Estimating Redevelopment Costs for High-Risk Components Using a Genetic Algorithm

Zeinab Faraji[1], Faraein Aeini[2*], Homayun Motameni[3]

Abstract–Component selection is a key challenge in component-based software systems, particularly when integrating a mix of commercially available off-the-shelf (COTS) and in-house components. This study emphasizes the impact of high-risk COTS components, which often require extensive modification to align with system requirements, potentially affecting cost, efficiency, and performance. To address this, high-risk components are initially identified based on reliability, and a genetic algorithm is then employed to compute a cost coefficient for their redevelopment. The proposed method led to improved outcomes in component selection, as demonstrated in a case study that reported increases in Intra-modular Coupling Density (ICD) by up to 0.05 and functionality gains of up to 1.08—validating the effectiveness of this cost-aware optimization strategy.

Keywords: Component-Based Software Engineering (CBSE); Component Selection; High-Risk Components; Genetic Coefficient; Software Cost Development

1. Introduction

This article gives you guidelines for preparing papers Component-based software engineering (CBSE) is a prevalent approach to software development, utilizing pre-existing software components. These components are largely independent and replaceable, designed to perform specific tasks within well-defined contexts. Developers employ CBSE to mitigate software complexities, streamline change management, and enhance software reusability [1,2,3]. Reusable software components are generally classified into four categories [4,5]. The first category comprises ready-made components, also known as commercially available off-the-shelf (COTS) components. These are obtained from third-party sources or are integrated into existing projects and can be used in the software development process without modification. The second category consists of complete components, which closely resemble new project components in terms of specifications, design, code, and testing. Development teams are typically familiar with these components, considering them low-risk. The third category includes incomplete components, or high-risk components, which share similarities with a current project but require substantial adjustments. Due to limited team expertise, these components are deemed high-risk. The fourth category is new components, which must be developed by the team to fulfill specific project requirements. COTS components encompass ready-made, complete, and incomplete components, with incomplete components classified as high-risk. High-risk components are often selected for their initial low cost; however, they can exhibit low reliability and negatively impact quality attributes such as Intra-modular Coupling Density (ICD) and cohesion, ultimately increasing system complexity. Therefore, effective component selection necessitates identifying component types and accurately estimating the redevelopment cost of high-risk components. This paper presents a method using a genetic algorithm to calculate the development cost factor for high-risk components. The results demonstrate the efficacy of this optimization and calculation approach in determining the software redevelopment cost factor for component selection. The primary objective of this study is to differentiate component selection and ordering strategies to enhance system performance, minimize the inclusion of high-risk components, and reduce redevelopment costs. Following the recommendation in [3], we optimized ICD and a performance function as objective functions, subject to constraints related to cost, delivery time, reliability, and ICD. This research investigates the following hypotheses:

1. Selecting fewer high-risk components results in a more optimal set of components.
2. Employing a genetic optimization approach leads to

1 Department of Computer Engineering, sari Branch, Islamic Azad University, sari, Iran. Email:zeinab.faraji84@gmail.com

2* Corresponding Author :Department of Computer Engineering, sari Branch, Islamic Azad University, sari, Iran. Email: aeini@ iausari.ac.ir

3 Department of Computer Engineering, sari Branch, Islamic Azad University, sari, Iran. Email: motameni@iausari.ac.ir

Received: 2025.05.08; Accepted: 2025.07.16

a more precise evaluation of the redevelopment effort required for COTS components.

While genetic algorithms are frequently used for component selection, their potential for optimizing objective functions and constraints is often underutilized. The method proposed in this paper leverages a genetic algorithm to determine the optimal

coefficients for the cost constraint function, owing to its robust output and mathematical foundation. The data collection for this study includes components from various device versions and internal parts assigned to different software modules. Table 1 show A summary of the methods presented by researchers in optimal components selection of software and figure 1 is showing pie chart of distribution of methods optimization, cost criteria and optimization types adopt to table 1.

The remainder of this paper is organized as follows: Section 2 reviews related work on optimal component selection, with a focus on cost considerations. Section 3 details the proposed approach. Section 4 presents the optimization problem within the case study. Section 5 presents the experimental results of the case study. Section 6 compares the results with other methods from the literature. Finally, Section 7 concludes the paper and outlines directions for future research.

Table 1: A summary of the methods presented by researchers in optimal components selection of software

Optimization type	*Optimization method*	*Cost criteria (objective/constraint)*
Single objective	Hierarchical clustering algorithms	Cost/budget in constraint
Single objective	AHP based on the access frequencies of the modules	Cost/budget in constraint
Single objective	fuzzy optimization model	Cost/budget in constraint
Single objective	optimization model based on decision variables	Cost in objective function
Multi objective	fuzzy mathematical programming	Cost in objective function
Multi objective	goal programming approach	Cost/budget in objective and constraint
Multi objective	Multi-criteria optimization approach in fuzzy environment	Cost in objective function
Multi objective	fuzzy optimization model	Cost/budget in constraint
Multi objective	fuzzy optimization model	Cost in objective constraint
Multi objective	fuzzy optimization model	Cost/budget in constraint
Multi objective	genetic algorithm (GA)-based hybrid approach with fuzzy exponential membership function	Cost/budget in constraint

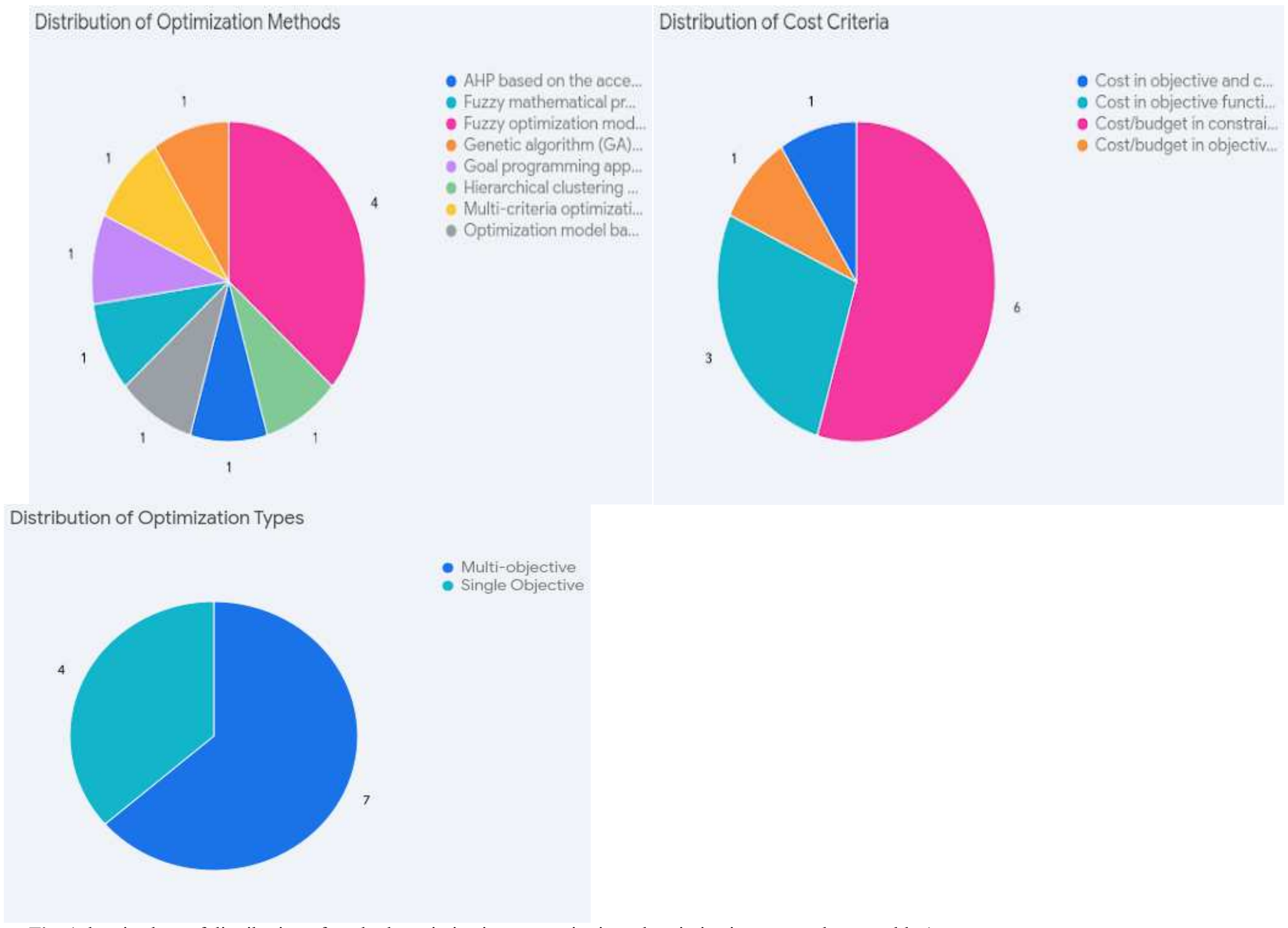

Fig. 1.the pie chart of distribution of methods optimization, cost criteria and optimization types adopt to table 1.

2. Related work

In research focused on selecting the best software components, cost frequently plays a key role as either a goal to minimize or a limitation within the optimization process. Table 1 provides an overview of how researchers have incorporated cost into their models. Table 2 presents statistical insights derived from analyzing approximately 40 common factors considered during software component selection for optimization [6]. As highlighted in Table 1, cost is the most prevalent factor in optimal component selection, appearing in 17.70 of the 130 studies that considered 41 different criteria. The data in Table 2 suggests that cost is a critical factor when deciding whether to acquire or develop components due to its impact on software development expenses. Consequently, software development organizations need to account for cost when choosing software components, potentially even making it the primary

goal of their optimization efforts for ideal selection. Various optimization techniques, including multi-objective, mathematical, and genetic algorithms, have been widely employed for optimal component selection [7,8,9,10,11,12,16]. Numerous researchers have reported successful outcomes using different heuristic methods, multi-objective optimization strategies, and genetic algorithms. Figure 2 is show Statistical results of the cost criterion according to the table 2.

Table 2: Statistical results on the practical use of the cost criterion in the problem of optimal selection of components

Criteria	*NP*	*MPP*	*Ref*
Cost	23	17.7	[17, 4, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28 ,29, 30, 31, 32, 33, 14, 34, 37, 38, 39, 41]
Reliability	12	9.23	[32, 4, 20, 33, 14, 34, 23, 26, 35, 36, 37, 31]
Number of components	9	6.92	[17, 18, 15, 21,23,25, 33, 34,35]
Delivery time	6	4.60	[1, 14, 20, 23, 30, 31]
Functionality	6	4.60	[1, 18, 21, 23, 30, 34]
Number of modules in the software	5	3.85	[29,14 ,31, 32,36]
Number of provide interfaces	5	3.85	[15,18 ,33, 37,38]
Number of alternative COTS available	5	3.85	[29,31 ,36, 32,38]
Coupling	3	2.30	[1 , 24,25]
Other	56	43.10	[16, 21, 33, 18, 15, 33, 30, 31, 20, 14, 30, 31 , 14, 19, 28, 1, 14, 30, 31, 29, 36, 31, 35, 22 ,35, 17]
	130	100	

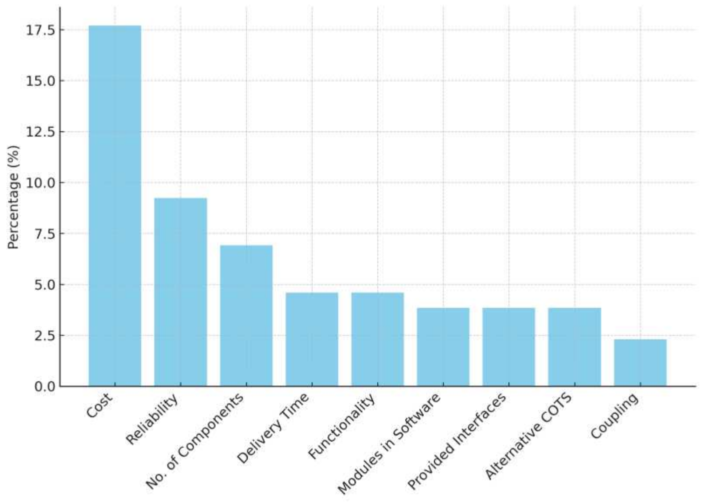

Fig. 2. Statistical results of the cost criterion according to the table 2.

Gholamshahi et al. [20] categorized different approaches to component identification and selection. In [11], the authors introduced a Genetic Algorithm (GA)-based hybrid approach with a fuzzy exponential membership function to select the best COTS component. The researchers in [42] used genetic algorithms and fuzzy techniques to create an optimization model for component selection in credit models. They solved the problem using a single-objective optimization method with three objective functions: cost, size, and execution time. The optimal component was determined using GA. In [44], the authors developed a model to optimize the selection of COTS components in the process of developing modular software systems. In more recent studies, researchers have increasingly focused on multi-period, multi-objective optimization frameworks that combine cost with other real-world constraints such as outsourcing, redundancy, and integration [48, 49]. These newer models reflect a growing trend toward holistic evaluation strategies that account for dynamic and time-dependent factors in component selection.

For instance, Gupta et al. (2019) proposed a framework that integrates customer relationships and outsourcing dynamics into component evaluation decisions [49]. Mehlawat et al. (2020) extended this approach by addressing software maintenance and enhancement through an adaptive multi-objective model, taking into account long-term integration costs and performance metrics [48].

Additionally, Nabot (2024) presented an optimized component selection criterion specifically tailored to modern CBSE environments. This recent study introduced refined metrics for cost, cohesion, and risk that align closely with commercial software development constraints [16].Table 3 presents the statistical results obtained by analyzing 35 articles about the best way to select components. These results reveal a variety of methods proposed in the literature for solving optimization problems. Among all, the evolutionary method using the genetic algorithm was found the most popular. All of these methods provided a solution in the form of a single-objective system, but the genetic algorithm had a positive impact on optimizing objective functions. This can lead to better results when selecting the optimal components for both single and multi-objective problems. Figure 3 illustrates Frequency chart methods that applied to solving optimal component selection problem.

Table 3. Statistical results on applied methods in the problem of optimal selection of components

Method	*NP*	*MPP*	*Reference*
Evolutionary MOO based on GA	7	20	[18,15 ,21,22,33, 34,39]
Fuzzy mathematical programming (FMP)	3	8.571	[14,30,31]
Customized MOO (used LINGO optimization model solver)	2	5.714	[21,41]
Goal programming	2	5.714	[36,27]
Both AHP and WSM	2	5.714	[19,45]
Fuzzy clustering	2	5.714	[35,24]
Integer Programming	2	5.714	[29,40]
Customized GA	1	2.858	[17]
Fuzzy MOO	1	2.858	[1]
Other methods (Lexicographic, Lagrange, MCDM, Evolutionary and WSM, AHP, WSM, SIREN method, Neural network, C4.5, Ontology-based, XML Query, backtracking algorithm	13	37.143	[23, 46, 25, 26, 28 , 32, 37, 38, 44, 45, 47 ,42, 43]
Total	35	100	

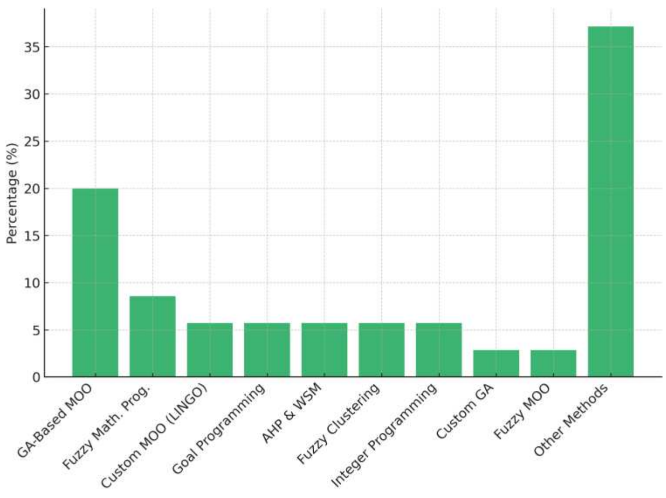

Fig. 3. Frequency chart methods to solving optimal component selection problem

3. Proposed approach: calculating the cost coefficients for the redevelopment of high-risk components using a genetic algorithm

Cost is a vital consideration for management and business in software development. Since it directly impacts their budgets, software companies must carefully consider cost when choosing software components. Cost can be factored into their decision-making as either a goal to minimize or a limitation they must stay within. In software systems built from components, overlooking compatibility, miscalculating extra costs, selecting unsuitable components, or choosing high-risk options can increase system vulnerabilities and threaten the stability of the underlying technology. Some inexpensive components available may also carry high risks due to a lack of extensive use and documented history. These might not integrate well and could require replacement to improve performance, ultimately driving up the system's cost. Therefore, a method to estimate the additional development costs associated with such components is necessary. After acquiring them, these components might need further development to function within the intended system, which increases the likelihood of selecting high-risk components when using optimization techniques. This can negatively affect the system's reliability and other related goals. To accurately determine the true cost of these components, this paper outlines a calculation method that considers both their initial purchase price and potential future development expenses. This will aid in selecting the most cost-effective option and improve the overall component selection process. The proposed technique employs genetic multi-objective optimization to estimate the cost of redeveloping components, specifically considering the impact of high-risk components on the total cost. Overall, this approach aims to provide a more precise cost calculation for component redevelopment, particularly for high-risk components, and enhance the system's reliability and other performance metrics. This paper proposes using cost coefficients and a genetic algorithm to more accurately assess the cost of high-risk components. Figure 4 illustrates the proposed approach, including the method for calculating the genetic cost of component redevelopment within a multi-objective optimization framework.

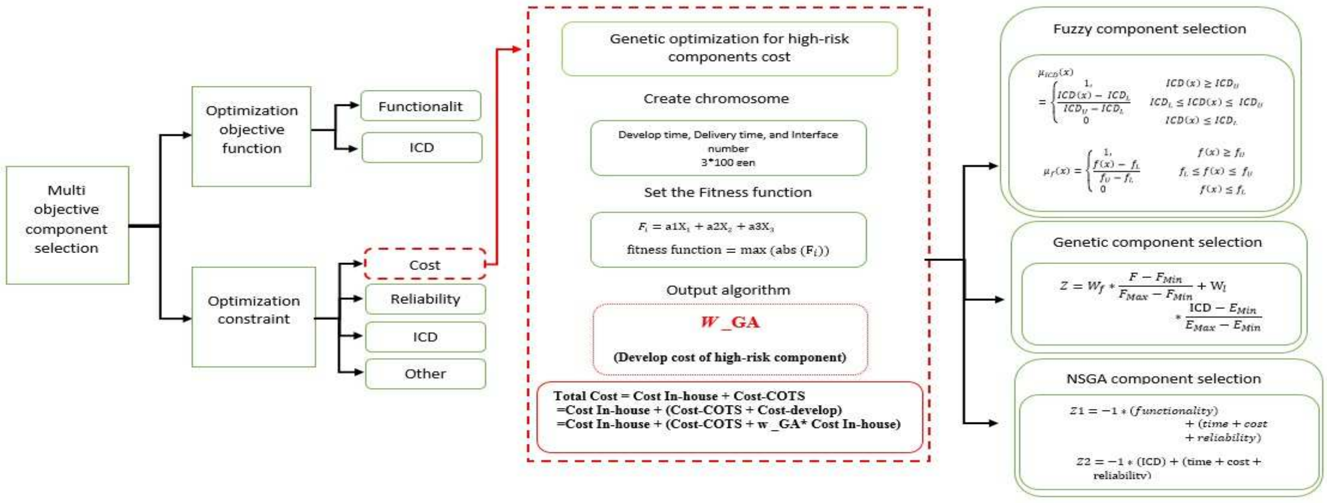
Multi objective component selection
Optimization objective function
Functionalit
ICD
Optimization constraint
Cost
Reliability
ICD
Other
Genetic optimization for high-risk components cost
Create chromosome
Develop time, Delivery time, and Interface number
3*100 een
Set the Fitness function
$F_i = a1X_1 + a2X_2 + a3X_3$
fitness function = max (abs (F_i))
Output algorithm
W_GA
(Develop cost of high-risk component)
Total Cost = Cost In-house + Cost-COTS
=Cost In-house + (Cost-COTS + Cost-develop)
=Cost In-house + (Cost-COTS + w_GA* Cost In-house)
Fuzzy component selection
$\mu_{ICD}(x) = \begin{cases} 1, & ICD(x) \geq ICD_U \\ \frac{ICD(x) - ICD_L}{ICD_U - ICD_L} & ICD_L \leq ICD(x) \leq ICD_U \\ 0 & ICD(x) \leq ICD_L \end{cases}$
$\mu_f(x) = \begin{cases} 1, & f(x) \geq f_U \\ \frac{f(x) - f_L}{f_U - f_L} & f_L \leq f(x) \leq f_U \\ 0 & f(x) \leq f_L \end{cases}$
Genetic component selection
$Z = W_f * \frac{F - F_{Min}}{F_{Max} - F_{Min}} + W_I * \frac{ICD - E_{Min}}{E_{Max} - E_{Min}}$
NSGA component selection
$Z1 = -1 * (functionality) + (time + cost + reliability)$
$Z2 = -1 * (ICD) + (time + cost + reliability)$

Fig. 4. The Schematic of the proposed approach with the approach of calculating the cost of genetics for the redevelopment of components in multi objective optimization.

3.1 Calculating the cost of components redevelopment

Component-based systems consist of two categories of components: build and buy. Choosing to use readily available COTS components from the market can shorten the system's delivery time, but it may also compromise security and reliability because of compatibility concerns. On the other hand, designing in-house components may take longer and cost more, but it results in a more reliable and compatible system. When dealing with multi objective systems and constraints, it is important to choose the appropriate component that can achieve the system's goals. When calculating costs, most references consider three factors. First, in [1,4] only COTS components are used, and the total cost is simply the cost of purchasing these components. This cost is determined by Equation 1, where $\mathrm{Cost}_{\mathrm{COTS}}$ equals the sum of all costs (k) of each component.

$$Cost_{COTS} = \sum_{k=1}^{vij} c_{ijk}\, x_{ijk} \qquad (1)$$

Second, in [1, 4], only components produced by the development team are used. The total cost is determined by Equation 2, where $\mathrm{Cost}_{\mathrm{inhouse}}$ equals the sum of all costs (i) of each component (j):

$$\mathrm{Cost}_{\mathrm{inhouse}} = \sum_{i=1}^{n}\sum_{j=1}^{m}(\tilde{C}_{ij}(t_{ij} + \tau_{ij}N_{ij}^{tot})y_{ij} \qquad (2)$$

Third, in [1,2,4], a combination of in-house and COTS components are used. This is determined by Equation 3

$$\mathrm{Cost}_{\mathrm{Total}} = \mathrm{Cost}_{\mathrm{cost}} + \mathrm{Cost}_{\mathrm{in\_house}} \qquad (3)$$

The final total cost is obtained by combining the buy-build expenditures of all modules in the build-or-buy strategy, as shown in Equation 4.

$$\text{Total cost} = \sum_{i=1}^{n}\sum_{j=1}^{m}(\tilde{C}_{ij}(t_{ij} + \tau_{ij}N_{ij}^{tot})y_{ij} + \sum_{k=1}^{vij} c_{ijk}\, x_{ijk} \qquad (4)$$

In Equation 4, the total cost is equal to the sum of all costs (i) of each component (j) and its manufacturing time (tij) plus any additional cost (τ(ij)) and quantity produced (Nijtot) multiplied by (yij)), plus the sum of all costs (k) of each COTS component.

3.2 Proposed approach: Calculating the cost of redevelopment of components based on the optimal calculation of coefficients with the genetic algorithm

As previously stated, certain COTS components are not compatible with software systems and are considered high-risk components. To make them compatible, changes and redevelopment are necessary. In the case study performed for the purposes of this study, low-cost and unreliable components were classified as high-risk components because of their affordability and cost constraints. However, selecting these components may reduce the reliability of the system and, specifically, ICD, which is directly related to reliability. Therefore, the selection of components needs to be evaluated based on their impacts on the objectives and constraints of the problem. To determine the ideal components that meet the goals of the problem, there is a need for considering the influencing factors of other criteria, whether they have a direct or indirect effect. Equation 5 is used to calculate the cost of a system, which includes the cost of high-risk components, including any costs associated with developing or adapting the component. Therefore, the cost of the bought component should be calculated using the following Equations:

Total Cost = Cost In-house + Cost-COTS =
Cost In-house + (Cost-COTS + Cost-develop)=
Cost In-house + (Cost-COTS + w _GA Cost In-house)* (5)

The cost of the component is calculated using Equation5, which is derived from Equations 3 and 4 and then using Equation6, which is derived from Equation 5.

$$\text{Total cost} = \sum_{i=1}^{n}\sum_{j=1}^{m}(\tilde{C}_{ij}(t_{ij} + \tau_{ij}N_{ij}^{tot})y_{ij} + \sum_{k=1}^{vij} c_{ijk}\,x_{ijk} + W_{GA} * (\tilde{C}_{ij}(t_{ij} + \tau_{ij}N_{ij}^{tot})y_{ij}) \quad (6)$$

This study analyzed three factors that affect the cost of software development: development time, delivery time, and the number of interactive interfaces. The goal was to determine the coefficient (W) that represents the development cost of high-risk components. The amount of time spent on development equals the time spent on developing bought components (COTS). The number of interfaces is equal to the number of connections between high-risk components and other modules, and the cost of redevelopment increases as these connections increase. Delivery time is the sum of the development and buy times of the components. We used three input parameters as the chromosomes' data structure and W, as the output's development cost coefficient in the genetic algorithm. The findings showed that the proposed approach is effective in selecting a combination of elements that improve the objectives of the problem.

3.3. Genetic parameters

To find the weighting coefficients of high-risk component costs, this paper used GA with the following features:

Chromosome allocation: To determine the weighting coefficient for high-risk component costs, data structures should be created for the chromosomes that represent the parameters affecting the component's cost. To do this, this study used an array with dimensions of m * n, where m represents the initial population of the algorithm and n represents the factors that affect software development costs. Because of the presence of three influencing parameters in this case (Develop time, Delivery time, and Number of interact), an array of 3 * 50 was designed. Finally, this variable was analyzed using the principle of maximization. Equation 7 is also defined for the cost influencing parameters variable as follows:

$$F_i = a1Y_1 + a2Y_2 + a3Y_3 \quad (7)$$

$$a1 + a2 + a3 = 1 \quad (8)$$

Where Yi denotes a component of the cost chromosome, and it is necessary to satisfy Equation 8 as well. To fulfill Equation 8, the average value of each row of chromosomes was calculated at every phase of the execution using Equation 9. Then, the number of chromosomes was replaced with this average value to normalize the random amounts of the chromosomes.

$$Y_i = \frac{Y_i}{\sum_{i=1}^{i=3} Y_i} \quad (9)$$

Fitness function: During each stage of the genetic algorithm, the cost of every chromosome is calculated based on the target functions defined in Equation 10.

$$W = \max(\mathrm{abs}(-F_i)) \quad (10)$$

At this stage, the chromosomes undergo evaluation through the cost function. In the proposed approach, the cost is a variable inversely proportional to the fitness function of the chromosomes. The presence of negativity in Equation 10 is a result of the potential for maximizing the function.

Selection: The selection operator improves the overall quality of chromosomes in the next generation by selecting the high-quality ones. This paper uses the tournament selection method [24, 25] because it is efficient and easy to implement.

Crossover: During the crossover operation, parts of the chromosomes are accidentally exchanged. This exchange results in children inheriting high-quality traits from either parent, which may lead to higher levels of confidence compared to both parents. To calculate the cost weight coefficient, a uniform composition for the intersection is used, which is called a comprehensive recombination intersection. In this approach, the chromosome points are selected as the intersection points. The procedure involves generating a random number between zero and one for each part of the chromosome. If this number is less than a constant value such as α, then the genes located after that point on the chromosomes will be displaced. The intersection rate in this paper is 1 [7].The comprehensive recombination intersection method was chosen due to its ability to explore a larger portion of the solution space, which increases diversity and helps avoid premature convergence. It is particularly effective in problems involving multiple influencing parameters, such as the cost-related coefficients in our model.

Mutation: To ensure diversity in the population after the intersection, a mutation threshold is established to calculate the best coefficients. The goal is to generate a random number between zero and one for every chromosome. If the number is less than a certain limit, a random number is generated for each gene, and the corresponding gene is mutated if the product number is smaller than a constant value β. The default mutation rate is 0.08, as mentioned in [7].

Termination criteria: Once the algorithm completes a certain number of iterations, it offers the optimal solution as an output. In this study, the number of repetitions is set to 100 [7].

Algorithm1: Initialize Population with Cost-Related Chromosomes

```
Input:
   - Population size (P)
   - Number of genes per chromosome (n = 3)
   - Range of values for each gene:
   DevTime ∈ [t_min, t_max]
   DelTime ∈ [d_min, d_max]
      Interfaces ∈ [i_min, i_max]
Output:
   - Initialized population of chromosomes: Pop[P][3]
Begin
   For i = 1 to P do
      // Generate one chromosome with 3 genes
      Chromosome[i][1] ← Random value in [t_min, t_max]   // Development Time
      Chromosome[i][2] ← Random value in [d_min, d_max]   // Delivery Time
      Chromosome[i][3] ← Random value in [i_min, i_max]   // Number of Interfaces
      // Normalize genes (optional step)
    For j = 1 to 3 do
       Chromosome[i][j] ← Normalize(Chromosome[i][j])
      End For
      Pop[i] ← Chromosome[i]
   End For
   Return Pop
End
```

3.4. Application of proposed approach to a case study

This research conducted a detailed analysis to investigate the way high-risk components affect costs and to determine how accurately the suggested approach estimates costs, especially when high-risk components are involved [1, 4]. The cost function was considered a constraint in the optimization problem of the case study, and this paper aimed to demonstrate the positive role of calculating cost coefficients in achieving the objectives of the problem.

3.5. Case study and dataset

The present research used a case study of financial and accounting software [2] to evaluate the effectiveness of the proposed approach, in selecting the optimal combination of COTS and in-house components for CBSS development. The case study involved a software system consisting of three modules, M1, M2, and M3. We had access to 20 software components in the market, numbered SC1 to SC20, which could be used to create a set of ten components, S1 to S10. Ten components could additionally be developed in-house which were labeled SB1 to SB10. For each software module, one component was needed to be selected from the alternatives available to meet operational needs. For example, S1 could be made up of SC1, SC2

, SC3, SC4, or SB1. Therefore, we had to choose one of these five components to satisfy the operational requirements of S1. SC1, SC2, SC3, and SC4 were COTS components, while SB1 was developed in-house. A more detailed description of the case study and dataset is presented in [1, 2]. In addition, the tables related to the data of this case study are given in the appendices section.

4. Optimization problem

As mentioned earlier, a case study of financial and accounting software [2] was used to assess proposed approach effectiveness in the selection of the optimal component of COTS or in-house components for CBSS development. This case study is based on a case study that considers the goals of the multi-objective optimization model, such as ICD and functionality, as well as constraints like cost, delivery time, reliability, and ICD.

4.1 Objective functions and constraints

When dealing with a software system made of various components, the levels of their individual performance should be assessed. The components can either be bought from vendors or created in-house, and their functional performance can be tailored to meet the unique requirements of the organization or client.

First objective function – Functionality: The overall functional performance of the system depends on the functionality of each module, which is calculated using Equation 11 [1,2]. In essence, functionality is a crucial measure that determines the efficacy of a modular software system.

$$F = \sum_{j=1}^{M} \sum_{i=1}^{N} (f_{ij} y_{ij} + \sum_{k=1}^{v_{ij}} f_{ijk} x_{ijk}) \qquad (11)$$

Second objective function – Intra-modular coupling density (ICD):

The intra coupling density is a measure of how coupling and cohesion of the modules are in a modular software system. Each module in a software system has multiple connections to other modules, and the ICD0s objective function is determined using Equation 12 based on both coupling and cohesion [1,2]. While Cohesion is the number of component interactions in the module, Coupling is the number of interactions between components in separate modules [1].

$$\mathrm{ICD} = \frac{\mathrm{Cohesion}}{\mathrm{Cuopling} + \mathrm{Cohesion}} \qquad (12)$$

First constraints – Threshold on ICD: This constraint expresses the minimum threshold H on the value of ICD for each module using Equation 13:

$$\mathrm{ICD} = \frac{\mathrm{CI_{in}} = \sum_{j=1}^{M}\sum_{i=1}^{N-1}\sum_{i'=i+1}^{N} r_{ii'} z_{ij} z_{i'j}}{\sum_{i=1}^{N-1}\sum_{i'=i+1}^{N} r_{ii'} \left(\sum_{i'=i+1}^{N} z_{ij}\right)\left(\sum_{j=1}^{M} z_{i'j'}\right)} ,$$

$$0 \le \mathrm{ICD} \le 1 \qquad (13)$$

Second constraints – Building decision versus buying decision: Developers can choose from multiple instances of COTS and one in-house example for each component, with a calculation of build and buy decisions included Equation 14.

$$y_{ij} + \sum_{k=2}^{v_{ij}} x_{ijk} = z_{ij}; \quad i = 1,2,\ldots,n \ j = 1,2,\ldots,m_i \qquad (14)$$

Third constraints – Budget constraint: One of the most crucial considerations in system design is the cost constraint. It is determined by adding up all the costs associated with various modules of the build-or-buy strategy using Equation 15.

$$\sum_{i=1}^{n}\sum_{j=1}^{m}\left(\tilde{C}_{ij}\left(t_{ij} + \tau_{ij} N_{ij}^{tot}\right) y_{ij} + \sum_{k=1}^{vij} \tilde{C}_{ijk}\, \tilde{x}_{ijk}\right) \le B$$
(15)

Fourth constraints – Delivery time constraint: When referring to a software component, the delivery time indicates the duration required to prepare and make the component available for use in a component-based software system. This includes the time taken for development, integration, and system testing. Commercially available components are represented by dij, while components developed in-house can be expressed as the delivery time of the ith component for the jth module using Equation 16.

Table 4.cost coefficient for the high-risk components calculation by GA

component	High-risk Cost weight																				
	8	9	12	13	14	19	21	22	23	24	33	38	41	43	45	46	48	49	50	54	60
Cost weight	0.9624	0.1768	0.122	0.948	0.2174	0.3125	0.2476	0.5963	0.7288	0.1606	0.5795	0.6187	0.8869	0.5905	0.1648	0.9055	0.9025	0.0896	0.9614	0.8533	0.7800

$$T_i = \left(C_{ij}\left(t_{ij} + \tau_{ij} N_{ij}^{tot}\right) y_{ij} + \sum_{k=1}^{Vij} d_{ijk} x_{ijk} \right) \qquad (16)$$

Fifth constraints – Threshold on the reliability constraint: To calculate the likelihood of the ith component developed in-house for the jth module failing, we can use Formula 1. In addition, by applying Equation 17, we can determine the average number of failures for both the ith component and the jth module, which is represented as qij.

$$q_{ij} = \left(1 - \rho_{ij}\right) y_{ij} + \sum_{k=1}^{V_{ij}} \mu_{ij}\, x_{ij} \qquad (17)$$

4.2. Results of the optimal components selection

The case study identified the components categorized as high-risk and low-risk, which included the following:

High-risk components: 8,9,12,13,14,19, 21, 22, 23, 24, 33, 38,41, 43, 45, 46, 48, 49, 50, 54, 60

Low-risk components: 1, 2, 3, 4, 5, 6, 7, 10, 11, 15, 16, 17, 18

The case study used low-cost and compatible components to identify high-risk components. By using Equation 18, the selection of such components during the optimum component selection process led to a lower ICD, which was owing to their reliability, cohesion, and affordability. The aim of this approach is to find components with low risk and high ICD (low cost).

Therefore, the key issue in selecting components with the ICD objective function is to identify low-risk components with high ICD.

The redevelopment process is more time-consuming and expensive because high-risk components need modification to work with the intended software system. The suggested approach to calculate the cost of redevelopment involves the calculation of cost coefficients. Using the genetic algorithm with the settings mentioned in the previous section, the cost coefficients for high-risk components were determined in the case study. Table 4 shows the values of these coefficients.

$$\text{Cost} \propto \text{Reliability} \propto \text{cohesion} \propto \text{ICD} \quad (18)$$

So: Min number of high-risk component selected $\propto$ max ICD (19)

5. The experimental results

To select the best components, there are two main methods: single-objective and multi-objective. Single-objective methods, which often use genetic algorithms or fuzzy methods, focus on changing the objective functions to create a single goal. On the other hand, multi-objective methods, which use the NSGA method, keep the objective functions the same. To ensure the effectiveness of proposed approach in optimal component selection, both multi-objective and single-objective methods were tested in this paper. Despite a shortage of case studies in this research area, this paper managed to compare GA-ICD to two other methods. To compare the effectiveness of the proposed approach with other methods, we implemented our own proposed approach using three methods that other authors used in selecting the optimal component and compared the results of this implementation with the results of others' methods. Also, to create a comparative standard environment, in addition to the proposed approach, we also implemented and implemented normal conditions (other than the approach) with these methods and presented the results in the form of relevant tables.

5.1. Multi-objective optimal component selection by fuzzy logic

To solve the optimal component selection problem as explained in [1,2], the problem was formulated based on the Belman-Zadeh maximization principle. First, the objective functions and constraints were calculated using the proposed approach, for high-risk cost. Next, the values of X1 and X2 were calculated using the two objectives in the problem. Afterward, the Functionality and ICD objective function optimization problem was solved (see the results in Tables 5 and 6). Then, the optimal lower limit (l) and upper limit (u), as well as the worst-case scenarios for each, were determined by evaluating the functionality and ICD while taking into account the constraints of the optimization problem obtained. The number of restrictions considered is based on the information provided in Table 7.

The membership functions for ICD and functionality in the case study are equation as 20 and 21, respectively.

Table 5. Result of solving the optimization problem with the objective of functionality in the case study

Functionality	*ICD*	*Time*	*Budget*
5.40	**0.8809**	**6s**	**25**

Table 6. Result of solving the optimization problem with the objective of ICD in the case study

Functionality	*ICD*	*Time*	*Budget*
7.82	**0.4940**	**6s**	**24**

Table7. Optimization solution set of the high-risk approach for the first and second objective function

	X1	*X2*
ICD	**0.8809**	**0.4940**
Functionality	**5.40**	**7.82**

$$\mu_{ICD}(x) = \begin{cases} 1, & ICD(x) \geq 0.8809 \\ \dfrac{ICD(x) - 0.4940}{0.3869}, & 0.4940 \leq ICD(x) \leq 0.8809 \\ 0, & ICD(x) \leq 0.4940 \end{cases} \quad (20)$$

$$\mu_f(x) = \begin{cases} 1, & f(x) \geq 7.82 \\ \dfrac{f(x) - 5.40}{2.42}, & 5.40 \leq f(x) \leq 7.82 \\ 0, & f(x) \leq 5.40 \end{cases} \quad (21)$$

The fuzzy multi-objective optimization model for high-risk components, based on the principle of maximizing introduced by Bellman-Zade and the fuzzy membership functions, defined implementing. The results of this implementation are displayed in Table 8.

5.2. Single-objective optimal component selection by GA approach

To ensure the efficiency of the proposed high-risk component approach for optimal component selection, this paper used an evolutionary and single-objective method to select the optimal component. To transform the probleminto a single objective problem, both objective functions were checked to be in the direction of maximization and parallel. Finally, the first and second goals were defined in the form of Equation 22 by integrating Equation 11 and 12 [7]. To ensure that both objective functions have an equal impact on the final result, the coefficients of each function were set to 0.5 when converting the multi-objective problem to a single-objective one [7]. As a result, Equation22 was redefined as Equation 24 Subject to condition 23.

$$Z = W_f * \frac{F-F_{Min}}{F_{Max}-F_{Min}} + W_l * \frac{\text{ICD}-E_{Min}}{E_{Max}-E_{Min}} \quad (22)$$

Thus:

$$W_f + W_l = 1 \quad (23)$$

$$Z = 0.5 * \text{ICD} + 0.5 * \text{F} - (\text{cost} + \text{time} + \text{reliability} + \text{ICD}) \quad (24)$$

In addition to the proposed risky cost approach, the component selection algorithm (GA) also formulated and solved the simple cost model presented in [1,4]. Table9 presents the results of both approaches in a comparative way.

5.3. Multi-objective optimal component selection by non-dominated sorting genetic algorithm (NSGA-II)

The process of selecting components using NSGA involves the following steps:

Step_1: An initial population of size N was randomly generated based on objective functions and constraints. The population of offspring Qt (t = 0) was generated from Ptusing GA operators such as selection, crossover, and mutation.

Step_2: Pt and Qt populations were combined to create a new population of size 2N. Non-dominant sorting of the population Rt was performed and classified into multiple classes (F1, F2, F3, etc.). Crowding distance was then calculated for a set of individuals in the population. Two fitness functions in the maximum state, the subject of equations 25 and 26, were defined for this problem.

$$\text{Z1} = -1 * (\text{functionality}) + (\text{time} + \text{cost} + \text{reliability}) \quad (25)$$

$$\text{Z2} = -1 * (\text{ICD}) + (\text{time} + \text{cost} + \text{reliability}) \quad (26)$$

Step_3: The best N individuals were selected according to the forward dominance order and crowding distance to form a new population Pt+1. Individuals from fronts with low mastery rank and high distance in the same front were chosen first.

Step_4: Selection, crossover, and mutation were performed on the population Pt+1 to create a new offspring Qt+1 of size N.

Step 5: If the termination criteria were met, the set of non-dominated solutions was considered as output and the process was stopped; otherwise, step2 would be repeated. The results of this implementation are shown in Table 10.

Table 8. The results of this implementation fuzzy component selection

approach	*Functionality*	*ICD*	*Time*	*Budget*	*Module1*	*Module2*	*Module3*
High-risk redevelop cost	7.75	0.85	7.4s	30	SC5,SC6,SC17	SC11,SC7,SC4	SC14,SB7,SB9,SB10
simple cost [1]	6.49	0.50	6s	25	SC5,SC6,SC17	SC2, SC8, SB5	SB9,SC14,SC15,SB10

Table 9. Optimal selection of components in the approach by the single-objective method of genetic algorithm

approach	*Functionality*	*ICD*	*Time*	*Budget*	*Module1*	*Module2*	*Module3*
High-risk redevelop cost by GA	7.1	0.68	6s	30	SC5,SC6,SC17	SC8,SC7,SC4	SC14,SB7,SB9,SC20
simple cost	6.43	0.52	6s	25	SC4, SC5, SC6	SC8, SC11, SB6	SC15,SB8,SB9,SB10

Table 10.result of optimal components selection in the propose approach by the NSGA-II

approach	functionality	ICD	Time	Budget	reliability	Module1
High-risk redevelop cost	7.13	0.76	7.1s	30	0.99	SC5,SC17,SC2, SC6,SC7,SC11,SC14,SC19,SC15,SC20

6. Results Comparison

The technique of Component-based software engineering (CBSE) involves using pre-made software components to develop software systems. These components can either be purchased as COTS or built from scratch by software developers. The use of high-risk components can lead to longer development time and increased costs. Given that cost is a crucial factor in software development, accurate computation is essential. This study used the genetic algorithm to determine the cost factor for redeveloping high-risk components. The results of the case study demonstrated that the genetic algorithm was effective in accurately estimating the cost of software redevelopment. Table 11 compares the proposed approach with other techniques used in the field for software system creation. According to the table, the proposed method selects an optimum component with a shorter execution time than that of others. Table 12 compares the unique features of the proposed method with the features of the other methods.

Table 11. The results of the optimal selection of the component using the multi-objective selection method based on the cost of redevelopment with the GA coefficient .Figure 5 demonstrate results to bar chart.

ref	Type of optimization	objective	constraint	approach	Novelty
[4]	Multi objective	Fuzzy-ICD, Functionality	Fuzzy-ICD, reliability, Cost, Delivery time	Fuzzy approach	Compute coupling and cohesion in Mayer's classification by fuzzy method for calculate ICD
[2]	Multi objective	ICD, Functionality	ICD, reliability, Cost, Delivery time	Fuzzy approach	Fuzzy optimization
[5]	single objective	Quality	ICD, reliability, Cost, Delivery time	GA approach	Joint objective using GA
[56]	Multi objective	ICD, Functionality	Cost, Consistency	Fuzzy approach	Weighted objective
Proposed method	Multi objective	ICD, Functionality	ICD, reliability, high-risk Cost, Delivery time	GA approach	Joint constraint (cost) using GA weight

Table 12. Comparison between the presented method and other previous methods in the field of optimal component selection and their innovation aspect

approach	Functionality	ICD	Time	Budget	Module1	Module2	Module3
High-risk redevelop cost + NSGA-IIselection method	7.75	0.85	7.4s	30	SC5,SC6,SC17	SC11,SC7,SC4	SC14,SB7,SB9,SB10
simple cost + NSGA-II selection method	6.49	0.50	6s	25	SC5,SC6,SC17	SC2, SC8, SB5	SB9,SC14,SC15,SB10
High-risk redevelop cost + fuzzy component selection	7.13	0.76	7.1s	30	SC5,SC17,SC6	SC2, SC7, SC11	SC14,SC19,SC15,SC20
simple cost + fuzzy component selection [1]	6.61	0.82	6s	24	SC5,SC17,SB6	SC1, SC7, SC11	SC14,SC15,SC19,SC20
High-risk redevelop cost + GA component selection	7.1	0.68	6s	30	SC5,SC6,SC17	SC8,SC7,SC4	SC14,SB7,SB9,SC20
simple cost + GA component selection	6.43	0.52	6s	25	SC4, SC5, SC6	SC8, SC11, SB6	SC15,SB8,SB9,SB10

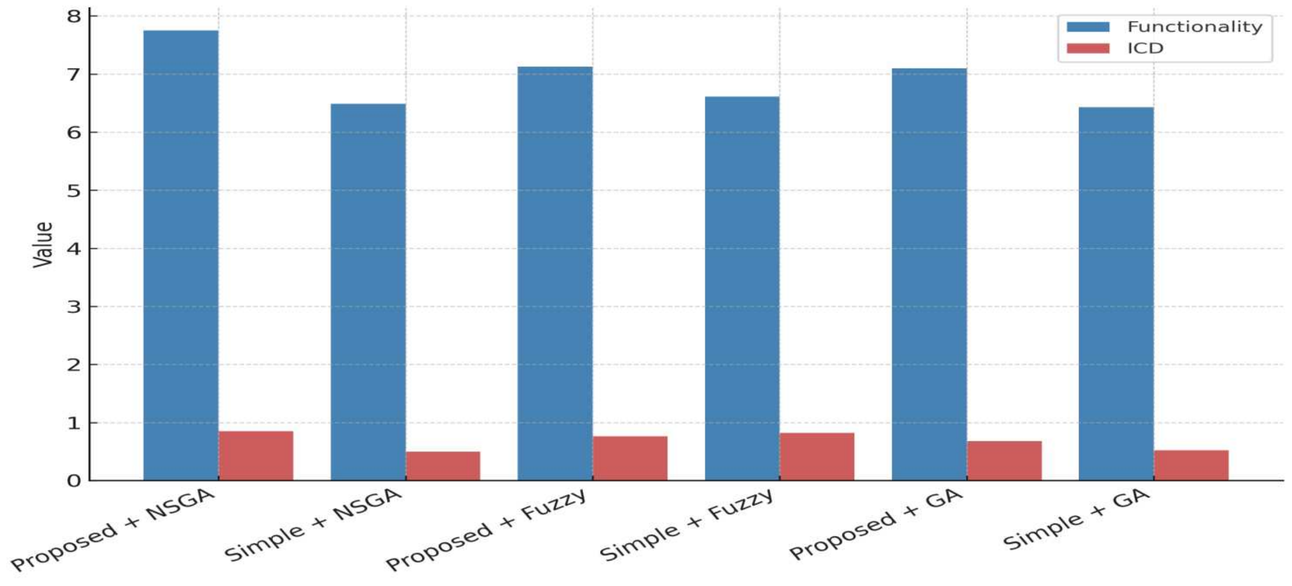

Fig. 5. The bar chart for results of optimal component selection by proposes approach and other method.

7. Conclusion and future work

When optimizing a component-based problem, it is important to consider various objective functions such as cost, reliability, and delivery time. This is because different types of components exist in optimization-related problems. These include COTS components such as ready, complete, incomplete, and new components. The aim of this study was to improve the optimization of software systems while reducing the developer's involvement in the calculation process. By improving cost calculations within the constraints, this study succeeded in reducing the selection of high-risk components with higher costs. As a result, the final value of performance significantly improved. The Strengths of proposed approach is accurate cost estimation for high-risk components, improved ICD and functionality, integration of real-world constraints in a multi-objective framework. The Weaknesses of proposed approach is Performance may depend on proper parameter tuning in GA; the method has been tested on a single case study and needs broader validation.

Future studies could conduct a more thorough analysis of high-risk components, along with evaluating the cost of other effective parameters, such as time and efficiency. They could also explore different types of evolutionary methods to optimize objective functions and constraints of the problem.

References

[1] Jha PC, Bali V, Narula S, Kalra M. Optimal component selection based on cohesion & coupling for component based software system under build-or-buy scheme. Journal of Computational Science. 2014 Mar 1; 5(2):233-42.

[2] Jha SK, Mishra RK. Predicting and accessing security features into component-based software development: a critical survey. In Software Engineering: Proceedings of CSI 2015 2019 (pp. 287-294). Springer Singapore.

[3] Mohan A, Jha SK. Predicting and accessing reliability of components in component based software development. In2019 International Conference on Intelligent Computing and Control Systems (ICCS) 2019 May 15 (pp. 1110-1114). IEEE.

[4] Kalantari S, Motameni H, Akbari E, Rabbani M. Optimal components selection based on fuzzy-intra coupling density for component-based software systems under build-or-buy scheme. Complex \&\ Intelligent Systems. 2021 Dec; 7(6):3111-34.

[5] Pressman RS. Software engineering: a practitioner's approach. Palgrave macmillan; 2005.

[6] Gholamshahi S, Hasheminejad SM. Software component identification and selection: A research review. Software: Practice and Experience. 2019 Jan; 49(1):40-69.

[7] Kwong CK, Mu LF, Tang JF, Luo XG. Optimization of software components selection for component-based software system development. Computers \&\ Industrial Engineering. 2010 May 1; 58(4):618-24.

[8] Kaliraj S, Bharathi A. Path testing based reliability analysis framework of component based software

system. Measurement.2019 Oct 1; 144:20-32.

[9] Yadav RK, Khan RA. Reliability Estimation of Object-Oriented Design.The IUP Journal of Systems Management. 2011 May 1; 9(2):28-41.

[10] Kaur R, Arora S, Jha PC, Madan S. Fuzzy multi-criteria approach for component selection of fault tolerant software system under consensus recovery block scheme. Procedia Computer Science.2015 Jan 1; 45: 842-51.

[11] Tailor AR, Dhodiya JM. GA-Based Hybrid Approach to Solve Fuzzy Multi-objective Optimization Model of Multi-application-Based COTS Selection Problem. InAdvanced Engineering Optimization through Intelligent Techniques: Select Proceedings of AEOTIT 2018 2020 (pp. 75-86). Springer Singapore.

[12] Tang JF, Mu LF, Kwong CK, Luo XG. An optimization model for software component selection under multiple applications development. European Journal of Operational Research. 2011 Jul 16; 212(2):301-11.

[13] Vescan A. Case study method and research design for the dynamic multilevel component selection problem. InService-Oriented Computing–ICSOC 2015 Workshops: WESOA, RMSOC, ISC, DISCO, WESE, BSCI, FOR-MOVES, Goa, India, November 16-19, 2015, Revised Selected Papers 13 2016 (pp. 130-141). Springer Berlin Heidelberg.

[14] Gupta P, Mehlawat MK, Verma S. COTS selection using fuzzy interactive approach. Optimization Letters.2012 Feb; 6: 273-89.

[15] Vescan A. An evolutionary multiobjective approach for the dynamic multilevel component selection problem. In Service-Oriented Computing–ICSOC 2015 Workshops: WESOA, RMSOC, ISC, DISCO, WESE, BSCI, FOR-MOVES, Goa, India, November 16-19, 2015, Revised Selected Papers 13 2016 (pp. 193-204). Springer Berlin Heidelberg.

[16] Nabot A. Software component selection: an optimized selection criterion for component-based software engineering (CBSE). Int. Arab J. Inf. Technol.. 2024;21(2):211-25.

[17] Tang JF, Mu LF, Kwong CK, Luo XG. An optimization model for software component selection under multiple applications development.European Journal of Operational Research. 2011 Jul 16; 212(2):301-11.

[18] Kontio J. A Systematic Process for Reusable Software Component Selection.Technical Report CS-TR-3478, University of Maryland; 1995.

[19] Cortellessa V, Marinelli F, Potena P. Automated selection of software components based on cost/reliability tradeoff. InSoftware Architecture: Third European Workshop, EWSA 2006, Nantes, France, September 4-5, 2006, Revised Selected Papers 3 2006 (pp. 66-81). Springer Berlin Heidelberg.

[20] Vescan A, Grosan C. Two evolutionary multiobjective approaches for the component selection problem. In2008 Eighth International Conference on Intelligent Systems Design and Applications 2008 Nov 26 (Vol. 2, pp. 395-400).IEEE.

[21] Vescan A. A metrics-based evolutionary approach for the component selection problem. In2009 11th International Conference on Computer Modelling and Simulation 2009 Mar 25 (pp. 83-88). IEEE.

[22] Khan MA, Mahmood S. Optimal component selection for component-based systems. In Innovations in Computing Sciences and Software Engineering 2010 May 20 (pp. 467-472). Dordrecht: Springer Netherlands.

[23] Vescan A, Şerban C. A fuzzy-based approach for the multilevel component selection problem.InHybrid Artificial Intelligent Systems: 11th International Conference, HAIS 2016, Seville, Spain, April 18-20, 2016, Proceedings 11 2016 (pp. 463-474). Springer International Publishing.

[24] Neubauer T, Stummer C. Interactive decision support for multiobjective COTS selection. In2007 40th Annual Hawaii International Conference on System Sciences (HICSS'07) 2007 Jan 3 (pp. 283b-283b).IEEE.

[25] Boonsiri S, Seacord RC, Bunting R. Automated component ensemble evaluation. International Journal of Information Technology. 2002 Aug; 8(1):40-53.

[26] Cortellessa V, Crnkovic I, Marinelli F, Potena P. Driving the selection of COTS components on the basis of system requirements. InProceedings of the 22nd IEEE/ACM International Conference on Automated Software Engineering 2007 Nov 5 (pp. 413-416).

[27] Lozano-Tello A, Gómez-Pérez A. BAREMO: how to choose the appropriate software component using the analytic hierarchy process. InProceedings of the 14th international conference on Software engineering and knowledge engineering 2002 Jul 15 (pp. 781-788).

[28] Jha PC, Kapur PK, Bali S, Kumar UD. Optimal component selection of COTS based software system under consensus recovery block scheme incorporating execution time. International Journal of Reliability, Quality and Safety Engineering. 2010 Jun; 17(03):209-22.

[29] Jha PC, KAUR R, BALI S, MADAN S. Optimal component selection approach for fault-tolerant software system under CRB incorporating build-or-

buy decision. International Journal of Reliability, Quality and Safety Engineering. 2013 Dec 26; 20(06):1350024.

[30] Kaur R, Arora S, Jha PC, Madan S. Fuzzy multi-criteria approach for component selection of fault tolerant software system under consensus recovery block scheme. Procedia Computer Science.2015 Jan 1; 45: 842-51.

[31] Verma S, Mehlawat MK. Multi-criteria optimization model integrated with AHP for evaluation and selection of COTS components. Optimization. 2017 Nov 2; 66(11):1879-94.

[32] Vescan A. An evolutionary multiobjective approach for the Component Selection Problem. In2008 First International Conference on the Applications of Digital Information and Web Technologies (ICADIWT) 2008 Aug 4 (pp. 252-257). IEEE.

[33] MOKARRAM AH, Isazadeh A, Izadkhah H. Early reliability assessment of component-based software system using coloredpetri net.Turkish Journal of Electrical Engineering and Computer Sciences. 2019; 27(4):2681-96.

[34] Şerban C, Vescan A, Pop HF. A new component selection algorithm based on metrics and fuzzy clustering analysis. InHybrid Artificial Intelligence Systems: 4th International Conference, HAIS 2009, Salamanca, Spain, June 10-12, 2009. Proceedings 4 2009 (pp. 621-628). Springer Berlin Heidelberg.

[35] Kumar D, Jha PC, Kapur PK, Kumar UD. Optimal component selection problem for COTS based software system under consensus recovery block scheme: a goal programming approach.

[36] Yessad L, Boufaida Z. A QoS ontology-based component selection.arXiv preprint arXiv:1109.0324. 2011 Sep 1.

[37] Iribarne L, Troya JM, Vallecillo A. Selecting software components with multiple interfaces. In Proceedings. 28th Euromicro Conference 2002 Sep 6 (pp. 26-32). IEEE.

[38] Vescan A, Grosan C. Evolutionary multiobjective approach for multilevel component composition. StudiaUniversitatis Babes-Bolyai Series Informatica. 2010 Dec 1; 55(4):18-32.

[39] Pande J, Garcia CJ, Pant D. Optimal component selection for component based software development using pliability metric. ACM SIGSOFT Software Engineering Notes. 2013 Jan 23; 38(1):1-6.

[40] Cortellessa V, Crnkovic I, Marinelli F, Potena P. Experimenting the Automated Selection of COTS Components Based on Cost and System Requirements. J. Univers. Comput.Sci . 2008 Jan 1; 14(8):1228-55.

[41] Dhodiya JM, Tailor AR. Genetic algorithm based hybrid approach to solve uncertain multi-objective COTS selection problem for modular software system. Journal of Intelligent \&\ Fuzzy Systems. 2018 Jan 1; 34(4):2103-20.

[42] Dhodiya JM, Tailor AR. Genetic algorithm based hybrid approach to solve fuzzy multi-objective assignment problem using exponential membership function. Springer Plus. 2016 Dec; 5(1):1-29.

[43] Gupta P, Verma S, Mehlawat MK. Optimization model of COTS selection based on cohesion and coupling for modular software systems under multiple applications environment. In Computational Science and Its Applications–ICCSA 2012: 12th International Conference, Salvador de Bahia, Brazil, June 18-21, 2012, Proceedings, Part III 12 2012 (pp. 87-102). Springer Berlin Heidelberg.

[44] Alves C, Castro J. CRE: A systematic method for COTS components selection. In Anais do XV SimpósioBrasileiro de Engenharia de Software 2001 Oct 3 (pp. 193-207). SBC.

[45] Martinez MA, Toval A. COTSRE: A components selection method based on requirements engineering. In Seventh International Conference on Composition-Based Software Systems (ICCBSS 2008) (pp. 220-223).IEEE.

[46] Maxville V, Armarego J, Lam CP. Intelligent component selection. In Proceedings of the 28th Annual International Computer Software and Applications Conference, 2004.COMPSAC 2004. 2004 Sep 28 (pp. 244-249). IEEE.

[47] Zhiqiao W, Kwong CK, Tang J, Chan JW. Integrated model for software component selection with simultaneous consideration of implementation and verification. Computers \&\ Operations Research. 2012 Dec 1; 39(12):3376-93.

[48] Mehlawat MK, Gupta P, Mahajan D. A multi-period multi-objective optimization framework for software enhancement and component evaluation, selection and integration.Information Sciences.2020 Jun 1; 523: 91-110.

[49] Gupta P, Mehlawat MK, Mahajan D. Multi-objective optimization framework for software maintenance, component evaluation and selection involving outsourcing, redundancy and customer to customer relationship. Information Sciences. 2019 May 1; 483: 21-52.