Knowledge and Information Systems (2020) 62:4569–4597
https://doi.org/10.1007/s10115-020-01496-7

REGULAR PAPER

A fuzzy-AHP-based approach to select software architecture based on quality attributes (FASSA)

Shahrouz Moaven[1] · **Jafar Habibi**[1]

Received: 8 October 2017 / Accepted: 14 July 2020 / Published online: 13 August 2020

Abstract
The software system design phase has recently received increasing attention due to continuous growth in both the size and complexity of software systems. As a key concept of this phase, software architecture plays an important role in the software extension cycle to the extent that the success of a software project is often determined by the degree of its design efficiency. In addition, software architecture evaluation is a fundamental step toward its subsequent validation. This paper is an attempt to propose an innovative method, based on fuzzy logic, to evaluate software architecture that addresses the inherent problems of existing methods found in the literature. The method can be used for complete design or even reconstruction of the architecture. Given the multi-faceted nature of the problem of evaluation and selection of an optimal architecture, we have employed a multi-objective decision technique, namely fuzzy hierarchical analysis process, which solves the problems associated with uncertainties and inaccuracies by incorporating fuzzy logic.

Keywords Architectural styles · Software architecture · Multi-objective decision method · Fuzzy logic · The fuzzy hierarchical analysis process

1 Introduction

Software architecture provides abstract and top-level models of a system as the software production line. It also plays a fundamental role in overcoming problems arising from the complexity and size of software projects. As technology, growth definitely increases the size and complexity of the software, the role of the architecture becomes more important in this process [17]. The most challenging problems in designing software architecture are related to the cost and time spent in the design and analysis. In addition, the architect's experience is also an important consideration in this regard. The use of design methods and architectural styles were introduced as a means to help software developers overcome these problems by reducing

✉ Shahrouz Moaven
moaven@ce.sharif.edu

Jafar Habibi
jhabibi@sharif.edu

[1] Department of Software Engineering, Faculty of Computer Engineering, Sharif University of Technology, Tehran, Iran

risk, cost and providing guidelines for less experienced architects. Although using architectural styles in designing the software system guarantees that some quality features of the existing style and architectural style may not satisfy the architectural requirements of some complex systems. To circumvent this issue, a combination of two or more different styles, referred to as *heterogeneous style,* can be used [35].

Since architecture is the first element in the software production cycle associated with the analysis of the quality attribute, providing a suitable architecture has a significant impact on meeting these kinds of requirements and the ultimate success of the system. On the other hand, the distribution and development of software systems, as well as the complexity of the full range of dependent requirements, make designing an efficient software architecture important, so much that the success or failure of software projects are largely determined by the design of efficient software architecture. Hence, the evaluation and verification of software architecture resulting from software architecture evaluation are basic requirements [5].

Due to the high costs of changes, the software architecture evaluation starts simultaneously with the design process and continues until its end. Predicting system quality, meeting its quality attributes and identifying potential risks in the design, are the goals of software architecture evaluation. Numerous methods have been proposed for software architecture evaluation, which differ in terms of the business objectives, desired quality attributes, techniques, activities and the relationship between stakeholders [10]. For example, from a measurement-theory point of view, having quantitative methods can add amounts of computational overhead and evaluation inaccuracies. Most of these inaccuracies are due to using a logic different from that used by the domain expert, which may cause many structural and semantic problems. Therefore, methods based on fuzzy logic could be used as an appropriate solution to improve the performance and precision of architecture evaluation. Moreover, in order to prevent making decisions based on incomplete criteria, taking advantage of information such as working platforms, application types, available quantitative records and many other effective parameters as complementary criteria can improve the decision-making process. The fact that software producers must consider many often overlapping and conflicting aspects and criteria turn the architecture selection process into complicated and multi-criteria enterprises. However, this complexity and vagueness arise much more in heterogeneous architectural styles, making the problem of their selection and evaluation more complicated.

In this study, we seek to propose an innovated method to evaluate different architectural styles based on their specific quality properties. To do so, we use a flexible multi-criteria decision-making approach, namely the hierarchical analysis process, as well as fuzzy logic in order to remove the vagueness and increase the precision in the comparison and evaluation of candidate architectural styles. The most important advantages of this method are accurate evaluation and the ability to extend the evaluation to aid in choosing among complicated heterogeneous architectural styles in a rational way. First, we introduce the required quality attributes and then we describe fuzzy logic. Next, the fuzzy hierarchical analysis process technique is described and after that, the proposed method is explained in detail. Finally, we present our conclusions in the last section.

2 Related work

In this section, we first discuss architecture evaluation and its various types, and then we shall deal with related works in the field of multi-criteria problems for architecture style selection.

There are different methods and structures to evaluate architectures, each adaptable to particular purposes. Current evaluation methods could be classified into two categories: quantitative and qualitative. To use special software architecture, research typically provides us with some guidelines by which qualitative attributes could be inferred [23]. Quantitative description is a means to compare the level of satisfying different quality attributes in different domain spaces. In other words, quantitative evaluations represent the significance level of described benefits and capabilities. The qualitative evaluation of architecture styles is, in fact, the combination of each architectural style with their inherent special quality attributes [23, 44]. A combination of the mentioned methods can be used. In the following, qualitative and quantitative methods are discussed in more detail.

In qualitative evaluation, questioning techniques are the most common methods used to evaluate architecture with every quality. Measuring techniques can be used to answer special questions and address special software qualities (e.g., performance or scalability); they are not as extensive as questioning techniques. There exist three types of questioning techniques, namely scenarios, questionnaires and checklists. There are some differences among these in terms of their applicability, but all have the same goal, namely to enhance our understanding of the degree of consistency between architecture and required quality attributes. In the scenario technique, quality attributes (e.g., maintainability, security, performance and reliability) become meaningful in the context. Scenarios are descriptive tools used to evaluate quality attributes in a context. In the questionnaire technique, a list of general and relatively open questions is provided which are applicable to all architectures [1]. Some questions are about architecture creation and documentation and some of them focus on descriptive details of architecture. In the checklist technique, a set of very detailed questions are used which have been obtained from several experiences about some similar domains. These questions focus on special quality attributes of a system. A checklist is helpful to keep a balanced focus on all areas of the system [29].

In quantitative evaluation, measuring techniques lead to quantitative results. These techniques answer the questions of the measurement team concerning the quality attributes of architecture instead of providing us with ways to design questions. These techniques are more complete and mature than questioning techniques. Metrics are quantitative interpretations based on particular observable measurements on the architecture, such as fan-in/fan-out of components [42]. In measuring techniques, the evaluation process should focus not only on the results of the metric but also on the assumptions underlying the use of the technique [1]. Constructing a prototype or performing a simulation of a system may help to establish and clarify architecture, but it is often costly. In other words, they are often a part of the development process.

Several other methods are used to evaluate software architecture. Scenario-based methods like SAAM (Scenario-based Architecture Analysis Method) [21, 38] were introduced in 1993 to provide a better understanding of the architecture and to prove that architecture deals with functional, in addition to qualitative requirements [1, 25]. This method was developed to ensure the compatibility of the architectural assumptions with the system's desirable attributes, understand risks associated with the architecture and potential conflicts among different quality attributes. Other methods like ATAM (Architecture Trade-off Analysis Method) [22, 38] deal with qualitative attributes and the balance between the attributes. Such methods evaluate the architecture to show to what extent specific quality objectives are met. They also focus on the contrast between qualitative attributes and their effects on each other. ATAM is based on SAAM.

There are other cost-based methods such as CBAM (Cost-Benefit Analysis Method) [38, 37], CBAM, which contrary to the other two methods, has created a bridge between

software development and economic issues. In this method, cost (especially budget) is considered as a qualitative attribute. With this strategy, the interaction between economic issues and other qualitative architectural concepts is measured. ALMA (Architecture-Level Analysis Method) [6, 25] uses maintenance cost estimation and risk assessment to analyze the variability of the system. FAAM (Family-Architecture Analysis Method) [47] is a method to measure the interoperability and expandability of a family of information systems.

Another category of methods focuses on formal methods. Formal methods include different types of architectural language descriptions, which are symbolic languages for expressing and describing the architecture of software systems [31]. In recent years, many studies have been conducted on formal architectural languages descriptions and the result is the creation of different architectural languages like Aesop [15], C2SADL [30], Darwin [27], MetaH [7], Rapide [26], UniCon [43] and ACME [16].

Now, we shall review the work related to multi-criteria problems for architecture style selection. (Tan et al. 2010) discusses the Choquet integral, which is a fuzzy operator enabling the selection of suitable architectural styles. The integral is a general tool in multi-criteria decision-making processes. Moaven et al. [33, 34] introduce a Decision Support System (DSS) which makes use of a fuzzy concept to represent concepts of quality attributes more precisely and efficiently. The system takes advantage of fuzzy inference support decisions of software architects. Moaven et al. [35] introduces a framework and a technique to provide an environment for evaluating heterogeneous architecture styles of a software system. The framework exploits the quantity attributes obtained during their use in order to increase the evaluation validity. Babu et al. [4] explain the ANP approach which is used to represent the various criteria and concepts of quality attributes more precisely and efficiently. One of the goals of this approach is to optimize software architecture design. Galster et al. [14] have introduced SYSAS as a method for systematic and traceable selection of architectural styles. This method is based on the characteristics of basic architectural elements relevant to the developer, and also those of the target system that are visible to the end-user. Moaven et al. [33, 34] explain that it is possible to use fuzzy inference to support the decisions made by software architects. The main objective of the study is to get the most out of the properties of styles. An integrated approach of SPL with architecture style selection and component-based design is presented in Zaki et al. [50]. This approach seeks to select the best architectural style using the fuzzy analytic hierarchy process. Dwivedi and Rath [13] present a library of styles to select an appropriate style for software systems. This study uses formal modeling language alloy for reusing and extensible modeling of a complex and highly distributed components. Tahmasebipour and Babamir [45] have evaluated the interaction between architectural tactics and architectural styles. By taking advantage of a new ranking scheme for the architectural styles, the best architectural style may be obtained for every individual quality attribute or their combinations. An intuition-based fuzzy Choquet integral is proposed in Tan and Chen [46] for multiple-criteria decision making. In fact, this integral takes account of interactions among the decision-making criteria. A method for the selection of the best software architecture styles, called SSAS, was suggested in Babu et al. [3] for improved selection methodology, reflecting interdependencies among the evaluation criteria using the analytic network process within a zero–one goal programming model. A meta-model called ADUAK is introduced in Dhaya and Zayaraz [12] for fuzzy-based quantitative architectural evaluation. The main objective of this study is to improve the efficiency of the architectural design process through architectural knowledge management (AKM) support. Saaty [39] explains that

many decision problems cannot be structured hierarchically because they involve interaction and dependence of higher-level elements on a lower-level element. The AHP and ANP are used to solve the problem of independence on alternatives or criteria and the problem of dependence among alternatives or criteria, respectively [18].

In the context of reviewing the evaluation methods, Abrahão and Insfran [2] are among the most insightful studies. In this work, the quality-driven architecture derivation and improvement (QUADAI) method is compared against the well-known architecture tradeoff analysis method (ATAM), showing that QUADAI obtains better results than ATAM in terms of effectiveness, perceived usefulness and intention to use.

Some research is done in Mahdavi-Hezavehi et al. [28] to demonstrate which architecture handles qualitative attributes to help the researchers know how to score and weight these attributes in self-adaptable systems. Some other studies have been conducted in the context of qualitative attributes of software product lines that may be generalizable in the architecture area [36]. This paper presents a systematic literature review with the objective of identifying and interpreting all studies from 1996 to 2010 that present quality attributes and measures for SPL. These attributes and measures have been classified using a set of criteria that includes the life cycle phase in which the measures are applied, the corresponding quality characteristics and their support for specific SPL characteristics. Moaven et al. [32] and Dasanayake et al. [11] provide other approaches that are useful for decision making about choosing architecture and its styles. In addition, Moaven et al. [33, 34] focuses on the decision-making procedure and using knowledge that could be useful in this selection.

In all mentioned methods, there are important challenges that remain open, like the overlapping of quality attributes or the contradiction between them. This is especially relevant when there is a need for selecting a suitable architectural style for the reconstruction of a legacy system. In selecting styles, due to multi-criteria selection and uncertainty (which depends on the experts and the system architecture), special attention must be paid for integrating system architecture experience and involving mechanisms able to learn. Using numerical concepts and binary thinking cannot reflect the real attributes of a style. Problems like these cannot be solved completely and always there are mechanisms for improving accuracy and effectiveness of choosing a suitable style. In this paper, we propose a method for covering the existing challenges.

3 Quality attributes

Quality attributes are a part of the requirements of application programs which express the concerns of users and other stakeholders to access the functional requirements of the systems [17]. Several efforts have been made to provide models (software quality models) for the description and classification of quality attributes. The ISO/IEC quality model is one of these software quality models which has had more coverage and covers more aspects of software quality. This model was presented in 1991 by the International Organization for Standardization (ISO) to meet the need of the software industry to develop a standardized software evaluation. It was modified again in 2001 [20]. In the ISO/IEC model, as shown in Fig. 1, software quality is described based on six main quality attributes, each of which have several sub-quality attributes. These attributes are as follows:

Functionality: The ability of the software product to provide functional requirements in a specific situation is called capability. Sub-features of this feature are suitability, accuracy, interoperability and security.
Reliability: The ability of the software product to remain at a certain level of performance when used in certain circumstances is called reliability and its sub-features are maturity, fault tolerance and recoverability.
Usability: This feature describes how much a software product can be learned, attractive and practical in specific usage conditions. Understandability, learnability and operability are sub-features of this characteristic.
Efficiency: Efficiency demonstrates the capability of the software product to provide appropriate performance, related to the amount of resources used in certain circumstances [20]. In other words, efficiency is the required system response time for a certain number of events in a given time interval. It shows how well the interactions between system components are conducted. It seeks to reduce time waste and response time in the system. The sub-features of efficiency are resource utilization and time behavior.

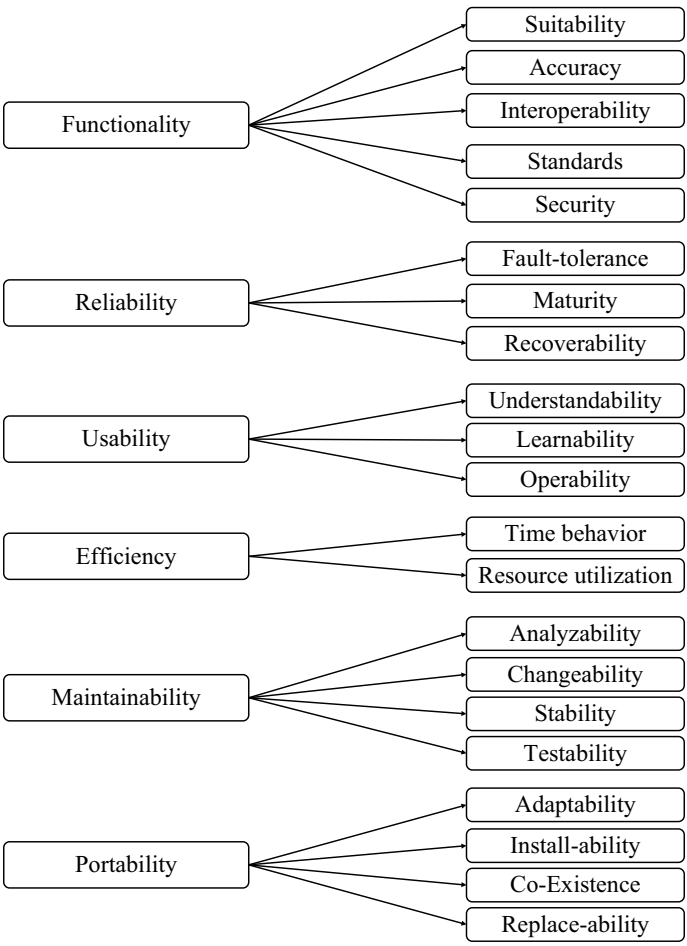

Fig. 1 ISO/IEC quality model

Maintainability: The ability of the software product to apply changes and modifications. These changes are analysis capability, variability, stability and testability.
Portability: This attribute expresses the ability to run the system under various computing environments, which include hardware, software and organizational environments. In order to evaluate portability adaption, installation and co-existence, sub-features should be examined.

4 Fuzzy logic

Fuzzy sets and fuzzy logic are powerful mathematical-based tools for modeling uncertain systems which also facilitate hypothetical reasoning in decision making where there is partial and inaccurate information. The main characteristic of a fuzzy set is its ability to represent vague information [49]. Given that triangular fuzzy numbers demonstrated in Fig. 2 lead to a simple process and simple computations in displaying data in a fuzzy environment, employing these numbers is suitable in most applications. The function of a triangular fuzzy number with the $M=(l, m, u)$ triple is defined with respect to membership function $\mu_A(x)$ as follows:

$$\mu_A(x) = \begin{cases} 0 & x < l; \\ \frac{x-l}{m-l} & l \le x \le m; \\ \frac{m-x}{u-m} & m \le x \le u; \\ 0 & x > u; \end{cases} \tag{1}$$

Consider two triangular numbers $M_1 = (l_1, m_1, u_1)$ and $M_2 = (l_2, m_2, u_2)$ which are plotted. Different operations are defined on triangular fuzzy numbers; the most important of them are as follows:

$$M_1 + M_2 = \left(l_1 + l_2, m_1 + m_2, u_1 + u_2\right) \tag{2}$$

$$M_1 * M_2 = \left(l_1 * l_2, m_1 * m_2, u_1 * u_2\right) \tag{3}$$

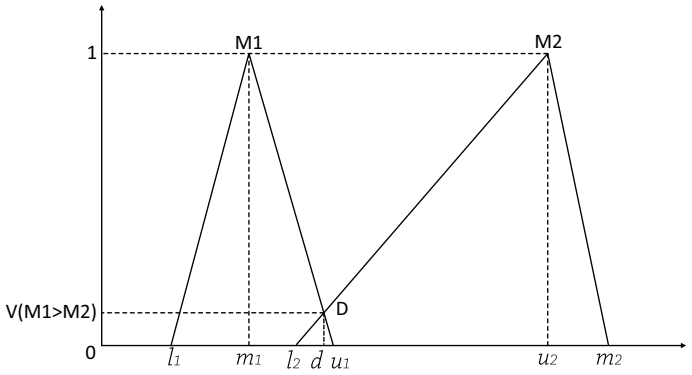

Fig. 2 Triangular numbers M_1 and M_2

$$M_1^{-1} = \left(\frac{1}{u_1}, \frac{1}{m_1}, \frac{1}{l_1}\right). M_2^{-1} = \left(\frac{1}{u_2}, \frac{1}{m_2}, \frac{1}{l_2}\right) \tag{4}$$

5 The fuzzy analytic hierarchy process (FAHP)

The analytic hierarchy process (AHP) is a multi-criteria decision-making method which selects the best option from all candidates [41] based on several criteria, which may be different or even in conflict with each other. In this method, after decomposing the problem by hand and constructing the hierarchical tree, completing a pairwise comparison between the available criteria in the same level, a certain number is assigned to each option to rank the options and allow the selection of the best one [40]. Based on AHP, the essential steps to solve complex and multi-criteria problems are:

1. Defining decision-making criteria in the form of a hierarchical structure so that the goal is placed in the topmost level, criteria and sub-criteria are in the middle levels, and the candidate solutions and options are in the lowest level.
2. Defining computing criteria, sub-criteria and replacements weights based on the relative importance of each member from its higher level.
3. Ranking the options.

In spite of AHP efficiency in the management of quantitative and qualitative criteria, the lack of specific standards for measuring and representing the numerical values of the qualitative criteria and ambiguity and uncertainty in relative comparisons leads to uncertainty in the ranking of options [9]. As a solution to this problem, a combination of analytic hierarchy process and fuzzy logic was first introduced in Van Laarhoven and Pedrycz [24] in order to adapt better the process to reality, to consider uncertainty and inaccuracy issues into account, and finally to increase precision in ranking options [24]. In this method, triangular fuzzy numbers and logarithmic least-squares were used to compute proportional weight. In addition, an extension to this method is FAHP, which was proposed by Chang et al. [9] to reduce computational overhead compared to the method. The development stages of FAHP based on the extended analysis method are as follows [8]:

1. Creation of the Fuzzy Pairwise Comparison Matrix. In this stage, a pairwise comparison matrix is determined based on triangular fuzzy numbers, and by decision-makers. In this case, if m_{ij} is the real value obtained from the comparison of two criteria i and j, the upper and lower bounds of the triangular fuzzy number, based on the fuzzification degree (δ), are determined by $(m_{ij} - \delta, m_{ij}, m_{ij} + \delta)$. So, the element m_{ij} of the matrix can be specified by the triangular fuzzy number, which the subscripts i and j refer to the row and column, respectively. Table 1 in Sect. 6 is based on this pairwise comparison. From Eq. 5 to 7, let $G = \{g_1, g_2, \ldots, g_n\}$ be a set of criteria, where n is the number of criteria.
2. Calculation of the Fuzzy Composite Extension S_i (the value of fuzzy synthetic extent with respect to the i$^{\text{th}}$ object). In fact for estimating of the weight values for each criterion and for each alternative with reference to a given criteria, S_i is calculated for each row of the pairwise comparison matrix according to the following equations:

Table 1 Allocated fuzzy values to any conversational values

Description	Linguistic values	Fuzzy numbers
The two compared elements have equal importance	Just equal	(1, 1, 1)
One of the elements preferred to the other by a narrow margin	Weakly more important	(1, 3/2, 2)
One of the elements strongly preferred to the other	Strongly more important	(3/2, 2, 5/2)
The preference of one element to another is practically demonstrated	Very strongly more important	(2, 5/2, 3)
The preference of one element is at the highest certainty level	Absolutely more important	(5/2, 3, 7/2)
If one of the above fuzzy numbers will be assigned to a comparison i to j, the reciprocal of that number is assigned to comparison j to i	Reciprocal value	

$$S_i = \sum_{j=1}^{m} M_{g_i}^j . \left[\sum_{i=1}^{n} \sum_{j=1}^{m} M_{g_i}^j \right]^{-1} \tag{5}$$

$$\sum_{j=1}^{m} M_{g_i}^j = \left[\sum_{j=1}^{m} l_{ij}, \sum_{j=1}^{m} m_{ij}, \sum_{j=1}^{m} u_{ij} \right] \tag{6}$$

$$\sum_{i=1}^{n} \sum_{j=1}^{m} M_{g_i}^j = \left[\sum_{i=1}^{n} l_{ij}, \sum_{i=1}^{n} m_{ij}, \sum_{i=1}^{n} u_{ij} \right] \tag{7}$$

Related parameters in these equations (l, m, u) are positive numbers and demonstrated in Fig. 2. Besides G is the goal set.

3. Calculation of the Priority Degree (Feasibility) of S_i over S_k. The degree of possibility for convex fuzzy number M to be greater than the number of k fuzzy numbers M_i $(i = 1, 2, \ldots, k)$ is given by the use of the operations max and min. Here the degree of importance of each S_i is calculated with respect to each other. In general, if M_1 and M_2 are triangular fuzzy numbers, in order to find the degree of importance of M_1 over M_2 or compare M_1 and M_2, where $(M_1 \geq M_2)$ or $(M_2 \geq M_1)$, the values of V are needed and are calculated by the following equation:

$$V(M_2 \geq M_1) = \begin{cases} 1 & \text{if} \quad m_2 \geq m_1; \\ 0 & \text{if} \quad l_1 \geq u_2; \\ \frac{l_1 - u_2}{(m_2 - u_2) - (m_1 - l_1)}, & \text{otherwise} \end{cases} \tag{8}$$

As it is clear, the degree of importance of a certain triangular fuzzy number compared to the other triangular fuzzy numbers is calculated by:

$$
\begin{aligned}
&V\left(M \geq M_1, M_2, \ldots, M_k\right) \\
&\quad = V\left(M \geq M_1\right), V\left(M \geq M_2\right), \ldots, V(M \geq M_k) \\
&\quad = \min V(M \geq M_i), i = 1, 2, \ldots, k
\end{aligned} \tag{9}
$$

4. Calculation weights of the criteria. The criteria's weight in the pairwise comparison matrix is calculated using the following equation:

$$W'\left(x_i\right) = \text{Min}\left\{V\left(S_i \geq S_k\right)\right\}, k = 1, 2, \ldots, n \quad \text{and} \quad k \neq i \tag{10}$$

Therefore, the criteria weight vector is:

$$W'(x) = [W'\left(x_1\right), W'\left(x_2\right), \ldots, W'\left(x_n\right)]^T \tag{11}$$

5. Weight Normalization. Results of the previous step are normalized as follows:

$$W_i = \frac{W'_1}{\sum W'_i} \tag{12}$$

6 Proposed method

In Moaven et al. [33, 34], architecture styles selection was introduced as a multiple-criteria problem and resolved it using a fuzzy integral. In this paper, the method that proposed helps software architects and developers to evaluate candidate architectures or architectural styles based on the system goals and stakeholder viewpoints and select the most useful architectural style with respect to their particular and required goals. To this aim, first, the required hierarchical structure is defined based on ISO/IEC software quality model; next, the fuzzy variables are used to explain the hypothetical reasoning about the relative importance of different criteria and options. Finally, candidate architectural styles are ranked by employing the FAHP method in order to select the best option. These steps are at length described and analyzed in the remainder of this section. Furthermore, the process of attributes weighting and method evaluation are discussed in the following.

6.1 Identifying criteria and sub-criteria

In order to choose an architectural style, we encounter different quality attributes that are viewed as decision-making criteria. Software hierarchical quality models, such as ISO/IEC, can be used to classify the structure of the quality attributes (criteria) and sub-criteria of the architectural style's selection problem. The hierarchical structure of this problem would then contain 6 criteria and 21 sub-criteria based on the ISO/IEC software quality model.

6.2 Constructing the hierarchical structure

Here, the architectural style selection problem is decomposed and described in a simple format according to the human mind. Also, based on a hierarchical structure, we see that the problem, comprising four levels:

- Goal: evaluating architectural style.
- Criterion: software quality attributes.
- Sub-criterion: secondary quality attributes corresponding to each quality attribute.
- Options: architectural styles.

To create a hierarchical structure, the decision-making goal, which is an evaluation of architectural style, is placed in the root of the tree. In the next level, the required quality attributes are appended to the tree as the branches of the root and the impact of each attribute in fulfilling the goal is considered as the weight of the edge. At the third level, the secondary attributes affecting each quality attribute are considered as the children of each attribute, with the impact of each attribute constituting the weight of the edge connecting them. Figure 3 illustrates the hierarchical structure for architectural style evaluation based on the ISO/IEC quality model.

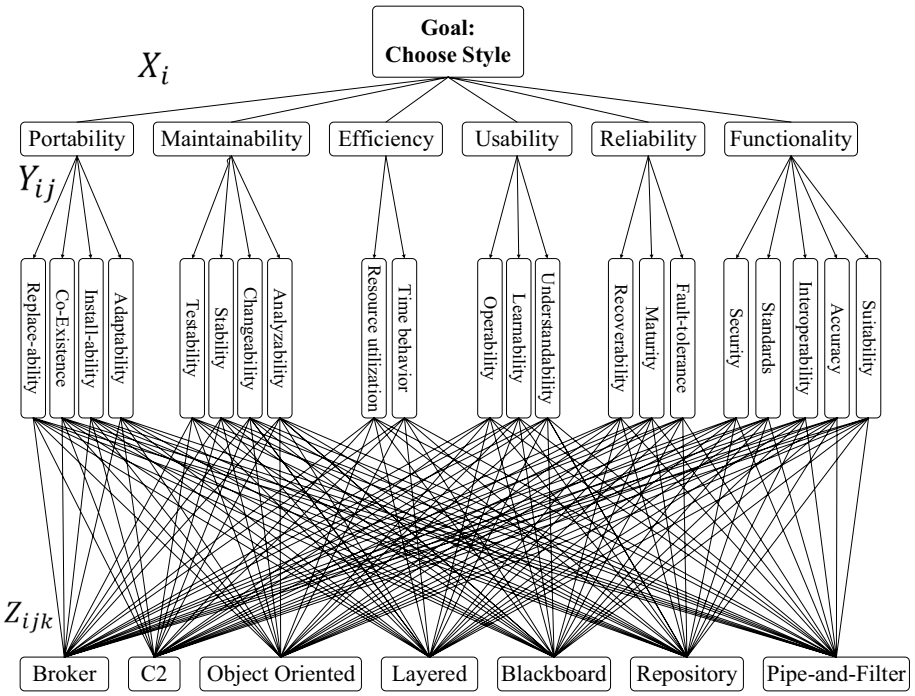

Fig. 3 Hierarchical structure for architectural style evaluation

6.3 Creating fuzzy decision-making matrices

After constructing the hierarchical structure for the problem, a pairwise comparison for the criteria, sub-criteria and options is carried out, and finally, the fuzzy decision-making matrices are developed. The collection of viewpoints of decision-makers and experts are required to define the pairwise comparisons among the members at any level of this hierarchical structure. Employing questionnaires is one of the ways of gathering information. In the process of selecting the most appropriate architectural style, a number of questionnaires should be prepared to be used by decision-makers and experts. These questionnaires compare the criteria, sub-criteria and options for each two of them and according to the existing upper-level members pairwisely and also explain their relative importance with respect to one another using items (such as the same importance, a bit more important, more important, very important and the most important). After this step, the conversational results of these comparisons are replaced with fuzzy numbers based on the prevalent pattern. For example, as shown in Table 1, in this method, the conversational values are replaced with triangular fuzzy values.

After collecting experts' viewpoints, the decision fuzzy matrices are created. Each member of the fuzzy decision-making matrix is indicated by a triple vector defining the comparison triangular fuzzy number between i and j. We suppose that is a pairwise comparison between i and j from the viewpoint of decision-maker k, denoted by D_{ijk}, as shown in Fig. 2. Then, the triangular fuzzy numbers for any pairwise comparisons are defined based on Eqs. (13) to (15) as follows.

$$l_{ij} = \min_{k} (D_{ijk}) \tag{13}$$

$$u_{ij} = \max_{k} \left(D_{ijk}\right) \tag{14}$$

$$m_{ij} = \sqrt[k]{\prod_{k=1}^{k} D_{ijk}} \tag{15}$$

These equations are used for solving equations such as (6) and (7).

6.4 Identifying criteria weight

The relative importance of each quality attribute depends on the conditions of the system and the type of the system architecture to be designed or evaluated. In other words, criteria weights are different for each system and depend on the problem domain and the importance of the quality attributes in the system. For example, portability may be the criterion with the highest degree of importance in one system, whereas it has the lowest degree of importance in another. Therefore, to explain the relative importance and make a pairwise comparison among the criteria, the evaluation process must be focused on a particular area, and the system or the proportional weights of different criteria must also be considered as parameters.

In this study, to evaluate the proposed method, a special domain is supposed for the problem at hand. Thus, the decision-making team fills out the questionnaires based on the

selected domain and uses this method to select the most appropriate style among candidate architectural styles. Suppose that we need to choose the most appropriate architectural style for a system with the following characteristics. The instance of this domain is the embedded system as a broker gateway between the banking system and different type of customer systems. Because of the platform, banking security is not the issue of our case study in this paper and it handled in another layer of general systems. The important things of this especial system are as below:

- The system must have high performance.
- System flexibility is the second priority in terms of importance ranking.
- Availability and fault tolerance are important factors in the system.
- The data transfer paths must be determined.
- The cost of planning, designing and other activities are also important.

Based on these assumptions, the fuzzy decision-making matrix used for pairwise comparison among the criteria is defined according to Table 2.

The relative weights of criteria are calculated according to Table 2 and the FAHP method as follows:

$$S_{c1} = (8.50, 11, 13.50) \otimes (1/50.83, 1/41.30, 1/33.17) = (0.17, 0.27, 0.14)$$
$$S_{c2} = (5.90, 7.50, 9.17) \otimes (1/50.83, 1/41.30, 1/33.17) = (0.12, 0.18, 0.28)$$
$$S_{c3} = (6.83, 8.40, 10) \otimes (1/50.83, 1/41.30, 1/33.17) = (0.13, 0.2, 0.3)$$
$$S_{c4} = (4.40, 5.33, 6.67) \otimes (1/50.83, 1/41.30, 1/33.17) = (0.09, 0.13, 0.2)$$
$$S_{c5} = (4.30, 5.17, 6.33) \otimes (1/50.83, 1/41.30, 1/33.17) = (0.08, 0.13, 0.19)$$
$$S_{c6} = (3.23, 3.90, 5.17) \otimes (1/50.83, 1/41.30, 1/33.17) = (0.06, 0.09, 0.16)$$

The degree of importance of the values was calculated in the previous step is defined with respect to other values.

$$V(S_{c1} \geq S_{c2}) = 1, \quad V(S_{c1} \geq S_{c3}) = 1, \quad V(S_{c1} \geq S_{c4}) = 1,$$
$$V(S_{c1} \geq S_{c5}) = 1, \quad V(S_{c1} \geq S_{c6}) = 1, \quad V(S_{c2} \geq S_{c1}) = 0.56,$$
$$V(S_{c2} \geq S_{c3}) = 0.87, \quad V(S_{c2} \geq S_{c4}) = 1, \quad V(S_{c2} \geq S_{c5}) = 1,$$
$$V(S_{c2} \geq S_{c6}) = 1, \quad V(S_{c3} \geq S_{c1}) = 0.68, \quad V(S_{c3} \geq S_{c2}) = 1,$$
$$V(S_{c3} \geq S_{c4}) = 1, \quad V(S_{c3} \geq S_{c5}) = 1, \quad V(S_{c3} \geq S_{c6}) = 1,$$
$$V(S_{c4} \geq S_{c1}) = 0.2, \quad V(S_{c4} \geq S_{c2}) = 0.62, \quad V(S_{c4} \geq S_{c3}) = 0.47,$$
$$V(S_{c4} \geq S_{c5}) = 1, \quad V(S_{c4} \geq S_{c6}) = 1, \quad V(S_{c5} \geq S_{c1}) = 0.14,$$
$$V(S_{c5} \geq S_{c2}) = 0.57, \quad V(S_{c5} \geq S_{c3}) = 0.42, \quad V(S_{c5} \geq S_{c4}) = 1,$$
$$V(S_{c5} \geq S_{c6}) = 1, \quad V(S_{c6} \geq S_{c1}) = 0.04, \quad V(S_{c6} \geq S_{c2}) = 0.31,$$
$$V(S_{c6} \geq S_{c3}) = 0.16, \quad V(S_{c6} \geq S_{c4}) = 0.67, \quad V(S_{c6} \geq S_{c5}) = 0.70$$

Table 2 Fuzzy decision-making matrix for comparing pair criteria

	C1			C2			C3			C4			C5			C6		
C1: functionality	1.00	1.00	1.00	1.50	2.00	2.50	2.00	2.5	3.00	1.00	1.50	2.00	1.50	2.00	2.50	1.50	2.00	2.50
C2: reliability	0.40	0.50	0.67	1.00	1.00	1.00	1.00	1.00	1.00	1.50	2.00	2.50	1.00	1.50	2.00	1.00	1.50	2.00
C3: usability	0.33	0.40	0.50	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.50	2.00	1.50	2.00	2.50	2.00	2.50	3.00
C4: efficiency	0.50	0.67	1.00	0.40	0.50	0.67	0.50	0.67	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.50	2.00
C5: maintainability	0.40	0.50	0.67	0.50	0.67	1.00	0.40	0.50	0.67	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.50	2.00
C6: portability	0.40	0.50	0.67	0.50	0.67	1.00	0.33	0.40	0.50	0.50	0.67	1.00	0.50	0.67	1.00	1.00	1.00	1.00

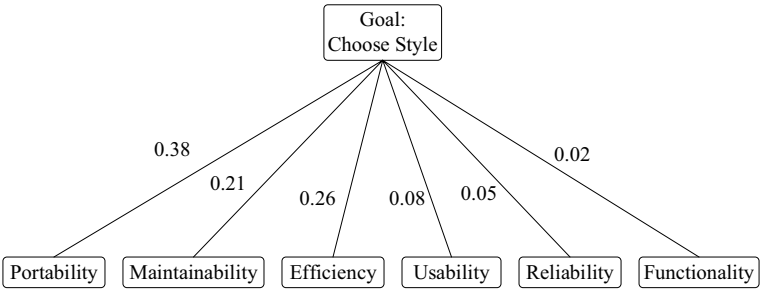

Fig. 4 Structure tree and weigh of primary criteria in architectural styles evaluation

$$W'(C_1) = \text{Min}\{1, 1, 1, 1, 1\} = 1$$
$$W'(C_2) = \text{Min}\{0.56, 0.87, 1, 1, 1\} = 0.56$$
$$W'(C_3) = \text{Min}\{0.68, 11, 1, 1\} = 0.68$$
$$W'(C_4) = \text{Min}\{0.2, 0.62, 0.47, 1, 1\} = 0.2$$
$$W'(C_5) = \text{Min}\{0.14, 0.57, 0.42, 1, 1\} = 0.14$$
$$W'(C_6) = \text{Min}\{0.04, 0.31, 0.16, 0.67, 0.7\} = 0.04$$

Based on the values obtained in the previous step, the following criteria weight vector is defined:

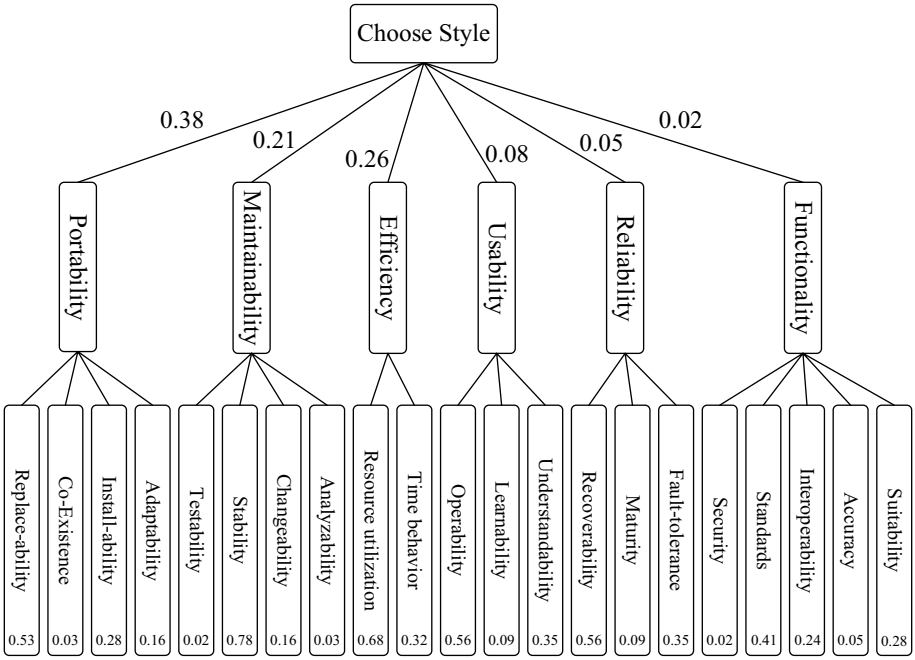

Fig. 5 Tree structural and proportional weight of sub-criteria in architectural styles evaluation problem

$$W' = (1, 0.56, 0.68, 0.2, 0.14, 0.04)$$

Finally, the criteria weights are normalized as follows to create the structure tree of architectural style evaluation in Fig. 4.

$$W = (0.38, 0.21, 0.26, 0.08, 0.05, 0.02).$$

6.5 Calculating sub-criteria and style weight or proportional priority

In this step, by conducting a pairwise comparison between the sub-criteria of each quality property and performing computations similar to what we observed in the previous step the weight of all the sub-criteria are calculated according to their corresponding criterion as shown in Fig. 5. After calculating the proportional weight of each sub-criterion, the proportional weight of any architectural style can be calculated based on any sub-criterion just as in the previous method.

6.6 Style scoring

In the final step, the architectural styles are scored based on both final weights of sub-criteria and relative weights of architectural styles. In more detail, the weight of any criteria must be multiplied by the relative weight of its parent (the weight of the main criterion related to the sub-criterion). Figure 5 illustrates the final weights of the criteria and sub-criteria. Finally, the weight or score of any architectural style is calculated using the following equation, which is an aggregation of the relative weights of all the styles in this example multiplied by the final weight of the corresponding sub-criterion:

$$OP(Q_i) = Y_i \tag{16}$$

$$OP(P_j) = \sum_{i}^{m} X_{ij} \times OP(Q_i) = \sum_{i}^{m} X_{ij} \times Y_i \tag{17}$$

$$OP(S_k) = \sum_{j}^{n} Z_{jk} \times OP(P_j) = \sum_{j}^{n} Z_{jk} \times Y_j \tag{18}$$

where n is the number of sub-criteria, Z_{jk} is the relative weight of style S_k compared to the sub-criterion P_j, and Y_j is the final weight of sub-criterion P_j. According to Table 7, Z_{jk} is calculated. The knowledge and experience of an expert in the questionnaires determine the behaviors of architectural styles in comparison with pairwise comparison method in every quality attribute and the computations will be done accordingly.

Table 3 Privileging architectural styles

Style	Client–server	C2	Object oriented	Layer	Blackboard	Storage	Pipe and filter
Rank	0.175	0.145	0.123	0.147	0.115	0.112	0.182

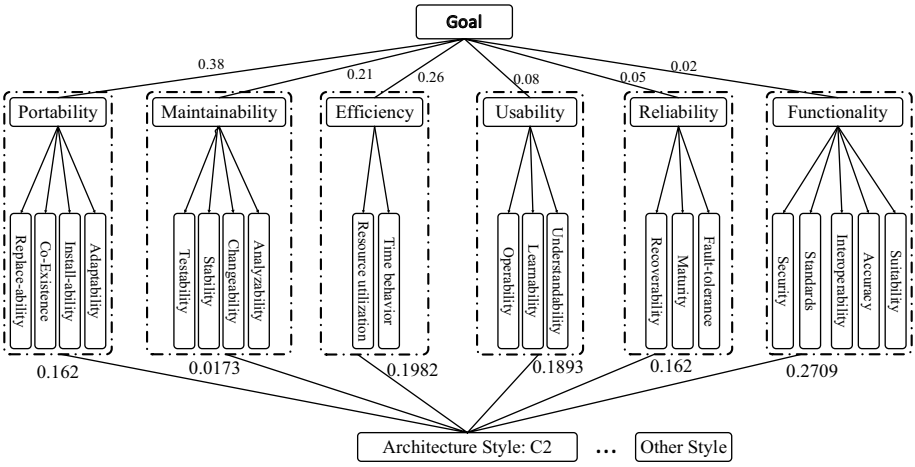

Fig. 6 The instance of tree structural and proportional weight

Table 3 shows the calculated score of all styles in this example. Therefore, based on this ranking, the most appropriate architectural styles in this special problem are Pipe and Filter, Client–Server, Layered, C2, Object Oriented, Blackboard and Storage styles, respectively.

Table 3 is calculated by the 18th formula which is shown below:

$$\begin{aligned} OP(C_2) = {} & 0.162 \times OP(Portability) + 0.1893 \times OP(Usability) \\ & + 0.162 \times OP(Reliability) + 0.1982 \times OP(Efficiency) \\ & + 0.0173 \times OP(Maintainability) + 0.271 \times OP(Functionality) \end{aligned}$$

As in the previous step, the Y_j is calculated for each quality attribute in the problem, Fig. 6 in this subsection helps clarify this step, which is taken from Fig. 3.

The value of other architectural styles is calculated in Table 3.

6.7 Weight and scoring

In the proposed method, we need to rank and compare styles and qualitative attributes and their sub-attributes. There are two general methods that could be used for this purpose: by using background and expert knowledge. The latter method utilizes questionnaires and methods to facilitate these issues and help to perform the work more accurate. We have used questionnaire forms and saved the results in a hierarchical format [32], which will be used as an example of this work.

In the following, Sect. 6.7.1 as mentioned before, we present a few examples of the forms (not all of them) which the required comparisons are derived from them. In fact, we provide some questionnaires (like the papers below) for all the quality attributes, its characteristics and for the comparison of both architectural styles.

Table 4 Correctness and its sub-attributes

	Relative importance of sub-criteria										According to correctness
	Absolute	Very strong	Fairly strong	Weak	Equal	Weak	Fairly strong	Very strong	Absolute		
Completeness										Traceability	1
Consistency										Traceability	2
Consistency										Completeness	3

Table 5 Usability and its sub-attributes

	Relative importance of sub-criteria to each other									Accord-ing to Usability
	Absolute	Very strong	Fairly strong	Weak	Equal	Weak	Fairly strong	Very strong	Absolute	
Training									Operability	1
Attractiveness									Operability	2
Attractiveness									Training	3

Table 6 Flexibility and its sub-attributes

	Relative importance of sub-criteria											According to flexibility
	Absolute	Very strong	Fairly strong	Weak	Weak	Equal	Weak	Fairly strong	Very strong	Absolute		
Expandability											Self containedness	1
Generality											Self containedness	2
Generality											Expandability	3

Table 7 The questionnaires for taking expert's opinions

Importance of styles with respect to each other										According to the sub-criteria	
Absolute	Very strong	Very strong	Fairly strong	Weak	Equal	Weak	Fairly strong	Very strong	Absolute	Sub-criteria of Security, flexibility, performance and … (only one sample comparison question for each sub-criteria is shown)	
C2										Broker	1
Object Oriented										Broker	2
Layered										Broker	3
Blackboard										Broker	4
Repository										Broker	5
Pipe & Filter										Broker	6
Object Oriented										C2	7
Layered										C2	8
Blackboard										C2	9
Repository										C2	10
Pipe & Filter										C2	11
Layered										Object Oriented	12
Blackboard										Object Oriented	13
Repository										Object Oriented	14
Pipe & Filter										Object Oriented	15
Blackboard										Layered	16
Repository										Layered	17
Pipe & Filter										Layered	18
Repository										Blackboard	19
Pipe & Filter										Blackboard	20

Table 7 (continued)

Importance of styles with respect to each other											According to the sub-criteria	
Absolute	Very strong	Very strong	Fairly strong	Weak	Equal	Weak	Fairly strong	Very strong	Absolute		Sub-criteria of Security, flexibility, performance and … (only one sample comparison question for each sub-criteria is shown)	
Pipe & Filter											Repository	21

Table 8 The questionnaires for taking expert's opinions

Number of people	Work experience (years)	Field of work	Academic level
5	5	Design & Analyze	BS
18	12	System Analyzer & architect	MS
7	20	Chief Architect	PhD

Table 9 Coefficients of each qualitative attribute

	Correctness	Usability	Flexibility
Broker	0.12	0.1089	0.1204
C2	0.1813	0.1166	0.0967
Object Oriented	0.1469	0.0979	0.1698
Layered	0.168	0.1562	0.0547
Blackboard	0.0709	0.1738	0.186
Repository	0.0709	0.1683	0.186
Pipe & Filter	0.2537	0.1639	0.184

6.7.1 Example of quality attribute sub-criteria questionnaires

The proposed method could be used in different models with respect to various quality attributes which leads us to give several samples (examples).

In Tables 4, 5, and 6, qualitative attributes like correctness, usability and flexibility are evaluated at the level of sub-attributes.

6.7.2 Example of questionnaires' style comparisons

In this subsection, a sample of common questionnaires of styles has been presented. In fact, in Table 7, the questionnaires have provided an implicit comparison between architectural styles based on architects' opinions and those of the system experts and in Table 8, the three categories of 30 experts (answered the questionnaires) are shown.

The results obtained by this method can help us find the required coefficients in various formulas and approaches. In addition, they can be used to extract reliable results for evaluating our output with regard to a specific problem. For example, for Broker, C2, Object Oriented, Layered, Blackboard, Repository and Pipe & Filter styles, qualitative attributes like Correctness, Usability and Flexibility in the scope of information systems (backup and restore) the following results (Table 9) were obtained after averaging the comments provided by 30 experts.

For using FAHP/AHP methods and also weighting each quality attribute related to the architectural styles, the different existing references could be helpful [1, 2, 4, 7, 9, 12, 19, 29, 44, 50]. The proposed methods divide the problem into small part for comparing and weighting but the user should be familiar with the architectural styles and their specification.

Table 10 Evaluation of the KPI's

KPI	Evaluation
Reusability	Both Conceptual reusability and structural are used in this approach, in a similar problem as conceptually this method has a suitable function and also the approach clearly provides reusability based on the hierarchical structure on its evaluation
Scalability	According to the quantitative method developed and used in the logical formulas, the proposed approach provides the possibility for evaluating many structures in a sequential or parallel manner
Understandability	This approach not only uses definitive methods and solves problems on a step-by-step basis, but also creates a white box that provides the ability of completion and high understanding in the procedure of architectural patterns evaluation
Usability	All activities can be used by any person who is familiar with this domain and using quantitative methods and their simplification such as using AHP and pairwise comparison for scoring provides for high usability
Completeness	The proposed approach is a comprehensive solution capable of covering all types of architecture, even heterogeneous and nested ones
Cost	No need for any implementation to learn new something. The cost of using this method in comparison with other quantitative and qualitative methods is significantly lower
Accuracy	Based on using readily reversible methods like cases, which are in relation to the AHP method, the correctness of the obtained outputs is guaranteed and included experts' opinions and existing cases in the literatures
Accountability	The proposed approach based on modularity, step-by-step process, separation capability and checking inputs and outputs has provided high accountability

Table 11 Comparison of the proposed method with other methods

KPI	Scenario-based	Cost-based	Formal method	Measuring method (simulation)
Reusability	–	I	I	I
Scalability	I	I	I	R
Understandability	–	I	I	–
Usability	I	I	I	I
Completeness	–	I	I	–
Cost	I	I	I	I
Accuracy	I	I	R	R
Accountability	I	I	R	–

7 Evaluation

Regarding the evaluation of the proposed methods in the previous step, we chose a real case (as a case study) and in this regard, the proposed method was described which its results can be considered as a reference to evaluate the proposed methods. Also, based on the existing methods in this context (mentioned in the related work), we examine the analysis of the proposed method.

Actually, to evaluate our method, we have examined several studies. As our method is based on an expert's opinion and questionnaires, and our goal is to analyze their influences.

In fact, our method is evaluated in different ways such as evaluation by experts, the Fuzzy-based AHP method and analytical evaluation based on previous studies.

7.1 Advantages of the proposed method

We discuss the advantages of the proposed method based on KPI in Table 10 and it is compared with the other methods from the literature. This comparison is based on eight quality attributes: *Reusability, Scalability, Understandability, Usability, Completeness, Cost, Accuracy, Accountability*. Tables 10 and 11 evaluate the proposed method from the perspective of each quality attribute, as presented in Hämäläinen [19], Wohlin et al. [48].

According to the related work section, we select four categories from architectural evaluation methods, which we review and compare with the proposed method. Table 11 presents the results of this comparison. "R" indicates that this method reduces the quality attribute and has a lower performance compared to other methods. "-" means the better method cannot be identified. "I" means that the proposed method improves the quality attribute.

As seen in Table 11, we survey the different viewpoint of the architectural evaluation method. In some of them there may be certain and customized cases which are difficult to compare against our method. In the comparison shown in Table 11, our method exhibits higher *reusability* compared to the other methods except for scenario-based methods; this is due to the fact that such methods, the scenario designer can plan it to be reusable, and this decision solely rests with the architect. As shown in the evaluation description table, for *Scalability*, our method provides sequential evaluation because of the quantitative capabilities and measurement methods based on a simulation that is easier to scale than our method because of already-existing simulation platforms and tools. The proposed method provides greater insight into the evaluation process and is better than both cost-based and formal methods. However, we definitively cannot compare our method to the scenario-based and measuring ones. According to Table 11, there is no need to know about simulators and formal hard methods or even determine the cost for different components. Therefore, our method is better than others. *Understandability* and *usability* in general, in this viewpoint, and by breaking the problem into small parts, the major advantages of our method is simplicity and *understandability* in which we do not use any tools, special language or specific knowledge, making it more useful than other procedures. Concerning *completeness*, since our approach fully depends on the designer and developer of the evaluation method, we cannot offer a definitive judgement about it when compared to scenario-based and measuring methods. However, in relation to the formal and Cost-based methods, we can take advantage of a wide range of issues. The *cost* associated with the proposed method for achieving acceptable results is lower than that of all the other. In fact, it does not include even the cost associated with the definition of specific scenarios and the expert merely makes a pairwise comparison. In terms of *accuracy,* it is clear that ours has lower accuracy than formal and measuring methods. Nevertheless, in regards to the use of hierarchical structure in the evaluation, in its category and compared to the other methods, the defects are identified and corrected at each step. As a result, this is the best method in its category. *Accountability* is one of the important features of it because it provides capabilities such as modularity, step-by-step process and input and output separation and checking. In spite of this, it is clear that our method has a lower performance than formal methods. As for the

measurement-based methods, since we do not consider readability and understandability here, no judgment can be made.

7.2 Threats of the proposed method

In general, any perspective in the context of software architecture evaluation comes with a certain set of pros and cons which should be considered. Our proposed approach, for weighting in the various domains has some risks such as increasing costs or decreasing accuracy (because of the involvement of human factor). In such situations, we should utilize some complementary methods to prevent these risks or reduce the probability of their occurrence and negative impacts.

The existing evaluation approaches related to the AHP questionnaires can be beneficial in preventing the inconsistencies in the answers. However, the lack of sufficient knowledge about weighting and the values of each quality attribute or architecture style can be considered as a potential risk. As we mentioned in the future works (refer to 8. Conclusion), in order to reduce the risks we can make use of a knowledge management system. It is worth mentioning that extracting and recording knowledge from architects would require more attention in order to prohibit misleading people and subjective judgments.

8 Conclusion

Selecting an appropriate software architecture is one of the key steps in any software development process, and satisfying quality requirements is one of the major challenges in this area. In addition, the relationships among these quality attributes must be studied when selecting the appropriate architecture from among a set of architectural styles or from among general candidate architectures. Hence, adding more information that must be analyzed and evaluated in the style selection process transforms the style selection issue into a multi-criteria problem. Prioritization uncertainty and the absence of quality metric evaluation standards are other problems in this context. To deal with these problems, a multi-criteria decision-making method based on fuzzy logic can be a useful solution for evaluating architectures and selecting the most suitable candidate. Therefore, this study proposes a fuzzy hierarchical analysis process (FHAP) in order to evaluate and compare candidate architectures. This process is a multi-criteria decision-making technique that resolves the aforementioned problems arising from the uncertainty of information using the fuzzy set theory. The proposed method ranks all options as well as providing a verifiable mathematical justification for this ranking.

Further work is needed to facilitate and improve the task of analyzing and evaluating software architectures, such as the design of Machine Learning mechanisms based on software reengineering activities. The proposed method could make this approach feasible by being integrated into decision support systems and prioritizing solutions based on information extracted and learned from previous cases. Therefore, by automating learning from domain information (i.e., a knowledge repository) instead of expert weighting and questionnaires, the method could generate better practical results to architects. In fact, the integration of the proposed method into the development tools and its use on the operational activities are also suitable future works.

References

1. Abowd G, Bass L, Clements P, Kazman R, Northrop L (1997) Recommended best industrial practice for software architecture evaluation (No. CMU/SEI-96-TR-025). Carnegie-Mellon University, Pittsburgh PA, Software Engineering Institute
2. Abrahão S, Insfran E (2017) Evaluating Software Architecture Evaluation Methods: An Internal Replication. In: Proceedings of the 21st international conference on evaluation and assessment in software engineering. ACM, pp 144–153
3. Babu KD, Govindaraju P, Reddy AR (2011) ANP-GP approach for selection of software architecture styles. Int J Softw Eng 1(5):91–104
4. Babu K, Rajulu PG, Reddy AR, Kumari AN (2010) Selection of architecture styles using analytic network process for the optimization of software architecture. arXiv preprint arXiv:1005.4271
5. Bass L, Clements P, Kazman R (2003) Software architecture in practice. Addison-Wesley Professional, New York
6. Bengtsson P (2002) Architecture-level modifiability analysis. Doctoral dissertation, Free University of Amsterdam
7. Binns P, Englehart M, Jackson M, Vestal S (1996) Domain-specific software architectures for guidance, navigation and control. Int J Softw Eng Knowl Eng 6(02):201–227
8. Chang DY (1996) Applications of the extent analysis method on fuzzy AHP. Eur J Oper Res 95(3):649–655
9. Chang CW, Wu CR, Lin HL (2009) Applying fuzzy hierarchy multiple attributes to construct an expert decision making process. Expert Syst Appl 36(4):7363–7368
10. Clements P, Garlan D, Bass L, Stafford J, Nord R, Ivers J, Little R (2002) Documenting software architectures: views and beyond. Pearson Education, London
11. Dasanayake, S., Markkula, J., Aaramaa, S., & Oivo, M. (2015, September). Software architecture decision-making practices and challenges: an industrial case study. In: 2015 24th Australasian software engineering conference (ASWEC). IEEE, pp. 88–97
12. Dhaya C, Zayaraz G (2012) Fuzzy based quantitative evaluation of architectures using architectural knowledge. Int J Adv Sci Technol 49:137–154
13. Dwivedi AK, Rath SK (2014) Selecting and formalizing an architectural style. In: 2014 seventh international conference on contemporary computing (IC3). IEEE, pp 364–369
14. Galster M, Eberlein A, Moussavi M (2010) Systematic selection of software architecture styles. IET Softw 4(5):349–360
15. Garlan D, Allen R, Ockerbloom J (1994) Exploiting style in architectural design environments. ACM SIGSOFT Softw Eng Notes 19(5):175–188
16. Garlan D, Monroe R, Wile D (2010) ACME: an architecture description interchange language. In: CASCON first decade high impact papers. IBM Corp., pp 159–173
17. Gorton I (2006) Essential software architecture. Springer, Berlin
18. Harker PT, Vargas LG (1987) The theory of ratio scale estimation: Saaty's analytic hierarchy process. Manag Sci 33(11):1383–1403
19. Hämäläinen N (2008) Evaluation and measurement in enterprise and software architecture management. Jyväskylä studies in computing 88
20. ISO/IEC, ISO/IEC CD 25010.3 (2009) Systems, software engineering - Software product Quality Requirements and Evaluation (SQuaRE) Software product quality, and system quality in use models, ISO
21. Kazman R, Bass L, Abowd G, Webb M (1994) SAAM: a method for analyzing the properties of software architectures. In: 16th international conference on software engineering, 1994. Proceedings. ICSE-16. IEEE, pp 81–90
22. Kazman R, Klein M, Clements P (2002) ATAM: method for architecture evaluation: ATAM—architecture trade-off analysis method report
23. Klein MH, Kazman R, Bass L, Carriere J, Barbacci M, Lipson H (1999) Attribute-based architecture styles. In: Software architecture. Springer Berlin, pp 225–243
24. Van Laarhoven PJM, Pedrycz W (1983) A fuzzy extension of Saaty's priority theory. Fuzzy Sets Syst 11(1–3):229–241
25. Lassing N (2002) Architecture-level modifiability analysis. Doctoral dissertation, Free University of Amsterdam
26. Luckham DC, Kenney JJ, Augustin LM, Vera J, Bryan D, Mann W (1995) Specification and analysis of system architecture using Rapide. IEEE Trans Softw Eng 21(4):336–354
27. Magee J, Dulay N, Eisenbach S, Kramer J (1995) Specifying distributed software architectures. In: Software engineering—ESEC'95, pp 137–153

28. Mahdavi-Hezavehi S, Durelli VH, Weyns D, Avgeriou P (2017) A systematic literature review on methods that handle multiple quality attributes in architecture-based self-adaptive systems. Inf Softw Technol 90:1–26
29. Maranzano J (1993) Best current practices: software architecture validation. AT&T Report
30. Medvidovic N, Oreizy P, Robbins JE, Taylor RN (1996) Using object-oriented typing to support architectural design in the C2 style. ACM SIGSOFT Softw Eng Notes 21(6):24–32
31. Medvidovic N, Taylor RN (2000) A classification and comparison framework for software architecture description languages. IEEE Trans Softw Eng 26(1):70–93
32. Moaven S, Habibi J, Alidoosti R, Mosaed AP (2015) Towards a knowledge based approach to style driven architecture design. Procedia Comput Sci 62:236–244
33. Moaven S, Habibi J, Ahmadi H, Kamandi A (2008) A decision support system for software architecture-style selection. In: Sixth international conference on software engineering research, management and applications, 2008. SERA'08. IEEE, pp 213–220
34. Moaven S, Habibi J, Ahmadi H, Kamandi A (2008) A fuzzy model for solving architecture styles selection multi-criteria problem. In: Second UKSIM European symposium on computer modeling and simulation, 2008. EMS'08. IEEE, pp 388–393
35. Moaven S, Kamandi A, Habibi J, Ahmadi H (2009) Toward a framework for evaluating heterogeneous architecture styles. In: First Asian conference on intelligent information and database systems, 2009. ACIIDS 2009. IEEE, pp 155–160
36. Montagud S, Abrahão S, Insfran E (2012) A systematic review of quality attributes and measures for software product lines. Softw Quality J 20(3–4):425–486
37. Nord RL, Barbacci MR, Clements P, Kazman R, Klein M (2003) Integrating the Architecture Tradeoff Analysis Method (ATAM) with the cost benefit analysis method (CBAM). Carnegie-Mellon Univ Pittsburgh Pa Software Engineering Inst
38. Paul C, Kazman R, Klein M (2002) Evaluating software architectures: methods and case studies. AddisonÇęWesley, Boston, MA
39. Saaty TL (1977) A scaling method for priorities in hierarchical structures. J Math Psychol 15(3):234–281
40. Saaty TL (1990) How to make a decision: the analytic hierarchy process. Euro J Oper Res 48(1):9–26
41. Saaty TL (1980) The analytic hierarchy process, paperback edition. RWS Publications, Pittsburgh (First appeared)
42. Schmidt D, Stal M, Rohnert H, Buschmann F (1996) Pattern-oriented software architecture, volume 1: a system of patterns
43. Shaw M, DeLine R, Klein DV, Ross TL, Young DM, Zelesnik G (1995) Abstractions for software architecture and tools to support them. IEEE Trans Softw Eng 21(4):314–335
44. Svahnberg M (2003) Supporting software architecture evolution Doctoral dissertation, Blekinge Institute of Technology
45. Tahmasebipour S, Babamir SM (2014) Ranking of common architectural styles based on availability, security and performance quality attributes. J Comput Secur 1(2):83–93
46. Tan C, Chen X (2010) Intuitionistic fuzzy Choquet integral operator for multi-criteria decision making. Expert Syst Appl 37(1):149–157
47. Thomas J (2002) Architecture assessment of information-system families. Doctoral dissertation, Eindhoven University of Technology
48. Wohlin C, Höst M, Henningsson K (2003) Empirical research methods in software engineering. In: Empirical methods and studies in software engineering. Springer, Berlin, pp 7–23
49. Zadeh LA (1965) Fuzzy sets. Inf Control 8(3):338–353
50. Zaki MZ, Jawawi DN, Hamdan NM, Halim SA, Mamat R, Mahat FS, Omar NA (2013) Multi-criteria architecture style selection for precision farming software product lines using fuzzy AHP. Int J Adv Soft Comput Appl 5(3):85

Shahrouz Moaven is a Ph.D. candidate at Sharif University of Technology. He received his BS degree in Computer Engineering in 2005 and MS degree in Software Engineering at Sharif University of Technology, Tehran, Iran in 2008. His research interests include software architecture, software engineering, decision support system and business intelligent. He can be contacted at Performance Evaluation Software Engineering Lab, No. 601, Computer Engineering Department, Sharif University of Technology.

Jafar Habibi received Ph.D. degree in computer science from University of Manchester in 1998. He is currently an associate professor and chairman of Computer Engineering Department at the Sharif University of Technology, where he has been a faculty member since 1989. His research interests include software engineering, software architecture and design, performance evaluation, system analysis and design, information systems and simulation. He can be contacted at No. 626, Department of Computer Engineering, Sharif University of Technology.