INTERNATIONAL JOURNAL OF INNOVATIVE STUDIES

A MONTHLY, INDEXED, REFEREED AND PEER REVIEWED OPEN

ACCESS INTERNATIONAL JOURNAL

RESEARCH ARTICLE

ISSN: 3027-2173

ARCHITECTURE OF AN ADVANCED SOFTWARE EVALUATION DSS USING MULTICRITERIA WEIGHTED SUM ALGORITHM

Benjamin, Ubokobong Effiong[1], Ogheneovo [2] & B. O. Eke [3]

[1, 2, 3] Department of computer science University of Port Harcourt Nigeria

D.O.I: 10.5281/zenodo.15465580

ARTICLE INFORMATION

Received: 11th February 2025

Accepted: 24th March 2025

Published: 24th April 2025

KEYWORDS: Software evaluation, decision support system, Weighted Sum Algorithm, multi-criteria decision analysis, decision-making

Publisher: Empirical Studies and Communication – (A Research Center)
Website: www.cescd.com.ng

ABSTRACT

The increasing complexity and variety of software solutions in modern industries necessitate effective decision-making tools to assist organizations in evaluating and selecting the most appropriate software for their needs. This paper explores the architecture of an advanced software evaluation decision support system (DSS) employing the Weighted Sum Algorithm (WSA) for multi-criteria decision analysis (MCDA). The system provides an efficient and transparent method for evaluating software based on multiple criteria, helping decision-makers prioritize factors such as performanceThis paper examines Architecture of an advanced software evaluation DSS using Weighted Sum, usability, cost, and technical support. The paper also discusses the integration of various decision-making factors, data collection methods, and the system's implementation, while highlighting its effectiveness in supporting complex decision-making processes in real-world scenarios.

INTRODUCTION

The selection of the right software solution is one of the most critical decisions that organizations face, particularly in highly competitive industries. As the number of software products increases, choosing the optimal solution based on multiple criteria becomes increasingly complex. Decision support systems (DSS) are often employed to aid this process by providing a structured, data-driven approach to evaluating software options (Jafari, 2020). One of the most widely used methods for evaluating software based on multiple criteria is the Weighted Sum Algorithm (WSA), a technique in multi-criteria decision analysis (MCDA).An advanced software evaluation Decision Support System

(DSS) using Weighted Sum Algorithm is profitable for determining best software alternative to go for. In the software market, there are numerous software for solving real life problems, however, the problem remains to choose the software that best solves the problem with least effort and high performance. There are few studies to determine software effectiveness and this does not encourage software developers and vendors to come up with highly productive software or to quickly identify flaws in their software for patches or update. Most software cost hundreds of thousands and even millions of naira, purchasing a software solution is a high expenditure activity that consumes a significant portion of companies' capital budget. Selecting the right solution is an exhausting process for companies. Therefore, selecting a software package that meets the requirements needs a full examination of many conflicting factors and it is a difficult task. Most times the software bought do not meet the needs of the institution or organization despite the huge amount. To avoid the problem of software ineffectiveness, this has led researchers to investigate better ways of evaluating and selecting software packages. Other problems that necessitated this research work are: not every software is easily usable due to usability flaws, security flaws, high cost and low benefits when considering cost-benefit analysis low benefits when considering cost-benefit analysis, without proper study it will be difficult to identify flaws in software application earlier and this can make an organization waste a lot of money which can make the firm to go bankrupt or close down abruptly and finally, Software users need to know the best application software to use to solve their problems and maximize profit where need arises. This is achieved using software evaluation decision support systems.

Decision Supports Systems (DSS) are computer-based information systems designed in such a way that helps managers to select one of the many alternative solutions to a problem. It is possible to automate some of the decision making processes in a large, computer-based DSS which is sophisticated and analyze huge amount of information fast. It helps organizations to increase market share, reduce costs, increase profitability and enhance quality. The nature of problem itself plays the main role in a process of decision making. A DSS is an interactive computer based information system with an organized collection of models, people, procedures, software, databases, telecommunication, and devices, which helps decision makers to solve unstructured or semi-structured business problems. Adopting decision support system to software evaluation guarantees accurate evaluation of the software (Del Vasto-Terrientes, Valls, Slowinski & Zielniewicz, 2015).Frequently employed approaches for evaluating and selecting components include the usage of simple scoring and weighted sum approaches, the Analytic Hierarchy Process (AHP), or iterative filtering.

Adopting Decision Support System (DSS) for Software Evaluation

A DSS is an interactive computer based information system with an organized collection of models, people, procedures, software, databases, telecommunication, and devices, which helps decision makers to solve unstructured or semi-structured business problems. Adopting decision support system to software evaluation guarantees accurate evaluation of the software (Del Vasto-Terrientes, Valls, Slowinski & Zielniewicz, 2015). Evaluation as a general endeavor can be characterized by the following features:Evaluation is a task, which results in one or more reported outcomes, evaluation is an aid for planning, and therefore the outcome is an evaluation of different possible actions and finally evaluation is goal oriented. The primary goal is to check results of actions or interventions, in order to improve the quality of the actions or to choose the best action alternative. Evaluation is dependent on the current knowledge of science and the methodological standards. Evaluation as an aid for software development has been applied since the last decade, when the comprehension of the role of evaluation within Human-Computer Interaction had changed. The activities "Task analysis", "Requirement specification", "Conceptual and formal design", "Prototyping", "Implementation" are each supplemented by an activity "Evaluation" which helps to decide progression to the next step. Software can be evaluated with respect to different aspects, for example, functionality, reliability, usability, efficiency, maintainability, portability (Benjamin, 2019). Adopting decision support system to software evaluation guarantees accurate evaluation of the software. Making decisions concerning software evaluation often strains our cognitive capabilities. Even though individual interactions among a system's variables may be well understood, predicting how the system will react to an external manipulation such as policy decision is often difficult. Many variables are involved in complex and often subtle interdependencies and predicting the total outcome may be daunting. There is a substantial amount of empirical evidence that human intuitive judgment and decision making can be far from optimal, and it deteriorates even further with complexity and stress. Because in many situations the quality of decisions is important, aiding the deficiencies of human judgment and decision making has been a major focus of science throughout history. Applying decision support systems to software evaluation therefore is a very important research area (Dias, Passeira, Malça & Freire, 2016).

Weighted Average Sum (WAS) Software Evaluation Technique

Another technique used for evaluation of software package is the weighted scoring method. In this method weights and rating scales are assigned to each criterion. The weight reflects the relative importance of each of the criteria while the rating scale indicates how easily each package is able to meet the specific criterion. The rating scales are then multiplied by weight factor of each criterion. Using this scheme a score is calculated for every criterion for each tool. These scores are then totaled to produce a score for each criteria category and the average is also computed. Finally, the categorical scores are compared to determine the highest (Benjamin, 2019).

The overall score for an alternative, is the weighted sum of the criteria utilities aggregated across the hierarchy of objectives. This approach combines objective evidence measured in specific scales, subjective assessment represented in explicitly defined scenario-specific utility functions, and relative weights across the goal hierarchy. As such, it is a flexible model, but it requires a profound understanding of the intricacies of decision making scenarios. Furthermore, a careful distinctionbetween the key concepts of evidence, utility, and weighting is expected from the decision maker (Rafique & Mišić, 2019).

Scoring Values:

The key to using weighted average sum is the raw score values. By using a defined and understood set of discrete values, the subjectivity of the evaluation is significantly reduced. In the prior examples, the raw values were based on the information shown in Table 2.2 There are only five values used, ranging from 1.0 to -1.0 in increments of 0.5. Note the use of negative values and the effects on the scoring. Instead of just assigning a value of 0, the use of negative values permits the application of a "penalty" value where not meeting the criterion would be detrimental. There are many different methods for deriving risk values, but descriptions of these methods are out of scope for this report. Additional references on risk can be found in the bibliography. Regardless of which risk calculation method you choose to follow, it is important to keep in mind that the scoring mechanism presented above is based on a "higher is better" score, and most risk calculations are based on a "lower is better" score. The two methods should be used individually and not combined into a single score for evaluation purposes. One consideration that must be addressed is how to handle scoring variances. Each potential evaluator has different experiences and perceptions that will ultimately affect the scoring. When using individual evaluators, the organization must have a scoring process that addresses (1) what constitutes a variance and (2) how to handle the differences in the scoring. Many organizations that use a similar process for evaluations will set a fixed value (e.g., less than 2 points on a 10-point scale) or a fixed percentage (e.g., 10% or more). When a scoring variance (or scoring split) occurs, the evaluators having a variance would then address the areas in the scoring that differed from the other evaluators. After the evaluators affected by the split have discussed their scoring and the rationale, each evaluator would take into consideration the new information and rescore the product (Miljković, Žižović, Petojević & Damljanović, 2018). For example, when performing an evaluation on a product, Evaluator A gives the product a total score of 78%, and Evaluator B gives the product a total score of 90%. Assuming the scoring process defines a split as 10% or more difference in scoring, both evaluators would discuss their individual scores for each range of criteria and their rationale for the individual scores; they would then rescore the product in the area(s) that differed until the scoring split was resolved.

Strengths of Weighted Average Sum Software Evaluation Technique:

i. Main advantage of WAS is its ease of use.

Weaknesses of Weighted Average Sum Software Evaluation Technique:

i. Weights to the attribute are assigned arbitrary and it is very difficult to assign weight when number of criteria is high.
ii. To obtain a score using this method a common numerical scaling is required.
iii. Difficulties emerge when WAS is applied to multi-dimensional MCDM problems

System Architecture of DSS for Software Evaluation

The architecture of the software evaluation DSS using WSA consists of several key components: data collection, criteria weighting, alternative scoring, aggregation and decision-making, and reporting.

1. Data Collection Module: The first step in the evaluation process involves the collection of data about various software alternatives. This typically includes both quantitative and qualitative data, such as performance metrics, costs, user ratings, technical support quality, and feature set. The data can be sourced from software documentation, expert reviews, end-user feedback, or direct trial runs of the software (Vargas & Alvarado, 2017). The data collection module is responsible for ensuring that the data is accurate, reliable, and up-to-date. It may integrate with existing databases or external systems that provide information about available software solutions. In some cases, manual entry by evaluators may be required for specialized or niche software options.

2. Criteria Weighting: In WSA, decision-makers must assign weights to each evaluation criterion based on its relative importance. The weighting of criteria reflects the priorities of the organization or the specific needs of the project (Chakraborty et al., 2013). For example, an organization may prioritize usability over cost if the software will be used by non-technical staff.

3. **Alternative Scoring**: Once the criteria weights have been established, the next step is to score each software alternative on each criterion. The scoring process evaluates how well each software option satisfies the defined criteria (Vargas & Alvarado, 2017). These scores may be numerical or qualitative and could be derived from user ratings, expert evaluations, or performance benchmarks. The scoring module aggregates data from various sources to generate scores for each alternative. In cases where the data is subjective, the module may employ normalization techniques to ensure consistency in scoring.

4. **Aggregation and Decision-Making**: The core of the system's architecture is the application of the WSA to aggregate the scores and determine the most suitable software. The WSA operates by multiplying the score for each alternative by the weight of the corresponding criterion and then summing these weighted scores to arrive at an overall score for each alternative (Jafari, 2020). The formula for the WSA is as follows:

$$S = \sum_{i=1}^{n} (w_i \times s_i)$$

Where:

- S is the overall score of an alternative,
- wi is the weight assigned to criterion iii,
- si is the score of the alternative on criterion iii,
- n is the number of criteria.

The alternatives are ranked according to their total scores, and the highest-ranking software is recommended. This transparent and objective approach helps minimize bias and ensures that decision-making is based on data rather than intuition.

5. **Reporting and Visualization**: The final component of the system is the reporting and visualization module. This component provides users with clear, comprehensible results that summarize the evaluation process. The system generates visualizations such as bar charts, radar charts, or decision matrices that display how each software alternative performed relative to the weighted criteria (Chakraborty et al., 2013). These reports help decision-makers understand the rationale behind the recommendations and facilitate discussions about trade-offs between different criteria.

Application and Benefits

The architecture described here is adaptable to various domains and industries. For example, in the healthcare sector, software evaluation might prioritize security and compliance with regulations, whereas in the education sector, ease of use and cost might be more important. By providing a systematic, data-driven approach to software selection, the DSS using the WSA helps organizations make informed decisions that align with their strategic goals and operational requirements.The use of WSA also provides several benefits over traditional methods of software evaluation, such as manual comparison or expert judgment. It offers consistency, transparency, and the ability to handle complex, multi-

dimensional data. Additionally, it supports collaboration among multiple stakeholders by allowing them to input their preferences and perspectives, which can lead to more robust decision-making outcomes (Vargas & Alvarado, 2017).

Software Architecture Evaluation

The software architect assembles component performance models from the repository for a planned architecture. This includes specifying the wiring of the components (assembly model), specifics of the component platform and hardware (deployment model), and the usage information, such as workload, input parameters, and internal state (runtime model). Service performance specifications potentially have to abstract from the actual performance behaviour of a component, to remain analyzable in an acceptable amount of time. The component developer creating a service performance specification must make a trade-off between the accuracy of the specification and its analysability (Kuwajima & Ishikawa, 2019). Babar et al., (2015), conducted a study on a framework for classifying and comparing software architecture evaluation methods. The researchers opined that different software engineering communities have developed different techniques for characterizing their respective quality attributes and the methods to evaluate software systems with respect to that particular quality attribute, e.g., real-time, reliability, and performance. These assessment techniques study a specific quality attribute in isolation. In reality, however, quality attributes interact with each other. For example, there is generally a conflict between configurability and performance; performance also impacts modifiability, availability affects safety, security conflicts with usability, and each quality attribute impacts cost. That is why it is important to find an appropriate balance of quality attributes in order to develop a successful product. One of the most significant features of method differentiation and classification is the number of quality attributes a method deals with.

The software architecture (SA) evaluation methods specifically studied by (Babar et al., 2015) are: Scenario-based Architecture Analysis Method (SAAM), Architecture Tradeoff Analysis Method (ATAM), Active Reviews for Intermediate Design (ARID), SAAM for Evolution and Reusability (SAAMER), Architecture-Level Modifiability Analysis (ALMA), Architecture-Level Prediction of Software Maintenance (ALPSM), Scenario-Based Architecture Reengineering (SBAR), SAAM for Complex Scenarios (SAAMCS), and integrating SAAM in domain-Centric and Reuse-based development (ISAAMCR). These are scenario-based methods, a category of evaluation methods considered quite mature. There are also some attribute model-based methods and quantitative models for SA evaluation, but, these methods are still being validated and are considered complementary techniques to scenario-based methods. There is, however, little consensus on the technical and non-technical issues that a method should fully address and which of the existing methods is most suitable for a particular issue. There is not much work done on systematic classification and comparison of the existing methods. Moreover, there is not much guidance on the desirable features of the evaluation methods and their usefulness.

Software Architecture (SA) evaluation can be performed for a number of purposes, e.g., risk assessment, maintenance cost prediction, architecture comparison, trade-off analysis and so forth. No one method can be considered as equally good for all types of assessment objectives as different methods are optimized to achieve different evaluation goals. The common goal of most of the evaluation methods is to evaluate the potential of the designed architecture to facilitate or inhibit the achievement of the required quality attributes. For example, some architectural styles, e.g., layered architectures, are less suitable for performance sensitive systems, even though they usually result in highly flexible and maintainable systems. In order to achieve maximum benefit from an assessment activity, the goals of the evaluation need to be explicitly defined. The goals of assessment help software architect make a number of critical decisions with regard to selection of a specific method and deliverable required. There is at least one common goal found in all surveyed methods, which is prediction-based assessment of the quality of a system at the architecture level. However, each method has a specific view and different approach to achieve the goal: SAAM and its variants (specifically SAAMCS and ISAAMCR) are mainly geared to identify the potential architectural risks. However, SAAMCS focuses on exposing the boundaries of SA with respect to flexibility using complex scenarios, while, ISAAMCR integrates SAAM in domain-centric reusable development process; SAAMER evaluates the designed SA for evolution and reusability and provide a framework for SA analysis; ALMA specializes in predicting one quality attribute (i.e., modifiability) and there are three possible objectives to be pursued: risk assessment, maintenance cost prediction, and SA comparison; ARID performs suitability analysis of intermediate design artifacts; ATAM identifies and analyses sensitivity and trade-off points as these can prevent the achievement of a desired quality attribute.

Architecture of an Advanced Software Evaluation Decision Support System using Weighted Sum Algorithm

Benjamin (2019) implemented an evaluation system that provide a grading or evaluation interface that will enable users carry out software evaluation of different software that are of the same application category. The advanced software are evaluated side by side (A and B). The system will also provide expert system remark after assessment. The evaluation wasdone based on five different criteria which are:

1. Vendor evaluation
2. Hardware/software evaluation
3. Cost/benefits evaluation.
4. Security evaluation

5. Reliability evaluation
6. Integrity Test

Each criterion for software evaluation outline above has sub-criteria that will be assigned evaluation rating values ranging from 2 to -2. At the end, the total sum is computed and the percentage sum is also computed. Expert system remark is also provided indicating which software is better.

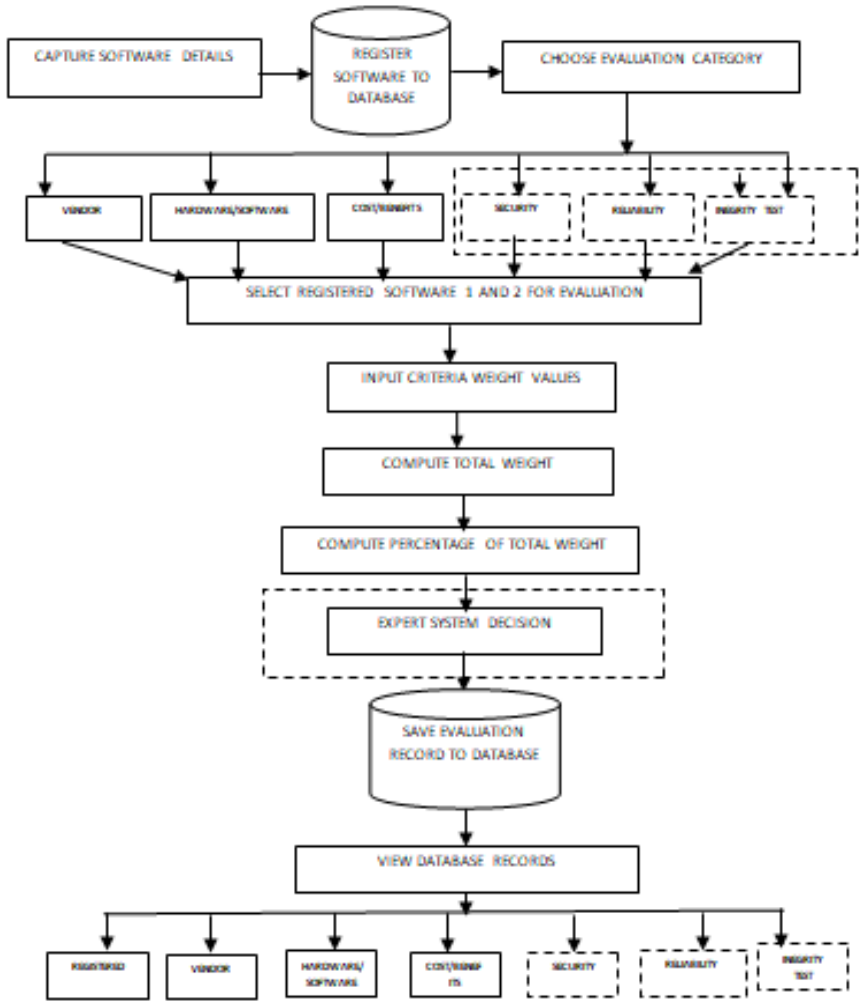

Figure 1: Architecture of proposed Decision Support System for software evaluation

The Use Case Model describes the proposed functionality of the new system. A Use Case represents a discrete unit of interaction between a user (human or machine) and the system. A Use Case is a single unit of meaningful work; for example, login to system, register with system, etc. Each Use Case has a description which describes the functionality that will be built in the proposed system. A Use Case may 'include' another Use Case's functionality or 'extend' another Use Case with its own behaviour.Use Cases are typically related to 'actors'. An actor is a human or machine entity that interacts with the system to perform meaningful work. The use case model of an advanced decision support system using weighted sum technique is presented below.

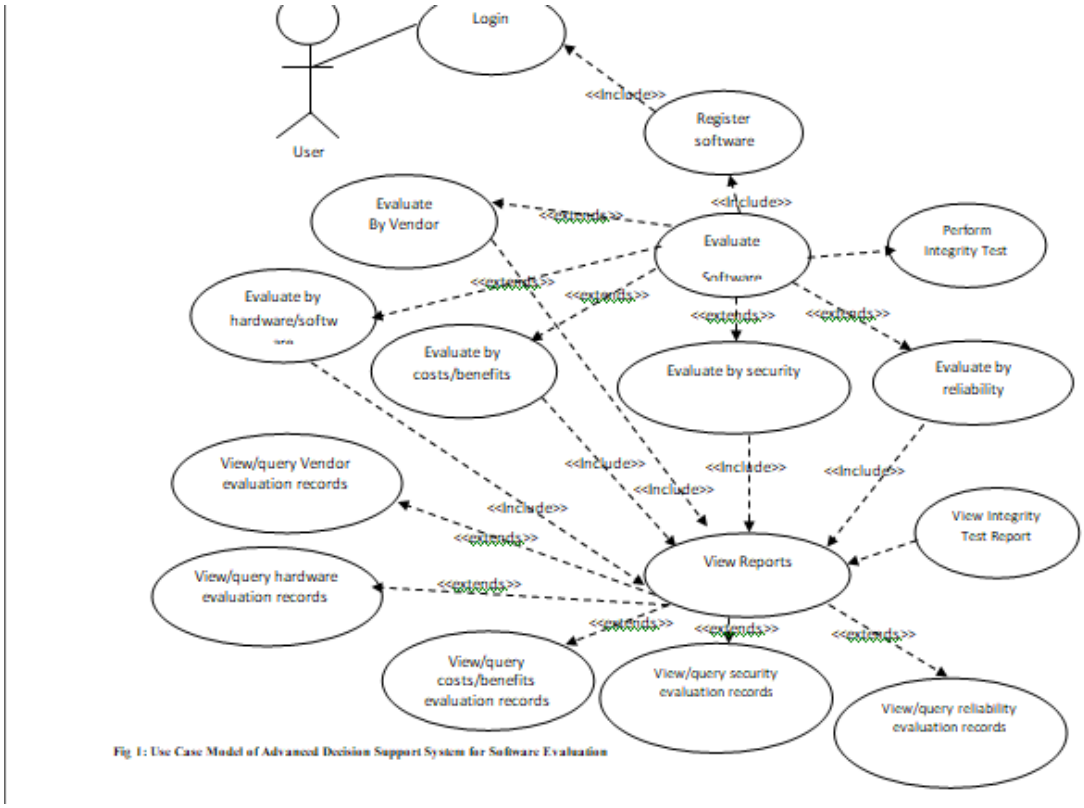

Fig 1: Use Case Model of Advanced Decision Support System for Software Evaluation

Fig 2: Use Case Model of Advanced Decision Support System for Software Evaluation

Figure 2 illustrates the Use Case Diagram of the Advanced Decision Support System (DSS) designed for software evaluation using the Weighted Sum Algorithm. It outlines the dynamic interaction between the primary user (referred to as the evaluator) and the system's various functional components. The evaluator can carry out essential actions such as accessing the system via login, registering software applications, and performing detailed evaluations based on multiple criteria. These criteria include vendor performance, hardware/software compatibility, cost-effectiveness, security features, reliability, and integrity assessment. The core use case—"Evaluate Software"—is systematically connected to each specialized evaluation aspect through inclusion and extension relationships, ensuring modular and comprehensive functionality. The user is also able to retrieve and review detailed evaluation reports and historical data related to each criterion. Furthermore, integrity checks and corresponding reports are integrated to support evidence-based analysis. This use case model provides a structured and user-centric representation of how the system facilitates multi-dimensional, data-driven software selection processes.

Conclusion

The architecture of an advanced software evaluation decision support system using the Weighted Sum Algorithm provides a structured and transparent approach to evaluating software alternatives. By integrating data collection, criteria weighting, alternative scoring, aggregation, and reporting, the system supports complex decision-making processes in real-world scenarios. The use of WSA enhances the objectivity and consistency of the evaluation process,

making it a powerful tool for organizations seeking to select the best software solution based on multiple criteria.Software evaluation was revealed to be the process of assessing the quality of different software systems so as to choose the most productive one. Evaluating and selecting software packages that meet an organization's requirements is a difficult software engineering process. Selection of a wrong software package can turn out to be costly and adversely affect business processes, making it very imperative to have an evaluation system. Software evaluation is important as it enables Software companies and individuals/organizations that make use of software to make sure they provide software solutions that are reliable and can meet the needs of users. This cannot be achieved without an effective decision support system for software evaluation. The need to evaluate the reliability of software system has become clearer in this era of intensive software application in different sectors of life. Many papers provide a preferred list of evaluation criteria for evaluation of specific software package; however, a lack of common list is apparent. Software evaluation criteria may differ from one evaluator to another. The exact meaning of a criterion is open to the evaluator's own interpretation. Sometimes the terminology used by author(s) for a criterion in one literature is different than another author(s) for the same criterion. This may lead to ambiguity and gives unclear picture to the evaluator. A common and effective evaluation technique than can be adopted by software evaluators to determine the effectiveness of the software is weighted sum evaluation technique. In this approach values or weights are assigned to the criteria used in determining the effectiveness of the software and the total weight is used to determine which is more effective than the other.

An advanced decision support system for software evaluation enables the determination of the best software alternative. Weighted sum evaluation technique can be adopted to easily ascertain the effectiveness of software based on defined criteria. From the foregoing, it can be seen that, requirements drive selection criteria, careful consideration must be given to the identification of selection criteria, pilots and demonstrations are essential selection tools, product and technology maturity must be considered. By systematically analyzing co-occurrence, correlation and impact of decision criteria across cases, it should be possible to integrate recommender systems into the decision making workflow that can provide increased guidance and warn decision makers of potential risks and opportunities based on others' experiences.

Recommendations

The following recommendations are offered based on the study:

1. More research should be encouraged on decision support system for software evaluation/assessment.
2. Software development companies should conduct survey programs to assess the reliability of their software product.
3. Raw data used for the assessment should be obtained from users of the software system.
4. The criteria utilized in making decision on the effectiveness of a software should be expanded to include more aspects so as to improve the reliability information.
5. Proper testing and debugging should be done before software systems are published or placed in the market.
6. More criteria categories should be added to boost software evaluation and selection.
7. Trial versions of software are important for users to assess the effectiveness of the software.

REFERENCES

Babar, M. A., Liming Z, Jeffery, R. (2015). A Framework for Classifying and Comparing Software Architecture Evaluation Methods. National ICT Australia Ltd. and University of New South Wales, Australia.

Benjamin, E. U. (2019).*An improved decision support system for software evaluation using weighted sum technique*. International Journal of Trend in Scientific Research and Development, 3(2), 1416-1419. IJTSRD

Chakraborty, S., Biswas, A., & Bandyopadhyay, S. (2013). *Multi-criteria decision-making methods for software selection: A review*. International Journal of Software Engineering and Its Applications, 7(1), 51-60.

Del Vasto-Terrientes, L. , Valls, A. , Slowinski, R. , & Zielniewicz, P. (2015). ELECTRE-I- II-H: An outranking-based decision aiding method for hierarchically structured criteria. Expert Systems with Applications, 42 , 4 910–4 926

Dias, L. C., Passeira, C., Malça, J., & Freire, F. (2016). Integrating life-cycle assessment and multi-criteria decision analysis to compare alternative biodiesel chains. *An- nals of Operations Research* .https://doi.org/10.1007/s10479-10016-12329-10477 .

Jafari, M. (2020). *Decision support systems in software evaluation: A case study approach*. International Journal of Computer Applications, 180(2), 29-34.

Kuwajima, H., & Ishikawa, F. (2019). Adapting SQuaRE for quality assessment of artificial intelligence systems. In *2019 IEEE International Symposium on Software Reliability Engineering Workshops* (pp. 13–18). IEEE. ojs.imeti.org

Miljković, B., Žižović, M. R., Petojević, A., & Damljanović, N. (2018).*New weighted sum model*. Filomat, 32(2), 379-386. ResearchGate

Rafique, S., & Mišić, V. B. (2019). "A framework for evaluating cloud-based software services using decision support systems." *Information and Software Technology, 113*, 115-127.

Vargas, L. G., & Alvarado, S. (2017). *A decision support system for software evaluation based on a multi-criteria approach*. Software Engineering Research, 8(4), 113-122.