] S. Verma, M. K. Mehlawat, and D. Mahajan, "Software component evaluation and selection using TOPSIS and fuzzy interactive approach under multiple applications development," *Annals of Operations Research*, vol. 312, no. 1, pp. 441–471, 2022, doi: 10.1007/s10479-018-3022-9.
- [36] V. Bali, S. Bali, D. Gaur, S. Rani, and R. Kumar, "Commercial-off-the shelf vendor selection: a multi-criteria decision-making approach using intuitionistic fuzzy sets and topsis," *Operational Research in Engineering Sciences: Theory and Applications*, vol. 6, no. 2, pp. 34–51, 2023, doi: 10.31181/oresta/060203.
- [37] P. A. Oberndorf, L. Brownsword, and E. Morris, "Workshop on COTS-based systems," CARNEGIE-Mellon University, Pittsburgh, PA, USA, 1997.
- [38] Z. Javed, A. R. Sattar, and M. S. Faridi, "Unsolved tricky Issues on COTS selection and evaluation," *Global Journal of Computer Science and Technology Neural & Artificial Intelligence*, vol. 12, no. 10, 2012.
- [39] S. Keele, "Guidelines for performing systematic literature reviews in software engineering," Technical report, Ver. 2.3 EBSE Technical Report. EBSE, 2007.
- [40] A. W. Khan and S. U. Khan, "Critical success factors for offshore software outsourcing contract management from vendors' perspective: An exploratory study using a systematic literature review," *IET Software*, vol. 7, no. 6, pp. 327–338, 2013, doi: 10.1049/iet-sen.2013.0013.
- [41] P. Gupta, M. K. Mehlawat, and S. Verma, "COTS selection using fuzzy interactive approach," *Optimization Letters*, vol. 6, no. 2, pp. 273–289, 2012, doi: 10.1007/s11590-010-0243-5.
- [42] S. Nazir, M. A. Khan, S. Anwar, H. Khan, and M. Nazir, "A novel fuzzy logic based software component selection modeling," in *2012 International Conference on Information Science and Applications*, May 2012, pp. 1–6, doi: 10.1109/ICISA.2012.6220925.
- [43] W. Zhiqiao, C. K. Kwong, J. Tang, and J. W. K. Chan, "Integrated model for software component selection with simultaneous consideration of implementation and verification," *Computers & Operations Research*, vol. 39, no. 12, pp. 3376–3393, Dec. 2012, doi: 10.1016/j.cor.2012.04.020.
- [44] M. A. Khan and S. Mahmood, "A graph based requirements clustering approach for component selection," *Advances in Engineering Software*, vol. 54, pp. 1–16, Dec. 2012, doi: 10.1016/j.advengsoft.2012.08.002.
- [45] P. Gupta, H. Pham, M. K. Mehlawat, and S. Verma, "A fuzzy optimization framework for COTS products selection of modular software systems," *International Journal of Fuzzy Systems*, vol. 15, no. 2, pp. 91–109, 2013.
- [46] C. Becker, M. Kraxner, M. Plangg, and A. Rauber, "Improving decision support for software component selection through systematic cross-referencing and analysis of multiple decision criteria," in *2013 46th Hawaii International Conference on System Sciences*, Jan. 2013, pp. 1193–1202, doi: 10.1109/HICSS.2013.263.
- [47] A. Vescan, "Case study method and research design for the dynamic multilevel component selection problem," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 9586, 2016, pp. 130–141.
- [48] J. Kaur and P. Tomar, "Four tier architecture of component selection process using clustering," *International Journal of Software Engineering, Technology and Applications*, vol. 1, no. 2/3/4, p. 155, 2015, doi: 10.1504/IJSETA.2015.075611.
- [49] A. Vescan and C. Şerban, "Multilevel component selection optimization toward an optimal architecture," *Soft Computing*, vol. 21, no. 15, pp. 4481–4495, Aug. 2017, doi: 10.1007/s00500-016-2472-8.
- [50] S. Verma and M. K. Mehlawat, "Multi-criteria optimization model integrated with AHP for evaluation and selection of COTS components," *Optimization*, vol. 66, no. 11, pp. 1879–1894, Nov. 2017, doi: 10.1080/02331934.2017.1316502.
- [51] M. W. Khan, J. Wang, L. Xiong, and M. Ma, "Modelling and optimal management of distributed microgrid using multi-agent systems," *Sustainable Cities and Society*, vol. 41, pp. 154–169, Aug. 2018, doi: 10.1016/j.scs.2018.05.018.
- [52] K. R. Sekar, J. Sethuraman, and R. Manikandan, "A novel software component selection through statistical models," *Proceedings of the 2nd International Conference on Trends in Electronics and Informatics, ICOEI 2018*, pp. 1391–1396, 2018, doi: 10.1109/ICOEI.2018.8553696.
- [53] P. Chatzipetrou, E. Papatheocharous, K. Wnuk, M. Borg, E. Alégroth, and T. Gorschek, "Component attributes and their importance in decisions and component selection," *Software Quality Journal*, vol. 28, no. 2, pp. 567–593, Jun. 2020, doi: 10.1007/s11219-019-09465-2.
- [54] Y. M. Roopa and A. R. M. Reddy, "An optimised component selection algorithm for self-adaptive software architecture using the component repository," *International Journal of Advanced Intelligence Paradigms*, vol. 14, no. 3/4, p. 236, 2019, doi: 10.1504/IJAIP.2019.103411.
- [55] A. Gusev, D. Ilin, P. Kolyasnikov, and E. Nikulchev, "Effective selection of software components based on experimental evaluations of quality of operation," *Engineering Letters*, vol. 28, no. 2, pp. 420–427, 2020.
- [56] R. Garg, "A ranking model for the selection and ranking of commercial off-the-shelf components," *IEEE Transactions on Engineering Management*, vol. 69, no. 5, pp. 2196–2204, Oct. 2022, doi: 10.1109/TEM.2020.3001943.
- [57] N. Patel and V. Hlupic, "A methodology for the selection of knowledge management (KM) tools," *Proceedings of the International Conference on Information Technology Interfaces, ITI*, pp. 369–374, 2002, doi: 10.1109/ITI.2002.1024701.
- [58] S. Gholamshahi and S. M. H. Hasheminejad, "Software component identification and selection: a research review," *Software: Practice and Experience*, vol. 49, no. 1, pp. 40–69, Jan. 2019, doi: 10.1002/spe.2656.

## BIOGRAPHIES OF AUTHORS



**Ahmad Nabot**    is Assistant Professor at Faculty of Information Technology, Zarqa University, Jordan. He Holds a Ph.D. degree in computer science with specialization in software engineering. His research areas are software development, decision making, machine learning, and software component reuse. He can be contacted at email: [anabot@zu.edu.jo](mailto:anabot@zu.edu.jo).