A Case Study in Applying a Systematic Method for COTS Selection

Jyrki Kontio[1]

University of Maryland
Department of Computer Science
A.V.Williams Building
College Park, MD 20742, U.S.A.
Email: jkontio@cs.umd.edu
http://www.cs.umd.edu/users/jkontio/

Abstract:

This paper describes a case study that used and evaluated key aspects of a method developed for systematic reusable off-the-shelf software selection. The paper presents a summary of the common problems in reusable off-the-shelf software selection, describes the method used and provides details about the case study carried out. The case study indicated that the evaluated aspects of the method are feasible, improve the quality and efficiency of reusable software selection and the decision makers have more confidence in the evaluation results, compared to traditional approaches. Furthermore, the case study also showed that the choice of evaluation data analysis method can influence the evaluation results.

Keywords: *software reuse, COTS, multiple criteria decision making*

1. Introduction

Reuse has been considered an important solution to many of the problems in software development. It has been claimed to be important in improving productivity and quality of software development [1,5,14,21,27,31] and significant benefits have been reported by many organizations [12,20]. A large volume of research has produced several useful tools to support reuse [14,27,30] but it is widely believed that successful reuse is not only dependent on technical issues, it also requires the solving of organizational, motivational and legal issues [2,4,31-33]. It has been argued that an important characteristic of the infrastructure supporting reuse is the existence of a "marketplace" that both provides access to reuse producers and consumers as well as provides a mechanism to transfer benefits between the parties [6,19,20,34].

Many organizations have implemented systematic reuse programs [12] which have resulted in in-house libraries of reusable components. The increased commercial offering of embeddable software components, standardization of basic software environments (e.g., MS-Windows, UNIX), and popularization of Internet have resulted in a new situation for reusable software consumers: there are many more accessible, potential reuse candidates. Given the high interest in reuse and motivation to the use of commercially available software, many software development projects include the evaluation and selection of reusable components as an important activity in the project, with a high potential impact on the product and project objectives. According to our observations in many organizations, the selection process typically is not defined, each project finds its own approach to it, often under schedule pressure, and there are no mechanisms to learn from previous selection cases. Yet the selection of the right reusable component is often a non-trivial task and requires careful consideration of multiple criteria and careful balancing between application requirements, technical characteristics and financial issues. It seems that there is a lot of potential for non-optimal or inconsistent COTS reuse decisions.

However, the issues and problems associated with the selection of suitable reusable components have not been addressed much in the software reuse community. Poulin et al. present an overall selection process [20] and include some general criteria for assessing the suitability of reuse candidate [29]. Some general criteria have been proposed to help in the search of potential reusable components [21,22]. Boloix and Robillard recently presented a general framework for assessing the software product, process and their impact on the organization [7]. However, none of this work is specific to COTS selection and the issues of how to define the evaluation criteria are not addressed.

[1] This work has been supported by the Hughes Information Technology Corporation and the EOS Program.

0270-5257/96 $5.00 © 1996 IEEE
Proceedings of ICSE-18

We have developed a method that addresses the selection process of packaged, reusable software, or COTS[2] as we refer to it in this paper, and provides detailed guidance for this process. The method, called OTSO[3], supports the search, evaluation and selection of reusable software and provides specific techniques for defining the evaluation criteria, comparing the costs and benefits of alternatives, and consolidating the evaluation results for decision making. The OTSO method has been documented separately [16] and this paper focuses in two specific aspects of the OTSO method: evaluation criteria definition and evaluation data analysis. These two aspects were evaluated in a case study in an active project with the Hughes Information Technology Corporation.

The structure of this paper is as follows. In the next chapter we will discuss the potential problems in reusable software selection. In chapter 3 we give an overview of the OTSO method and present the two aspects that are addressed in the case study in detail. Chapter 4 will present the case study: its objectives, how it was carried out and its results. Finally, the conclusion chapter discusses the relevance of our results.

2. Problems in Reusable Software Selection

Based on our observations of the practices in several organizations, the COTS software selection process is prone to some potentially serious problems.

The task of COTS selection is often assigned under schedule pressure and, as many organizations are just now increasing their COTS usage, evaluators are often first-timers at the task. They may not have the time or experience to plan the selection process in detail and, therefore, they may not use the most appropriate methods in the selection process. Furthermore, when the selection process is not defined, it is reinvented each time, it is performed inconsistently and learning from previous cases is difficult. An organization that has to evaluate COTS frequently will benefit from a well-defined, repeatable selection process. Such a process will make the planning easier, allow accumulation of experience, make different selection processes consistent, support the use of validated evaluation methods and increase the efficiency of evaluation.

Another common problem in COTS selection is the lack of attention to application requirements. This may be somewhat understandable mistake as the COTS selection often takes place early in the project where requirements may be fuzzy. At the same time the technical characteristics of COTS are very concrete. Unless this is consciously avoided, the combination of fuzzy requirements and easy access to some technical information easily lead to bottom-up approach to evaluation: it is much easier to evaluate what you happen to see than taking the effort to determine and search for important characteristics. This may result in an overemphasis of some technical characteristics that may not be important from the point of view of the end user. For example, the evaluators may find it easy to compare CPU usage and some advanced feature of COTS alternatives although they may be irrelevant from the point of view of the real application requirements.

Furthermore, when the evaluation criteria are defined, they are seldom well defined. The criteria may be listed and documented with one or two words, e.g., "memory usage", "ease of use" or "reliability". This leaves the exact meaning of each criterion very open to each evaluator's own interpretations. This problem becomes more relevant when there are more than one evaluator. Operational definitions [9] for all criteria are required so that all COTS alternatives are compared against a common yardstick.

Once the evaluation of COTS alternatives has been done, a common approach to consolidating evaluation results is to use some kind of weighted scoring method (WSM) to rank the alternatives. The WSM method is typically applied in the following fashion: criteria are defined and each criterion is assigned a weight or a score. In the case of using weights, they may be normalized so that their total is one. If "scoring" is used, this is done, e.g., by assigning a "weight score" between one and five for each criterion. Then, each alternative is given a score on each criterion. The score for each alternative is calculated by the following formula:

$$\text{score}_a = \sum_{j=1}^{n} (weight_j * \text{score}_{aj})$$

where subscript a represents an alternative and n represents the number of criteria.

The application is the WSM is straight-forward and presents results that intuitively make sense. However, the WSM as presented above has several serious limitations that are often ignored when it is applied. First, as the WSM produces real numbers as results, these are easily interpreted as if they represented truly the differences between alternatives in ratio or distance scales. However, this would be true only if all the criteria scores have been

[2] Originally, COTS stands for commercial off-the-shelf software. In this paper we use the term to refer to software packages that have been developed or are suitable for reuse, commercial or in-house. We often use the term as a noun and it should be "read" as *"off-the-shelf software"*.

[3] OTSO stands, among other things, for the Off-The-Shelf Option.

given using distance or ratio scale. Our experience strongly suggests that this is rarely the case. If even one of the criteria was rated on an ordinal scale score, the resulting scores only represent relative ranking of the alternatives and the differences in their value do not give indication of their relative superiority.

Second, assigning of weights for the criteria is very difficult when the number of criteria is high. It seems that people are able to deal with less than ten alternatives at a time [23]. With large number of issues to keep in mind people have difficulties in mentally coping with the dependencies of individual factors. The assigning of scores instead of weights is even more limiting. If this method is used, this effectively sets lower and upper limits to the weights one can assign to criteria. For example, if a scale of 1 to 5 is used and we have 20 criteria, the minimum and maximum weights given to a single criterion can be calculated by

$$\min = \frac{1}{\sum_{j=1}^{20} weight_j},$$

$$\max = \frac{5}{\sum_{j=1}^{20} weight_j}$$

In other words, the weight range possible for each criterion is within 1.04% and 4.17%. In practice the range is even smaller as weight scores are likely be more distributed than in the extreme cases used in the above calculations. In our case study the maximum weight happened to be 3.4% with the WSM. The problem with this method is that in practice some of the criteria often are much more important than the maximum would allow.

Third, it is rather 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 overlapping areas.

The weighted scoring method has also been criticized in the DSS literature [28].

The problems in the COTS selection can be summarized in the following main points:

- lack of a defined, systematic, and repeatable process,
- potential disregard of the application requirements, and
- misuse of data consolidation methods in decision making.

Given the increasing role of COTS in software development, we believe that the above issues need to be addressed to ensure more effective use of COTS in software development.

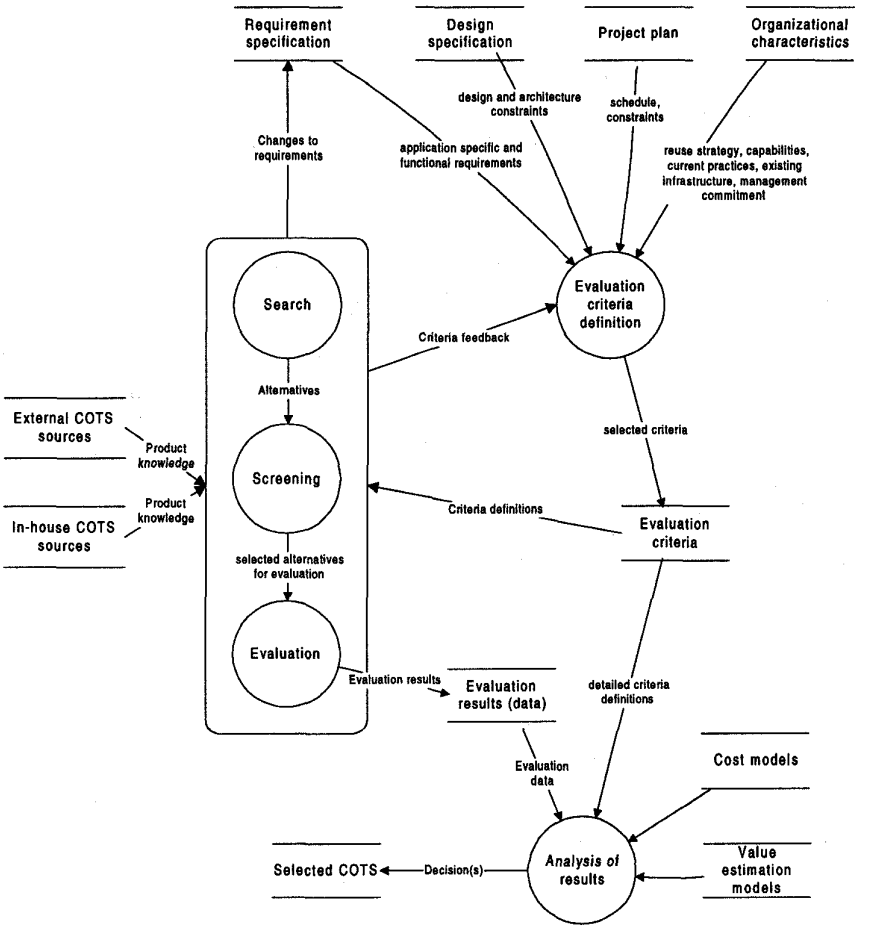

Figure 1: Overview of the OTSO method process

3. The OTSO Method

The OTSO method was developed to facilitate a systematic, repeatable and requirements-driven COTS selection process. The main principles of the OTSO method are the following:

- explicit definitions of tasks in the selection process, including entry and exit criteria
- incremental, hierarchical and detailed definition of evaluation criteria

- a model for comparing the costs and value associated with each alternative, making them comparable with each other
- use of appropriate decision making methods to analyze and summarize evaluation results

The full OTSO method has been documented separately [16] and it is not explained in detail in this paper. However, the two specific aspects of the method -- evaluation criteria definition and analysis of evaluation data -- are explained in more detail as they were addressed in our case study. These two aspects are represented by the two right-most processes (circles) in Figure 1.

3.1 Evaluation criteria definition

The evaluation criteria definition process essentially decomposes the requirements for the COTS into a hierarchical criteria set. Each branch in this hierarchy ends in an *evaluation attribute*: a well-defined measurement or a piece of information that will be determined during evaluation. This hierarchical decomposition principle is analogous to the GQM method [3]. The evaluation attributes should have clear operational definitions so that consistency can be maintained during evaluation.

The criteria set is specific to each COTS selection case but most of the criteria can be categorized into four groups: functional requirements for the COTS; required quality characteristics, such as reliability, maintainability and portability [15]; business concerns, such as cost, reliability of the vendor, and future development prospects; and relevant software architecture, such as constraints presented by operating system, division of functionality in the system or specific communication mechanisms between modules.

It is possible to identify three different subprocesses in the definition of evaluation criteria. Figure 2 presents these processes graphically using the modified dataflow

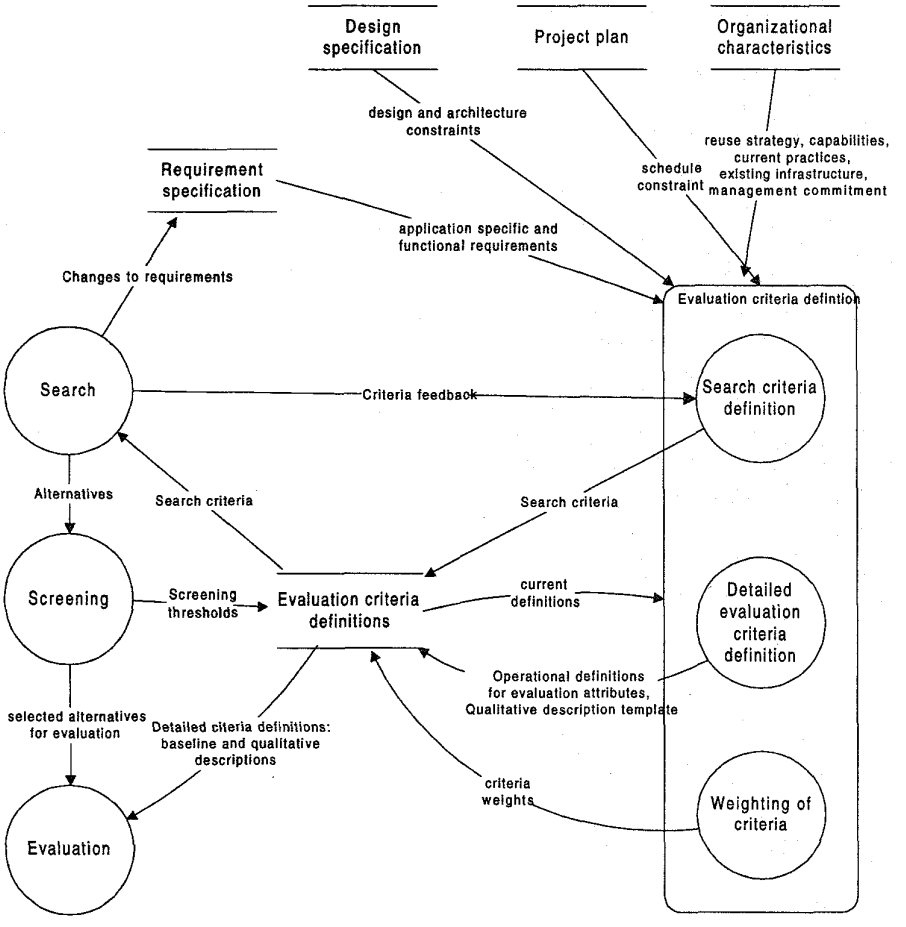

Figure 2: Evaluation Criteria Definition Process

diagram (DFD) notation. First, when the available alternatives are searched and surveyed it is necessary to define the main search criteria and the information that needs to be collected for each alternative. The search criteria are typically based on the required main functionality (e.g., "visualization of earth's surface" or "hypertext browser") and some key constraints (e.g., "must run on Unix and MS-Windows" or "cost must be less than $X"). An effective way to communicate such requirements is to use an existing product or COTS as a reference point, i.e., defining the functionality search criteria as "look for COTS that are similar to our prototype".

The search of alternatives should try to cover breath more than depth. It is enough to define the survey criteria broadly so that the search is not unnecessarily limited by too many constraints. The search phase uses the criteria and determines the "qualifying thresholds", which are in deciding which alternatives are selected for closer

evaluation. These threshold values will be documented together with the criteria definitions.

The search criteria, however, often are not detailed enough to act as a basis for detailed technical evaluation. Therefore, the criteria will need to be refined and formalized before initiating the technical evaluation. Without such operational definitions it is difficult to conduct a consistent and systematic evaluation, let alone consolidate the evaluation results for decision making. We have defined a template for evaluation attribute definition that helps in defining criteria and evaluation attributes in adequate detail. The template is presented in Table 1.

3.2 Analysis of Qualitative Evaluation Data

The OTSO method relies on the use of the Analytic Hierarchy Process (AHP) for consolidating the evaluation data for decision making purposes. The AHP technique was developed by Thomas Saaty for multiple criteria decision making situations [23,24]. The technique has been widely and successfully used in several fields [25], including software engineering [10] and software selection [13,18]. It has been reported to be an effective technique in multiple criteria decision making situations in several case studies and experiments [8,11,25,28]. Due to the hierarchical treatment of our criteria, AHP fits well into our evaluation process as well. AHP is supported by a commercial tool that supports the entering of judgments and performs all the necessary calculations [26].

The AHP is based on the idea of decomposing a multiple criteria decision making problem into a criteria hierarchy. At each level in the hierarchy the relative importance of factors is assessed by comparing them in pairs. Finally, the alternatives are compared in pairs with respect to the criteria. The following are the main steps in applying AHP [10,24]:

1. Define a hierarchy of factors that influence the decision, resulting in a hierarchical structure of factors that have alternatives as the leaf nodes in the hierarchy
2. Define the importance of factors on each level
3. Define the preferences of alternatives
4. Check the consistency of rankings and revise the ranking if rankings are too inconsistent
5. Present the results of the evaluation, the alternative with the highest priority being the one that is recommended as the best alternative.

The rankings obtained through paired comparisons between the alternatives are converted to normalized rankings using the eigenvalue method [24], i.e., the relative rankings of alternatives are presented in ratio

Item name	Description
Heading	Heading for each evaluation attribute acts as a unique identifier.
Definition	A definition of the evaluation attribute.
Rationale	Description of the rationale for the evaluation attribute and how it relates to the evaluation criteria.
Scale	The scale or type of description used. • *Free format description:* The evaluation attribute will be documented with a free format description. • *List:* A list of features, characteristics, functions etc. Is produced. • *Structured description:* There is a template or a checklist that defines what should be described for each alternative. • *Nominal:* Classes are identified but they are not ordered. • *Ordered:* Classes are identified and they are ordered. • *Interval:* The scale has meaningful interpretation of distance between entities, but their ratios cannot be calculated, i.e., "there is no meaningful zero point". • *Ratio:* Entities can have ratios, "zero is a meaningful concept". • *Absolute:* The number of entities is counted.
Unit/classes	Definition of the unit of measure or the classes used, which ever is applicable.
Screening rule	Definition of a possible level that is required for an alternative to be selected for detailed evaluation. This field is used for documenting which criteria were used in the screening phase.
Baseline	Baseline is the minimum required level of functionality and features that the application must satisfy when it is delivered. See reference [16] for details.
Qualitative description	Guidelines how additional information about the evaluation attribute should be documented.
Source	How the value for the evaluation attribute can be determined for each alternative.
Priority	Description of how important the particular evaluation attribute is. E.g. required, recommended or optional.

Table 1: Evaluation attribute definition template

scale values which total one.

Saaty argues that hierarchies are a natural way for humans to organize their view of the world and they represent real world phenomena well [24]. Criteria and alternatives are compared in pairs, which results in more reliable comparison results. This way we are able to avoid the problem of having to assign absolute values to alternatives, only their relative preference or values are compared.

From our perspective the main advantage of AHP is that it provides a systematic, validated approach for consolidating information about alternatives using multiple criteria. AHP can be used to "add up" the characteristics of each alternative. Furthermore, an additional benefit of AHP is that we can choose the level of consolidation. We recommend that consolidation is only carried out to the level that is possible without sacrificing important information. On the other hand, some consolidation may be necessary in order not to overflow the decision makers with too much detailed, unstructured information.

The evaluation of alternatives in the OTSO method concentrates in producing consistent data about the alternatives. We deliberately want to separate the analysis of this data from producing the data. This allows the use of appropriate techniques in evaluation data analysis for decision making.

The weighting of alternatives is done using the AHP method, preferably using a supporting tool [26]. Preferences are collected and consolidated to the level stakeholders prefer. The AHP allows the consolidation of all qualitative information and the financial information into a single ranking of alternatives. However, we believe that this would condense valuable information too much. Instead, we recommend that information about the evaluation is consolidated to a level where a few main items remain and stakeholders can discuss their impact and preferences verbally. The full consolidation can be done at the end as a sanity check, if desired. The selection of what are the right main items depends on the reuse goals and the criteria hierarchy used.

4. Case Study

4.1 Case Study Objectives and Arrangements

We have applied the OTSO method in two case studies. This paper reports the experiences from the second case study. Both case studies were carried out with Hughes corporation in the EOS program being developed for NASA. The EOS program is developing systems that integrate Earth environmental data from satellites and makes this data available to scientists all over the world. The project required a hypertext browser to provide easy-to-use access to the system.

The objectives of the case study were to (i) validate the feasibility of the evaluation criteria definition approach in the OTSO method and (ii) compare the AHP and WSM methods for analyzing the data. Our hypotheses were that the more detailed evaluation criteria definition will result in more effective, consistent and reliable evaluation process. We also expected that the AHP method would give decision makers more confidence in the decisions they made.

A total of over 48 tools were found during the search for possible tools. Based on the screening criteria, four of them were selected for hands on evaluation: Mosaic for X, Netscape, Webworks for Mosaic, and HotJava. These tools were each evaluated by two independent evaluators, most of whom evaluated two tools. This arrangement allowed each evaluator to have more than one reference point and enabled discussion and comparison of tools. All evaluators were Hughes project personnel and two of them acted as "key evaluators", i.e., they had previous experience in COTS selection and they coordinated much of the evaluation and analysis in the case study.

The evaluation criteria were derived from existing, broad requirements. However, it was soon discovered that the level of detail in the documented requirements was clearly insufficient for detailed technical evaluation of the COTS selected. The requirements had to be elaborated and detailed substantially during this process.

The hands-on evaluation was based on the evaluation criteria defined earlier and evaluators wrote reports addressing all the evaluation criteria for each tool. The results were discussed in a joint meeting with all evaluators, and all open issues or conflicting evaluation results were logged and assigned as action items.

In the joint evaluation meeting the differences between tools were first discussed qualitatively. The scores for the WSM were also assigned for each tool were assigned after this discussion. In most cases only the "high" and "low" values were explicitly defined for the scores verbally before assigning a score. Note that the scores were, therefore, mostly based on ordinal scales.

The qualitative differences between tools were documented separately [17]. This report represents the "raw" differences between the tools without any judgments of their importance or ranking to each other.

After the evaluation meeting we waited for two weeks before continuing with the analysis, partially because of the logistics of completing the action items on missing information, partially to allow the two key evaluators to "forget" the numerical scores that were assigned to alternatives.

The final step in the analysis was to complete the WSM by defining the criteria weights and to apply the AHP method. The WSM criteria weights were assigned using a scoring method, i.e., assigning a value between one and five to each criterion. For this purpose, a "flat" table of criteria was produced from our originally hierarchical criteria definition. The two key evaluators did this independently and they discussed the scores until they reached a consensus on each. The moderator in the session was responsible for making sure that both evaluators' opinions had an equal weight in the discussion. The results of the WSM are presented in Table 2, which also includes some of the criteria, their weight score, weight in percentages and tool scores for some criteria. Note that we have only included five examples of the total of 38 criteria actually used.

Criteria/tests	weight score	weight%	Mosaic for X	Netscape	Webworks	HotJava
Level 2 compatibility	5	3.4%	3	3	3	3
HTML Level 3 compatibility	5	3.4%	0	3	0	0
.....	...	...	...	...	...	...
Required disk space	2	1.4%	1	2	2	5
Ease of installation	3	2.1%	5	5	5	5
Popularity of the tool	4	2.8%	3	5	0	0
Total of weight scores	145	1				
Score			470	591	467	427

Table 2: The results of the weighted scoring method (WSM)

The AHP method was applied using the Expert Choice tool [26] in an on-line session. We first ranked the criteria hierarchically and then proceeded to rank the alternatives against each leaf-level criterion. The results of the AHP analysis method are presented in Figure 3.

The total effort distribution for the evaluation process is presented in Table 3. After the case study was completed, we interviewed the key evaluators for their experiences, observations and perception of the process. Specifically, they were asked to state how useful they found the criteria definition process and the two analysis methods used.

4.2 Case Study Analysis

As far as the OTSO criteria definition process is concerned, we noticed that the effort spent in criteria definition, 44 hours (i.e., 28% of the total effort) was within the range we expected it to be. According to our interviews with the evaluators, it clearly had a positive impact in the efficiency, consistency and quality of evaluations. Evaluators were given a well-defined template that allowed them to focus on important characteristics of the tools, they were able to discuss the features with each other and evaluation results were relatively consistent. However, despite our efforts, there were still some vague definitions for some of the criteria: a couple of criteria definitions were misunderstood and one was found to be irrelevant during evaluation.

The comparison of WSM and AHP methods supported our hypothesis. The results of the WSM seemed reasonable for the evaluators but the WSM table did not provide any insights to the sensitivity of the results. Also, with 38 criteria, it was rather difficult to see the big picture in the data.

The AHP method was initially perceived as difficult as its calculation model is more complex and it involves a high number of paired comparisons. In fact, during the evaluation we made 322 such comparisons, not including revisions made to some comparisons. This would not have been practical without a graphical, easy to use tool that allowed this to take place within a single session.

The large volume of individual assessments in the AHP method is perhaps its main weakness. Even though the overall duration of the assessment session was not too long, the repetitive assessments may cause fatigue in evaluators. Fortunately, the paired assessment method automatically produces information on how consistent the evaluations are, which would be the likely result of fatigue. The moderator can monitor the consistency, as we did in our case study, and call a break if consistency rates become alarming.

The AHP method allowed the key evaluators to analyze the results from various perspectives and play what-if scenarios by changing weights for different criteria groups. Also, the redundancy built into the paired ranking method and possibility to get feedback on the consistency of comparisons further increased the confidence in results. As the AHP method produced ratio scale rankings, the method produced more information for decision making. The key evaluators agreed that the AHP method produced

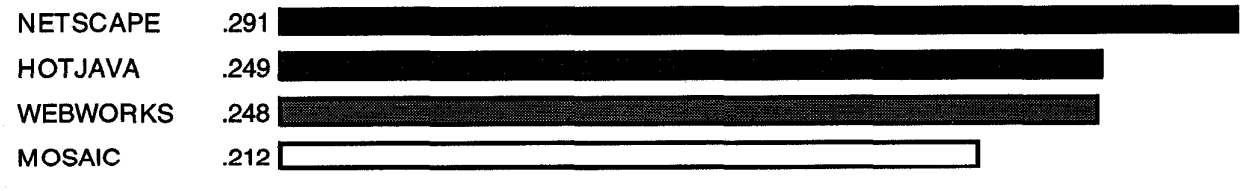

Figure 3: Results of the AHP method

more information and more reliable information for decision making.

The surprising result in our case study was that the relative ranking of browsers was different using the two analysis methods. Both methods ranked Netscape as the best alternative, but the remaining order of tools was different as Table 2 and Figure 3 show. WSP ranked HotJava the worst but AHP ranked it the second, although very close to Webworks. The WSP could not differentiate between Webworks and Mosaic for X but AHP found clear differences between them.

Given the serious limitations of the WSP approach, we believe that the AHP results are more reliable and they better represent the real rankings between the tools. We draw this conclusion primarily based on the confidence the evaluators had with the analysis results. Unfortunately, it is practically impossible to verify this conclusions with certainty. As in all multiple criteria decision making situations involving future behavior of a system, preferences change over time and between initial information may have contained errors, individuals, situations and information available may change and objectives may also change.

The fact that the analysis method can have such a big impact on the evaluation results is striking. The last couple of hours in COTS selection can be decisive. Therefore, it seems that the selection of the analysis method should be done with care in COTS evaluation.

Activity	Effort (hrs)		%
Search		20	14%
Screening		8	6%
Evaluation		79	55%
Criteria definition	40		28%
Mosaic for X	10		7%
Netscape	9		6%
Webworks	9.5		7%
HotJava	10.5		7%
Analysis/WSM		5	3%
Analysis/AHP		7	5%
Administration (planning meetings, writing reports)		20	14%
Learning about the methods		1	1%
other (vendor contacts, installations)		4	3%
Total		**144**	

Table 3: Effort distribution in the OTSO case study

5. Conclusions

The OTSO method was developed to consolidate some of the best practices we have been able to identify for COTS selection. We believe that the primary benefit of formalizing the COTS selection process is that it allows further improvement of it. A repeatable process supports learning through experience.

Our limited case study was intended to provide practical experience in applying the method and to provide some indication of its feasibility in practice. The case study seemed to suggest that the method is practical and it may improve the COTS selection process if it is currently conducted in an ad hoc manner.

In particular, the requirements driven, detailed evaluation criteria definition seemed to have a positive impact on the evaluation process. Furthermore, the marginal cost of more formal criteria definition seems to be within acceptable range as it can be accomplished within person days of effort. It also can have a positive effect on the definition of the application requirements.

Second, our case study showed that the AHP method can produce more relevant information for COTS selection and this information is perceived as more reliable by decision makers. At the same time, the additional cost of applying AHP is small, compared to the WSM approach. However, when the number of alternatives and criteria are small, WSP may still be a reasonable method to use, provided that its limitations are taken into account and compensated.

Third, our case study also showed that the choice of the evaluation data analysis method can have more than a minor impact on evaluation results. If this is true in the general case as well, this has strong implications on the way evaluation data should be handled in COTS selection cases.

The case study reported in this paper provided initial results and practical feedback on main aspects of the OTSO method. It seems that the OTSO method addresses important and often ignored problems in COTS usage. However, due to the limited number of data points, i.e., evaluators and cases, the results are not conclusive. We plan to carry out additional case studies and experiments to validate our method further.

6. Acknowledgments

I would like to thank the Hughes Information Technology Corporation and Mike Deutsch, Show-Fune Chen and Kevin Limperos for the support and cooperation that made this research possible. I am also grateful for the discussions with Victor Basili and Gianluigi Caldiera during the course of this research.

7. References

[1] B. Barnes, T. Durek, J. Gaffney, and A. Pyster. A Framework and Economic Foundation for Software Reuse. In: *Tutorial: Software Reuse: Emerging Technology*, ed. W. Tracz. Washington: IEEE Computer Society, 1988. pp. 77-88.

[2] V. R. Basili, G. Caldiera, and G. Cantone, A Reference Architecture for the Component Factory, *ACM Transactions on Software Engineering and Methodology*, vol. 1, 1. pp. 53-80, 1992.

[3] V. R. Basili, G. Caldiera, and H. D. Rombach. Goal Question Metric Paradigm. In: *Ensyclopedia of Software Engineering*, ed. J. J. Marciniak. New York: John Wiley & Sons, 1994. pp. 528-532.

[4] V. R. Basili, G. Caldiera, and H. D. Rombach. The Experience Factory. In: *Encyclopedia of Software Engineering*, Anonymous New York: John Wiley & Sons, 1994. pp. 470-476.

[5] T. Birgerstaff and C. Richter, Reusability Framework, Assessment, and Directions, *IEEE Software*, vol. March. pp. 41-49, 1987.

[6] T. B. Bollinger and S. L. Pfleeger, Economics of Reuse: issues and alternatives, *Information and Software Technology*, vol. 32, 10. pp. 643-652, 1991.

[7] G. Boloix and P. N. Robillard, A Software System Evaluation Framework, *IEEE Computer*, vol. 28, 12. pp. 17-26, 1995.

[8] A. T. W. Chu and R. E. Kalaba, A Comparison of Two Methods for Determining the Weights Belonging to Fuzzy Sets, *Journal of Optimization Theory and Applications*, vol. 27, 4. pp. 531-538, 1979.

[9] W. E. Deming. *Out of the Crisis*, Cambridge: Massachusetts Institute of Technology, 1986. 507 pages.

[10] G. R. Finnie, G. E. Wittig, and D. I. Petkov, Prioritizing Software Development Productivity Factors Using the Analytic Hierarchy Process, *Journal of Systems and Software*, vol. 22, pp. 129-139, 1995.

[11] E. H. Forman, Facts and Fictions about the Analytic Hierarchy Process, *Mathematical and Computer Modelling*, vol. 17, 4-5. pp. 19-26, 1993.

[12] M. L. Griss, Software reuse: From library to factory, *IBM Systems Journal*, vol. 32, 4. pp. 548-566, 1993.

[13] S. Hong and R. Nigam. Analytic Hierarchy Process Applied to Evaluation of Financial Modeling Software. In: *Proceedings of the 1st International Conference on Decision Support Systems, Atlanta, GA*, Anonymous1981.

[14] J. W. Hooper and R. O. Chester. Software Reuse: Guidelines and Methods, R.A. Demillo (Ed). New York: Plenum Press, 1991.

[15] ISO. *Information technology - Software product evaluation - Quality characteristics and quidelines for their use, ISO/IEC 9126:1991(E)*, Geneve, Switzerland: International Standards Organization, 1991.

[16] J. Kontio, OTSO: A Systematic Process for Reusable Software Component Selection CS-TR-3478, 1995. University of Maryland Technical Reports. University of Maryland. College Park, MD.

[17] J. Kontio and S. Chen, Hypertext Document Viewing Tool Trade Study: Summary of Evaluation Results 441-TP-002-001, 1995. EOS project Technical Paper. Hughes Corporation, EOS project.

[18] H. Min, Selection of Software: The Analytic Hierarchy Process, *International Journal of Physical Distribution & Logistics Management*, vol. 22, 1. pp. 42-52, 1992.

[19] S. L. Pfleeger and T. B. Bollinger, The Economics of Reuse: New Approaches to Modeling and Assessing Cost, *Information and Software Technology*, vol. 1994.

[20] J. S. Poulin, J. M. Caruso, and D. R. Hancock, The business case for software reuse, *IBM Systems Journal*, vol. 32, 4. pp. 567-594, 1993.

[21] R. Prieto-Díaz, Implementing faceted classification for software reuse, *Communications of the ACM*, vol. 34, 5.1991.

[22] C. V. Ramamoorthy, V. Garg, and A. Prakash, Support for Reusability in Genesis, pp. 299-305, 1986. Proceedings of Compsac 86. Chicago.

[23] T. L. Saaty. *Decision Making for Leaders*, Belmont, California: Lifetime Learning Publications, 1982. 291 pages.

[24] T. L. Saaty. *The Analytic Hierarchy Process*, New York: McGraw-Hill, 1990. 287 pages.

[25] T. L. Saaty. Analytic Hierarchy. In: *Encyclopedia of Science & Technology*, Anonymous McGraw-Hill, 1992. pp. 559-563.

[26] T. L. Saaty, Expert Choice software 1995, ver. 9, rel. 1995. Expert Choice Inc. IBM. DOS.

[27] W. Schäfer, R. Prieto-Díaz, and M. Matsumoto. *Software Reusability*, W. Schäfer, R. Prieto-Díaz, and M. Matsumoto (Eds). Hemel Hempstead: Ellis Horwood, 1994.

[28] P. J. Schoemaker and C. C. Waid, An Experimental Comparison of Different Approaches to Determining Weights in Additive Utility Models, *Management Science*, vol. 28, 2. pp. 182-196, 1982.

[29] W. Tracz, Reusability Comes of Age, *IEEE Software*, vol. July. pp. 6-8, 1987.

[30] W. Tracz. *Tutorial: Software Reuse: Emerging Technology*, W. Tracz (Ed). Washington: IEEE Computer Society, 1988.

[31] W. Tracz. Software Reuse: Motivators and Inhibitors. In: Tutorial: Software Reuse: Emerging Technology, ed. W. Tracz. Washington: IEEE Computer Society, 1988. pp. 62-67.

[32] W. Tracz, Legal obligations for software reuse, American Programmer, vol. March.1991.

[33] M. Wasmund, Implementing Critical Success Factors in software reuse, *IBM Systems Journal*, vol. 32, 4. pp. 595-611, 1993.

[34] F. Wolff, Long-term controlling of software reuse, Information and Software Technology, vol. 34, 3. pp. 178-184, 1992.