The Limitations of Current Decision-Making Techniques in the Procurement of COTS Software Components.

Ncube, C.; Dean, John

The Limitations of Current Decision-Making Techniques in the Procurement of COTS Software Components*

Ncube, C. and Dean, J.C.
February 2002

* published in Proceedings of the 1st International Conference on COTS-Based Software System, Orlando, FL. February 4-6, 2002. pp. 176-187. NRC 44926. (LNCS 2255, Springer).

The Limitations of Current Decision-Making Techniques in the Procurement of COTS Software Components

[1]Cornelius Ncube, [2]John C Dean

[1]Institute for Technological Innovation (ITI), Zayed University, P.O. Box 19282, Dubai, United Arab Emirates
Cornelius.Ncube@zu.ac.ae
[2]Software Engineering Group, Institute for Information Technology, National Research Council Canada K1A 0R6
John.Dean@nrc.ca

Abstract. The fundamentals of good decision-making are, first, a clear understanding of the decision itself and second the availability of properly focused information to support the decision. Decision-making techniques help with both these problems. However, the techniques should be thought of as aids to decision-making and not the substitutes for it. Numerous decision-making techniques have been proposed as effective methods of ranking software products for selection for use as components in large-scale systems. Many of these techniques have been developed and successfully applied in other arenas and have been either used directly or adapted to be applied to COTS product evaluation and selection. This paper will show that many of these techniques are not valid when applied in this manner. We will describe an alternate requirements-driven technique that could be more effective.

1 Introduction

One of the critical issues for the COTS-based development process is COTS component assessment and decision-making. In order to select or recommend a suitable required component, the evaluated alternatives must be ranked according to their perceived relative importance to meet the customer's requirements. Decision-making techniques have been used for this purpose. As in other activities, decision making in COTS selection may occur under conditions of (1) Certainty; (2) Risk-where each action has several possible outcomes for which the probabilities are known; and (3) Uncertainty – where each action has several possible outcomes for which the probabilities are not known' (Jha, 1997).

Therefore, making a decision that does not help to achieve the goal of selecting the required component can lead to long lasting user disappointments. Decision-making in component evaluation is a very complex process that combines probability judgments that may be affected by the evaluator's beliefs and underlying preferences. Figure 1 depicts the principles of decision-making in COTS component evaluation and selection that the decision-makers usually have to contend with, represented as a hierarchy of three levels taken from Saaty (1990). At the first level is the main goal for the decision making process (e.g. selecting a suitable component among the alternatives). At the second level there are some criteria for selecting the component. The suitable component will be judged by these criteria. At the third level are the actual alternative candidate components in which the criteria will be applied to achieve the main goal. The figure clearly shows how complex decision-making becomes if there are many components to compare and criteria to apply to each component.

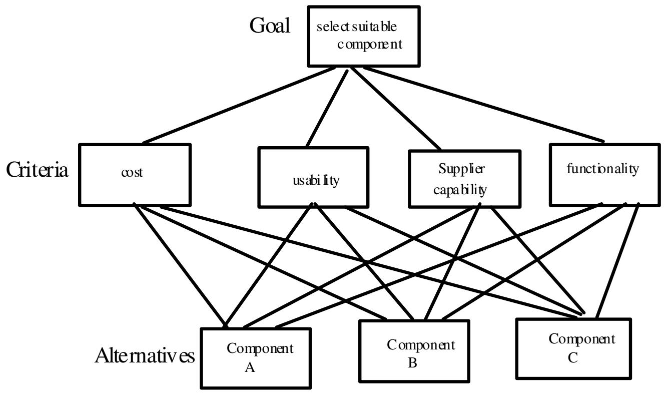

Fig. 1. Principles of a decision-making problem

2. General difficulties with current decision-making techniques

Some factors that give rise to problems in evaluating and assessing COTS software are:

- that there is a large number of component attributes or features that have to be considered;
- that various combinations of hardware platforms, operating systems and application software need to be considered;
- that there is rapid technological changes in all aspects of computing, the business environment and the needs of the users;
- that most users lack the technical expertise or time to develop criteria, measurements and testing procedures for performance assessments and to conduct the actual evaluations;
- that there are considerable variations in performance between the attributes of each component and across the components for each attribute.

Mistree & Allen (1997) further suggest that characteristics of decisions for selecting COTS software are governed by decision characteristics such as the following:

1. Selection decisions are invariably multileveled and multidimensional in nature;
2. Decisions involve information that comes from different sources;
3. Decisions are governed by multiple measures of merit and performance;
4. All the information required to make a decision may not be available;
5. Some of the information used in making a decision may be hard, that is, based on scientific principles and some information may be soft, that is, based on the selectors judgment and experience;
6. The problem for which a decision is being made is invariably loosely defined and open and is characterized by the lack of a singular, unique solution. The decisions are less than optimal and represent satisfying solutions, that is, not the 'best' but 'good enough',

Tkach & Simonovic, (2000) furthermore suggest that selecting the best COTS product from a number of potential alternatives is a complex decision making process that may include conflicting quantitative, qualitative criteria and multiple decision-makers. Currently, a number of decision-making techniques that can be used in COTS component evaluation and assessment are available in the market. However, almost all of these techniques are found not to be suitable for assessing software components due to their fundamental underlying assumptions in their judgment value system. Most currently existing traditional decision-making approaches rely on compensatory models such as the linear weighted score model which sums the weighted ratings of the component's capability attributes to arrive at a single score for each component. The end result is either an aggregate total score for the component or a group of scores representing various attributes of the component. However, aggregate total scores tend to mask individual attributes of the component that may represent particular strengths or weaknesses in a component. These models are problematic in that they can permit very good performance on one attribute to offset poor performance on another.

3. Current proposed decision-making techniques

The techniques summarized below all attempt to consolidate the evaluation results or rank the alternatives in one way or another. The following sections give brief discussions of four such decision-making techniques.

3.1 The Multi-Attribute Utility Theory (MAUT)

The MAUT decision-making model deals with choosing among a set of alternatives which are described in terms of their attributes (MacCrimmon, 1972). A typical multi-attribute decision problem is choosing among COTS software components described by such attributes as cost, usability, functionality, size, portability, supplier capability, etc. To deal with the multi-attribute situations, the MAUT technique requires information about:

- the decision-makers preference among values of a given attribute (i.e. how much does s/he prefer a commercial database over a proprietary database), and;
- the decision-maker's preference across attributes (i.e. how much important is the database than cost). A marginal value function is associated with each criterion and a global value function is computed in an additive or multiplicative form.

The MAUT technique asks the decision-maker for an assessment on the strength of preferences. The decision may be reduced to a number of independent attributes that involve making trade-offs between different goals or criteria. MAUT uses a reductionist approach to a problem and it is up to the decision-maker to split the problem into a number of dimensions that are perceived to be independent. This independence is essential for MAUT because without it, certain attributes could be over represented in the final result. This is the fundamental weakness of the method.

3.2 The Multi-Criteria Decision Aid (MCDA)

The MCDA approach is in the category of the utility theory. Morisio et al (1997) proposes the following advantages for using MCDA in COTS component evaluation and selection:

- the MCDA approach makes explicit reference to the decision process so as to take into account the different actors involved in the process, their different objectives and the partiality of the information provided;
- the MCDA approach allows handling judgments based on qualitative, partial information and the subjective nature of the problem of evaluating and selecting software components. This is done by adopting appropriate specific techniques to help in the decision making process (including multi-attribute utility theory, multi-objective interactive techniques and out-ranking technique) and provide the evaluator with both formal and substantial reasons for any choice;

the MCDA approach combines strictness (non-redundancy of the set of criteria, appropriateness of the aggregation procedure, etc) with flexibility, different problem statements, different aggregation techniques, custom evaluation attributes and measures.

With MCDA, a list of criteria that the component should meet is established first, then scores are assigned to each criterion based on its relative importance in the decision. Each alternative is then given a number of scores according to how it fully meets the criterion. For the scores, a scale of 1 to 5, or 1 to 7, etc can be used. An example is shown in Table 1.

In the example, component A is rated 25 out of 30 points for the "cost" criterion, while component C is rated a little less favourable. Once all the alternatives have been assigned their points for each criterion, all points for each alternative are added together and the alternative with the highest total is the one chosen. In the example, this would be component A. The main weakness of the method is that if the criteria set is large, it quickly becomes very complicated.

Criteria	Possible Points	Component A	Component B	Component C
Cost	30	25	20	15
Functionality	40	35	10	20
Supplier	20	15	5	10
Usability	10	5	3	2
Total	100	80	38	47

Table 1. A list of criteria that the component should meet and scores assigned to each criterion.

3.3 Weighted Score Method (WSM) or Weighted Average Sum (WAS)

The WSM/WAS is an aggregation technique and the most commonly used technique in many decision-making situations. Its 'weights' are trade-offs between the criteria; i.e. they are ratios between the scales of each criterion. 'Criteria are defined and each criterion is assigned a weight or a score' (Kontio, 1996).

The scales themselves represent preferences relative to each attribute. The WSM/WAS technique is a fully compensatory model in that each preference relative to a criterion can be totally compensated for by a countervailing preference on another criterion. This trade-off between criteria may result in any big difference that may exist being compensated for, so that an indifferent situation is created instead of the actual incomparability situation (Morisio et al 1997). This scenario is one of the many weaknesses of this technique. Although weighting methods seem very diverse, they all have the following characteristics:

- a set of available alternatives with specified attributes and attribute values;
- a process for comparing attributes by obtaining numerical scalings of attribute values (intra-attribute preferences) and numerical weights across attributes (inter-attribute preferences);
- an objective function for aggregating the preferences into a single number for each alternative;
- a rule for choosing or rating the alternatives on the basis of the highest weight.

Table 2 below shows an example application of the WSM and its limitations. The criteria weights were assigned using a scoring method by assigning a value of between 1 and 5 to each criterion. The overall score of each alternative was calculated using the following formula (Kontio, 1996):

$$score_a = \sum_{j=1}^{n} (weight_j * score_{aj})$$

where a = alternative, n = number of criteria, j = criteria

The problem with the method is in assigning the scores. For example, the security and compatibility could be interpreted as twice as important as ease of use, whereas in reality this might not be the case.

Criteria	Weight Score	Component A	Component B	Component C
Ease of use	2	3	3	3
Compatibility	4	1	5	2
Cost	3	3	5	1
Functionality	5	4	4	3
Security	4	1	2	5
Supplier	5	2	5	3
Score		53	94	67

Table 2. Example of the weighted score method.

However, WSM/WAS techniques have additional serious limitations that are often ignored when they are applied in COTS component evaluation and assessment (Kontio 1996):

- As the Weighted Score Method produces real numbers as results, these results can easily be interpreted as if they represent the true differences between the alternatives. In actual fact, the resulting scores only represent relative ranking of the alternatives and the differences in their value does not give any indication of their relative superiority;
- Assigning weights for the criteria is very difficult when the number of criteria is large. If the number of attributes is large, it is very difficult to mentally cope with the dependencies between individual attributes. Assigning scores instead of weights is even more limiting because it effectively sets predetermined lower and upper limits to the weights that can be assigned to the criteria;
- It is very difficult to define a set of criteria and their weights so that they are either independent from each other or if they overlap, their weights are adjusted to compensate for the overlap.

3.4 The Analytical Hierarchy Process (AHP)

The AHP (Saaty 1990) is a multiple criteria decision-making technique that is based on the idea of decomposing a multiple criteria decision-making problem into a hierarchy. The decisional goal is decomposed into a hierarchy of goals and ratio comparisons are performed on a fixed ratio scale. The overall priorities are computed using an eigenvalue technique on the comparison matrix. The factors are arranged in a hierarchic structure descending from an overall goal to criteria, sub-criteria and alternatives in successive levels as shown in figure 1. At each level of the hierarchy, the relative importance of each component attribute is assessed by comparing them in pairs. The rankings obtained through the paired comparisons between the alternatives are converted to normalised rankings using the eigenvalue method, i.e. the relative rankings of alternatives are presented in ratio scale values which total to one as shown in the Priority Vector column of Table 3. The technique suggests that comparing criteria in pairs result in more reliable comparison results and that in this way, it is possible to avoid the problem of having to assign absolute values to alternatives, but only their relative preferences or values are compared. A typical application of the AHP method is shown in Table 3. Functionality is shown to have the highest total score and priority vector and therefore ranked more important.

			Level 1 Priority Vector				
	Cost	Functionality	Usability	Technical	Supplier	Total Scores	Priority Vector
Cost	1	4	5	4	6	20	0.339
Functionality	0.25	1	7	7	7	22.25	0.377
Usability	0.2	0.143	1	5	3	9.343	0.158
Technical	0.25	0.143	0.2	1	4	5.593	0.095
Supplier	0.167	0.143	0.333	0.25	1	1.893	0.032
						59.079	1

Table 3: An example of applying the AHP method.

However, the AHP technique has some fundamental drawbacks when applied to COTS component evaluation. One of its main problems is that it assumes total independence between the component attributes, i.e. in order to do a pair-wise comparison, the technique assumes that the component attributes/features are independent of each other and this is rarely the case with software requirements. Also, especially for large complex systems, it is difficult to apply the AHP technique as its calculation model involves a very high number of pair-wise comparisons. The large number of individual assessments is also one of its main weaknesses. Even if the overall duration of the assessment sessions are not very long, the repetitive assessments cause tiredness and boredom. Furthermore, the assumption that there should be complete comparability and the imposition of the ratio scales at all levels of the hierarchy is very demanding (Kontio 1996).

4. Limitations of the current decision-making techniques

As Tkach & Simonovic (1997) state, most current decision-making techniques are characterized by a great diversity with three main groups: out-ranking techniques, multi-attribute utility techniques, and mathematical programming techniques. Out raking techniques require pair-wise or global comparisons among alternatives, which is not practical when the number of alternatives is large. Multi-attribute utility techniques rely on linear additive or simple multiplicative models for aggregating single criterion evaluations. They are not appropriate for the analysis of complex software systems. Mathematical programming techniques, on the other hand, are used in continuous context. They identify solutions that are closest to the ideal solution as determined by some measure of distance. The solutions identified to be closest to the ideal solutions are called compromise solutions and constitute the compromise set. The ideal solution is one that provides the extreme value for each of the criteria considered in the analysis. The distance from the ideal solution for each alternative is measured by the distance metric. This value, which is calculated for each alternative solution, is a function of the criteria values themselves, the relative importance of the various criteria to the decision makers, and the importance of the maximal deviation from the ideal solution.

Therefore the selection of a decision-making technique for COTS evaluation and assessment should be done with care. Current decision-making techniques do not adopt a requirement-driven approach to component selection decisions and are therefore inadequate and not suitable for the COTS based decision-process. The fundamental criticism of these techniques is their underlying theory and their value judgment system, i.e. where the values come from.

For example, the idea of producing a single number from the individual scores (e.g. by some arithmetic combination formula such as weighted ranking) is misleading because many different combinations of numbers can produce the same aggregate score. Furthermore, certain features may attract higher average scores than others because an assessor may understand them better and be more able to recognize support in the component. There are also deeper reasons concerning the nature of the ordinal scales that are usually used to assess component features. For instance, a score of 4 is not necessarily twice as good as a score of 2.

5. An Alternative Approach

There is an alternative approach that one can consider that is requirements-driven and that mitigates the loss of detail encountered when employing a weighted aggregation approach. The approach applies the principles of gap analysis to evaluation and allows selection based on the cost of bridging the gap. This requires a novel viewpoint where we examine products, not for the purposes of eliminating them should they not meet the requirements, but for the purpose to attempting to determine what capabilities the products lack in terms of the requirements context. We then analyze these capability deficiencies to determine the cost of implementing supplemental functionality for each product or product set under evaluation. These are the fulfillment costs and there will be a set of these for each evaluation that we undertake. The fulfillment costs are used as a basis for selection of an appropriate product set.

5.1 Gap Analysis

Gap Analysis is a technique adapted for our purposes from environmental GAP analysis. The Dictionary of Business [Dict. Bus] defines gap analysis is "A methodical tabulation of all the known requirements of consumers in a particular category of products, together with a cross-listing of all the features provided by existing products to satisfy these requirements." In order to apply gap analysis to COTS product selection we still need to assess a product's capabilities against requirements. The determination of the gap requires that we highlight and record those capabilities that are not fulfilled by the product under consideration. This can be accomplished by constructing a 2-dimensional matrix of requirements versus products. The cells of the matrix would contain information about the gap. Figure 2 shows a simple example of such a matrix.

Product / Requirement	P1	P2	P3
R1	Limited Java support	Complete solution	Complete solution
R2	Inaccurate math	Precision only to 2 decimals	No Math engine
R3	10%< required reliability	No reliability figures available	Complete solution
R4	Complete solution	Vendor out of country	Vendor Canadian
R5	Complete solution	Linux platform required	Windows only

Fig. 2. A Gap Analysis Evaluation Matrix example.

One matrix would be required for each evaluation conducted and, in a large scale system employing multiple COTS products, the number of evaluations, and thus the number of matrices derived, could be significant. However for each individual evaluation it is assumed that the number of competing products examined in detail would be in the range of three to five so that the internal complexity of the matrices is not a factor. A further assumption is made that the evaluations can be conducted relatively independently. The implication of this assumption is that the system requirements have been stated in such a way as to minimize overlap.

There are three potential results that might be obtained during an evaluation. The first is the trivial case in which the capabilities of the product and the requirements match exactly as shown in Figure 3(a).The second is the case where the product partially fulfills the requirements and does not provide any inherent capabilities that exceed the requirements as in Figure 3(b). The third case is where the product fulfills some or all of the requirements but also incorporates capabilities that fall outside the boundaries of the original system needs as shown in Figure 3(c).

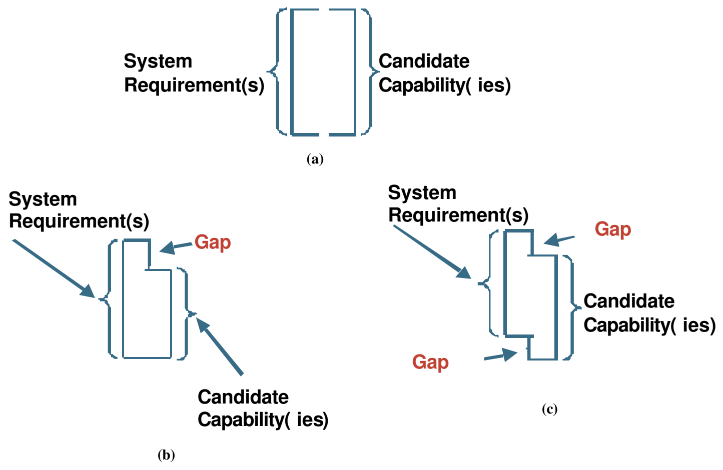

Fig. 3. Gap analysis results

5.2 Fulfillment Cost Calculations

Once the gaps between product and system requirements have been determined a further step is required to establish the cost of reducing the gap to an acceptable solution. At this point there are a number of strategies that could be followed, all of which depend on the parameters and nature of the gap. These strategies can be derived from Figure 3 as well.

In the trivial case of an exact match between requirements and products capabilities the cost of gap reduction is zero. In the case where a product does not fully meet a requirement and that requirement is firm, i.e. it cannot be restated or relaxed, then the strategy which must be followed is to determine the cost of adding functionality to account for the product's deficiencies with respect to the requirement. In a COTS-based system this must be accomplished without modifying the COTS product itself. Thus we are restricted to the strategy of implementing a custom code solution to the deficiencies.

A third case occurs when the product does not fully meet a requirement but that requirement is less rigid. In this case one might negotiate a change in the requirement so that the product's capabilities and the requirement match more closely. The most common situation would be that a combination of requirements adjustment and adding functionality is required to meet the deficiency. The cost of employing this product would be calculated from both the cost of negotiation as well as the cost of custom code implementation.

A final case is the situation where the product's capabilities fall outside the boundaries established by the known requirements. Here we have the situation where we either accept the excess capability, (i.e. the capability is a benefit), and provide it as a part of the system, or we must attempt to inhibit access to that capability (i.e. the capability is a liability) from within the system. There are, of course, various degrees of acceptance or rejection that can be negotiated as with the previous case. The costs here are calculated from the cost of custom coding, the cost of adding beneficial functionality and the cost of restating the requirements.

If we consider the matrix shown of Figure 2 we can visualize a transformation of the information contained in individual cells into a fulfillment cost for each product-requirement pair.

5.3 Aggregation

There remains the issue of aggregation of the results of the cost calculations. Gap analysis leads to the creation of multiple matrices corresponding to the number of evaluations performed. Each of these matrices is transformed into a fulfillment cost matrix during the fulfillment cost determination. For each matrix we can find a preferred product There is a final step required to select the optimal combination of these products with which to construct the system. This problem has already been described by Briand [Briand 98] as a operational research optimization problem. The goal is to find the optimal path through the matrix series based on selecting the most suitable product from each matrix.

6. Conclusions

Therefore, the concluding view of this paper is that most of the current decision-making techniques available are not adequate for component-based evaluations and assessments due to their underlying assumptions and their judgement value systems. There is a need for new requirements-driven decision-making techniques for the component-based development paradigm. The alternative paradigm presented here

provides the appropriate relationship between requirements and products in the system context has the potential to provide accurate recommendations for product selection. It can be adapted to systems of varying complexity and size.
Research into fulfillment cost determination is currently underway.

References

1. Dictionary of Business, Oxford University Press, © Market House Books Ltd 1996.
2. Briand, L.C., "COTS evaluation and selection," Proc. International Conference on Software Maintenance (Bethesda, Nov, 1998) IEEE Comput. Soc, Los Alamitos, pp 222-223.
3. Nand K. Jha, Decision-Based Design Value Design, Position paper, Decision Based Design Open Workshop 1997.
4. Farrokh Mistree and Janet K. Allen, Optimization in Decision-Based Design, Position Paper, Decision-Based Design Workshop, Orlando, Florida, April 1997.
5. Robert J. Tkach & Slobodan P. Simonovic, A New Approach to Multi-Criteria Decision, Journal of Geographic Information and Decision Analysis, vol. 1, no. 1, pp. 25-43.
6. Anderson E. E. (1989). A Heuristic for Software Evaluation and Selection, Software Practice and Experience, vol 19(8), August 1989.
7. Fenton N. (1994). Out Ranking Method for Multi-Criteria Decision Aid: with emphasis on its role in systems dependability assessment, Centre for Software Reliability, City University, London, UK.
8. Frair L. (1995). Student Peer Evaluations Using the AHP Method, Foundation Coalition, Department of Industrial Engineering, University of Alabama Tuscaloosa, 1995
9. Kontio J. (1996). A Case Study in Applying a Systematic Method for COTS Selection, Proceedings of the 18th International Conference on Software Engineering, IEEE Computer Society Press.
10. MacCrimmon R. K. (1972). Proceedings of Multiple Criteria Decision Making, University of South Carolina, October 26-27, 1972
11. Morisio M and Tsoukias A. (1997). A Methodology for the Evaluation and Selection of Software Products, Dipartimento di Automatica e Informatica, Politicnico di Torino, Italy.
12. Saaty L.T. (1990). The Analytic Hierarchy Process (AHP): How to make a decision, European Journal of Operational Research, 48(1990), 9-26