Quantifying the Value of Architecture Design Decisions: Lessons from the Field

Mike Moore
NASA Goddard Space Flight Center
mike.moore@nasa.gov

Rick Kazman,
Software Engineering Institute Carnegie Mellon University and *University of Hawaii*
kazman@sei.cmu.edu

Mark Klein
Software Engineering Institute, Carnegie Mellon University
mk@sei.cmu.edu

Jai Asundi
School of Management University of Texas, Dallas
asundi@utdallas.edu

Abstract

This paper outlines experiences with using economic criteria to make architecture design decisions. It briefly describes the CBAM (Cost Benefit Analysis Method) framework applied to estimate the value of architectural strategies in a NASA project, the ECS. This paper describes the practical difficulties and experiences in applying the method to a large real-world system. It concludes with some lessons learned from the experience.

1. Introduction

For the past three years we have been developing and road-testing a method for doing architecture-based economic analyses of software-intensive systems. We have called this method the CBAM (Cost Benefit Analysis Method). The method was first described in [6] and revisions have been described in [1], and [7]. The purpose of this paper is not to discuss the CBAM, but rather to discuss the lessons learned from applying it to a large real-world project over several years. We applied the CBAM to NASA's ECS project in two distinct phases of the project's development, and the results from these experiences have helped us improve the method. In this paper we outline our experiences in eliciting economic data from stakeholders and in analyzing and interpreting this data. We describe the changes made to the CBAM based on our successes and failures, and we describe the benefits that it brings to a large development effort.

2. The ECS Project

The ECS problem was framed as follows: we had a limited budget to maintain and enhance the current system. From a prior analysis, in this case an architecture evaluation using the ATAM (Architecture Tradeoff Analysis Method) [5], a large set of desirable changes to the system was elicited from the stakeholders of the system, resulting in a large set of ASs (Architectural Strategies). The problem facing us was to choose a (much) smaller subset of ASs for implementation. The CBAM was applied to the problem so that we could justify decisions based on the economic criterion of return on investment (ROI).

The CBAM begins with the assumption that the *business goals* of a system influence the architectural decisions made. For example, the choice of a tri-modular redundant server will be influenced by a business need for a highly reliable/available system that the system's stakeholders are presumably demanding. This AS will in turn influence various quality attributes of the system (such as performance, modifiability, availability, security etc.) These quality attributes (QAs) will in turn accrue benefit to the system as defined by the business goals. For example, the high availability server will result in $X worth of more business due to additional sales, or an unstable system will result in $Y worth of lost business per year, or $Z worth of lawsuits filed against the corporation each year.

Each architectural decision will also incur a cost. They can be considered to be software investments as described in [2] or more precisely, investments under uncertainty [3]. The aim of the CBAM was to maximize the ROI for the architecture design decisions and explore the possibility of embedded options within these decisions—options that are opened/closed due to the design choice being made. For example, the implementation of one particular architectural strategy could preclude the implementation of another.

2.1. Operationalizing the CBAM

Any simple description of the steps of a method glosses over many of the problems with actually *operationalizing* such a method. Creating a method that asked questions that stakeholders could actually answer, and imposing only small demands on the stakeholders (in terms of the amount of work that they had to do) turned out to be non-trivial.

For example, within the original version of the CBAM, we assumed that the effects of the ASs on the QAs are additive. We also assumed that each QA has a weight proportional to its importance in the system and that it was possible to elicit these weights from stakeholders. To do this, we asked the stakeholders to individually rate each QA such that the sum of their scores was 100. Hence, by design, S($QAScore_j$) = 100 and for all j, $QAScore_j >= 0$.

We then elicited from the stakeholders an estimate of the impact resulting from the application of each AS, in terms of how it affects each quality attribute of the system. This impact estimate was termed the "contribution score", $Contrib_{i,j}$ where this describes the impact of AS_i on QA_j. Based upon the *QAScore* and the *Contrib* score the total benefit of each AS was calculated as:

$$Benefit(AS_i) = ?\,_j(Contrib_{i,j} \times QAscore_j)$$

The units of benefit are 'utils' and the return for a single AS (in units of utils/person-month) is calculated as:

$$Return(AS_i) = Benefit(AS_i)/Cost(AS_i)$$

This return score is relative to all other ASs and hence is only useful for rank ordering them. A relative scoring (such as a rank ordering) was sufficient for our investigations, since one of NASA's concerns was to decide how to spend its budget, by implementing the top *r* ASs.

3. The CBAM Experience

Using the above framework we conducted the initial CBAM exercise with three facilitators from the SEI (Software Engineering Institute) and seven stakeholders from the ECS project. For the initial exercise the facilitators spent 8 hours meeting with the stakeholders, spread over 2 days. The stakeholders also spent time on their own for rating the ASs and estimating the costs of the ASs.

Eight QAs were elicited (Maintainability, Operability, Relavailability, Scalability, Performance, User satisfaction, Flextensibility and Security). There were 58 distinct ASs put forth by the architect of the ECS. We estimated that we had the resources to implement only 6 to 10 of these ASs (depending on the specific costs of strategies chosen)!

In this exercise the stakeholders' responses were kept from each other to prevent undue influence of some stakeholders on the others. The concordance of the stakeholders on the QA ratings was 0.838 which reflects a high degree of agreement. On the other hand the AS contribution ratings varied considerably. The concordance rating here was only 0.326 which indicates a low level of agreement regarding the effects of the AS on the respective QAs. This experience taught us a number of important lessons about operationalizing the CBAM.

3.1. Dealing with large numbers of ASs

The experience taught us that we needed to explicitly compare all the stakeholders' judgments to understand, and to further probe, areas where they differed. Such differences of judgment could be interpreted in two ways:

1. They are true differences of opinion among the stakeholders, and hence represent uncertainty, or

2. They are based on differing understanding of the AS and QA, and hence the stakeholders are actually rating different things.

Differences of the second kind need to be eradicated as much as possible. But how to do this? Karlsonn and Ryan [4] describe the Analytical Hierarchy Process for rank ordering requirements. This method, though effective, is quite cumbersome for dealing with a large number of strategies each of which affect multiple quality attributes. With the 58 ASs that we had in the ECS study, this would mean making 1653 comparisons! Even if we could prune the number of ASs to 25, it would mean 300 comparisons which is still quite large and for more time consuming than the stakeholders would put up with.

3.2. Short attention spans

Stakeholders typically have very short attention spans and tolerances for new "methods". The stakeholders are typically very busy and their time working on elicitation exercise is highly valued. We needed to be able to show them the benefits of the exercise itself and justify their time spent on it. Thus, when eliciting information, we had to be realistic about the time that it will take the stakeholders to complete any exercise and must strive to keep this time to a minimum.

3.3. The "Interpretation" problem

In dealing with a multi-attribute decision problem such as ours, it is theoretically convenient to view QAs like performance, security, availability etc. as attribute dimensions that we can reason about and for which we can choose desirable or undesirable levels. However, the notion of a QA is an abstract concept. In our experience, despite providing precise definitions for the QAs, the stakeholders showed considerable subjectivity in their interpretations of the QAs. Furthermore, from one context to another (for example, from one scenario to another) the interpretation of the QA could change, thus exacerbating the "interpretation" problem.

3.4. Stakeholders "game" the system

Stakeholders, once they realize how the analysis is going to be carried out will try to "game" the system. They might do this by providing favorable values for their favorite scenario or AS. This is typically done to further some hidden agenda. This tendency can mitigated by making the elicitation procedure open to discussion and by making the recorded values arise from *consensus*, rather than a voting procedure. This further has the benefit that it reduces the time to dispel misunderstandings and the resultant variance in scores. Though consensus-based valuation runs the risk of the values being unduly influenced by a few powerful personalities, it is the duty of the project manager to ensure that the right personnel have appropriate influence in the proceedings. This, in turn, ensures that the right stakeholders have appropriate influence on the final value recorded.

3.5. Knowing the state of the system

Though it seemed obvious to us that stakeholders would be able to quantify the effect of a change to their system, we found that these judgments were highly variable. We discovered that this was due to considerable variation among the stakeholders in their understanding of the *current* state of the system. It is typical in large systems that stakeholders will be experts only in their particular subsystem and will be only partially aware of the other subsystems. As a consequence they are not fully capable of characterizing the current state of the system and hence the precise effect of a particular change. Thus, characterizing the current state of the system is an important exercise before trying to measure the effect of a change.

3.6. The dialogue is important

What are the important benefits from the CBAM exercise? We assumed, when we embarked on the exercise, that the most important benefit was having a disciplined process that would result in a well-reasoned choice of ASs for NASA to pursue. However, it turns out that just as important was the dialogue and the discussion that the method fosters amongst the stakeholders. This dialogue forces them to discuss the various alternative strategies in terms of their utilities, and this discussion results in greater understanding of the scenarios that motivate the ASs, the utility of various response levels for these scenarios, and the ASs themselves. These discussions lead to a better understanding of how particular scenarios are important or how some set of ASs affect a particular QA. In our experience this exchange of ideas leads to results that better appeal to intuition.

4. CBAM Version 2

Armed with the experience of the first attempt at quantifying the benefit information we have developed a new version of the CBAM, described in [7]. The salient features of CBAM 2 are:

• It is now an *iterative* procedure where we refine our understanding of the system's benefits via the elicitation of progressively more information (scenarios, ASs, and utility levels). This iterative process helps the stakeholders understand the method's calculations as well as giving them preliminary results so that they see the benefit of performing the exercise. The steps of the CBAM are run in iterations. Each iteration progressively adds information via additional elicitation from the stakeholders. For example after the initial iteration, follow-on iterations add estimates of risk and uncertainty and correlations among the ASs.

• We have introduced scenarios to represent each of the QAs. Instead of talking about the QA of "relavailability", for example, we introduce a specific scenario that addresses the QA, e.g. "Reduce data distribution failures that result in hung distribution requests requiring manual intervention." This helps in providing a concrete context for the stakeholders to understand the impact of an AS.

• We now elicit the current state of the system, to provide all stakeholders with a common basis of understanding to inform their decision-making.

• Perhaps most important, we now explicitly determine the sensitivity of Utility to each QA. Rather than elicit a single value for an AS's effect on a QA, we ask the stakeholders to give a utility value for several different response measures (e.g. for the data distribution scenario above, the stakeholders might give {response measure, utility value} pairs of {10% hung, 10}, {5% hung, 80}, {1% hung, 95}, and {0% hung, 100}). These sets of pairs define Utility-QA level *curves*. These curves aid the stakeholders in understanding how utility varies with changing QA responses, and hence how the value of the system changes as different response levels are achieved.

An example of such a graph is shown in Figure 1. As we see in the graph on the left, we have elicited the QA response levels for the worst-case (W), Best case (B), Current (C) and Desired (D) cases. The respective utility values for these response levels are elicited. Depending on the AS chosen, the expected case (E) is plotted on the curve.

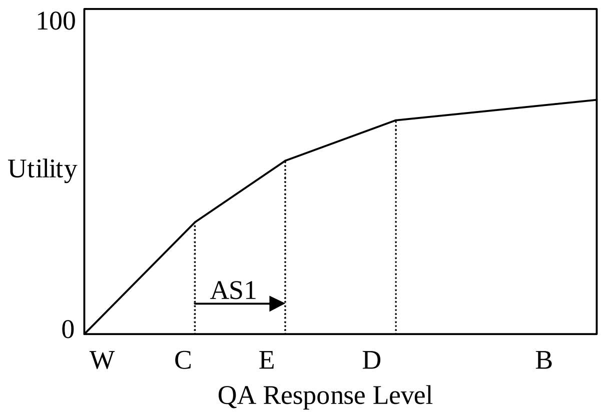

Figure 1. A QA Response/Utility Curve

5. Applying CBAM 2

We now turn to an application of the CBAM to a real-world system as an example of the method in action. The Earth Observing System is a constellation of NASA satellites that gathers data about the Earth for the U. S. Global Change Research Program and other scientific communities worldwide. The Earth Observing System Data Information System (EOSDIS) Core System (ECS) collects data from various satellite downlink stations for further processing. The mission of the ECS is to process the data into higher-form information and make it available in searchable form to scientists world-wide.

The ECS problem was framed as follows: management had a limited budget to maintain and enhance the system. From a prior ATAM analysis [5] a set of desired changes to the system was elicited from the stakeholders, resulting in a large set of ASs. The problem facing management was to choose a (much) smaller subset of ASs for implementation, as only 10-20% of what was being proposed could actually be funded. Management used the CBAM to help make a rational decision based on the economic criterion of return on investment.

5.1. Collate Scenarios

We began by eliciting a set of scenarios from the assembled stakeholders. Because the ECS stakeholders had previously been used ATAM, this step was relatively straightforward for them. A subset of the raw scenarios put forward by the stakeholders were as shown in Table 1. Note that the scenarios are not yet all well formed and some do not even have defined responses.

5.2. Refine and Prioritize Scenarios

These scenarios were then refined, paying particular attention to precisely specifying their stimulus/response measures. The worst, current case, desired, and best case response goals for each scenario were elicited. Having the team of stakeholders perform this refinement together helped drive out any differences in interpretation of the scenarios. It also helped the team members reach a common understanding of the current system state.

TABLE 1. Collected Scenarios

Scenario	Scenario Description
1	Reduce data distribution failures that result in hung distribution requests requiring manual intervention.
2	Reduce data distribution failures that result in lost distribution requests.
3	Reduce the number of orders that fail on the order submission process.
4	Reduce order failures that result in hung orders that require manual intervention.
5	Reduce order failures that result in lost orders.

Based on this representation of the scenarios, the stakeholder team proceeded to vote. This close-knit group chose to discuss each scenario and arrive at a determination of its weight via consensus. The votes allocated to the entire set of scenarios were constrained to total 100, as shown in Table 2. Although the stakeholders were not constrained to make the votes multiples of 5, they felt that this was a reasonable resolution, and that more precision in the votes was not needed and could not be justified. It also helped the team quickly reach consensus on the relative importance of the scnearios.

TABLE 2. Refined Scenarios with Votes

Sce-nario		Response Goals			
	Votes	Worst	Current	Desired	Best
1	10	10% hung	5% hung	1% hung	0% hung
2	15	> 5% lost	<1% lost	0% lost	0% lost
3	15	10% fail	5% fail	1% fail	0% fail
4	10	10% hung	5% hung	1% hung	0% hung
5	15	10% lost	<1% lost	0% lost	0% lost

5.3. Assign Utility

In this step the utility for each scenario was determined by the stakeholders, again by a consensus process. A utility score of 0 represents no utility; a score of 100 represents the most utility possible. The results of this process are given in Table 3. This step proved to be one of the most valuable because it helped the team quickly reach a consensus on how they thought the system should evolve.

TABLE 3. Adding Votes and Utility Scores

Scenario	Votes	Utility Scores			
		Worst	Current	Desired	Best
1	10	10	80	95	100
2	15	0	70	100	100
3	15	25	70	100	100
4	10	10	80	95	100
5	15	0	70	100	100

5.4. Develop ASs for Scenarios and Elicit Expected Response Levels

Based upon the requirements implied by the above scenarios, a set of 10 ASs was developed by the ECS architects. Recall that an AS may affect more than one scenario. To account for these complex relationships, we determined the expected quality attribute response level that each AS was predicted to achieve with respect to *each* relevant scenario. A subset of the ASs, along with the determination of the scenarios that they address is shown in Table 4. For each AS/scenario pair, the response levels that the AS is expected to achieve with respect to that scenario is shown (along with the current response, for comparison purposes).

TABLE 4. Architectural Strategies and Scenarios Addressed

AS	AS Name	Scenarios Affected	Current Response	Expected Response
1	Order persis tence	3	5% fail	2% Fail
		5	<1% lost	0% lost
4	Order seg-mentation	4	5% hung	2% hung
5	Order re-assignment	1	5% hung	2% hung
6	Order retry	4	5% hung	3% hung
7	Forced order completion	1	5% hung	3% hung

5.5. Determine Expected Utility and ROI

Once the expected response level of each AS with respect to a set of scenarios was characterized, we could calculate the *utility* of this expected response level. To do this we consulted the utility scores for each scenario's worst, current, desired, and best responses for all of the affected attributes. Using these scores we could interpolate the utility of the expected response levels for the AS/scenario pair. Using the votes from Table 2 as weights and the delta benefit scores, a total benefit for each AS could now be calculated.

To complete the analysis, a cost estimate for each AS was done, and based on these a return on investment was calculated for each architectural strategy. Using the ROI, we are then able to rank each of the architectural strategies. This is shown in Table 5. Note that these ranks differ from the AS numbers (which were the stakeholders' intuitive initial rankings of the ASs). For example, AS 7 has the third highest rank. If the stakeholders were using intuition alone they may well have overlooked this AS. This is evidence of the usefulness of a *method*, as opposed to ad hoc design decision making.

TABLE 5. ROI of Architectural Strategies

AS	Cost	Total AS Benefit	AS ROI	AS Rank
1	1200	950	0.79	1
4	200	100	0.5	3
5	400	120	0.3	7
6	200	50	0.25	8
7	200	70	0.35	6

5.6 Iteration II

In the second iteration, a number of important risk factors were discussed by the team that were not given significant consideration in iteration I, due to time constraints (and the triage focus of iteration I). Each risk pertained to one or more of the ASs that had been previously proposed. For each risk factor its probability of occurrence, the impact that it would have if it did occur, and any mitigation strategy were discussed and assessed.

The outcomes of the risks as well as the probabilities of their occurence were elicited in iteration II. Based on these elicited values expected cost and benefit ranges were determined. (Note that during iteration I we had *single values* for cost and benefit; now we have ranges.) The min and max values of these ranges are interpreted as end points of confidence intervals. These probability functions allowed us to produce a matrix that shows the confidence that we have in the rank orderings (from Table 5) given the additional information that we have collected regarding the uncertainties. This matrix is shown in Table 6.

TABLE 6. Probability that AS(row) > AS (column)

	AS1	AS4	AS5	AS6	AS7
AS1	0.5	0.64	0.82	0.94	0.69
AS4	0.36	0.5	0.89	0.99	0.6
AS5	0.18	0.11	0.5	0.89	0.05
AS6	0.06	0.01	0.11	0.5	0
AS7	0.31	0.4	0.95	1	0.5

Management, viewing these rankings and their associated probability values, used this information in some cases to commit to certain ASs, and in other cases to get better estimates of costs and benefits of two attractive alternatives that were too close to differentiate (i.e. when the probability that one is greater than the other is between 0.4 and 0.6).

5.7. Results

The most obvious results of the CBAM were an ordering of architectural strategies based upon their predicted ROI and a probability that shows the confidence in this position. But the benefits of the CBAM extend far beyond the qualitative outcomes. There have been palpable social and cultural benefits as well. The CBAM process provided a great deal of structure to what was largely unstructured discussions where requirements and architectural strategies and personal opinions are freely mixed together, and where stimuli and response goals are not clearly articulated. The CBAM process forced our stakeholders to make their scenarios clear in advance, to assign utility levels to specific response goals, and to prioritize scenarios based on the resulting determination of utility. The CBAM forced our stakeholders to address risks and their resulting effects explicitly, rather than simply stating an “unease” with a particular technical direction.

6. Conclusions

Elicitation of information from real world projects is difficult. As researchers we are charged with creating methods that are usable, by real-world engineers in real projects. These methods need to produce useful results quickly, and at a reasonable “price”, in terms of the time of the stakeholders. As we have discovered in our experiences with the CBAM, solving a problem in theory and in practice are very different. We have already modified the CBAM considerably as a result of the two applications of this method to the ECS.

The new version of the CBAM is an iterative elicitation process combined with a decision analysis framework. It incorporates scenarios to represent the various QAs. The stakeholders explore the decision space by eliciting Utility-QA response level curves to understand how the system’s utility varies with changing attributes. The consensus basis of the method allows for active discussion and clarification amongst the stakeholders. The traceability of the design decision permits updating and continuous improvement of the design process over time.

In spite of the practical difficulties in running such methods, we believe that the application of economic techniques is inherently better than the ad-hoc approaches that projects (even quite sophisticated projects) employ today. We cannot prove the optimality of our techniques Because we use expert judgments our techniques may not be optimal but they are at least satisficing. Our experience with the CBAM tells us that giving people appropriate tools with which to frame and structure their discussion and their decision making is an enormous benefit to the development of a software system.

7. References

[1] J. Asundi, R. Kazman and M. Klein, “Using Economic Considerations to Choose Amongst Architecture Design Alternatives”, CMU/SEI-2001-TR-035, SEI, Carnegie Mellon University, 2001.

[2] B. Boehm, *Software Engineering Economics*, Prentice Hall, 1981.

[3] A. Dixit, R. S. Pindyck, *Investment Under Uncertainty*, Princeton University Press, New Jersey, 1994.

[4] J. Karlsson, K. Ryan, “A Cost-Value Approach for Prioritizing Requirements”, *IEEE Computer*, September/ October 1997.

[5] R. Kazman, M. Barbacci, M. Klein, S. J. Carriere, S. G. Woods, “Experience with Performing Architecture Tradeoff Analysis”, *Proceedings of the ICSE 21*, Los Angeles, CA, May 1999, 54-63.

[6] R. Kazman, J. Asundi, M. Klein, “Quantifying the Costs and Benefits of Architectural Decisions”, *Proceedings of the ICSE 23*, Toronto, Canada, May 2001, 297-306.

[7] R. Kazman, J. Asundi, M. Klein, “Making Architecture Design Decisions: An Economic Approach”, CMU/SEI-2002-TR-035, SEI, Carnegie Mellon University, 2002.