Multi-criteria Optimization Approach for Component Selection under Build-or-Buy Scheme

P.C. Jha[1], Vikram Bali[2], Sonam Narula[3], Mala Kalra[4]
[1,3] Department of Operational Research, University of Delhi
[2] Rayat Bahra Institute of Engineering & Bio-Technology, Mohali, Punjab
[4] National Institute of Technical Teachers Training & Research, Chandigarh
{ jhapc@yahoo.com; vikramgcet@gmail.com; sonam.narula88@gmail.com; malakalra2004@yahoo.co.in }

Abstract - **As the size and complexity of the software system has increased, there is a great need for the development of modular software system. For designing these modular software systems, Component Based Software System (CBSS) approach is most likely to adopt by software developers due to large number of favorable factors. In CBSS approach components are selected for each module depending upon various attributes and then these modules are integrated to form the software system. These software components can either be purchased from the outsourcing companies or can be built in-house. Therefore while designing a software system; one has to make a fundamental decision whether to build the components or to buy them as COTS. In this paper, a framework of build-or-buy components is formulated that discusses a multi-objective optimization problem which simultaneously maximizes three objective functions, ICD, functionality and quality, and minimizes overall cost of building a software system with respect to minimum threshold on ICD and Reliability, and maximum threshold on delivery time of the software system. A case study of Restaurant and Take away Joints is discussed and solved using fuzzy mathematical programming. The solution of the model gives the optimal mix of COTS or in-house built components selected for each module.**

Keywords - **Commercial-off-the shelf (COTS), Intra-modular Coupling Density (ICD), Build-or-Buy (B-B), Fuzzy Mathematical Programming (FMP).**

1. INTRODUCTION

The roadmap to building high quality software is software process. Software processes are adapted to meet the needs of software engineers and managers as they undertake the development process. In software development, quality of design encompasses requirements, specifications and the design of the system. Even the most jaded software developers will agree that high quality software is an important goal. Quality of a software system depends on 'conformance to specification' and 'meeting customer needs' (Cai et al., 2011). Several software quality models discussed in literature list numerous attributes that accounts for quality of the software system. The quality level of the component can be estimated using any of the quality models available in the literature such as ISO/IEC 9126 or ISO 25010. ISO/IEC 9126 classifies software quality in a structured set of characteristics as shown in Fig 1.

Increasing demand of large software system is the major driving force for the adoption of COTS based approach. COTS based approach promises faster delivery with lower resource costs. Here instead of building the whole software system from the scratch, a new system can be assembled and constructed by using existing vendor supported COTS product. Component-based software systems (CBSS) development focuses on the decomposition of a software system into functional or logical components with well defined interfaces. It allows a software system to be developed using appropriate software components. Since CBSS development approach was introduced, more and more software components have been provided by various software component suppliers such as ILOG and Rogue Wave Software (Land, Alvaro, & Crnkovic, 2008). Using COTS product does offer many advantages; however there are also disadvantages like incomplete knowledge of inner working code, incompatibility among COTS components, misunderstood specifications and incompatibility of modification in code etc. The developers have different options for the development of these small independent components such as choosing from already available COTS components, in-house development, or modifying the functioning of some existing components. The decision to choose right mix of components for a CBSS is not an easy task; it in fact involves consideration of several critical issues. For selection process developers has to rely upon the scientific way of decision making. Mathematical modeling and optimization usually offers solution to such problems.

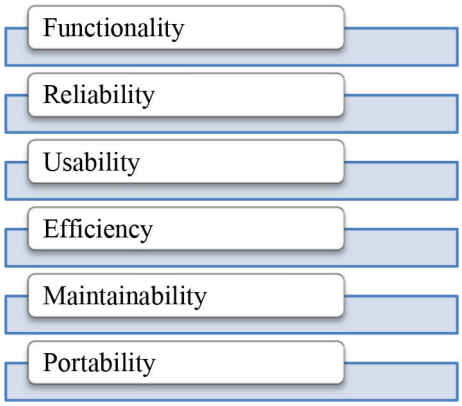

FIGURE 1: ISO/IEC 9126 CHARACTERISTICS

Some software systems are critical to human life. Thus, a trifling fault of components in such system would largely affect the functions of the entire software systems. Therefore, the reliability of the software component and ultimately of software system is of prime importance. It has become a major concern for software industry today. Another major restriction that can affect the development of software system is limited budget. Sometimes the best solution for software development may not be feasible due to higher costs. Cortellessa et al. developed an optimization model that supports "build-or-buy" decisions in selecting software components based on cost-reliability trade off. Jung and Choi introduced two optimization models for the COTS selection in the development of modular software systems considering cost reliability trade-off. Software components within a set of alternative software components are often assumed to have same functionality in CBSS development. But since these components are derived from different origin like they may be obtained from different software component supplier or developed by different software developers, there functions could be different from one another. Therefore, functional contributions of software components towards the functional requirements of a CBSS should be considered. Kwong, et al., (2010) have addressed an optimization problem of selection of software components for software modules in CBSS development considering functionality as one of their objective; they have used heuristic technique of Genetic Algorithm for the same.

It has already been proved in literature that modularity is an important quality attributes and hence should be applied at all level of abstraction, ranging from requirements specifications to the executable code level. The importance of high cohesion and low coupling are the main drivers to module identification. Modularization is a way to reduce efforts to understand and maintain the software system. A quantitative criterion was given by Abreu et al. for measuring cohesion and coupling of modular software systems. Authors in their paper have introduced ICD (Intra-modular coupling density) metric which simultaneously accounts for high cohesion and low coupling. Some previous studies attempted to define criteria and develop metrics for software modularity (Brito e Abreu & Goulao, 2001; Carlo, Mehdi, & Dino, 1991; Ian, 1993; Parsa & Bushehrian, 2004). Coupling is about the measure of interactions among software modules while cohesion is about the measure of interactions among the software components which are within a software module. A good software system should possess software modules with high cohesion and low coupling. A highly cohesive module exhibits high reusability and loosely coupled systems enable easy maintenance of a software system (Seker, van der Merwe, Kotze, Tanik, & Paul, 2004). The software quality metrics of coupling and cohesion were proposed by Larry Constantine (Stevens, Myers, & Constantine, 1974) based on characteristics of ''good" programming practices that aimed at reducing maintenance and modification costs. Parsa and Bushehrian (2004) discussed the criteria used in dividing a software system into modules, and evaluate their proposed criteria in the conventional and unconventional decomposition. Sarkar, Kak, and Nagaraja (2005) presented a new set of metrics for analyzing the interaction between the modules of a large software system. Sarkar et al. (2007) later presented a new set of metrics for assessing the quality of software modularity.

In literature large number of models discussed were based on the assumption that decision maker has complete knowledge about the software development. But it may be noted that software development is not an exact science as it requires estimating several attributes of component selection in a given module. Hence solving such problems under crisp optimization is not a best decision. Solving such problems using fuzzy selection model is proved to be a satisfactory solution to the decision maker in different scenario of input data uncertainty. Shen et al. presented a fuzzy optimization model subject to a parameterized budgetary level constraint for selecting the best COTS product considering weighted quality as an objective function. Gupta et al. (2009) formulated fuzzy multiple objective optimization model for COTS selection using non-linear S-shaped functions describing vague aspiration levels of the decision maker in respect of the weighted quality and cost. Gupta et al. (2010) developed an interactive fuzzy approach using linear membership functions for the COTS selection problem considering weighted quality and cost objectives. Gupta et al. (2011) proposed a solution method for the uncertain COTS selection problem whose imprecise parameters are expressed by triangular fuzzy numbers.

2. RELATED WORK

Most of the work done in past is in the area of component selection for CBSS. This has been the interest of academicians as well as industry people. Several optimization models have been formulated and are applied to real life software projects. In this section we will discuss only few recent articles. Cortellessa, et al. (2008) have proposed a general framework of the optimization model to support "build-or-buy" decisions based on cost and quality attributes. Gupta et al. (2009, 2010) formulated fuzzy multi-objective optimization models for selection of COTS components. Kwong et al. (2010) formulated optimization model for COTS selection considering cohesion and coupling of modular software system. The latter authors have not considered build-or-buy strategy in their work. Jha et al. (2013a) have formulated a joint optimization model for selection of components in design of fault tolerant modular software system incorporating build-or-buy strategy. Jha et al. (2013b) have formulated a bi-criteria optimization model considering intra-modular coupling density and functionality. The work done in the present paper is generalization of the work done by Jha et al. (2013b).

The novelty of our paper is that we have formulated a multi-criteria optimization model which has not been studied in literature with rigor. Very few authors have worked on optimal component selection in design of CBSS incorporating cohesion & coupling and build-or-buy strategy simultaneously. By doing this we are improving the cost effectiveness and maintainability of our system.

In this paper we have formulated a multi-criteria optimization model under build-or-buy framework. The model has four objectives:

1. 1Maximize ICD
2. Maximize Functionality
3. Minimize Cost
4. Maximize Quality

Along with the multiple objectives threshold on the value of reliability, delivery time of components and on ICD of each module is set by the decision maker. Contingent decision constraints are also described for dealing with incompatibility among COTS components. For COTS components different alternatives are available in market depending on different generation of technology that may provide same functionality but for components that are developed in-house we always use latest technology available and these components perform specific functionality for which they are developed. Based on the model a case study is solved, due to the imprecision involved in the data set fuzzy interactive approach is used for finding the solution. Solution of the model gives the optimal set of components selected (COTS or in-house built) that satisfies all the functional requirements of our model and gives the optimal value of objective functions.

The rest of the paper is organized as follows:
The list of symbols used throughout the paper is discussed in section 3. In section 4, we describe the component selection problem along with the numerous assumptions made for the model development. In section 5, we present the solution methodology using fuzzy membership function approach. Section 6 presents a case study along with the data set used for which the solution is given in section 7. Finally, concluding remarks are furnished in section 8.

3. LIST OF SYMBOL USED IN THE PAPER

Let S be a software architecture made of M modules, with a maximum number of N components available for each module. Figure 2 shows how CBSS can be developed using software components.

M	: the number of software modules
N	: the number of software components
Vij	: the number of alternatives of ith COTS component of jth software module
Si	: the ith component COTS or in-house; i = 1,2 . . . , N
SCi	: the ith COTS component; i = 1,2 . . . , N
SBi	: the ith in-house component; i = 1,2 . . . , N
L	: the number of functional requirements that our software system have to perform
Sk	: a set of alternative software components for the kth functional requirement of a CBSS. Only one software component in Sk is selected to implement the kth requirement; k=1,2,..,L i∈Sk denotes that Si belongs to the kth set
mj	: the jth software module; j=1,2,…,M
Rj	: reliability of jth module

rii'	: the number of interactions between Sci and Sci'; i; i' = 1,2 . . . , N as the coupling and cohesion are undirected relations, rii' = ri'i
fij	: fij are real numbers ranging from 0 to 1 depicting the function rating of Sci to mj for in-house components; i = 1,2 . . . , N; j = 1,2 . . . , M
fijk	: function rating of kth alternative of Sci to mj for COTS components; i = 1,2 . . . , N; j = 1,2 . . . , M
H	: a threshold value of ICDj of each module that needs to be set by decision makers. The constraint, ICDj ≥ H, j = 1, 2, . , M, needs to be satisfied.
cij	: cost of ith in-house component developed for jth module
cijk	: cost of kth alternative of ith COTS component available for jth module
tij	: development time of ith in-house component developed for jth module
τij	: average time required to perform a test case for ith in-house component of jth module
dijk	: delivery time of kth alternative of ith COTS component available for jth module
N_{ij}^{Tot}	: total number of tests performed on the ith in-house component of jth module
N_{ij}^{Suc}	: number of successful tests performed on the ith in-house component of jth module
πij	: the probability that a single execution of a software fails on a test case chosen from a certain input distribution.
ρij	: the probability that the in-house component is failure free during a single run given that test cases have been successfully performed
μijk	: the probability of failure on demand of COTS component
φ_{ij}	: probability that no failure occurs during the execution of the ith component of jth modu le
gij	: probability of no failures occurring in a Poisson distribution
qij	: quality level of ith in-house component of jth module
qijk	: quality level of kth alternative of ith COTS component of jth module
wj	: weight assigned to jth module; j=1,2,..,M
Ti	: delivery time of ith component
T	: maximum threshold given on delivery time of the software system
Scij	: the ith software component of jth software module, s.t. Scij = Scij' = Sci for all j,j'=1,2,….M
xijk	$\begin{cases}1 ; \text{ if } kth \text{ alternative of } ith \text{ COTS component of } jth \text{ module is selected} \\ 0 ; \text{ otherwise}\end{cases}$
yij	$\begin{cases}1 ; \text{ if } ith \text{ in-house component of } jth \text{ module is selected} \\ 0 ; \text{ otherwise}\end{cases}$
zij	$\begin{cases}1 ; \text{ if } ith \text{ component of } jth \text{ module is selected} \\ 0 ; \text{ otherwise}\end{cases}$

4. COMPONENT SELECTION PROBLEM

Consider the design of a modular software system. In module based systems, software modules are first identified for developing component based software system with the assumption that each module must contain atleast one software component. Software components are concrete software products that contain executable program codes and could be provided by various third party vendors. These components can also be developed in-house. The diagrammatic depiction of such a software system is given in Fig 2. One of the major problems of CBSS development is how to select software components to formulate a software module. The goal of the study is to select the best fit components for each module satisfying the system requirements. We propose an optimization model of component selection under the following assumptions:

1. 1For a given functional requirement several COTS alternatives and one in-house build component is available.

2. At least one component is supposed to get selected from each module.

3. A minimum threshold value on ICDj and reliability of each module and maximum threshold value on delivery time of each component is set by the decision makers.

4. The cost of an alternative component is the development cost and testing cost, if developed in-house; otherwise it is the buying price for the COTS product.

5. Reliability of in-house component is calculated which depends on the amount of testing performed whereas for COTS component reliability is known prior.

6. Redundancy is not allowed i.e. exactly one software component in Sk may get selected to implement kth requirement.

7. Interaction data for components is exactly same for all modules; irrespective of the selection happened for COTS or in-house component. Interaction associated is set by the software development team.

8. Different COTS alternatives w.r.t. cost, reliability, functionality and delivery time of a component are available.

9. Different in-house alternatives w.r.t. unitary development cost, estimated development time, average testability time of a component are available.

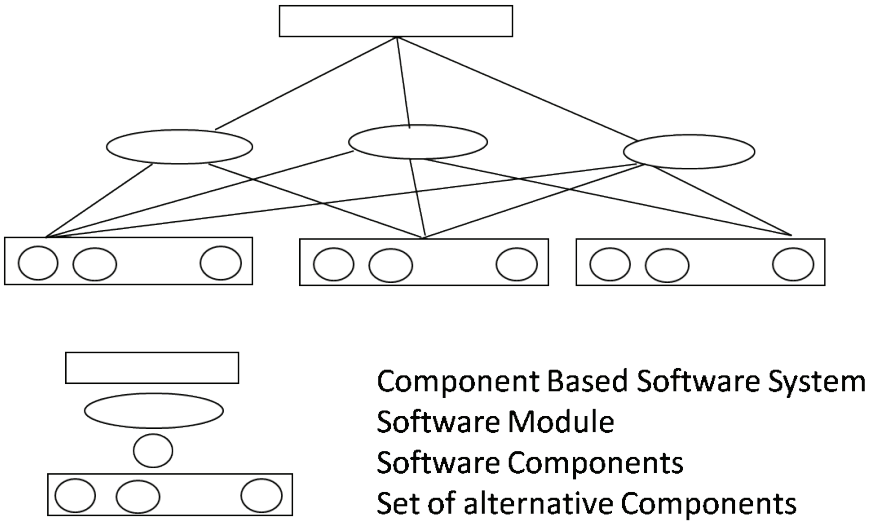

FIGURE 2: A CBSS SYSTEM

4.1 Problem description

In this section objective functions and constraints of the model are described for selection of optimal components under build-or-buy framework.

4.1.1 Objectives

1. Intra-modular Coupling Density (ICD)

In this research, we have employed Abreu and Gaulao's [8] approach which yields the quantitative measures of cohesion and coupling. The authors in their work presented intra-modular coupling density (ICD) to measure the relationship between cohesion and coupling of modules in design of modular software system and is given as follows:

$$ICD = \frac{CI_{IN}}{CI_{IN} + CI_{OUT}} \quad \text{(i)}$$

where, CIIN is the number of component interactions within modules, and CIOUT is the number of interactions between component of distinct modules.

Referring to Eq. (i), the ratio of cohesion to all interactions within the jth module can be expressed as ICDj. However, it can be found if any module contains only one component, the values of ICD for that module becomes zero. To make up for the deficiency 1 is added to the numerator of Eq. (i) to form another measure of ICD as follows:

$$ICD_j = \frac{(CI_{IN})_j + 1}{(CI_{IN})_j + (CI_{OUT})_j} \quad \text{(ii)}$$

where, ICDj is the intra-modular coupling density for the jth module; (CIIN)j is the number of component interactions within the jth module; and (CIOUT)j is the number of component interactions between the jth module and

other modules. Figure 3 (replicated from [7]) shows diagrammatic depiction of cohesion and coupling of software modules in the development of modular software system. The value of cohesion ([8]; Kwong et al., [7]) for both COTS and in-house developed components within the jth module (CIIN)j can be devised as follows:

$$(CI_{IN})_j = \sum_{i=1}^{N-1} \sum_{i'=i+1}^{N} r_{ii'} z_{ij} z_{i'j}$$

All interactions including cohesion and coupling associated with the jth module, CAj , can be expressed as:

$$CA_j = (CI_{IN})_j + (CI_{OUT})_j = \sum_{i=1}^{N-1} \sum_{i'=i+1}^{N} r_{ii'} z_{ij} \left(\sum_{j'=1}^{M} z_{i'j'} \right)$$

All interactions including cohesion and coupling of a software system, CA can be expressed as shown below:

$$CA = \sum_{i=1}^{N-1} \sum_{i'=i+1}^{N} r_{ii'} \left(\sum_{j=1}^{M} z_{ij} \right) \left(\sum_{j'=1}^{M} z_{i'j'} \right)$$

The sum of cohesions within all modules can be expressed as shown below:

$$CI_{IN} = \sum_{j=1}^{M} \sum_{i=1}^{N-1} \sum_{i'=i+1}^{N} r_{ii'} z_{ij} z_{i'j}$$

It is widely recognized that loose coupling and tight cohesion can achieve high maintainability of a software system. Therefore, the values of ICD incorporating "build-or-buy" concept and can be expressed as:

$$Max \;\; ICD = \frac{\sum_{j=1}^{M} \sum_{i=1}^{N-1} \sum_{i'=i+1}^{N} r_{ii'} x_{ij} x_{i'j}}{\sum_{i=1}^{N-1} \sum_{i'=i+1}^{N} r_{ii'} \left(\sum_{j=1}^{M} x_{ij} \right) \left(\sum_{j=1}^{M} x_{i'j} \right)} \quad ; \quad 0 \le ICD \le 1$$

By maximizing ICD, we can simultaneously maximize cohesion and minimize coupling.

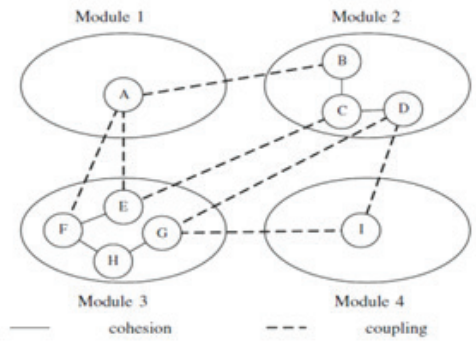

FIGURE 3: COHESION AND COUPLING OF SOFTWARE MODULES IN CBSS

2. Functional Performance

Functionality of the COTS or in-house built components can be defined as the ability of the component to perform according to the specific needs of the customer/organization requirements. The functional capabilities of the components are different. We use functional ratings of the components to the software module as coefficients in the objective function corresponding to maximizing the functional performance of the modular software system. These ratings are assumed to be given by the software development team.

$$Max \;\; F = \sum_{j=1}^{M} \sum_{i=1}^{N} \left(f_{ij} y_{ij} + \sum_{k=1}^{Vij} f_{ijk} x_{ijk} \right)$$

3. Cost

Cost constraint represents the overall cost of the system. It is the major criteria in determining the selection of components. We have considered cost based on procurement and adaptation of COTS components and development and testing cost of in-house components.

$$Min \; C = \sum_{j=1}^{M} \sum_{i=1}^{N} \left(c_{ij} \left(t_{ij} + \tau_{ij} N_{ij}^{Tot} \right) y_{ij} + \sum_{k=1}^{Vij} C_{ijk} x_{ijk} \right)$$

4 Quality

In the present study, in order to evaluate the overall quality of the software system the quality level of each component is determined using ISO/IEC 9126 [2] model. The objective we have defined maximizes weighted quality of the software system. Since the modules have different level of importance, therefore, weights are assigned to each module according to their access frequency.

$$Max \; Q = \sum_{j=1}^{M} w_j \sum_{i=1}^{N} \left(q_{ij} \; y_{ij} + \sum_{k=1}^{Vij} q_{ijk} \; x_{ijk} \right)$$

4.1.2 Constraints

1. Build versus buy decision

The equation stated below guarantees that redundancy is allowed in the module for both the build and buy components (i.e. in-house and COTS components). For a module, more than one component can be selected.

$$y_{ij} + \sum_{k=1}^{Vij} x_{ijk} = z_{ij} ; i = 1, 2, .., N; j = 1, 2, ..., M$$

$$\sum_{i=1}^{N} z_{ij} \ge 1 \; ; \;\; j = 1, 2, ..., M$$

If ith component of jth module is bought (i.e. some xijk =1) then there will be no in-house development (i.e. yij=0) and vice versa.

$$\sum_{j=1}^{M} z_{ij} = 1; i = 1, 2, \ldots, N$$

2. Threshold on ICD Constraint

This constraint shows the minimum threshold H on ICD value of each module.

$$\frac{\sum_{i=1}^{N-1}\sum_{i'=i+1}^{N} r_{ii'}x_{ij}x_{i'j}+1}{\sum_{i=1}^{N-1}\sum_{i'=i+1}^{N} r_{ii'}x_{ij}\sum_{j'=1}^{M} x_{i'j'}} \geq H;\ \text{j=1,2,..,M , j'=1,2,..,M}$$

3. Threshold on delivery Time constraint

The delivery time of the component is the time of acquiring and integrating the components within and amongst the modules of the software system. The total delivery time includes development, integration and system testing time.

The maximum threshold T has been given on the delivery time of the whole system. In case of COTS components the delivery time is simply given by dijk, whereas for an in-house developed component the delivery time of ith component available for jth module is $(t_{ij}+\tau_{ij}N_{ij}^{tot})$. So the delivery time of the ith component can be expressed as:

$$T_i=\left(t_{ij}+\tau_{ij}N_{ij}^{Tot}\right)y_{ij}+\sum_{k=1}^{Vij} d_{ijk}x_{ijk};i=1,2,....,N;j=1,...,M$$

It is assumed that manpower is available to independently develop in-house component instances. Therefore, the delivery time constraint can be reformulated as follows:

$$\max_{i=1,2,....,N}(T_i)\leq T$$

The effect of testing on cost, reliability and delivery time of COTS product is instead assumed to be accounted in the COTS parameters. Basing on the testability definition, we can assume that the number of successful (i.e. failure free) test performed on the same component can be obtained as

$$N_{ij}^{Suc}=(1-\pi_{ij})N_{ij}^{Tot};i=1,2,...,N;j=1,2,...,M$$

and it will be used to develop reliability constraint.

5. Reliability of in-house Components

Software reliability is defined as the probability of failure-free software operation for a specified period of time in a specified environment [20]. Software developers aim at building software with minimum resources and simultaneously achieving a maximum reliability. The reliability of a COTS component for a specified module is provided by its vendor whereas the reliability of the built component is estimated by the software development team.

Authors in [5] have defined the probability of failure on demand of an in-house developed, the ith component of jth module under the assumption that the on-field users' operational profile is the same as the one adopted for testing in [8]. Let A be the event " failure – free test cases have been performed" and B be the event " the alternative is failure free during a single run ".If ρij is the probability that the in-house developed alternative is failure free during a single run given that test cases have been successfully performed, from the Baye's Theorem we get

$$\rho_{ij}=P(B/A)=\frac{P(A/B)P(B)}{P(A/B)P(B)+P(A/\bar{B})P(\bar{B})}$$

The following equalities come straightforwardly:

$$P(A/B)=1$$

$$P(B)=1-\pi_{ij}$$

$$P(A/\bar{B})=(1-\pi_{ij})^{N_{ij}^{suc}}$$

$$P(\bar{B})=\pi_{ij}$$

therefore, we have $\rho_{ij}=\frac{(1-\pi_{ij})}{(1-\pi_{ij})+\pi_{ij}(1-\pi_{ij})^{N_{ij}^{suc}}}$

6. Average number of failures

The probability of failure on demand of an in-house developed component i of module j can be expressed as 1-ρij. Now we can write the average number of failures qij of ith component of jth module as follows:

$$g_{ij}=(1-\rho_{ij})y_{ij}+\sum_{k=1}^{V_{ij}}\mu_{ijk}x_{ijk}$$

7. Threshold on Reliability Constraint

The probability that no failure occurs during the execution of the ith component of jth module is given by , which represents the probability of no failures occurring in a Poisson distribution with parameter qij

$$\prod_{i=1}^{N} e^{-q_{ij}}\geq R_j;j=1,2,...,M$$

8. Selection or rejection of COTS or in-house build products

$$x_{ijk}\in\{0,1\}\ ;\ i=1,2,...,N;j=1,2,...,M;k=1,2,...,V_{ij}$$
$$y_{ij}\in\{0,1\}\ ;\ i=1,2,...,N;j=1,2,....,M$$
$$z_{ij}\in\{0,1\}\ ;\ i=1,2,...,N;j=1,2,....,M$$

9. Contingent decision constraint

CBSS is a technique based on wide use of COTS components. COTS software market contains hundreds, thousands components, that makes component selection an extremely difficult and time expensive task. Often the COTS component selected for one module is incompatible with the alternative COTS components for other modules due to problem such as implementation technology, interfaces and licensing. Therefore, the issue of compatibility amongst the COTS components is addressed in this section by incorporating the additional compatibility constraints in the optimization model.

Compatibility constraints are used to deal with incompatibility amongst COTS components. The incompatibility constraint can be denoted by xrs ≤ xu1t , that is if the module s chooses COTS component r, then the module t must choose the COTS component u1. This decision is called contingent decision constraint (Jung and Choi, 1999). Suppose that there are two contingent decisions in the model, such as, the COTS alternative for the module s is only compatible with the COTS products u1 and u2 for the module t, i.e. either xu1t =1 if xrs =1 or xu2t =1 if xrs=1, these constraint can be represented as either xrs ≤ xu1t or xrs ≤ xu2t. Since the presence of "either-or" constraint makes the optimization problem non-linear, it can be linearized by binary variable yk' as follows:

$$y_{k'} = \begin{cases} 0, \text{ if } k'\text{th constraint is active} \\ 1, \text{ if } k'\text{th constraint is inactive} \end{cases}$$

Thus, only one out of z contingent decision constraints for any COTS products between two modules is guaranteed to be active if M¯is sufficiently large.

$$x_{rs} - x_{u_k \cdot th} \leq \bar{M} y_{k'}$$
$$\sum_{k'=1}^{\bar{z}} y_{k'} = z - 1$$
$$y_{k'} \in \{0,1\} \; ; \; k' = 1,2,\ldots,z$$

4.2 The Decision Problem

The multi-objective optimization model for COTS selection using CBSS development can be formulated as follows: **Problem (P1)**

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

$$Max\ F = \sum_{j=1}^{M}\sum_{i=1}^{N}\left(f_{ij} y_{ij} + \sum_{k=1}^{Vij} f_{ijk}\, x_{ijk}\right) \quad (2)$$

$$Min\ C = \sum_{j=1}^{M}\sum_{i=1}^{N}\left(c_{ij}\,(t_{ij} + \tau_{ij} N_{ij}^{Tot}) y_{ij} + \sum_{k=1}^{Vij} c_{ijk}\, x_{ijk}\right) \quad (3)$$

$$Max\ Q = \sum_{j=1}^{M} w_j \sum_{i=1}^{N}\left(q_{ij} y_{ij} + \left(\sum_{k=1}^{Vij} q_{ijk} x_{ijk}\right)\right) \quad (4)$$

$$Subject\ to\ X \in S = \{x_{ijk},\ y_{ij},\ z_{ij}\ are\ binary\ \text{var}iables\ /$$

$$\frac{\sum_{i=1}^{N-1}\sum_{i'=i+1}^{N} r_{ii'} z_{ij} z_{i'j} + 1}{\sum_{i=1}^{N-1}\sum_{i'=i+1}^{N} r_{ii'} z_{ij} \sum_{j'=1}^{M} z_{i'j'}} \geq H;\ j, j' = 1,2,\ldots,M \quad (5)$$

$$\left(t_{ij} + \tau_{ij} N_{ij}^{Tot}\right) y_{ij} + \sum_{k=1}^{Vij} d_{ijk} x_{ijk} = T_i; i = 1,2,\ldots,N; j = 1,\ldots,M \quad (6)$$

$$\max_{i=1,2,\ldots,N}(T_i) \leq T \quad (7)$$

$$N_{ij}^{Suc} = (1-\pi_{ij}) N_{ij}^{Tot}; i = 1,2,\ldots,N; j = 1,2,\ldots,M \quad (8)$$

$$\rho_{ij} = \frac{(1-\pi_{ij})}{(1-\pi_{ij}) + \pi_{ij}(1-\pi_{ij})^{N_{ij}^{Suc}}} \quad (9)$$

$$g_{ij} = (1-\rho_{ij}) y_{ij} + \sum_{k=1}^{V_{ij}} \mu_{ijk}\, x_{ijk} \quad (10)$$

$$\prod_{i=1}^{N} \varphi_{ij} = e^{-g_{ij} x_{ij}} \geq R_j; j = 1,2,\ldots,M \quad (11)$$

$$y_{ij} + \sum_{k=1}^{Vij} x_{ijk} = z_{ij}; i = 1,2,..,N; j = 1,2,\ldots,M \quad (12)$$

$$\sum_{j=1}^{M} z_{ij} = 1; i = 1,2,\ldots,N \quad (13)$$

$$\sum_{i=1}^{N} z_{ij} \geq 1 \; ; \; j = 1,2,\ldots,M \quad (14)$$

$$x_{rs} - x_{u_k \cdot th} \leq \bar{M} y_{k'} \quad (15)$$

$$\sum_{k'=1}^{\bar{z}} y_{k'} = z - 1 \quad (16)$$

$$y_{k'} \in \{0,1\} \; ; \; k' = 1,2,\ldots,z \quad (17)$$

$$x_{ijk} \in \{0,1\} \; ; \; i = 1,2,\ldots,N; j = 1,2,\ldots,M; k = 1,2,\ldots,V_{ij} \quad (18)$$

$$y_{ij} \in \{0,1\} \; ; \; i = 1,2,\ldots,N; j = 1,2,\ldots,M \quad (19)$$

$$z_{ij} \in \{0,1\} \; ; \; i = 1,2,\ldots,N; j = 1,2,\ldots,M \; \} \quad (20)$$

5 SOLUTION METHODOLOGY USING FUZZY INTERACTIVE APPROACH

The fuzzy set theory appears as an important tool to provide a multi-criteria decision framework that incorporates vagueness and imprecision inherent in the justification and selection of COTS or in-house components for CBSS. An effective way to express factors such as quality, ICD, functionality and cost which can neither be assessed by crisp values nor random processes, is by using fuzzy numbers. Following steps are required to perform for solving fuzzy multi-objective optimization problem.

5 SOLUTION METHODOLOGY USING FUZZY INTERACTIVE APPROACH

The fuzzy set theory appears as an important tool to provide a multi-criteria decision framework that incorporates vagueness and imprecision inherent in the justification and selection of COTS or in-house components for CBSS. An effective way to express factors such as quality, ICD, functionality and cost which can neither be assessed by crisp values nor random processes, is by using fuzzy numbers. Following steps are required to perform for solving fuzzy multi-objective optimization problem.

Step 1 : Construct multi-objective optimization problem. Refer problem (P1)

Step 2 : Solve multi-objective optimization problem by considering first objective function. This process is repeated for all the remaining objective functions. If all the solutions (i.e. X1=X2=…= Xk= xij , i=1,…,N; j=1,…,M) are same, select one of them as an optimal compromise solution and stop. Otherwise, go to step 3.

Step 3 : Evaluate the kth objective function at all solutions obtained and determine the best (worst) lower bound (Lk) and best (worst) upper bound (Uk) as the case may be.

Step 4 : Define membership function of each objective of optimization model. The membership function for ICD is given as follows.

$$\mu_{ICD}(x)=\begin{cases}1, & if\ ICD(x)\geq ICD_u,\\ \dfrac{ICD(x)-ICD_l}{ICD_u-ICD_l}, & if\ ICD_l<ICD(x)<ICD_u,\\ 0, & if\ ICD(x)\leq ICD_l\end{cases}$$

where ICDl is the worst lower bound and ICDu is the best upper bound of ICD objective function.

The membership function for cost is given as follows.

$$\mu_{C}(x)=\begin{cases}1, & if\ C(x)\leq C_l,\\ \dfrac{C_u-C(x)}{C_u-C_l}, & if\ C_l<C(x)<C_u,\\ 0, & if\ C(x)\geq C_u\end{cases}$$

where Cl is the best lower bound and Cu is the worst upper bound of cost objective function.

The membership function for functionality is given as follows.

$$\mu_{F}(x)=\begin{cases}1, & if\ F(x)\geq F_u,\\ \dfrac{F(x)-F_l}{F_u-F_l}, & if\ F_l<F(x)<F_u,\\ 0, & if\ F(x)\leq F_l\end{cases}$$

where Fl is the worst lower bound and Fu is the best upper bound of functionality objective function.

The membership function for quality is given as follows.

$$\mu_{Q}(x)=\begin{cases}1, & if\ Q(x)\geq Q_u,\\ \dfrac{Q(x)-Q_l}{Q_u-Q_l}, & if\ Q_l<Q(x)<Q_u,\\ 0, & if\ Q(x)\leq Q_l\end{cases}$$

where Ql is the worst lower bound and Qu is the best upper bound of quality objective function.

Step 5: Develop fuzzy multi-objective optimization model.

Following Bellaman-Zadeh's maximization principle [21] and using the above defined fuzzy membership functions, the fuzzy multi-objective optimization model for COTS or in-house selection is formulated as follows:

$$\begin{aligned}&Maximize\ \ \lambda\\ &Subject\ to\ \ \lambda\leq\mu_{ICD(x)}\\ &\qquad\lambda\leq\mu_{C(x)}\\ &\qquad\lambda\leq\mu_{F(x)}\\ &\qquad\lambda\leq\mu_{Q(x)}\\ &\qquad 0\leq\lambda\leq 1\\ &\qquad X\in S\end{aligned}$$

Solve the above model. Present the solution to the decision maker. If the decision maker accepts it then stop. Otherwise, evaluate each objective function at the solution. Compare the upper (lower) bound of each objective function with new value of the objective function. If the new value is lower (higher) than the upper (lower) bound, consider it as a new upper (lower) bound. Otherwise, use the old values as it is. If there are no changes in current bounds of all the objective functions then stop otherwise go to step 4.

The solution process terminates when decision maker accepts the obtained solution and considers it as the preferred compromise solution which is in fact a compromise feasible solution that meets the decision maker's preference.

6 AN ILLUSTRATIVE CASE STUDY

A case study of CBSS development is presented to illustrate the proposed methodology of optimizing the selection of COTS or in-house components for CBSS development. A local software system supplier planned to develop a software system for Restaurant and Take Away Food Joints. In this case, a software system is decomposed into three modules M1, M2 and M3. A total of twenty eight

software components (Sc1–Sc28) are available in market and at the same time the company has resources and expertise to develop twelve components in-house (SB1-SB12) to make up twelve sets of alternative software component (S1–S12) for each module. Total components available for selection are ninety six. Exactly one software component in each set of alternatives may get selected for a particular software module for fulfilling a given functional requirement. For example Sc1, Sc2, Sc3 and SB1 all belong to the set of alternative software components S1. Hence only one of the four components will be selected for fulfilling the S1 functional requirement.

The data set for case study is given below.

In table 1, functional requirements of the software system and functionality associated with COTS (fij for all i,j) and In-house (fijk for all i,j,k) components is given.

TABLE 1: EXAMPLE DESCRIPTION AND FUNCTIONALITY OF COTS COMPONENTS

Functional Requirements	**Sk**	**Software Components**	**Module 1**	**Module 2**	**Module 3**
			fi1	fi2	fi3
For fine Dining	S1	Sc1	0.45	0.54	0.23
		Sc2	0.67	0.62	0.66
		Sc3	0.48	0.48	0.40
		SB1	0.53	0.55	0.43
For Take Away & Home Delivery	S2	Sc4	0.38	0.49	0.51
		Sc5	0.83	0.35	0.67
		Sc6	0.62	0.65	0.54
		Sc7	0.71	0.58	0.63
		SB2	0.63	0.52	0.58
Recipe & Inventory Management	S3	Sc8	0.23	0.82	0.08
		Sc9	0.46	0.85	0.16
		SB3	0.34	0.82	0.12
Order Procurement & Material Rep.	S4	Sc10	0.62	0.73	0.21
		Sc11	0.49	0.81	0.18
		SB4	0.55	0.77	0.19
Schemes & Promotions	S5	Sc12	0.72	0.56	0.15
		Sc13	0.83	0.79	0.06
		Sc14	0.70	0.62	0.13
		SB5	0.75	0.66	0.11
Accounting	S6	Sc15	0.42	0.72	0.85
		Sc16	0.39	0.53	0.91
		SB6	0.40	0.62	0.88
Other Features	S7	Sc17	0.15	0.51	0.35
		SB7	0.16	0.52	0.37
MIS Reports	S8	Sc18	0.33	0.75	0.23
		Sc19	0.41	0.89	0.05
		SB8	0.37	0.82	0.14
Purchase Reports	S9	Sc20	0.53	0.78	0.75
		Sc21	0.42	0.89	0.52
		Sc22	0.31	0.90	0.48
		SB9	0.42	0.86	0.58
Stock Reports	S10	Sc23	0.65	0.79	0.42
		Sc24	0.78	0.82	0.51
		SB10	0.71	0.80	0.46

Sale Reports	S11	Sc25	0.79	0.42	0.90
		Sc26	0.88	0.35	0.87
		SB11	0.83	0.38	0.88
Profitability Reports	S12	Sc27	0.53	0.23	0.92
		Sc28	0.61	0.19	0.88
		SB12	0.57	0.21	0.90

In table 2, Cost (cijk for all i,j,k) in 100$ unit; delivery time (dijk for all i,j,k) in days, probability of failure on demand (μijk for all i,j,k) and quality level (qijk for all i,j,k) associated with COTS components is given.

TABLE 2: COST, DELIVERY TIME AND PROBABILITY OF FAILURE ON DEMAND DATA SET OF COTS COMPONENTS

Software Comp.	**Module 1**				**Module 2**				**Module 3**			
	cijk	dijk	μijk	qijk	cijk	dijk	μijk	qijk	cijk	dijk	μijk	qijk
Sc1	8.2	3	0.002	0.61	9.4	3	0.001	0.67	6.3	5	0.003	0.51
Sc2	10.1	2	0.002	0.70	10.2	2	0.002	0.71	10.6	3	0.001	0.70
Sc3	8.3	3	0.001	0.62	8.8	3	0.003	0.64	8.0	4	0.001	0.60
Sc4	7.2	4	0.004	0.55	8.9	3	0.004	0.64	9.1	3	0.003	0.65
Sc5	12.2	1	0.002	0.82	7.5	4	0.002	0.56	10.7	3	0.001	0.71
Sc6	10.2	2	0.005	0.71	10.5	2	0.002	0.73	9.4	4	0.002	0.69
Sc7	11.1	1	0.001	0.74	9.8	3	0.002	0.68	10.3	3	0.002	0.70
Sc8	6.3	5	0.002	0.51	12.2	1	0.003	0.82	4.9	6	0.001	0.46
Sc9	8.6	3	0.002	0.63	12.5	1	0.001	0.85	5.0	5	0.002	0.49
Sc10	10.2	2	0.001	0.71	11.3	2	0.002	0.75	6.1	5	0.001	0.50
Sc11	8.9	4	0.004	0.65	12.1	1	0.002	0.81	5.8	5	0.003	0.50
Sc12	11.2	2	0.003	0.75	9.6	3	0.001	0.67	5.5	6	0.002	0.51
Sc13	12.3	2	0.001	0.83	11.9	1	0.002	0.79	4.8	6	0.001	0.45
Sc14	11.0	2	0.001	0.74	10.2	2	0.002	0.72	5.3	5	0.002	0.49
Sc15	8.2	3	0.002	0.61	11.2	2	0.002	0.75	12.5	2	0.003	0.85
Sc16	7.9	3	0.002	0.59	9.3	2	0.001	0.67	13.1	1	0.001	0.90
Sc17	5.1	5	0.003	0.49	9.1	3	0.001	0.65	7.5	4	0.001	0.56
Sc18	7.8	4	0.002	0.59	11.5	2	0.002	0.78	6.3	5	0.002	0.51
Sc19	7.3	3	0.002	0.55	12.9	1	0.002	0.88	4.0	5	0.003	0.40
Sc20	8.1	2	0.004	0.61	11.8	1	0.001	0.78	11.5	2	0.001	0.75
Sc21	8.2	2	0.002	0.61	12.9	1	0.002	0.89	9.2	3	0.002	0.65
Sc22	7.1	3	0.002	0.55	13.0	1	0.003	0.89	8.8	3	0.003	0.64
Sc23	10.5	1	0.002	0.73	11.9	2	0.002	0.80	8.2	3	0.004	0.61
Sc24	11.8	1	0.002	0.79	12.2	1	0.002	0.82	9.1	2	0.003	0.65
Sc25	11.9	1	0.003	0.80	8.2	3	0.003	0.61	13.1	1	0.004	0.90
Sc26	12.8	1	0.004	0.88	7.5	3	0.002	0.56	12.7	1	0.004	0.87
Sc27	9.3	3	0.002	0.67	6.3	4	0.002	0.51	13.4	1	0.002	0.92
Sc28	10.1	2	0.002	0.70	5.9	5	0.001	0.50	12.8	1	0.001	0.88

In table 3, Cost (cij for all i,j) in 100$ unit; development time (tij for all i, j) in days and quality level (qij for all i,j) associated with In-house components is given.

TABLE 3: COST AND DEVELOPMENT TIME DATA SET FOR IN-HOUSE COMPONENTS

Software Comp.	**Module1**			**Module 2**			**Module 3**		
SB1	8.0	3	0.80	8.5	3	0.85	7.0	6	0.70
SB2	9.5	4	0.88	8.5	4	0.84	9.1	5	0.85
SB3	7.5	4	0.75	11.0	2	0.90	4.5	7	0.65
SB4	8.5	4	0.82	11.0	2	0.91	5.5	6	0.69
SB5	10.0	3	0.89	9.5	3	0.87	4.5	7	0.65
SB6	7.5	3	0.75	9.0	3	0.86	12.0	3	0.95
SB7	5.0	5	0.50	9.0	3	0.86	7.0	5	0.71
SB8	7.0	4	0.70	11.0	3	0.90	3.5	6	0.61
SB9	7.0	3	0.70	11.5	2	0.93	8.5	4	0.82
SB10	10.5	2	0.90	11.5	2	0.92	8.0	3	0.80
SB11	11.5	2	0.93	7.0	4	0.70	12.0	2	0.94
SB12	9.0	3	0.86	5.0	6	0.50	12.0	2	0.95

In table 4, Degree of Interaction among components is given. The range of the degrees is 1–10. The degree '1' means a very low degree of interaction while the degree '10' refers to a very high degree of interaction

TABLE 4: INTERACTION DATA SET AMONG COMPONENTS

	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9	S_{10}	S_{11}	S_{12}	S_{13}	S_{14}	S_{15}	S_{16}	S_{17}	S_{18}	S_{19}	S_{20}	S_{21}	S_{22}	S_{23}	S_{24}	S_{25}	S_{26}	S_{27}	S_{28}
S_1	0	4	1	7	1	4	0	7	8	9	9	6	6	5	6	1	8	4	7	1	10	9	3	0	2	2	1	4
S_2		0	2	3	4	1	2	8	6	7	10	4	7	8	10	1	0	6	5	0	2	1	2	3	1	1	7	2
S_3			0	6	5	7	6	7	9	6	8	5	3	6	2	2	2	0	0	4	10	0	1	0	3	9	0	5
S_4				0	10	0	3	5	6	9	7	7	7	6	2	0	1	3	6	3	5	1	8	5	6	4	0	3
S_5					0	0	8	8	9	8	6	6	4	8	3	10	0	7	0	2	2	1	7	6	1	4	3	5
S_6						0	0	6	7	7	8	7	1	2	0	1	0	2	3	1	1	0	6	7	2	1	2	2
S_7							0	7	6	7	8	8	9	8	3	0	2	3	9	2	0	1	7	9	0	1	2	1
S_8								0	1	2	5	1	3	4	6	1	2	5	1	1	10	0	7	8	1	5	5	9
S_9									0	9	3	0	0	7	8	0	2	1	1	0	0	2	8	6	7	6	0	3
S_{10}										0	8	1	5	4	3	9	0	0	6	2	5	3	6	3	4	1	2	4
S_{11}											0	2	1	6	5	1	0	4	1	0	1	1	5	4	0	1	3	0
S_{12}												0	6	0	4	4	8	5	0	0	2	2	3	1	0	0	0	1
S_{13}													0	7	1	9	2	1	8	1	2	0	2	0	8	2	1	0
S_{14}														0	3	3	0	2	4	2	1	2	7	4	0	10	2	2
S_{15}															0	7	5	2	6	3	0	0	1	1	3	1	9	7
S_{16}																0	1	3	10	1	2	9	1	2	2	1	8	8
S_{17}																	0	2	3	2	1	6	5	2	4	7	3	3
S_{18}																		0	5	2	7	8	4	6	8	7	1	10
S_{19}																			0	9	5	5	7	3	6	4	2	8
S_{20}																				0	7	5	6	5	1	3	6	1
S_{21}																					0	2	4	3	0	3	0	1
S_{22}																						0	0	4	0	0	1	2
S_{23}																							0	7	0	1	0	3
S_{24}																								0	1	2	2	5
S_{25}																									0	3	7	9
S_{26}																										0	8	8
S_{27}																											0	1
S_{28}																												0

7. SOLUTION

The solution to the optimization model gives the optimal mix of components selected for the software system along with the corresponding ICD, Functionality, Cost and Quality with the threshold on ICD and reliability of each module and threshold on delivery time of the software system under fuzzy environment.

Equal weights are given to each module and other initial parameters were assumed as:

H	T	Rj
0.4	7	0.85

After forming multi-objective programming using the above data, the problem is solved using LINGO [22] and the solution of each single objective problem is found as follows:

	X1	X2	X3	X4
ICD(Z1)	0.87	0.43	0.33	0.37
F(Z2)	5.77	9.1	9.85	2.79
C(Z3)	303.9	400	163.9	74.7
Q(Z3)	8.99	10.88	10.18	6.18

where X1, X2, X3 and X4 are optimal sets of components selected for ICD, Functionality and Cost objectives. Thus upper and lower bounds of each objective function are derived as follows:

$0.33 \leq ICD \leq 0.87$
$2.79 \leq F \leq 9.85$
$74.7 \leq C \leq 400$
$6.18 \leq Q \leq 10.88$

The membership function of three objectives using lower and upper bounds are constructed and fuzzy multi-objective optimization model is developed and solved.

Owing to compatibility constraint 10th component of module 1 is found to be compatible with the 18th component of module 2.

The solution thus obtained is:

λ	ICD	Functionality	Cost	Quality
0.68	0.71	7.98		9.4
Q(Z3)	8.99	10.88	10.18	6.18

COTS or In-house Components Selected:

Module 1	Module 2	Module 3
• SC5, SC9, SC10, SC13, SC17, SC26 • SB3, SB15, SB23	SC18, SC22	SC27

8 CONCLUSION

In this paper, we have discussed a framework where a software system is required to perform certain given number of functionalities. For each functional requirement we require exactly one software component to perform its intended operation. The required component may belong to any of the given module. Since we have discussed the framework under build-or-buy scheme therefore the component can either be purchased from outside as COTS or can be developed in-house. For components that are developed in-house additional expenditure and time are required in terms of testing efforts. Based on the proposed methodology, an optimization model is formulated to perform the selection of COTS or in-house components to obtain an optimal solution. A case study of Restaurant and Food Joints is discussed to illustrate the formulated problem. However, the developed methodology involves some subjective judgments from software development teams, such as the determination of the scores of interaction, function rating etc. in this regard; the fuzzy mathematical programming is incorporated to deal with fuzziness caused by subjective judgments. The solution obtained gives the optimal values of objective function within the given set of constraints. Also the solution consists of optimal mix of COTS and in-house developed components. The proposed model is applicable for automation of operations in any business organization which in turn improves the productivity of the business. Same methodology can be adopted for software development of other sectors such as retail, airline, academic and many more.

REFERENCES

1. L. Cai, X. Xie, S. Huang, "Software Quality Model Development-An Introduction" Elsevier:13(2011) 8749-8758
2. ISO/IEC 9126: Information Technology-Software Product Evaluation-Quality Characteristics and Guidelines for their use: International standard, ISO, Geneva, Switzerland, 1991
3. R. Land, A. Alvaro, I. Crnkovic, "Towards Efficient Software Component Evaluation: An Examination of Component Selection and Certification", Software Engineering and Advanced Applications, 2008. SEAA'08. 34th Euromicro Conference. IEEE, 2008
4. V. Cortellessa, F. Marinelli, and P. Potena, "An optimization framework for "build-or-buy" decisions in software architecture", Com-

puters and Operations Research, Elsevier Science, Vol. 35, No. 10, pp.3090-3106, 2008.

5. H.W. Jung, and B. Choi, "Optimization models for quality and cost of modular software systems", European Journal of Operational Research, Vol. 112, No.3, pp.613-619,1999
6. C.K. Kwong, T.J.F. Mu, and X.G. Luo, "Optimization of software components selection for component-based software system development", Comput. Ind. Eng. Vol. 58, pp. 618-624, 2010
7. F. Brito e Abreu, & M. Goulao, "Coupling and cohesion as modularization drivers: Are we being over-persuaded?", In Proceedings of the fifth European conference on software maintenance and reengineering, 2001.
8. G. Carlo, J. Mehdi, & M. Dino, "Fundamentals of software engineering", Prentice-Hall, Inc., 2001
9. Ian, S.:Software engineering. Addison-Wesley Longman Publishing Co., Inc., 2001
10. S. Parsa, & O. Bushehrian, "A framework to investigate and evaluate genetic clustering algorithms for automatic modularization of software systems", Lecture Notes in Computer Science, pp. 699–702, 2004.
11. R. Seker, A.J. van der Merwe, P. Kotze, M.M Tanik, & R. Paul, "Assessment of coupling and cohesion for component-based software by using Shannon languages", Journal of Integrated Design & Process Science, Vol. 8, No. 4, pp. 33–43, 2004.
12. W. Stevens, G. Myers, & L. Constantine, Structured design, IBM Systems Journal, Vol. 13, No. 2, pp. 115–139, 1974.
13. S. Sarkar, G.M. Rama, & A.C. Kak, "API-based and information-theoretic metrics for measuring the quality of software modularization", IEEE Transactions on Software Engineering, Vol. 33, No. 1, pp. 14–32, 2007.
14. X. Shen, Y. Chen, L. Xing, "Fuzzy Optimization Models for Quality and Cost of Software systems Based on COTS", In: Proceedings of the Sixth International Symposium on Operations Research and Its Applications (ISORA'06), Xinjiang, China, August 8-12, 2006, ORSC & APORC, pp. 312-318, 2006.
15. P. Gupta, M. Mehlawat, S. Verma, "A hybrid approach for selecting optimal COTS products", Lect. Notes Computer Sci. 5592, pp. 949-962, 2009.
16. P. Gupta, M. Mehlawat, G. Mittal, S. Verma, "COTS selection using fuzzy interactive approach", Springer-Verlang, pp. 273-289, 2010.
17. P. Gupta, S. Verma, M.K. Mehlawat, "A membership function approach for cost-reliability trade-off of COTS selection in Fuzzy Environment", Int. J. Reliab. Qual. Safety Eng., Vol. 18, No. 6, pp. 573-595, 2011.
18. P.C. Jha, V. Bali, S. Narula, M. Kalra, "Optimal Component selection based on cohesion & coupling for component based software systems under build-or-buy scheme", Journal of Computational Science 10.1016/j.jocs.2013.07.003, 2013.
19. P.C. Jha, S. Bali, U.D. Kumar, H. Pham, "Fuzzy optimization approach to component selection of fault tolerant software system", Memetic Computing, DOI.10.1007/s12293-013-0116-4, 2013.
20. ANSI/ IEEE, Standard Glossary of Software Engineering Terminology, STD-729-1991, 1991.
21. R.E. Bellman, and L.A. Zadeh, "Decision-making in a fuzzy environment", Management Science.17, 4, pp. B141–B164, 1970.
22. H. Thiriez, OR software LINGO. European Journal of Operation Research. 124, 655-656, 2000.

ABOUT THE AUTHORS

Dr. P.C. Jha obtained his Ph.D., M. Phil and M.Sc. degrees in Operational Research (OR) from the University of Delhi. He is the Reader in the Department of OR, University of Delhi. He has published more than 50 research papers in the areas of software reliability, marketing and optimization in Indian and international journals and edited books. He has guided master's projects, MBA and M. Phil dissertations and is supervising Ph.D. students in OR. His research interests include modeling and optimization in software reliability, marketing and supply chain management.

Vikram Bali is the Head of the Department of Information Technology at Rayat Bahra Institute Engineering and Bio- Technology, Mohali, Punjab. His area of interest lies in the area of Software Engineering. He has authored few books and articles in national and international conferences. He is a life member of CSI and ISTE

Sonam Narula has done her M.Sc. and M. Phil from the Department of Operational Research, University of Delhi. She is working as lecturer in Krishna Institute of Engineering and Technology, Ghaziabad. Her area of interest is in the field of engineering mathematics, optimization and software reliability. She has presented papers at national and international conferences.

Mala Kalra is working as assistant professor in the Department of Computer science & Engineering at National Institute of Technical Teachers Training & Research, Chandigarh. She has presented papers at national and international conferences. Currently is working in the area of cloud computing.