Quantitative Methods for Software Selection and Evaluation

Michael S. Bandor

September 2006

Acquisition Support Program

Technical Note
CMU/SEI-2006-TN-026

This work is sponsored by the U.S. Department of Defense.

The Software Engineering Institute is a federally funded research and development center sponsored by the U.S. Department of Defense.

This work was created in the performance of Federal Government Contract Number FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center. The Government of the United States has a royalty-free government-purpose license to use, duplicate, or disclose the work, in whole or in part and in any manner, and to have or permit others to do so, for government purposes pursuant to the copyright license under the clause at 252.227-7013.

For information about purchasing paper copies of SEI reports, please visit the publications portion of our Web site (http://www.sei.cmu.edu/publications/pubweb.html).

Contents

List of Tables

Acknowledgments

I would like to thank the following Software Engineering Institute (SEI) personnel for their assistance in reviewing and producing this technical note: Mary Ann Lapham, Harry Levinson, Bud Hammons, Linda Levine, Suzanne Couturiaux, and John Foreman.

Abstract

When performing a “buy” analysis and selecting a product as part of a software acquisition strategy, most organizations will consider primarily the requirements (the ability of the product to meet the need) and the cost. The method used for the analysis and selection activities can range from the use of basic intuition to counting the number of requirements fulfilled, or something in between. The selection and evaluation of the product must be done in a consistent, quantifiable manner to be effective. By using a formal method, it is possible to mix very different criteria into a cohesive decision; the justification for the selection decision is not just based on technical, intuitive, or political factors. This report describes various methods for selecting candidate commercial off-the-shelf packages for further evaluation, possible methods for evaluation, and other factors besides requirements to be considered. It also describes the use of a decision analysis spreadsheet as one possible tool for use in the evaluation process.

1 Software Package Selection

Many organizations are attempting to save costs by integrating third-party, commercial off-the-shelf (COTS) packages (e.g., component libraries or extensions) or complete COTS-based solutions (e.g., enterprise resource planning [ERP] applications). The methods used to identify a set of possible candidate solutions are, for the most part, rather subjective. The individual or individuals performing the evaluation have various, distinct experiences that will factor into the decision process, either consciously or subconsciously. To have a successful COTS evaluation, a formal process is needed to properly evaluate COTS products and vendors supplying them [SEI 05]. In this instance, the term *formal* means having an established and documented process to perform the selection and evaluation activities in a consistent, repeatable manner.

1.1 Initial Selection

How does an organization conduct the initial research into products that might be candidates for use on their project? How is the initial selection performed? Some organizations use an "intuitive approach" to select the initial list of products. This approach uses an individual who has had past experience with the product or who has "heard good things" about the product. An inappropriate selection strategy for COTS products can lead to adverse effects. It could result in a short list of COTS products that may not be able to fulfill the required functionality; in addition, it might introduce overhead costs in the system integration and maintenance phases [Leung 02].

One successful method for selecting products is the use of a selection team. When selecting a COTS component,[1] the use of a team of technical experts—systems/software engineers and several developers—is recommended. When selecting a COTS-based system,[2] however, the inclusion of business domain experts and potential end users is recommended [Sai 04]. The use of a team virtually eliminates a single-person perspective or bias and takes into account the viewpoints and experiences of the evaluators in the selection and evaluation process.

Table 1 describes several approaches that can be used to conduct the initial market research.

[1] A COTS component, in this context, would be something like a third-party graphics library or report generation tool. They are building blocks integrated into a larger system.

[2] An example of a COTS-based system is an enterprise resource management (ERP) package.

Table 1: Approaches for Conducting the Initial Market Research

Approach	Usage
Vendor surveys	The survey is designed to evaluate the usefulness to the vendor of the request for proposal (RFP) and related documents. It also provides information about the vendors themselves [Sai 04].
Vendor white papers	A significant number of vendors will produce "white papers" giving information about their products and, sometimes, case-study information related to successful implementation.
Product/component technical specifications	In the case of most COTS components (e.g., libraries, graphics packages) and COTS-based solutions, the vendor will have detailed technical information available for review. The technical specifications may or may not list specific constraints.
Representation at key information technology (IT) conferences	The larger the vendor, the more visible they will be in the marketplace. This visibility is especially evident at IT conferences. If you are researching a vendor-specific solution, find out if the vendor sponsors or is present at one or more large conferences. Attending a conference allows you to talk directly to competing vendors and affords you the opportunity to talk with other users of the product and other companies that provide additional support for the product (e.g., product extensions).
Communication with other customers using the product/component	The satisfaction of other customers using the product can provide additional insight you might not be able to get through other methods (e.g., customer/technical support issues related to the product).
Conducting a pre-bid conference	This type of event (sometimes referred to as an "industry day") allows potential vendors to visit your organization to discuss your needs and how their products might fulfill the stated requirements. Again, this type of event affords your organization the opportunity to ask the vendor questions directly.

As an example of using these approaches, the Carnegie Mellon® Software Engineering Institute (SEI) used the vendor-survey approach, among several others listed, to select a new ERP application. The SEI needed to replace a long-lived, faltering budget system that was built internally and had many shortcomings relating to the budget and business goals of the SEI. The system required substantial modifications to accommodate several new needs created by the advent of a new Oracle ERP system in use by Carnegie Mellon University [Sai 04]. In a case of "practicing what you preach," the SEI put into practice the principles taught in the *COTS-Based Systems for Program Managers* and *COTS Software Product Evaluation for Practitioners* training courses. The approach and subsequent results were captured in the technical note, *COTS Acquisition Evaluation Process: Preacher's Practice* [Sai 04].

® Carnegie Mellon is registered in the U.S. Patent and Trademark Office by Carnegie Mellon University.

In this documented example, the SEI evaluation team established some high-level criteria and capability statements, along with some basic expectations. A grading scale (shown in Table 2 [Sai 04]) was established by the evaluation team for the vendors to rate their own products against the specified criteria.

Table 2: Vendor Self-Evaluation Scale—Sample

Score Value	Definition
10	Fully addressed in current version
8	Partially addressed in current version; low-cost, no-impact/low-impact minor modifications will return fully desired functionality.
7	Not addressed in current version; low-cost, no-impact/low-impact modifications will return fully desired functionality.
6	Partially addressed in current version; high-cost, no-impact/low-impact modifications will return fully desired functionality.

Sai identified some interesting characteristics of this process:

- Evaluators felt respected by the level of participation afforded.
- Evaluators were allowed to evaluate not only criteria that mapped to their field of expertise but also other aspects of the proposal if they chose.
- Core technical staff members voiced happiness about being involved in the process.
- A common understanding of the capabilities of the solution existed.
- Most evaluators turned in valuable evaluation comments.
- Scores appeared to be based on the evaluators' understanding of the proposal.
- Experts were used to review the proposals for better understanding.
- New questions were generated for the vendors' clarifications.

Other useful mechanisms for performing initial evaluations are the use of pilot programs and obtaining a trial-use copy of the product being evaluated. These mechanisms allow an organization to evaluate the robustness[3] of the product, critical aspects of the system, and the tailoring and customization capabilities of the product [SEI 05]. They also demonstrate how well the product works in the target environment and allow the organization to determine what tradeoffs are necessary in the evaluation criteria.

[3] *Robustness*, in this use of the term, means "the degree to which a system or component can function correctly in the presence of invalid inputs or stressful environmental conditions" [IEEE 90].

1.2 Evaluation Criteria

When evaluating a possible software solution, most organizations are likely to consider the ability of the product to meet the functional requirements. Although it is a significant first step in the evaluation process, this should not be the only criterion that is considered. Two additional criteria that should be considered are intangible factors and risk.

1.2.1 Intangible Factors

Intangible factors are not the traditional "quality" factors (e.g., the various "-ilities"), but rather factors that are programmatic decisions (i.e., decisions that can or will affect the overall program during its life span) and that have an effect on the system utilizing the software. Most of the decisions also depend on intangible factors that are difficult to quantify. According to Litke and Pelletier, some costs can be identified up-front, but others—the ones that organizations need to worry about for the long term—are hidden. Some examples of intangible factors cited by Litke and Pelletier and DeVries are shown in Table 3 [Litke 02, DeVries 05].

Table 3: Examples of Intangible Factors

Intangible Factor	**Consideration**
Can other people work on it?	Does the software require specialized language training or techniques to use it or integrate it into the system?
Are you going to change?	Are your organization's business processes/requirements subject to a large amount of change?
What is the scope?	Is this software being applied to only one area of the system, or is it being reused across many areas?
Is it overkill?	Are you buying more "bells and whistles" than you really need? You may be paying for many features that can't be used or that could have a detrimental effect on the architecture.
Remember the end user.	The end user is the person who is most likely affected by your decision. Will integrating this software require additional training or changes to the process?
What is the additional time/cost to modify or interface to the software?	Interface development may still be needed to integrate the software or fully take advantage of its features.
How well does it integrate or "play well" with the other applications within the architecture?	If the software doesn't integrate well, it may be necessary to make a significant change to the architecture. Remember that time is not on your side!
What kind of documentation and support are available?	If there is a lack of documentation and support, the integration may be difficult and your organization may need a significant amount of time to understand how the software works.

Table 3: Examples of Intangible Factors (continued)

Intangible Factor	Consideration
Are all of the costs known up front?	Many corporate customers purchase a COTS application, only to find that they have to pay a large consulting firm three times as much to come in and customize the application.
Do you have or will you have the correct mix of skill sets for the aggregate product?	Each piece of the system may be covered, but when the pieces are aggregated, will you need additional skill sets to operate and maintain the end product?

1.2.2 Risk

Risk[4] is another element that should be part of the selection criteria. Many of the risks associated with system management and operation are not in your direct control. Each vendor that plays a role in the design, development, acquisition, integration, deployment, maintenance, operation, or evolution of part (or all) of your system affects the risks you face in your attempt to survive cyber attacks, accidents, and subsystem failures [Lipson 01]. Some possible risk factors that should be considered are listed below:

- Is the company well established?
- What is the longevity of the company?
- Is there support (training, developer, etc.) offered?
- Is your vendor flexible enough to make changes in the middle of development?
- Is the vendor financially stable?
- How mature is the technology used?

Another risk to consider is the volatility of the COTS components. COTS-based systems are always subject to the volatility of the COTS components (i.e., frequency with which vendors release new versions of their products). Expect volatility to increase exponentially with time and the number of components used [Lipson 01].

After a product or component has been selected, continuous risk management[5] should be applied for the life cycle of the system that uses it. Continuous risk management is especially

[4] *Risk*, in this usage, is defined as "the possibility of suffering loss. In a development project, the loss describes the impact to the project, which could be in the form of diminished quality of the end product, increased costs, delayed completion, or failure" [Dorofee 96].

[5] *Continuous risk management* is defined as "…a software engineering practice with processes, methods, and tools for managing risks in a project. It provides a disciplined environment for proactive decision making to assess continuously what could go wrong (risks), determine which risks are important to deal with, and implement strategies to deal with those risks" [Dorofee 96].

important if the product or component is being used as part of a framework.[6] Unlike other software-selection decisions, the selection of a framework is a long-term decision—possibly lasting 10–15 years [Fayad 00]. After a final selection has been made, the risks associated with the product or component should be fed back into the risk management plan.

One method for mitigating the risk is to perform continual vendor-based risk evaluations. This type of evaluation focuses only on the vendor or vendors supplying the third-party components. Continual risk evaluation is especially important if the component is a critical part of the system life cycle for a mission-critical system [Lipson 01]. This activity should also be addressed as part of a risk management plan.

[6] "In software development, a framework is a defined support structure in which another software project can be organized and developed. A framework may include support programs, code libraries, a scripting language, or other software to help develop and glue together the different components of a software project. The word *framework* has become a buzzword due to recent continuous and unfettered use of the term for any generic type of libraries" (Wikipedia [http://en.wikipedia.org/wiki/Framework]).

2 Evaluation Methods

After you have determined your selection criteria, you will need a mechanism to score and compare the potential products for suitability. One tool that is well suited to this task is a decision analysis spreadsheet.

2.1 Decision Analysis Spreadsheet

A decision analysis spreadsheet allows an organization to compare various products by using the selection criteria and assigning a weighted value to the criteria [Litke 02]. The product with the best score (based on the values) is the preferred product. There are two variations on this method. The first variation can be seen in Table 4 [Litke 02]. This example shows two products (System 1 and System 2) being compared based on a range of criteria (Items A through I). Each criterion has its own weight, and the individuals performing the evaluation assign a raw value to each product, which results in a weighed score. The weighted scores are then totaled and compared. The key to this method is that the total weights must add up to 100%.

Table 4: Decision Analysis Spreadsheet: Example 1

			Software Alternatives			
			System 1		**System 2**	
Item	**Decision Criterion**	**Weight**	**Raw**	**Weighted**	**Raw**	**Weighted**
A	Rule-based presentation	20%	1.0	20.00%	1.0	20.00%
B	Reliable/fault tolerant	10%	1.0	10.00%	1.0	10.00%
C	Scalable	10%	1.0	10.00%	1.0	10.00%
D	Product/vendor maturity	10%	0.5	5.00%	1.0	10.00%
E	Vendor support	10%	0.5	5.00%	1.0	10.00%
F	Low total cost of ownership	10%	0.0	0.00%	1.0	10.00%
G	Extensible	10%	1.0	10.00%	1.0	10.00%
H	Single-vendor solution	5%	-0.5	-2.50%	1.0	5.00%
I	Visual rules definition/administration	15%	1.0	15.00%	0.5	7.50%
	Total	**100%**		**72.5%**		**92.50%**

The second variation uses subgroups of criterion. An example of this variation can be seen in Table 5. Each subgroup is further decomposed one level further and weights are assigned. Again, the total of the weights for the subgroup must add up to 100%. The score for this variation differs slightly in that the final score for the subgroup is calculated by multiplying the total weighted score for each of the subcriteria by the total weighted value for the subgroup. In the example shown in Table 5, the subgroup weight is 20%, and the weighted value for System 1 is 18% (90% of 20%). The key to this variation is not to overly decompose the requirements. Start with the high-level groupings and decompose the criteria by only one additional level.

Table 5: Decision Analysis Spreadsheet: Example 2

			Software Alternatives			
			System 1		**System 2**	
Item	**Decision Criterion**	**Weight**	**Raw**	**Weighted**	**Raw**	**Weighted**
A	**Graphical user interface**	**20%**		**18.00%**		**11.50%**
A.1	Multiple window use	50%	1.0	50%	0.5	25%
A.2	Resizable windows	30%	1.0	30%	1.0	30%
A.3	Remembers user's screen settings	10%	0.5	5%	-0.5	-3%
A.4	Provides keyboard shortcuts	10%	0.5	5%	1.0	5%
	Subtotal			**90%**		**58%**

2.2 Scoring Values

The key to using a decision analysis spreadsheet is the raw score values. By using a defined and understood set of discrete values, the subjectivity of the evaluation is significantly reduced. In the prior examples, the raw values were based on the information shown in Table 6 [Litke 02]. There are only five values used, ranging from 1.0 to -1.0 in increments of 0.5. Note the use of negative values and the effects on the scoring. Instead of just assigning a value of 0, the use of negative values permits the application of a "penalty" value where not meeting the criterion would be detrimental.

There are many different methods for deriving risk values, but descriptions of these methods are out of scope for this report. Additional references on risk can be found in the bibliography. Regardless of which risk calculation method you choose to follow, it is important to keep in mind that the scoring mechanism presented above is based on a "higher is better" score, and most risk calculations are based on a "lower is better" score. The two methods should be used individually and not combined into a single score for evaluation purposes.

Table 6: Example Legend for Scoring Requirements

Score Value	Definition
1.0	Alternative fully satisfies business requirement or decision criterion.
0.5	Alternative partially satisfies business requirement or decision criterion.
0.0	Unknown or null/balanced (The alternative neither satisfies nor dissatisfies business requirement or decision criterion.)
-0.5	Alternative partially dissatisfies business requirement or decision criterion.
-1.0	Alternative fully dissatisfies business requirement or decision criterion [Litke 02].

One consideration that must be addressed is how to handle scoring variances. Each potential evaluator has different experiences and perceptions that will ultimately affect the scoring. When using individual evaluators, the organization must have a scoring process that addresses (1) what constitutes a variance and (2) how to handle the differences in the scoring.

Many organizations that use a similar process for evaluations will set a fixed value (e.g., less than 2 points on a 10-point scale) or a fixed percentage (e.g., 10% or more). When a scoring variance (or scoring split) occurs, the evaluators having a variance would then address the areas in the scoring that differed from the other evaluators. After the evaluators affected by the split have discussed their scoring and the rationale, each evaluator would take into consideration the new information and rescore the product. For example, when performing an evaluation on a product, Evaluator A (using the sample found in Table 1) gives the product a total score of 78%, and Evaluator B gives the product a total score of 90%. Assuming the scoring process defines a split as 10% or more difference in scoring, both evaluators would discuss their individual scores for each range of criteria and their rationale for the individual scores; they would then rescore the product in the area(s) that differed until the scoring split was resolved.

3 Conclusion

A successful evaluation is not simply picking a product based on intuition. It involves a formal process, the right mixture of evaluators, and a specific quantifiable set of evaluation criteria. The process should include how to handle differences in scoring by the evaluators. The SEI, in going through its own selection process, offers the following lessons learned [SEI 05]:

- Every off-the-shelf item used in the system should be the subject of an appropriate evaluation and selection process.
- A sound evaluation process for COTS products must support the selection.
- Requirements drive selection criteria, especially initially.
- Careful consideration must be given to the identification of selection criteria.
- Pilots and demonstrations are essential selection tools.
- Product and technology maturity must be considered.

Bibliography

URLs are valid as of the publication date of this document.

[Alberts 06] Alberts, Christopher J. *Common Elements of Risk* (CMU/SEI-2006-TN-014). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2006. http://www.sei.cmu.edu/publications/documents/06.reports/06tn014.html.

[Carney 03] Carney, David J.; Morris, Edwin J.; & Place, Patrick R. H. *Identifying Commercial Off-the-Shelf (COTS) Product Risks: The COTS Risk Usage Evaluation* (CMU/SEI-2003-TR-023, ADA418382). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2003. http://www.sei.cmu.edu/publications/documents/03.reports/03tr023.html.

[Dorofee 96] Dorofee, Audrey; Walker, Julie; Alberts, Christopher; Higuera, Ronald; Murphy, Richard; & Willams, Ray. *Continuous Risk Management Guidebook*. Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 1996. http://www.sei.cmu.edu/publications/books/other-books/crm.guidebk.html.

[DeVries 05] DeVries, Michael. "To Buy? Or To Build? … That Is The Question!" *ISnare*. http://www.isnare.com/?id=5434&ca=Computers+and+Technology (2005).

[Fayad 00] Fayad, Mohamed E. & Hamu, David S. *Enterprise Frameworks: Guidelines for Selection*. New York, NY: Association for Computing Machinery (ACM), 2000.

[IEEE 90] Institute of Electrical and Electronics Engineers (IEEE). *IEEE Standard Glossary of Software Engineering Terminology* (IEEE Standard 610.12-1990). New York, NY: IEEE, 1990.

[Leung 02] Leung, Karl R. P. H. & Leung, Hareton K. N. "On the Efficiency of Domain-Based COTS Product Selection Method." *Information and Software Technology 44*, 12 (Sept. 2002): 703–715.

[Lipson 01] Lipson, Howard F.; Mead, Nancy R.; & Moore, Andrew P. *Can We Ever Build Survivable Systems from COTS Components?* (CMU/SEI-2001-TN-030, ADA3399238). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2001. http://www.sei.cmu.edu/publications/documents/01.reports/01tn030.html.

[Litke 02] Litke, Christian & Pelletier, Michael. "Build it or Buy it? How to perform a cost-benefit analysis for IT projects." *The Fabricator*. http://www.thefabricator.com/ShopManagement/ShopManagement_Article.cfm?ID=166 (March 28, 2002).

[Sai 04] Sai, Vijay. *COTS Acquisition Evaluation Process: Preacher's Practice* (CMU/SEI-2004-TN-001, ADA421675). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2004. http://www.sei.cmu.edu/publications/documents/04.reports/04tn001.html.

[SEI 05] Software Engineering Institute (SEI). *Product Evaluation & Selection.* http://www.sei.cmu.edu/cbs/lessons/evaluation-selection/lessons.htm. (2005).

[Zizakovic 04] Zizakovic, Lubo. *Buy or Build: Corporate Software Dilemma*. Toronto, Canada: Insidus Custom Software Systems, August 2004. http://www.insidus.com/BuyorBuild.pdf.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
(Leave Blank)	September 2006	Final

4. TITLE AND SUBTITLE	5. FUNDING NUMBERS
Quantitative Methods for Software Selection and Evaluation	FA8721-05-C-0003

6. AUTHOR(S)

Michael S. Bandor

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

8. PERFORMING ORGANIZATION REPORT NUMBER

CMU/SEI-2006-TN-026

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

HQ ESC/XPK
5 Eglin Street
Hanscom AFB, MA 01731-2116

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12A DISTRIBUTION/AVAILABILITY STATEMENT	12B DISTRIBUTION CODE
Unclassified/Unlimited, DTIC, NTIS	

13. ABSTRACT (MAXIMUM 200 WORDS)

When performing a "buy" analysis and selecting a product as part of a software acquisition strategy, most organizations will consider primarily the requirements (the ability of the product to meet the need) and the cost. The method used for the analysis and selection activities can range from the use of basic intuition to counting the number of requirements fulfilled, or something in between. The selection and evaluation of the product must be done in a consistent, quantifiable manner to be effective. By using a formal method, it is possible to mix very different criteria into a cohesive decision; the justification for the selection decision is not just based on technical, intuitive, or political factors. This report describes various methods for selecting candidate commercial off-the-shelf packages for further evaluation, possible methods for evaluation, and other factors besides requirements to be considered. It also describes the use of a decision analysis spreadsheet as one possible tool for use in the evaluation process.

14. SUBJECT TERMS	15. NUMBER OF PAGES
acquisition, buy versus build, COTS software, software evaluation, software selection	22

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL