sustainability

MDPI

Article

A Fuzzy-TOPSIS-Based Approach to Assessing Sustainability in Software Engineering: An Industry 5.0 Perspective

Samar Hussni Anbarkhan

Information Systems Department, Northern Border University, Rafha 76322, Saudi Arabia; samar.hussni@nbu.edu.sa

Abstract: New possibilities and challenges have evolved in the setting of the software engineering sector's rapid transition to Industry 5.0, wherein sustainability takes centre stage. Appropriate evaluation approaches are required for analysing the long-term viability of software engineering practices within this paradigm. This study proposes an innovative approach to evaluating sustainability in software engineering within Industry 5.0 by utilising the fuzzy technique for order of preference by similarity to ideal solution (fuzzy TOPSIS) methodology. The fuzzy TOPSIS approach is effective at accounting for the inherent uncertainties as well as imprecisions related to sustainability assessments, allowing for informed decision-making. This approach helps in the recognition of the most sustainable software engineering practices in Industry 5.0 by taking into account a defined set of sustainability parameters. We rigorously analyse the current literature and expert views to provide an extensive set of relevant sustainability standards for the area of software engineering. Following that, we develop an evaluation methodology based on fuzzy TOPSIS that can handle the subjectivity as well as fuzziness inherent in sustainability evaluations. A case study with a software development company functioning in Industry 5.0 demonstrates the utility and efficacy of our suggested framework. The case study outcomes reveal the benefits and drawbacks of various software engineering methodologies in terms of sustainability. The study's findings provide substantial information for decision-makers in the software engineering field, assisting them in making educated decisions about sustainable. Finally, this study helps to establish environmentally and socially appropriate techniques within the context of Industry 5.0.

Keywords: sustainability; software engineering; Industry 5.0; fuzzy TOPSIS; sustainable software development; decision-making

Citation: Anbarkhan, S.H. A Fuzzy-TOPSIS-Based Approach to Assessing Sustainability in Software Engineering: An Industry 5.0 Perspective. *Sustainability* **2023**, *15*, 13844. https://doi.org/10.3390/su151813844

Academic Editors: Jurgita Antuchevičienė, Brian Vejrum Wæhrens, Rodrigo Salvador, Peder Veng Søberg and Samuel Brüning Larsen

Received: 8 July 2023
Revised: 6 September 2023
Accepted: 12 September 2023
Published: 18 September 2023

1. Introduction

With the introduction of Industry 5.0, the software engineering sector is continuing to evolve after years of notable change. The fusion of cyber-physical systems, artificial intelligence and big data analytics is the hallmark of this most recent industrial revolution, presenting software engineering practices with both unheard-of potential and difficulties. Sustainability has become a critical component that necessitates consideration and review in this dynamic environment. It is crucial to evaluate the sustainability of software engineering practices in Industry 5.0 as organisations work to align their operations with sustainable development goals. In the context of software engineering, sustainability refers to a range of factors, including social, economic and environmental considerations. It comprises lowering carbon emissions, maximising resource use, fostering long-term economic viability and promoting ethical principles. However, because it comprises a number of interrelated criteria and subjective assessments, evaluating sustainability is a difficult undertaking. More advanced methodologies are required since traditional evaluation approaches frequently fall short of capturing the inherent uncertainties as well as imprecisions involved in sustainability assessments [1–5].

Despite certain attempts that have been made, there are still widespread erroneous assumptions and misconceptions about this developing discipline, despite the growing importance of software sustainability. As the software industry and software engineers must be aware of and can be influenced by all aspects of software sustainability, our particular objective is to highlight the significance of software sustainability within the field of software engineering. Descriptions of green software in the literature often lack conceptual consistency, resulting in the use of words like "green software", "green through software", "green in software", and other variations. The difference between Green BY (where IT is used as a tool to support sustainability goals) and Green IN (when the term "green" applies to the IT aspects themselves, including software and hardware) must be determined. Generally speaking, these two viewpoints are frequently combined in definitions of green software [6–8]. Figure 1 incorporates all the definitions, from software sustainability to green in software, to address this.

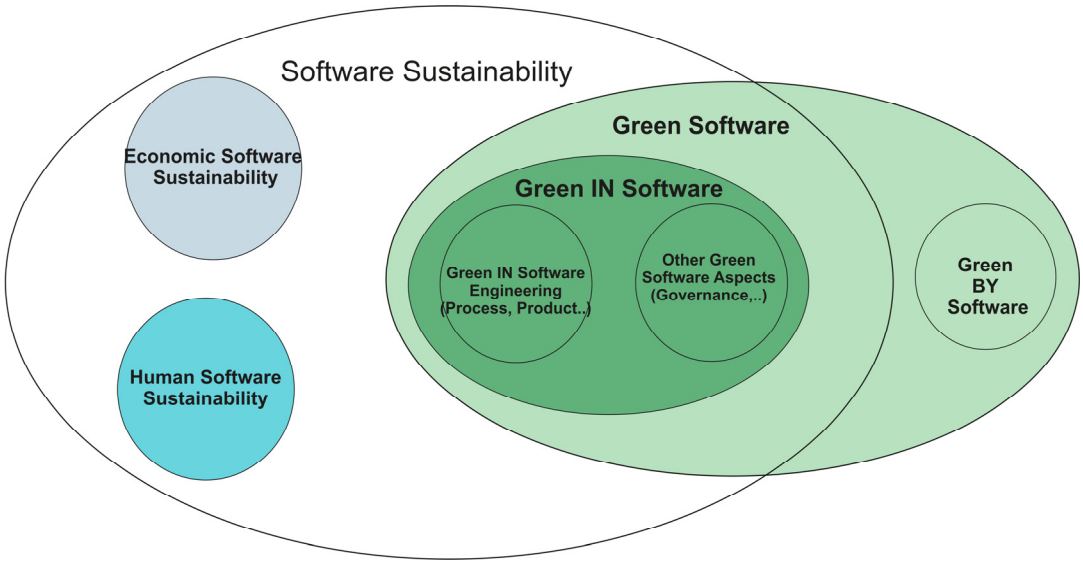

Figure 1. The evolution from software sustainability to green in software.

The rapid shift of the sector to Industry 5.0 has shown both bright potential and significant obstacles in the direction of sustainable software engineering. However, a noteworthy research gap remains within this dynamic transition regarding a systematic and thorough evaluation technique to evaluate sustainability in software engineering practices within the setting of Industry 5.0. While previous research has looked into various areas of sustainable software development, there is still a clear lack of a comprehensive methodology that effectively addresses the complex relationship of sustainability variables in Industry 5.0. This work tries to fill that need by introducing an innovative evaluation approach focused on the fuzzy technique for order of preference by similarity to ideal solution (fuzzy TOPSIS) methodology. This method is customised to the intricacies of Industry 5.0, allowing for an extensive evaluation of sustainability in the area of software engineering. Numerous industries now use the fuzzy TOPSIS procedure as a tool for making decisions. Given the inherent ambiguity and subjectivity in decision-making processes, fuzzy TOPSIS offers a solid framework for evaluating and ranking alternatives based on a variety of factors. Fuzzy TOPSIS enables decision-makers to successfully handle uncertain and imprecise data by utilising fuzzy logic and pairwise comparisons. Fuzzy TOPSIS is a potential method for assessing sustainability in the software engineering sector. Decision-makers can use it to take into account a variety of sustainability criteria, account for their relative importance and generate useful rankings. Fuzzy TOPSIS incorporates the inherent subjectivity and uncertainty that are common in sustainability assessments by using fuzzy logic [9–15].

Our work contributes to the larger Industry 5.0 conversation by offering a solution to this important problem and by promoting well-informed decision-making in the field of software engineering. This research paper's goal is to suggest a fresh method for assessing sustainability in the field of software engineering, especially in the setting of Industry 5.0, by employing the fuzzy TOPSIS methodology. Our goal is to give decision-makers a thorough and useful framework for evaluating the sustainability aspects of various software engineering practices and making wise decisions. To accomplish this goal, we first perform a thorough assessment of the literature to find and develop a set of pertinent sustainability standards for the software engineering sector in the context of the Industry 5.0 paradigm. These standards cover ethical issues, resource efficiency, social responsibility and environmental effects. Then, using fuzzy logic to account for uncertainty and subjectivity, we create an evaluation framework founded on fuzzy TOPSIS that combines these criteria.

A case study with a software development company functioning in Industry 5.0 is used to apply the suggested methodology. Through the case study, we show the fuzzy TOPSIS methodology's usefulness and potency in assessing sustainability in the software engineering sector. We offer the evaluation's findings, showing the advantages and disadvantages of various software engineering techniques in terms of sustainability. The research's outcomes add to the body of understanding regarding how to evaluate sustainability in the software engineering sector, and they provide useful information and direction for decision-makers trying to implement sustainable practices in the context of Industry 5.0. This study suggests a fuzzy-TOPSIS-based evaluation framework to answer the requirement for sustainability assessment in the software engineering sector. Our method gives decision-makers a useful and efficient tool to evaluate and contrast various software engineering practices in terms of sustainability in the Industry 5.0 paradigm by utilising fuzzy logic and including a thorough set of sustainability criteria.

The rest of the paper is organised as follows: Section 2 delves into related works, providing an overview of the existing research in the field. Section 3, titled "Materials and Methods," discusses the hierarchical structure for evaluation and the fuzzy TOPSIS method, explaining the methodology employed in the study. In Section 4, "Statistical Findings and Comparative Analysis" are presented, highlighting the results obtained from the evaluation process. Section 5 offers a comprehensive discussion of the findings, providing insights and interpretations. Finally, in Section 6, the paper concludes with a summary of the main findings and their implications, presented in the "Conclusions" section.

2. Related Works

A number of research studies have examined sustainability in the software engineering sector with the goal of directing businesses towards actions that are more socially and ecologically responsible. These studies investigated numerous frameworks and approaches to evaluate sustainability in diverse contexts, offering insightful information and adding to the expanding body of knowledge in this area. In this section, we evaluate pertinent studies on the evaluation of software engineering sustainability with a focus on Industry 5.0. We look at the methods, concepts and approaches now in use in these studies, noting their benefits and drawbacks as well as how well they apply to the field of software engineering. We may find research gaps while expanding on past studies' foundations by comprehending the existing body of work, eventually facilitating the growth of sustainability evaluation in the software engineering sector within the setting of Industry 5.0.

Penzenstadler et al.'s [16] thorough analysis of sustainability in software engineering research was published. They looked at many different aspects, including research activity, themes, limits, proposed solutions, techniques, existing research and domains. By using a systematic literature review (SLR) technique, the authors adhered to the commonly used guidelines set forth by Kitchenham et al. [17]. They examined the top 100 outcomes from five credible, widely used databases, arranged according to how relevant they were to the search query.

In a research study, Zada et al. [18] designed and created OntoSuSD (ontology for sustainable software development), an integrated ontology that incorporates software engineering methodologies like agile, lean and green. OntoSuSD's main goal is to encourage people to learn about, be aware of and use sustainable software development techniques. This calls for the establishment of a formal, extensive and collaborative knowledge base that uses semantic terminology and specifies the ideas and connections necessary for the representation and implementation of lean, agile and green techniques in software development processes. OntoSuSD enables sustainable software development by facilitating the simultaneous implementation and evaluation of these methods. OntoSuSD was built using practical ontology engineering techniques, combining pertinent ontologies and defining precise concepts and characteristics to satisfy the domain's knowledge requirements and representational needs. Evaluation of OntoSuSD showed that the ontology had achieved its development goals and had a strong ontological design, broad coverage of the domain and potential for a variety of applications.

Manteuffel and Ioakeimidis [19] carried out methodical mapping research in their work to provide an overview of sustainability in software engineering. The study's objectives were to categorise the published literature and offer insights into the current research environment. The purpose of the study, the research questions, the description of the search strategy, the selection criteria and data extraction are only a few of the topics covered in the paper. Furthermore, the search string evaluation's preliminary findings are shown. In order to incorporate sustainability issues into any software development methodology, Dick and Naumann [20] developed a general upgrade for software development processes. With this method, many software development practices have been able to incorporate sustainability considerations. The words "Green and Sustainable Software" as well as "Green and Sustainable Software Engineering" are defined by Naumann et al. [21]. The GREENSOFT Model, a conceptual reference model that includes a cradle-to-grave product life cycle for software, sustainability metrics and criteria, software engineering extensions for sustainable software design and development and suitable guidance, is also described in further detail. In an associated investigation, Seacord et al. [22] contextualise the assessment of sustainability and cover a set of metrics created at Carnegie Mellon University's Software Engineering Institute. These metrics offer a framework for evaluating the sustainability of software engineering practices.

In a different investigation, Venters et al. [23] proposed a definition of software sustainability and investigated empirical techniques for measuring it during the design and engineering of software applications. Their research focuses on the conception of software sustainability and its actual evaluation. The Generic Sustainable Software Star Model (GS3M), which provides a comprehensive view of sustainable software, was introduced by Amri and Saoud [24]. This model includes characteristics of sustainability from economic, social, technical and environmental perspectives. Every aspect has a unique set of software sustainability values attached to it and these values are in turn supported by matching software attributes. Subqualities of these attributes are also possible and they can each be associated with specific metrics. Mourão et al.'s systematic mapping study [25] was used to compile and analyse the most recent methods for sustainable software engineering practices. They focused on approaches, processes, tools and metrics offered to promote sustainable software development and examined 75 primary research pieces that were published before 2017. Different classification criteria, such as contribution forms, SDLC phases, evidence types, research types, application domains, publication venues, distribution between academia and industry and research techniques served as the basis for the assessment. The research results showed that the field of software engineering research is becoming more and more interested in green and sustainable software. The outcomes also demonstrated the requirement for additional research on methodologies, instruments and metrics that specifically address the phases of construction, testing and maintenance. They also emphasised how crucial it is in the field of sustainable software engineering to coordinate research efforts with actual implementation.

Numerous other authors [25–31] have also added to the conversation about sustainable software production. Their research examines several facets of sustainability in software engineering, from social responsibility and ethical considerations to energy efficiency and lowering environmental effects. These authors have clarified the significance of incorporating sustainability principles into software development processes through their research and conversations and they have also put forth strategies and guidelines for doing so. Their efforts and insights have greatly improved our understanding of sustainable software development and given scholars and business professionals looking to improve the sustainability of software engineering useful guidance.

There is a need for more specialised methodologies that explicitly meet the difficulties and demands of Industry 5.0, even though earlier research has significantly aided in the evaluation of sustainability in the software engineering business. Industry 5.0's incorporation of cyber-physical systems, AI and big data analytics calls for a re-evaluation of sustainability standards and the use of more advanced examination techniques. The fuzzy TOPSIS methodology offers a potential solution because it uses fuzzy logic and enables subjectivity and uncertainty in decision-making. Due to its exceptional applicability in tackling the complexities of sustainability evaluation within the software engineering mechanism of Industry 5.0, fuzzy TOPSIS was selected as the recommended methodology in this research. Fuzzy TOPSIS, in contrast to other approaches like fuzzy AHP, thrives in its capacity to concurrently take into account both the positive and negative elements of alternatives, which is crucial in sustainability evaluations where trade-offs and thorough assessments are crucial [30–32]. It is the best option for the study's objectives in this dynamic and varied field due to its ability to manage the inherent subjectivity and uncertainties in sustainability assessments as well as its sturdiness in handling complicated relationships among criteria and alternatives. Because of its clear advantages in managing the difficulties involved in sustainability evaluations within the software engineering sector of Industry 5.0, fuzzy TOPSIS was chosen as the analytical approach in our research. Fuzzy TOPSIS enables us to express and quantify these elements in a way that standard crisp approaches like fuzzy AHP might not be able to capture adequately given the intricate interplay of many sustainability dimensions. Additionally, fuzzy TOPSIS provides a more complete evaluation because it inherently takes into account both the advantages and disadvantages of alternatives. This quality is crucial in sustainability evaluations, where a software engineering practice's possible drawbacks are just as important as its advantages. Fuzzy TOPSIS excels at managing the complicated interactions between various criteria and alternatives, making it the perfect method for our research in the setting of Industry 5.0, where software engineering practices are becoming more interconnected and sophisticated. In our work, this decision is highlighted to show the transparency and justification for the technique choices. The aim of this research article is to create a thorough and efficient framework for assessing sustainability in the software engineering business, particularly within the setting of Industry 5.0, by developing upon the current body of information and utilising the advantages of fuzzy TOPSIS.

3. Materials and Methods

This section outlines the methodology used to assess sustainability in the software engineering sector. The hierarchical structure for evaluation and the fuzzy TOPSIS procedure are the two subsections that make up this section. In the beginning, the hierarchical structure for assessment offers a structure for classifying and organising the assessment criteria necessary to judge the sustainability of software engineering practices. This hierarchical structure allows for a thorough examination by taking into account many sustainability-related dimensions. Second, using the fuzzy TOPSIS approach as a decision-making tool, it is determined where various software engineering practices stand in relation to one another in terms of performance with regard to sustainability. The fuzzy TOPSIS method handles ambiguity and uncertainty in the evaluation process by combining fuzzy set theory and multicriteria decision-making procedures.

3.1. Hierarchical Structure for the Evaluation

3.1.1. Criteria Identification

An essential first step in assessing sustainability in the context of Industry 5.0 for the software engineering sector is the establishment of acceptable criteria. These standards form the basis for evaluating the economic, social, environmental and resource aspects of software engineering practices. We provide an overview of the criteria identification approach in this part, building on a thorough literature analysis and industry best practices [33–36]. We want to build a comprehensive set of criteria that captures the essential elements of sustainability in the context of software engineering by combining knowledge from existing studies and industry standards. These standards will make it possible to evaluate and compare software engineering practices in depth, allowing for more informed decision-making and encouraging the implementation of sustainable practices in the sector. Table 1 lists several evaluation criteria that were determined after consultation with domain specialists and a literature review.

Table 1. Different identified criteria for the evaluation.

Criteria	Description
Environmental Impact (SC1)	This criterion evaluates how software engineering practices are environmentally sustainable. It takes into account variables including energy use, carbon emissions, trash production and the utilisation of sustainable resources. A positive environmental impact is the result of reduced energy use and carbon emissions, effective waste management and the use of renewable energy sources. The evaluation of this criterion aids in the discovery of procedures that minimise environmental damage and advance sustainable resource management.
Social Responsibility (SC2)	The ethical and societal ramifications of software engineering practices are the main emphasis of this criterion. It takes into account elements like inclusivity and diversity, privacy and security of information, fair labour practices and community engagement. Strong social responsibility is demonstrated by practices that place a high priority on diversity, equity and inclusion in the workforce, guarantee user privacy and data security, offer fair working conditions and actively involve local communities. Assessing this criterion aids in determining the ethical implications and societal consequences of software engineering practices.
Resource Efficiency (SC3)	This criterion assesses how well resources are used during the software engineering processes. It takes into account things like optimised code and algorithms, optimal resource allocation and effective utilisation of computational resources. Higher resource efficiency is demonstrated by techniques that reduce resource consumption, encourage the recycling and reuse of software components and optimise resource allocation. Identifying practices that maximise resource utilisation and minimise waste is made easier by evaluating this criterion.
Economic Viability (SC4)	This criterion looks at how software engineering practices can remain profitable. Cost-effectiveness, return on investment and long-term financial viability are among the things it takes into account. Economically feasible practices are those that show cost-effectiveness, produce dependable revenue sources and offer long-term financial advantages. Assessing this criterion aids in determining the costs and long-term viability of software engineering practices.

3.1.2. Alternatives Identification

It is crucial to establish the pertinent alternatives for comparability in order to assess the sustainability of software engineering practices within the framework of Industry 5.0 [37–40]. These alternatives show various methods or approaches that are frequently used in software development. We may learn more about these alternatives' individual advantages and disadvantages in terms of sustainability performance by analysing and contrasting them. In this section, we provide an overview of the procedure for identifying alternatives while taking into consideration current market trends, technology breakthroughs and Industry-5.0-specific traits. We aim to provide a thorough evaluation framework that helps software engineering organisations make educated judgements on sustainable practices in the constantly changing environment of Industry 5.0 through the rigorous evaluation and analysis of these alternatives. The several selected alternatives for the assessment in this study are displayed in Table 2.

Table 2. Different identified alternatives for the evaluation.

Alternatives	Description
Agile Methodologies (Alternative 1)	Agile methodologies like Scrum, Kanban or Extreme Programming (XP) are included in this alternative. Agile techniques encourage flexibility as well as customer involvement across the development procedure by focusing on iterative development, cooperation and responsiveness to changing requirements.
Cloud-Native Development (Alternative 2)	The adoption of cloud-native architectures and technologies is represented by this alternative. Utilising cloud platforms as well as services, cloud-native development helps create and deploy applications that are scalable, flexible and resource-efficient.
DevOps (Alternative 3)	The DevOps methodology, which integrates software development and operations, is represented by this alternative. It aims to improve cooperation, efficiency and automation throughout the software development lifecycle. Continuous integration, regular delivery and strong cooperation between the development and operations teams are all stressed by DevOps.
Traditional Waterfall Model (Alternative 4)	In this alternative, each phase (requirements, design, programming, testing and deployment) is finished before proceeding to the next, following the conventional sequential method of software development. It is distinguished by linear development and constrained flexibility.
Sustainable Software Engineering Practices (Alternative 5)	This option consists of a collection of methods chosen for their emphasis on sustainability. It might cover a method for coding that uses less energy, optimising resource use, using green technologies and using sustainable development practices.

These options reflect several software engineering strategies or methods that can be contrasted according to how well they support sustainability. This research intends to offer insights into the environmentally friendly aspects of software engineering practices within the framework of Industry 5.0 by analysing these alternatives using the established criteria. In this study, the choice of sustainable characteristics was made after a thorough process that included consultation with domain specialists in the area of sustainable practices and a thorough examination of the body of current literature. The selection of the parameters was made with the study's objective in mind. The ultimate choice attempted to take into account a broad spectrum of standards that jointly reflect the complexity of sustainability. Eighty-five software development professionals participated in the decision-making process for assessing sustainability in the software engineering sector within the context of Industry 5.0. These specialists were chosen due to their depth of knowledge and proficiency in the fields of sustainability and software engineering. Their varied backgrounds and in-

depth knowledge of the sector enabled them to conduct a thorough and well-rounded evaluation of the alternatives. Such participants' experience was crucial to the decision-making process since their knowledge and viewpoints influenced the formulation of the fuzzy TOPSIS methodology's evaluation criteria, weightings and general framework. Their active participation and contributions helped to ensure that the evaluation process and its results were based on actual knowledge of the industry, which improved the validity and application of the research findings. The combined knowledge and experience of these 85 seasoned specialists provided a solid basis for making wise choices and coming to insightful conclusions about sustainability in the software engineering sector 5.0. The hierarchical arrangement of the diagram utilised in the assessment in the present research study is shown in Figure 2 below.

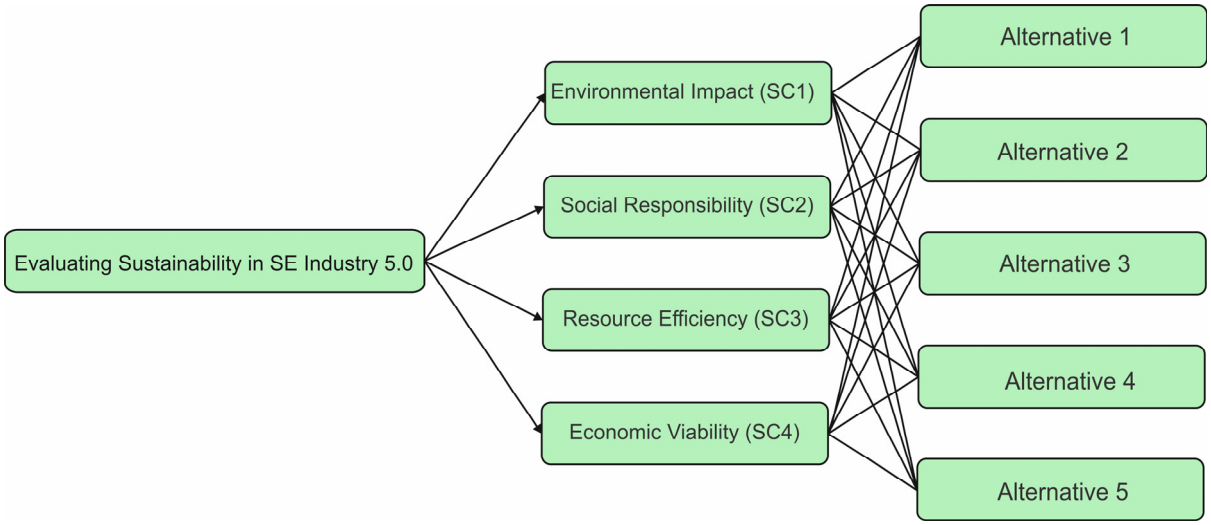

Figure 2. Hierarchical structure for the evaluation.

3.2. Fuzzy TOPSIS Method

To address uncertainty and linguistic evaluations in decision-making processes, the fuzzy technique for order preference by similarity to ideal solution (TOPSIS) is a potent multicriteria decision-making approach that combines fuzzy set theory. When comparing options based on a variety of factors, fuzzy TOPSIS is very helpful because it enables a thorough evaluation of each option's performance. The procedure entails defining the criteria and alternatives, distributing fuzzy membership functions to represent linguistic evaluations, creating a fuzzy decision matrix, normalising the matrix to remove scale differences, identifying the fuzzy positive ideal as well as fuzzy negative ideal solutions, computing fuzzy closeness coefficients to gauge how similar the alternatives are to the ideal solutions and ultimately ranking the alternatives on the basis of these coefficients [38–42].

Fuzzy TOPSIS provides an organised strategy for decision analysis by including fuzzy logic within the decision-making procedure, enabling a more flexible and nuanced evaluation that may accommodate inaccurate and subjective judgements. By taking into account a wide range of criteria and their uncertainties, it provides a reliable framework for evaluating sustainability in software engineering practices within the context of Industry 5.0. In order to assess sustainability in software engineering practices within the context of Industry 5.0, the fuzzy TOPSIS approach is applied along with a thorough description and mathematical derivation of each step. Figure 3 shows the flow diagram of the fuzzy TOPSIS approach used in this research study.

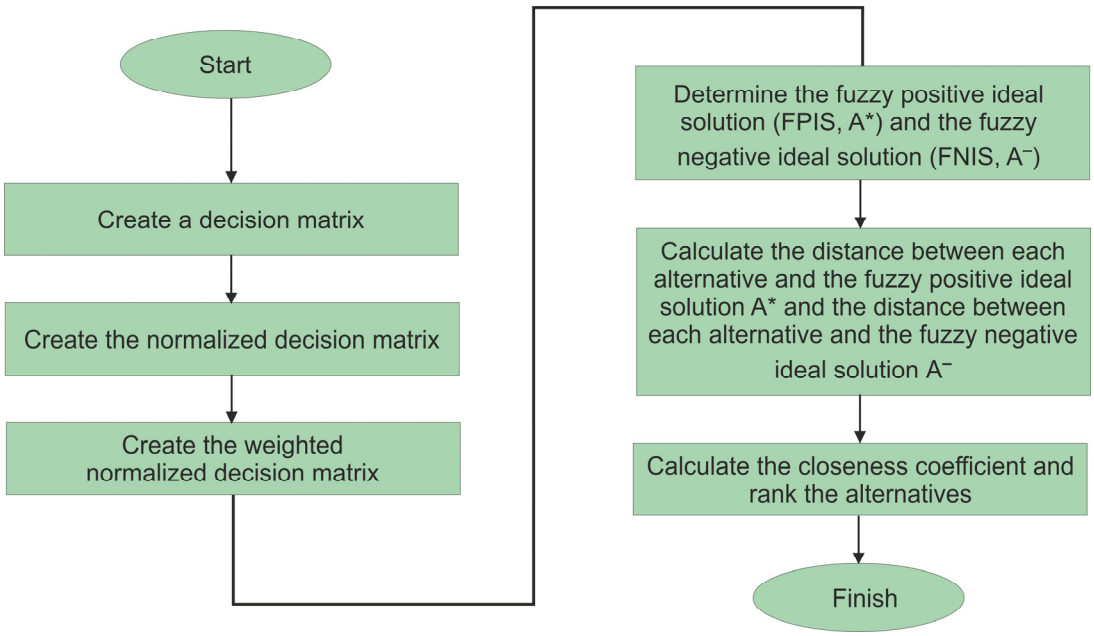

Figure 3. Flow diagram of fuzzy TOPSIS approach.

4. Results

The method of assessment based on the use of the fuzzy TOPSIS technique for evaluating sustainability in the software engineering sector within the context of Industry 5.0 is presented in the Section 4. In order to shed light on the relative performance of the alternatives in the context of sustainability criteria, the aim of this section is to provide a thorough study of the alternatives and their rankings. For a better understanding of the advantages and disadvantages of each alternative and to guide decision-making processes in relation to sustainable software engineering practices, the findings gathered will be provided, analysed and understood. This section emphasises key findings, trends and consequences by analysing the rankings and assessing the effectiveness of the alternatives with the aim of improving sustainability and aiding stakeholders in selecting software development methodologies and practices for Industry 5.0. The entirety of the data generated through the fuzzy-TOPSIS-based MCDM evaluation, in collaboration with software industry experts, has been meticulously documented within the paper. This encompasses a comprehensive presentation of the evaluation outcomes, showcasing the performance of different software engineering techniques in terms of sustainability. While this research does not include specific plots, the collected data are thoroughly analysed and discussed, offering a comprehensive understanding of the implications of the proposed methodology. These findings enable decision-makers to make well-informed choices for sustainable practices, which is a crucial facet of our research's contribution to the field.

4.1. Statistical Finding

The statistical results based on the fuzzy TOPSIS evaluation offer insightful information about the sustainability performance of the alternatives in the context of Industry 5.0 for the software engineering industry. Quantitative measurements were established to calculate the relative efficiency of the alternatives through the use of fuzzy set theory and the computation of fuzzy proximity coefficients. To summarise the overall sustainability ratings for each alternative, descriptive statistics were computed, such as the average fuzzy closeness coefficients and their standard deviations. These statistical results allow for a thorough evaluation of the alternatives' sustainability levels and the discovery of major variances among them. They also provide empirical evidence to help decision-making and establish a priority for the implementation of more sustainable software engineering practices in Industry 5.0.

When using the fuzzy TOPSIS technique, alternatives are ranked and evaluated according to how closely they resemble the best possible solution. Decision-makers can use this strategy to analyse difficult problems and make wise decisions by taking a step-by-step approach. The approach begins with the creation of a decision matrix that assesses how well options perform in relation to a number of criteria. This matrix is then weighted and normalised to identify the ideal and nonideal solutions, resulting in a thorough rating of available options based on how closely they adhere to the ideal answer. The fuzzy TOPSIS method's development in detail and results are given below.

Step 1: Create a decision matrix

The fuzzy TOPSIS approach was used to rank the four criteria and five alternatives in this research investigation. The criteria types and their corresponding weights are shown in Table 3 below.

Table 3. Characteristics of criteria.

	Name	Type	Weight
1	SC1	+	(0.250,0.250,0.250)
2	SC2	+	(0.250,0.250,0.250)
3	SC3	+	(0.250,0.250,0.250)
4	SC4	+	(0.250,0.250,0.250)

Table 4 below illustrates the fuzzy scale employed in the model.

Table 4. Fuzzy scale.

Code	Linguistic Terms	L	M	U
1	Very low	1	1	3
2	Low	1	3	5
3	Medium	3	5	7
4	High	5	7	9
5	Very high	7	9	9

Following are the findings of the decision matrix's examination of alternatives based on several criteria. It is significant to note that the matrix in Table 5 below provides the average ratings calculated by all experts in situations when numerous experts participated in the examination.

Table 5. Decision matrix.

	SC1	SC2	SC3	SC4
Alternative 1	(4.120,6.120,7.800)	(3.480,5.400,6.760)	(3.560,5.560,6.920)	(3.080,5.080,6.600)
Alternative 2	(2.760,4.600,6.280)	(2.680,4.600,6.200)	(3.720,5.720,7.080)	(3.000,5.000,6.520)
Alternative 3	(4.360,6.360,7.480)	(2.040,3.960,5.800)	(4.200,6.120,7.240)	(3.160,5.160,6.760)
Alternative 4	(2.840,4.760,6.360)	(2.520,4.280,6.040)	(2.920,4.840,6.440)	(2.120,3.960,5.720)
Alternative 5	(3.880,5.640,7.160)	(2.040,3.960,5.800)	(3.640,5.640,7.080)	(2.440,4.280,5.960)

Step 2: Create the normalised decision matrix

The normalised decision matrix can be calculated using the following relation, considering the positive and negative ideal solutions:

$$\widetilde{r}_{ij} = \left(\frac{a_{ij}}{c_j^*}, \frac{b_{ij}}{c_j^*}, \frac{c_{ij}}{c_j^*} \right); \; c_j^* = max_i \, c_{ij}; \text{ Positive ideal solution}$$

$$\widetilde{r}_{ij} = \left(\frac{a_j^-}{c_{ij}}, \frac{a_j^-}{b_{ij}}, \frac{a_j^-}{a_{ij}} \right); \; a_j^- = min_i \, a_{ij}; \text{ Negative ideal solution}$$

The Table 6 below displays the normalised decision matrix.

Table 6. A decision matrix that has been normalised.

	SC1	SC2	SC3	SC4
Alternative 1	(0.528,0.785,1.000)	(0.515,0.799,1.000)	(0.492,0.768,0.956)	(0.456,0.751,0.976)
Alternative 2	(0.354,0.590,0.805)	(0.396,0.680,0.917)	(0.514,0.790,0.978)	(0.444,0.740,0.964)
Alternative 3	(0.559,0.815,0.959)	(0.302,0.586,0.858)	(0.580,0.845,1.000)	(0.467,0.763,1.000)
Alternative 4	(0.364,0.610,0.815)	(0.373,0.633,0.893)	(0.403,0.669,0.890)	(0.314,0.586,0.846)
Alternative 5	(0.497,0.723,0.918)	(0.302,0.586,0.858)	(0.503,0.779,0.978)	(0.361,0.633,0.882)

Step 3: Generate the weighted normalised decision matrix

The weighted normalised decision matrix is calculated by multiplying the weight of each criterion with the corresponding normalised fuzzy decision matrix, as shown in the following formula.

$$\widetilde{v}_{ij} = \widetilde{r}_{ij} \cdot \widetilde{w}_{ij}$$

where $\widetilde{w}_{ij}$ represents weight of criterion c_j.

Table 7 below illustrates the weighted normalised decision matrix:

Table 7. Using a weighted, normalised decision matrix.

	SC1	SC2	SC3	SC4
Alternative 1	(0.132,0.196,0.250)	(0.129,0.200,0.250)	(0.123,0.192,0.239)	(0.114,0.188,0.244)
Alternative 2	(0.088,0.147,0.201)	(0.099,0.170,0.229)	(0.128,0.198,0.244)	(0.111,0.185,0.241)
Alternative 3	(0.140,0.204,0.240)	(0.075,0.146,0.214)	(0.145,0.211,0.250)	(0.117,0.191,0.250)
Alternative 4	(0.091,0.153,0.204)	(0.093,0.158,0.223)	(0.101,0.167,0.222)	(0.078,0.146,0.212)
Alternative 5	(0.124,0.181,0.229)	(0.075,0.146,0.214)	(0.126,0.195,0.244)	(0.090,0.158,0.220)

Step 4: Determine the fuzzy positive ideal solution (FPIS, A*) and the fuzzy negative ideal solution (*FNIS*, A^-)

The definition of FPIS (fuzzy positive ideal solution) and FNIS (fuzzy negative ideal solution) for the alternatives is as follows:

$$A^* = \{\widetilde{v}_1^*, \widetilde{v}_2^*, \ldots, \widetilde{v}_n^*\} = \left\{ \left(\max_j v_{ij} \middle| i \in B \right), \left(\min_j v_{ij} \middle| i \in C \right) \right\}$$

$$A^- = \{\widetilde{v}_1^-, \widetilde{v}_2^-, \ldots, \widetilde{v}_n^-\} = \left\{ \left(\min_j v_{ij} \middle| i \in B \right), \left(\max_j v_{ij} \middle| i \in C \right) \right\}$$

where $\widetilde{v}_i^*$ is the max value of i for all the alternatives and $\widetilde{v}_1^-$ is the min value of i for all the alternatives. B and C characterise the positive and negative ideal solutions, correspondingly.

Table 8 below illustrates the positive and negative ideal solutions.

Table 8. The positive and negative ideal solutions.

	Positive Ideal	Negative Ideal
SC1	(0.140,0.204,0.250)	(0.088,0.147,0.201)
SC2	(0.129,0.200,0.250)	(0.075,0.146,0.214)
SC3	(0.145,0.211,0.250)	(0.101,0.167,0.222)
SC4	(0.117,0.191,0.250)	(0.078,0.146,0.212)

Step 5: Determine the distance between each alternative and the fuzzy positive ideal solution A^* and the distance between each alternative and the fuzzy negative ideal solution A^-

The calculation of the distance between each alternative and the FPIS and the distance between each alternative and the FNIS is performed as follows:

$$S_i^* = \sum_{j=1}^{n} d(\widetilde{v}_{ij}, \widetilde{v}_j^*)\ i\ =\ 1, 2 \ldots,\ m$$

$$S_i^- = \sum_{j=1}^{n} d(\widetilde{v}_{ij}, \widetilde{v}_j^-)\ i\ =\ 1, 2 \ldots,\ m$$

d is the distance between two fuzzy numbers, when given two triangular fuzzy numbers (a_1, b_1, c_1) and (a_2, b_2, c_2), e distance between the two can be calculated as follows:

$$gap\ among d_v\left(\widetilde{M}_1, \widetilde{M}_2\right) = \sqrt{\frac{1}{3}\left[(a_1 - a_2)^2 + (b_1 - b_2)^2 + (c_1 - c_2)^2\right]}$$

Note that $d\left(\widetilde{v}_{ij}, \widetilde{v}_j^*\right)$ and $d\left(\widetilde{v}_{ij}, \widetilde{v}_j^-\right)$ are crisp numbers.

Table 9 presented below illustrates the distances from the positive and negative ideal solutions.

Table 9. Distance from positive and negative ideal solutions.

	Distance from Positive Ideal	Distance from Negative Ideal
Alternative 1	0.029	0.153
Alternative 2	0.099	0.082
Alternative 3	0.054	0.129
Alternative 4	0.164	0.017
Alternative 5	0.113	0.069

Step 6: Estimate the closeness coefficient and rank the alternatives

The calculation of the closeness coefficient for each alternative is as follows:

$$CC_i = \frac{S_i^-}{S_i^+ + S_i^-}$$

Table 10 below presents the ranking order of alternatives based on their closeness coefficient, where the best alternative is closest to the fuzzy positive ideal solution and farthest from the fuzzy negative ideal solution.

Table 10. Closeness coefficient.

	Ci	Rank
Alternative 1	0.843	1
Alternative2	0.452	3
Alternative 3	0.705	2
Alternative 4	0.094	5
Alternative 5	0.378	4

The graph presented below in Figure 4 illustrates the closeness coefficient of each alternative.

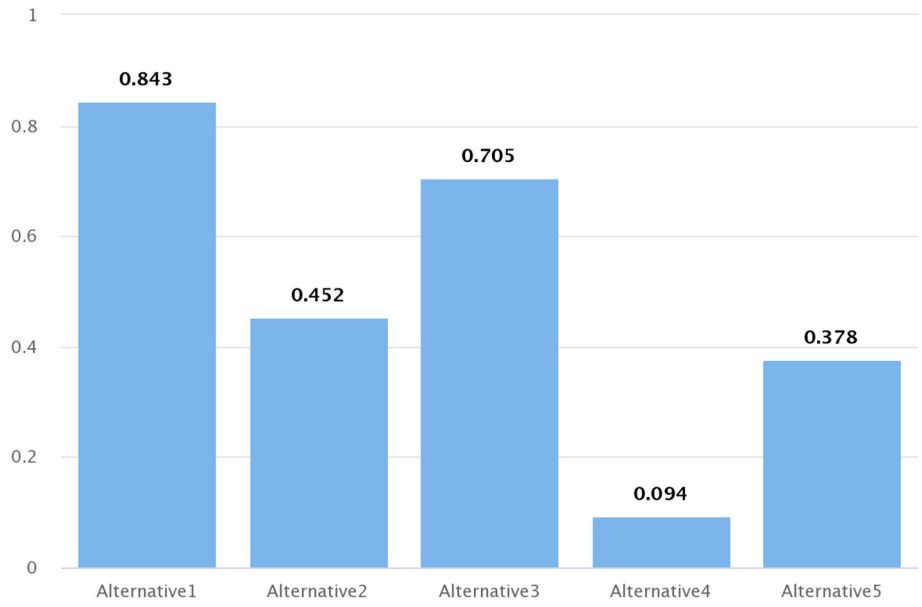

Figure 4. Closeness coefficient graph.

The results show how the examined alternatives ranked and how well they performed in terms of sustainability (Ci). Agile methodologies (Alternative 1), according to the outcomes, have the best sustainability performance of the alternatives, with a closeness coefficient of 0.843 and a top ranking of 1, respectively. DevOps (Alternative 3) earned the second position (Rank 2) and a moderately good closeness coefficient (Ci = 0.705). Cloud-native development (Alternative 2) came in third (Rank 3) with a lower closeness coefficient (Ci = 0.452). With a considerably lower closeness coefficient (Ci = 0.378), sustainable software engineering practices (Alternative 5) earned the fourth position (Rank 4). The traditional waterfall model (Alternative 4) had the least sustainable performance among all assessed options, as indicated by its ranking of fifth (Rank 5) as well as lowest closeness coefficient (Ci = 0.094).

4.2. Comparative Analysis

A key step in validating the outcomes provided in this research paper concerning the assessment of sustainability in the software engineering sector within the setting of Industry 5.0 is the comparison of the fuzzy TOPSIS and analytic hierarchy process (AHP) techniques. We may discover more about the consistency, reliability and resilience of the outcomes through the comparison of the rankings derived from these two different evaluation methodologies [40–45].

The comparison study not only demonstrates how sensitive the outcomes are to the selected evaluation methodology but it also offers the chance to spot any differences or parallels that might appear. We can strengthen the credibility and validity of the findings, enabling better decision-making and encouraging a more thorough comprehension of the environmental effectiveness of the evaluated alternatives, by closely examining the variations in the rankings and looking into the fundamental criteria as well as weighting variables used in every approach.

The comparison of the fuzzy TOPSIS and AHP methodologies presented in Table 11 demonstrates that the alternatives score differently in terms of sustainability capability. Agile methodologies had the top rank in the fuzzy TOPSIS technique (Rank 1), suggesting their outstanding sustainability performance. DevOps, cloud-native development, sustainable software engineering practices and the traditional waterfall model were next in line. Agile methodologies did, however, maintain their top ranking (Rank 1) when employing the AHP technique, which is consistent with the fuzzy TOPSIS results. The ranks of the other options, however, show some clear variations. DevOps is ranked second by AHP

(Rank 2), followed by cloud-native development (Rank 4), sustainable software engineering practices (Rank 3) and the traditional waterfall model (Rank 5), in that order.

Table 11. Comparative analysis findings.

Rank Order	1	2	3	4	5
AHP	Alternative 1	Alternative 3	Alternative 5	Alternative 2	Alternative 4
Fuzzy TOPSIS	Alternative 1	Alternative 3	Alternative 2	Alternative 5	Alternative 4

This comparative analysis shows how different ranks and conclusions might result from using a different evaluation technique. Both approaches take into account a variety of factors and offer insights on sustainability but they do so from various angles. While the AHP technique involves pairwise assessments of alternatives determined by relative criteria weights, the fuzzy TOPSIS approach concentrates on the proximity coefficients to the ideal solutions. Such differences show how delicate the evaluation process is and how crucial it is to choose a methodology that is suited to the research environment and needs [45–48].

The outcomes of the evaluation utilising the fuzzy TOPSIS technique offer insightful information on the viability of various alternatives in the context of Industry 5.0 in the software engineering sector. Stakeholders may pick the most sustainable choices and give them the highest priority for adoption in their software development processes using the rankings derived from the research. The results emphasise the value of taking sustainability factors into account and applying them to decision-making procedures. The outcomes also show the promise of agile approaches, DevOps, cloud-native development and sustainable software engineering practices for encouraging sustainability and dealing with environmental, social and economic problems. Industry experts, researchers and policymakers can use these insights to help them make educated decisions, implement best practices and support a more sustainable software engineering environment in Industry 5.0. To assist the ongoing development and improvement of sustainable software engineering practices in the future, additional analysis, validation and refining of the results may be carried out.

5. Discussion

The software engineering industry is undergoing a significant transformation within the fast-evolving Industry 5.0 paradigm, which is characterised by cutting-edge technologies, constantly changing information sharing and the integration of cyber-physical systems. In the wake of these profound changes, the integration of sustainability into software engineering practices poses a substantial challenge. This change calls for a thorough evaluation approach that measures software engineering practices' sustainability performance and ensures that they comply with Industry 5.0's ecological and social requirements. There is currently no systematic methodology in the academic environment that can handle the complex dynamics of sustainability in software engineering practices inside Industry 5.0. By providing a cutting-edge technique built on fuzzy TOPSIS and adeptly navigating the intricacies of sustainability evaluation within the developing Industry 5.0 setting, our research seeks to fill this gap.

Due to their adaptability, iterative process and focus on customer collaboration, agile techniques like Scrum, Kanban or Extreme Programming (XP) have grown significantly in favour in the software engineering sector. Agile approaches promote resource efficiency and responsiveness to changing environmental and social concerns, which can contribute to sustainability by enabling quicker delivery, adaptation to changing requirements and greater team communication. In order to achieve continuous delivery, continuous integration and quicker deployment cycles, DevOps focuses on improving communication and automation between development and operations teams. By reducing waste, improving resource management and encouraging dependable and frequent software releases, DevOps

practices can streamline processes, enhance efficiency and shorten time-to-market. These benefits are consistent with sustainability aims.

Building and deploying scalable, robust and effective software applications requires the use of cloud technologies as well as architectures. Organisations can improve resource utilisation, scalability and sustainability by making optimal use of cloud resources. Economic viability is aided by the cost-optimisation and improved fault tolerance provided by cloud-native development. An intentional effort to include sustainability ideas and practices into software engineering processes is represented by the sustainable software engineering practices option. This strategy places a strong emphasis on resource optimisation, green technology adoption and energy-efficient coding. Sustainable software engineering techniques have the potential to considerably lessen environmental impact and improve long-term sustainability, even if they could involve more up-front investment as well as specialised skills. Requirements gathering, design, development, testing and deployment are just a few of the distinct steps that make up the traditional waterfall model's linear and sequential software development process. Although this paradigm has been widely employed in the past, its lack of flexibility and adaptation may make it less sustainable. However, it is important to remember that the waterfall approach may still be advantageous, in particular situations where rigorous adherence to specifications and documentation is required.

The complete dataset obtained through the fuzzy-TOPSIS-based MCDM evaluation carried out in cooperation with professionals from the software sector has been thoroughly described in this paper. This dataset offers a thorough summary of the evaluation findings, illustrating how different software engineering methodologies perform in terms of sustainability. Even though this study does not contain any particular graphical plots, we have thoroughly analysed the data that were gathered and provided a full discussion of the consequences of the suggested methodology. These results are crucial in enabling decision-makers in the software engineering industry to prioritise sustainable practices in their decisions. The practical significance of this research within the software engineering field is improved by this contribution.

6. Conclusions

This study used the fuzzy TOPSIS methodology to assess the viability of different options in the context of Industry 5.0 for the software engineering sector. By using this way of decision-making, we were able to rank and gain insight into how other alternatives—such as the conventional waterfall model, agile methodologies, DevOps, cloud-native creation and sustainable software engineering practices—performed. The results of this study advance knowledge of sustainable software engineering practices and their potential for resolving social, economic and environmental issues. The findings underlined the benefits of DevOps, cloud-native development and agile approaches in supporting sustainability, including resource optimisation, accelerated delivery cycles and improved scalability. Furthermore, it became clear that employing sustainable software engineering practices was essential for minimising environmental impact and implementing green technologies. The findings of our study considerably assist in filling the existing research gap by offering a structured and practical framework for assessing sustainability in software engineering practices within the setting of Industry 5.0. This method tackles the complexity of sustainability assessment in a systematic manner, bridging the gap between the evolving software engineering industry and the need for environmentally and socially responsible practices.

However, there are some restrictions of this research. First of all, the assessment criteria as well as their weights are arbitrary and dependent on the situation. Different stakeholders might have different views on sustainability and the rankings may change depending on the weights given to each factor. Second, the evaluation is based on data that are currently accessible and does not fully account for the difficulty of sustainability in software engineering. It can be difficult to acquire correct and thorough data; thus, future studies should think about doing so in order to improve the evaluation. These issues could

be resolved and new directions for investigating sustainability in the software engineering sector could be explored in the future research of this area. Initially, broader and more uniform standards might be created to account for several aspects of sustainability, such as energy efficiency, carbon footprint, societal consequences and ethical issues. Secondly, to increase the precision and objectivity of the review process, sophisticated modelling methods and data analytics strategies could be incorporated. Furthermore, longitudinal studies may be carried out to determine potential synergies or trade-offs over time and evaluate the long-term sustainability impacts of various solutions.

Funding: The author extend their appreciation to the Deanship of Scientific Research at Northern Border University, Arar, KSA for funding this research work through the project number "NBU-FFR-2023-0096".

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The authors confirm that the data supporting the findings of this study are available within the article.

Acknowledgments: The author would like to express her gratitude to Deanship of Scientific Research at Northern Border University, Arar, KSA for funding this work.

Conflicts of Interest: The author declares no conflict of interest.

References

1. Kristin, R.; Richardson, D. A proposed recommender system for eliciting software sustainability requirements. In Proceedings of the 2013 2nd International Workshop on User Evaluations for Software Engineering Researchers (USER), San Francisco, CA, USA, 26 May 2013.
2. Calero, C.; Piattini, M. Introduction to green in software engineering. In *Green in Software Engineering*; Springer: Cham, Switzerland; Berlin/Heidelberg, Germany, 2015; pp. 3–27.
3. Pham, Y.D.; Bouraffa, A.; Maalej, W. Shapere: Towards a multi-dimensional representation for requirements of sustainable software. In Proceedings of the 2020 IEEE 28th International Requirements Engineering Conference (RE), Zurich, Switzerland, 31 August 2020–4 September 2020; pp. 358–363.
4. Ansari, M.T.J.; Pandey, D.; Alenezi, M. STORE: Security threat oriented requirements engineering methodology. *J. King Saud Univ.-Comput. Inf. Sci.* **2022**, *34*, 191–203. [CrossRef]
5. Roher, K.; Richardson, D. Sustainability requirement patterns. In Proceedings of the 2013 3rd International Workshop on Requirements Patterns (RePa), Rio de Janeiro, Brazil, 15 July 2013; pp. 8–11.
6. Rashid, N.; Khan, S.U. Developing Green and Sustainable Software using Agile Methods in Global Software Development: Risk Factors for Vendors. In Proceedings of the 11th International Conference on Evaluation of Novel Software Approaches to Software Engineering, Rome, Italy, 27–28 April 2016.
7. Calero, C.; Piattini, M. Puzzling out software sustainability. *Sustain. Comput. Inform. Syst.* **2017**, *16*, 117–124. [CrossRef]
8. Torre, D.; Procaccianti, G.; Fucci, D.; Lutovac, S.; Scanniello, G. On the presence of green and sustainable software engineering in higher education curricula. In Proceedings of the 2017 IEEE/ACM 1st International Workshop on Software Engineering Curricula for Millennials (SECM), Buenos Aires, Argentina, 27 May 2017; pp. 54–60.
9. Koontz, R.J.; Nord, R.L. *Architecting for Sustainable Software Delivery*; CrossTalk; Carnegie Mellon University: Pittsburgh, PA, USA, 2012; pp. 14–19.
10. Kern, E.; Hilty, L.M.; Guldner, A.; Maksimov, Y.V.; Filler, A.; Gröger, J.; Naumann, S. Sustainable software products—Towards assessment criteria for resource and energy efficiency. *Future Gener. Comput. Syst.* **2018**, *86*, 199–210. [CrossRef]
11. Ansari, M.T.J.; Al-Zahrani, F.A.; Pandey, D.; Agrawal, A. A fuzzy TOPSIS based analysis toward selection of effective security requirements engineering approach for trustworthy healthcare software development. *BMC Med. Inform. Decis. Mak.* **2020**, *20*, 236. [CrossRef] [PubMed]
12. Gibson, M.L.; Venters, C.; Duboc, L.; Betz, S.; Chitchyan, R.; Silva, V.P.; Penzenstadler, B.; Seyff, N. Mind the chasm: A UK fisheye lens view of sustainable software engineering. In Proceedings of the International Conference on Software Engineering: Software Engineering in Society Track, Buenos Aires, Argentina, 20–28 May 2017.
13. Yazdani, M.; Tavana, M.; Pamučar, D.; Chatterjee, P. A rough based multi-criteria evaluation method for healthcare waste disposal location decisions. *Comput. Ind. Eng.* **2020**, *143*, 106394. [CrossRef]
14. Eakin, H.; Bojórquez-Tapia, L.A. Insights into the composition of household vulnerability from multicriteria decision analysis. *Glob. Environ. Chang.* **2008**, *18*, 112–127. [CrossRef]
15. Li, J.; Fang, H.; Song, W. Sustainable supplier selection based on SSCM practices: A rough cloud TOPSIS approach. *J. Clean. Prod.* **2019**, *222*, 606–621. [CrossRef]

16. Penzenstadler, B.; Bauer, V.; Calero, C.; Franch, X. Sustainability in software engineering: A systematic literature review. In Proceedings of the 16th International Conference on Evaluation & Assessment in Software Engineering (EASE 2012), Ciudad Real, Spain, 14–15 May 2012.
17. Kitchenham, B.; Pretorius, R.; Budgen, D.; Brereton, O.P.; Turner, M.; Niazi, M.; Linkman, S. Systematic literature reviews in software engineering–a tertiary study. *Inf. Softw. Technol.* **2010**, *52*, 792–805. [CrossRef]
18. Zada, I.; Shahzad, S.; Ali, S.; Mehmood, R.M. OntoSuSD: Software engineering approaches integration ontology for sustainable software development. *Softw. Pract. Exp.* **2023**, *53*, 283–317. [CrossRef]
19. Manteuffel, C.; Ioakeimidis, S. A Systematic Mapping Study on Sustainable Software Engineering: A Research Preview. 9th SC@ RUG 2011–2012. 2012. Available online: https://www.researchgate.net/profile/Rein-Smedinga/publication/235704105_proceedings_of_the_nineth_StudCol_2012_18_and_20_April_2012_Groningen/links/02bfe512c6f5d17ff1000000/proceedings-of-the-nineth-StudCol-2012-18-and-20-April-2012-Groningen.pdf#page=36 (accessed on 22 August 2023).
20. Dick, M.; Naumann, S. Enhancing Software Engineering Processes towards Sustainable Software Product Design. In *EnviroInfo*; Shaker: Aachen, Germany, 2010; pp. 706–715.
21. Naumann, S.; Dick, M.; Kern, E.; Johann, T. The GREENSOFT Model: A reference model for green and sustainable software and its engineering. *Sustain. Comput. Inform. Syst.* **2011**, *1*, 294–304. [CrossRef]
22. Seacord, R.C.; Elm, J.; Goethert, W.; Lewis, G.A.; Plakosh, D.; Robert, J.; Wrage, L.; Lindvall, M. Measuring software sustainability. In Proceedings of the International Conference on Software Maintenance, ICSM 2003 Proceedings, Amsterdam, The Netherlands, 22–26 September 2003; pp. 450–459.
23. Venters, C.; Lau, L.; Griffiths, M.; Holmes, V.; Ward, R.; Jay, C.; Dibsdale, C.E.; Xu, J. The blind men and the elephant: Towards an empirical evaluation framework for software sustainability. *J. Open Res. Softw.* **2014**, *2*, e8. [CrossRef]
24. Amri, R.; Saoud, N.B.B. Towards a generic sustainable software model. In Proceedings of the 2014 Fourth International Conference on Advances in Computing and Communications, Cochin, India, 27–29 August 2014; pp. 231–234.
25. Mourão, B.C.; Karita, L.; do Carmo Machado, I. Green and sustainable software engineering-a systematic mapping study. In Proceedings of the 17th Brazilian Symposium on Software Quality, SBQS 2018, Curitiba, Brazil, 17–19 October 2018; pp. 121–130.
26. Johann, T.; Dick, M.; Kern, E.; Naumann, S. Sustainable development, sustainable software, and sustainable software engineering: An integrated approach. In Proceedings of the 2011 International Symposium on Humanities, Science and Engineering Research, Kuala Lumpur, Malaysia, 6–7 June 2011; pp. 34–39.
27. Oyedeji, S.; Seffah, A.; Penzenstadler, B. Classifying the measures of software sustainability. In Proceedings of the 4th International Workshop on Measurement and Metrics for Green and Sustainable Software Systems Co-Located with 12th International Symposium on Empirical Software Engineering and Measurement (ESEM 2018), Oulu, Finland, 9 October 2018.
28. Penzenstadler, B.; Mehrabi, J.; Richardson, D.J. Supporting physicians by re4s: Evaluating requirements engineering for sustainability in the medical domain. In Proceedings of the IEEE/ACM 4th International Workshop on Green and Sustainable Software (GREENS), Florence, Italy, 16–24 May 2015; pp. 36–42.
29. Swacha, J. Models of sustainable software: A scoping review. *Sustainability* **2022**, *14*, 551. [CrossRef]
30. Nazim, M.; Mohammad, C.W.; Sadiq, M. A comparison between fuzzy AHP and fuzzy TOPSIS methods to software requirements selection. *Alex. Eng. J.* **2022**, *61*, 10851–10870. [CrossRef]
31. Junior FR, L.; Osiro, L.; Carpinetti, L.C.R. A comparison between Fuzzy AHP and Fuzzy TOPSIS methods to supplier selection. *Appl. Soft Comput.* **2014**, *21*, 194–209. [CrossRef]
32. Ertuğrul, İ.; Karakaşoğlu, N. Comparison of fuzzy AHP and fuzzy TOPSIS methods for facility location selection. *Int. J. Adv. Manuf. Technol.* **2008**, *39*, 783–795. [CrossRef]
33. Ansari, M.T.J.; Baz, A.; Alhakami, H.; Alhakami, W.; Kumar, R.; Khan, R.A. P-STORE: Extension of STORE methodology to elicit privacy requirements. *Arab. J. Sci. Eng.* **2021**, *46*, 8287–8310. [CrossRef]
34. Alzahrani, F.A.; Ahmad, M.; Ansari, M.T.J. Towards design and development of security assessment framework for internet of medical things. *Appl. Sci.* **2022**, *12*, 8148. [CrossRef]
35. Çifçi, G.; Büyüközkan, G. A fuzzy MCDM approach to evaluate green suppliers. *Int. J. Comput. Intell. Syst.* **2011**, *4*, 894–909. [CrossRef]
36. Alshahrani, H.M.; Alotaibi, S.S.; Ansari, T.J.; Asiri, M.M.; Agrawal, A.; Khan, R.A.; Mohsen, H.; Hilal, A.M. Analysis and Ranking of IT Risk Factors Using Fuzzy TOPSIS-Based Approach. *Appl. Sci.* **2022**, *12*, 5911. [CrossRef]
37. Petrović, G.; Mihajlović, J.; Ćojbašić, Ž.; Madić, M.; Marinković, D. Comparison of three fuzzy MCDM methods for solving the supplier selection problem. *Facta Univ. Ser. Mech. Eng.* **2019**, *17*, 455–469. [CrossRef]
38. Alassery, F.; Alzahrani, A.; Khan, A.I.; Khan, A.; Nadeem, M.; Ansari, T.J. Quantitative Evaluation of Mental-Health in Type-2 Diabetes Patients Through Computational Model. *Intell. Autom. Soft Comput.* **2022**, *32*, 1701–1715. [CrossRef]
39. Henig, M.I.; Buchanan, J.T. Solving MCDM problems: Process concepts. *J. Multi-Criteria Decis. Anal.* **1996**, *5*, 3–21. [CrossRef]
40. Agrawal, A.; Khan, R.A.; Ansari, M.T.J. Empowering Indian citizens through the secure e-governance: The digital India initiative context. In *Emerging Technologies in Data Mining and Information Security: Proceedings of IEMIS 2022, Volume 3*; Springer Nature: Singapore, 2022; pp. 3–11.
41. Awodi, N.J.; Liu, Y.K.; Ayo-Imoru, R.M.; Ayodeji, A. Fuzzy TOPSIS-based risk assessment model for effective nuclear decommissioning risk management. *Prog. Nucl. Energy* **2023**, *155*, 104524. [CrossRef]

42. Hooshangi, N.; Gharakhanlou, N.M.; Razin, S.R.G. Evaluation of potential sites in Iran to localize solar farms using a GIS-based Fermatean Fuzzy TOPSIS. *J. Clean. Prod.* **2023**, *384*, 135481. [CrossRef]
43. Alharbi, A.; Ansari, T.J.; Alosaimi, W.; Alyami, H.; Alshammari, M.; Agrawal, A.; Kumar, R.; Pandey, D.; Khan, R.A. An Empirical Investigation to Understand the Issues of Distributed Software Testing amid COVID-19 Pandemic. *Processes* **2022**, *10*, 838. [CrossRef]
44. Attaallah, A.; Al-Sulbi, K.; Alasiry, A.; Marzougui, M.; Ansar, S.A.; Agrawal, A.; Ansari, T.J.; Khan, R.A. Fuzzy-Based Unified Decision-Making Technique to Evaluate Security Risks: A Healthcare Perspective. *Mathematics* **2023**, *11*, 2554. [CrossRef]
45. Gurmani, S.H.; Chen, H.; Bai, Y. Multi-attribute group decision-making model for selecting the most suitable construction company using the linguistic interval-valued T-spherical fuzzy TOPSIS method. *Appl. Intell.* **2023**, *53*, 11768–11785. [CrossRef]
46. Vadivel, R.; Saravanan, S.; Unyong, B.; Hammachukiattikul, P.; Hong, K.S.; Lee, G.M. Stabilization of delayed fuzzy neutral-type systems under intermittent control. *Int. J. Control Autom. Syst.* **2021**, *19*, 1408–1425. [CrossRef]
47. Shanmugam, S.; Hong, K.S. An event-triggered extended dissipative control for takagi-sugeno fuzzy systems with time-varying delay via free-matrix-based integral inequality. *J. Frankl. Inst.* **2020**, *357*, 7696–7717. [CrossRef]
48. Kacprzyk, J.; Bozhenyuk, A.; Gerasimenko, E. Lexicographic maximum dynamic evacuation modelling with partial lane reversal based on hesitant fuzzy TOPSIS. *Appl. Soft Comput.* **2023**, *144*, 110435. [CrossRef]