Using Additive Multiple-Objective Value Functions for Value-Based Software Testing Prioritization

Qi Li

University of Southern California Computer Science Department,

University Park Campus, Los Angeles, CA 90089, USA

qli1@usc.edu

1. Introduction

Testing is one way of validating that the customers have specified the right product and verifying that the developers have built the product right. Traditional testing methodologies such as: path, branch, instruction, mutation, scenario, or requirement testing usually treat all aspects of software as equally important [1], and is often reduced to a purely technical issue leaving the close relationship between testing and business decisions unlinked and the potential value contribution of testing unexploited [2]. Value-based software testing serves to bridge the gap between the internal software testing process and business value that comes from customers and market [3]. The essential of Value-based software testing is feature prioritization that brings business importance that comes from CRACK key customers, software quality risk from developing team, software testing cost, process estimation and controlling from the testing team, and the external commercial factors, e.g., market pressure from the market into consideration. This decision making process could be seen as a multiple-objective decision making process that tries to maximize the business importance, identify the most risky features, minimize the testing cost and minimize the market erosion. The prioritization strategy could be reflected and influenced by different multiple-objective decision value functions. In [3], a multiplicative multiple-objective value function is used to generate the testing priority ranking. In this paper, we will use additive multiple-objective value function to generate testing priority ranking and compare the result in the end.

2. Related Work

2.1. Value-based Software Testing Process with the Previous Case Study

Figure 1 illustrates the whole process of this value-based software testing method. This method helps test manager consider all the win-conditions from Success-Critical Stakeholders (SCSs), enact the testing plan and adjust it during testing execution. The main steps are as follows:

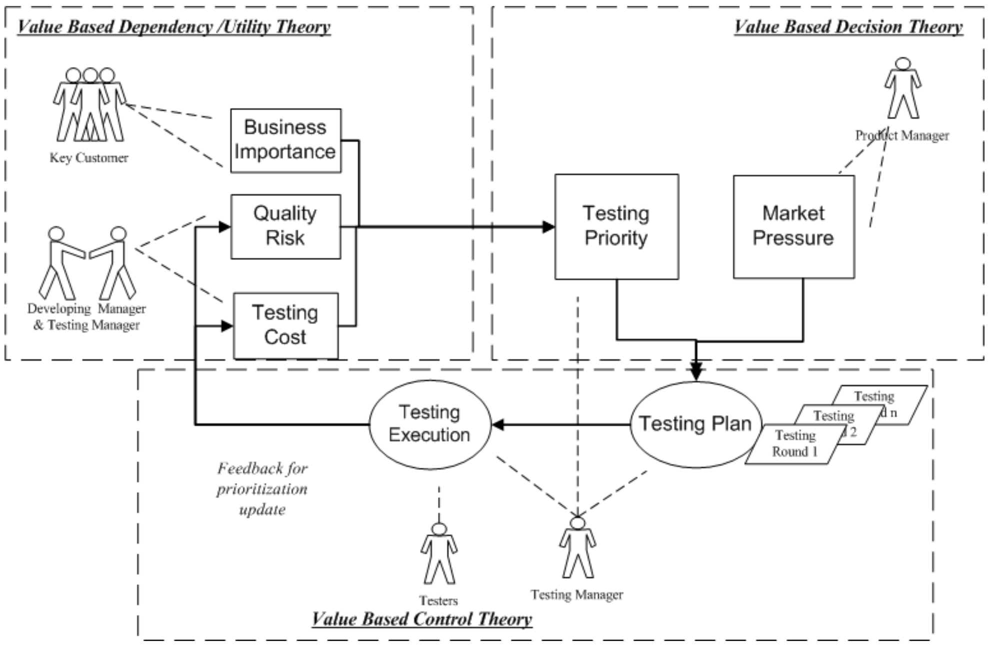


Figure 1. Software Testing Process-Oriented Expansion of "4+1" VBSE Framework [3]

1) *Define Utility Function of Business Importance, Quality Risk Probability and Cost.*

- With the assist of the key CRACK customer, the testing manager use a method first proposed by Karl Wiegers [4] to get the relative Business Importance for each feature as shown in Table 1.
- The developing manager and the test manager accompanied with some experienced developers calculate the quality risk probability of each feature as displayed in Table 2 .
- The test manager with the developing team estimate the testing cost for each feature as shown in Table 3.

2) *Testing Prioritization Decision for Testing Plan*

Then we combine Business Importance, Quality Risk Probability and Testing Cost to get the Value Priority by using the following multi-objective multiplicative formula. Basically, as shown in the following multiplicative formula

$$Value\,\Pr iority = \frac{BI * \Pr obability}{Cost}$$

The single value functions of "Business Importance" and "Risk Probability" are numerators of this division, because they are increasing preference, the larger the "Business Importance" or "Risk Probability", the higher the testing priority. For the single value function of "Testing Cost", it is the denominator of the division equation, because it is decreasing preference, the larger the Testing Cost, the lower the testing priority value. Process implication is that testing features with higher business importance to CRACK representative customers, higher quality risk probability and lower testing cost should have higher priority to be tested. This value function for testing priority displays the win conditions from different stakeholders, e.g. testing team, customers, and developing team. The value priority results are displayed in Table 4.

2.2. Data Resource

Table 1. Relative Business Importance Calculation

	Business Importance			
	Benefit	Penalty	Total BI	BI %
Weights	2	1		
F1	9	7	25	30.9%
F2	8	7	23	28.4%
F3	1	3	5	6.2%
F4	2	1	5	6.2%
F5	1	1	3	3.7%
F6	2	1	5	6.2%
F7	3	2	8	9.9%
F8	1	2	4	4.9%
F9	1	1	3	3.7%
SUM	28	25	81	1

Table 2. Quality Risk Probability Calculation (Before System Testing)

	Initial Risks				**Feedback Risks**		**Probability**
	Personnel Proficiency	Size	Complexity	Design Quality	Defects Proportion	Defects Density	
weights	**0.09**	**0.19**	**0.03**	**0.04**	**0.27**	**0.38**	
F1	5	3	1	1			**0.13**
F2	4	9	5	2			**0.26**
F3	3	3	5	5			**0.14**
F4	5	4	7	5			**0.19**
F5	5	2	3	3			**0.12**
F6	5	2	5	6			**0.14**
F7	5	4	5	2			**0.17**
F8	1	2	1	1			**0.06**
F9	1	1	1	1			**0.04**

Table 3. Relative Testing Cost Estimation

	Cost	Cost%
F1	2	4.8%
F2	5	11.9%
F3	5	11.9%
F4	9	21.4%
F5	6	14.3%
F6	4	9.5%
F7	5	11.9%
F8	3	7.1%
F9	3	7.1%
SUM	42	1

Table 4. Value Priority Calculation

	BI %	Probability	Cost%	Value Priority
F1	31	0.13	5	0.81
F2	28	0.26	12	0.63
F7	10	0.17	12	0.14
F6	6	0.14	10	0.09
F3	6	0.14	12	0.07
F4	6	0.19	21	0.05
F8	5	0.06	7	0.04
F5	4	0.12	14	0.03
F9	4	0.04	7	0.02

3. Using Additive Multiple-Objective Value Functions for Value-Based Software Testing Prioritization

In this part, we change the testing priority value function from multi-objective multiplicative multi-objective additive value function, and the formula is:

$$V\,(X_{BI}+X_C+X_{RP}) = W_{BI} * V\,(X_{BI}) + W_C * V\,(X_C) + W_{RP} * V\,(X_{RP})$$

V (X_{BI}), V (X_C) and V (X_{RP}) are single value functions for "Business Importance", "Cost" and "Risk Probability". W_{BI}, W_C and W_{RP} are relative weights for them respectively. V ($X_{BI}+X_C+X_{RP}$) is the multi-objective additive value function for testing priority. For simplicity, here we assume that $W_{BI} = W_C = W_{RP}$, In fact, we did the same way for the priority multiplicative value function. Sections 3.1-3.3 show the steps of how to use additive multiple-objective value functions to generate the testing priority result and we continue using the data from Table 1 to 3 from previous case study [3].

3.1. Define Objective Hierarchy

The first step is to define the objective hierarchy as shown in Figure 2, the decision making goal is to prioritize the testing features. Three important factors should be considered when prioritizing as we have discussed in section 3: the higher the business importance the feature is of, the higher the priority is; the higher the quality risk probability the feature is of, the higher the priority is; the lower the testing cost is, the higher the priority is. So for the sub-goals, they should be maximize business importance, maximize quality risk, and minimize testing cost.

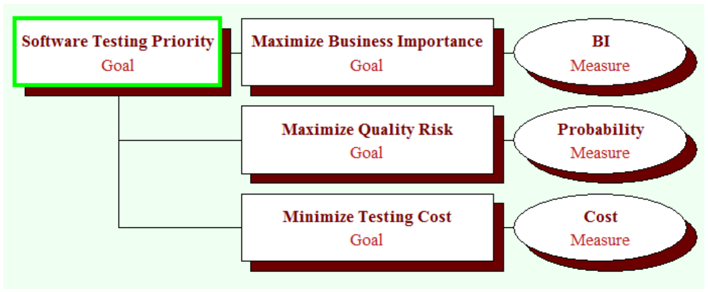
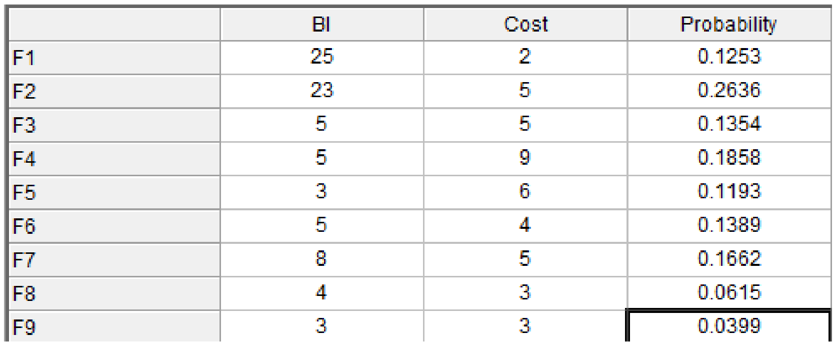


Figure 2. Software Testing Priority Objective Hierarchy

3.2. Define the Range for Each Measure

The second step is to define the range for each measure. The Business Importance is composed of two parts, one is the benefit of having this feature, the other is the penalty of absence of this feature, and benefit is twice important as penalty in this case study. Because we score importance from 1-9, so the total Business Importance from these two sides is in the range 3-27. We get the total business importance values from the "Total BI" in Table 1. For the cost, it is obvious that the range is from 1 to 9 and we get the value from Table 3. For the Probability, as it has been defined in step 3 of section 4.1.2, its range is 0 to 1 and we get the value from Table 2. So we get the value matrix for measures before system test in Table 5 and measures' range in Table 6.

Table 5.Concequence Matrix for measures



	BI	Cost	Probability
F1	25	2	0.1253
F2	23	5	0.2636
F3	5	5	0.1354
F4	5	9	0.1858
F5	3	6	0.1193
F6	5	4	0.1389
F7	8	5	0.1662
F8	4	3	0.0615
F9	3	3	0.0399

Table 6. Measures' Range

Measures	**Range**
Business Importance	[3-27], ↑ the testing priority is higher
Quality Risk Probability	[0-1], ↑ the testing priority is higher
Testing Cost	[1-9], ↓ the testing priority is higher

3.3. Define Exponential Value Function for Each Measure

The third step is to define the single value function for each measure. Usually, the value function of measure is not simple linear function, and the more reasonable form is the exponential value function [5]. The equations for the exponential have a particular form, which depends on the range of the evaluation measure and the exponential constant ρ. ρ determines the shape of the exponential value function. If ρ is infinity, the exponential value function tends to be linear.

If preference is monotonically increasing over an evaluation measure x, that is, higher amounts of x are preferred to lower amounts, the measures "Business Importance" and "Quality Risk Probability" are just the case. Then the exponential single dimensional value function v(x) can be written as follows [5]:

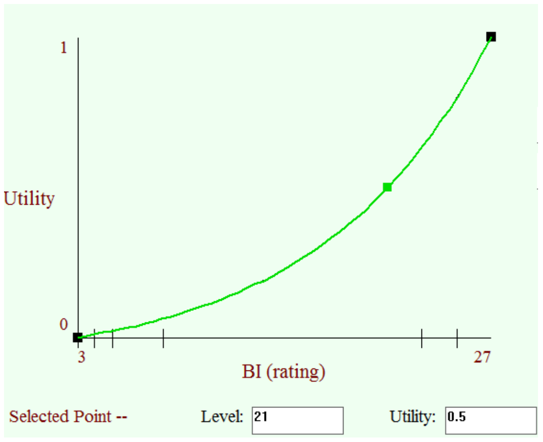
$$V(x)=\begin{cases}\dfrac{1-\exp[-(x-Low)/\rho]}{1-\exp[-(High-Low)/\rho]}, \rho \neq Infinity\\ \dfrac{x-Low}{High-Low}, otherwise\end{cases}$$

And if preference is monotonically decreasing over x, that is, lower amounts of x are preferred to higher amounts, the measure "Testing Cost" is just the case. Then the value function v(x) can be written as follows [5]:

$$V(x)=\begin{cases}\dfrac{1-\exp[-(High-x)/\rho]}{1-\exp[-(High-Low)/\rho]}, \rho \neq Infinity\\ \dfrac{High-x}{High-Low}, otherwise\end{cases}$$

The procedure to determine the value of ρ for a specific exponential single dimensional value function depends on the concept of the mid-value for the range of evaluation measure scores such that the difference in value between the lowest score in the range and the mid-value is the same as the difference in value between the mid-value and the highest score [5]:

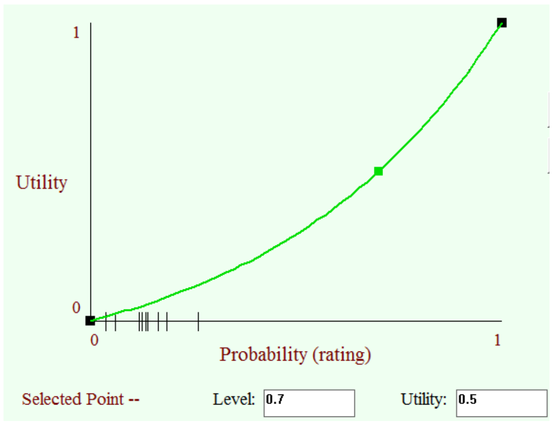
For Business Importance, it is increasing preference, the larger the BI, the higher the priority value. The key customer agrees that the difference in value between the lowest score 3 and the mid-value 21 is the same as the difference in value between the mid-value 21 and the highest score 27. This would imply the key customer's emphasis on high business importance features.



Selected Point -- Level: 21 Utility: 0.5

Figure 3. Value Function for "Business Importance"

For Quality Risk Probability, it is also increasing preference, the larger the Probability, the higher the priority value. The test manager agrees that the difference in value between the lowest score 0 and the mid-value 0.7 is the same as the difference in value between the mid-value 0.7 and the highest score 1. This could imply the test manager's emphasis on features with high risk probability.



Selected Point -- Level: 0.7 Utility: 0.5

Figure 4. Value Function for "Quality Risk Probability"

For Testing Cost, it is decreasing preference, the larger the Cost, the lower the priority value. The test manager agrees that the difference in value between the lowest score 1 and the mid-value 7 is the same as the difference in value between the mid-value 7 and the highest score 9. This could imply the test manager's avoidance on features with high testing cost.

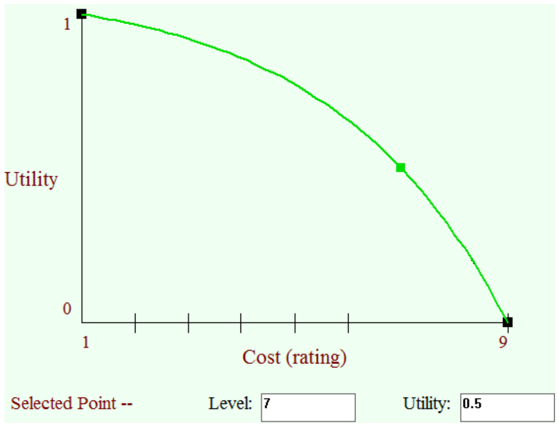


Figure 5. Value Function for "Testing Cost"

4. Result

The multi-objective additive value rankings are displayed as shown in Figure 6. We could see that F1 and F2's priorities are still far ahead of others. Because this prioritization is conducted before system testing, it is the same that we should test F1 and F2 in the test round 1 as we use multiplicative value function form. That means that in both ways, we could get the similar prioritizations that maximize the ROI of the testing investment.

It should also be noted that after using the exponential value function, decision makers' preferences, the emphasis on high business importance (e.g. F1, F2) and avoidance of high cost (e.g. F4) are obviously displayed in the stacked bar ranking graph. Because this prioritization is conducted before system testing, the quality risk probability for all features is low as shown in Table 2, all the 9 features shows narrow value bars on Quality Risk Probability. For Testing Cost, we could see that except F4, all others' value bars have no obvious differences, this shows that testing manager's tolerance of testing cost, however, if the testing cost is very high, e.g. F4 with score 9, the value decreases dramatically. We think that exponential single value functions can more realistically reflect the decision maker's value.

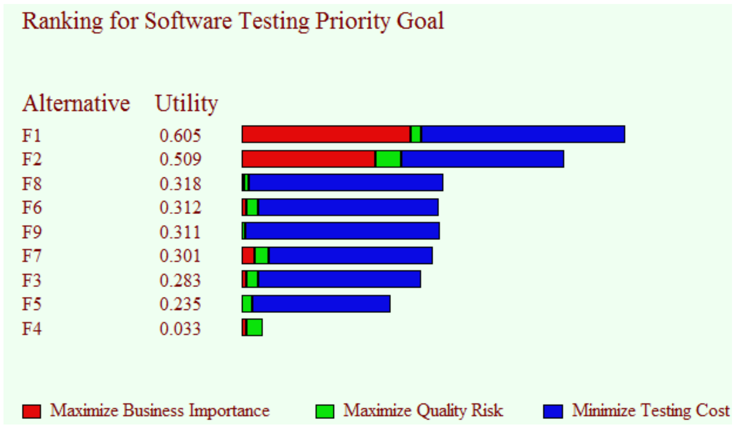


Figure 6. Stacked Bar Ranking for Software Testing Priority

5. Sensitivity Analysis

We do a sensitivity analysis of the measure “Business Importance” as shown in Figure 7, we can see that if we don’t consider about “Business Importance” at all, F1 is still the one with highest priority for testing, because it has the lowest testing cost and not small quality risk probability. As more weight is placed on “Business Importance”, F1 and F2 are still the most preferred.

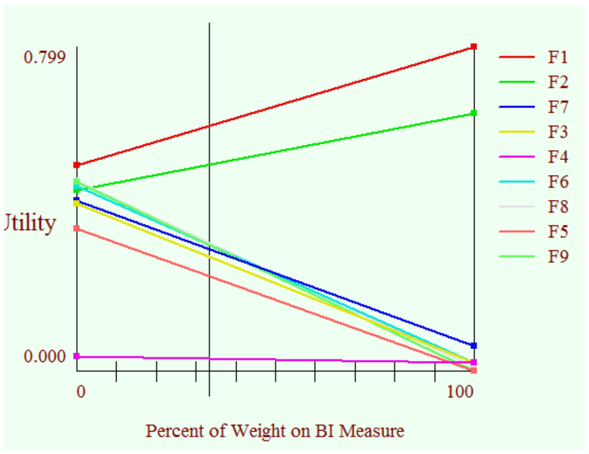


Figure 7. Sensitivity Analysis for “BI” Weights

The same sensitivity analysis are also done for other two measures “Quality Risk Probability” and “Testing Cost” as shown in Figure 8 and Figure 9, we could see that F1 and F2 still dominate at most time. It should be noted that Quality Risk Probability, not like other two measures, as the weight of the measure increases, the 9 features shows less and less differences. That is because in terms of “Quality Risk Probability”, these 9 features don’t have much difference before system testing starts. However, since the prioritization for testing features is not a static process, after the testing starts and defects are collected, the probability differences become obvious, and these differences could finally be reflected in larger value bars in Figure 6, and wider differences when the weight reaches 100% in Figure 8.

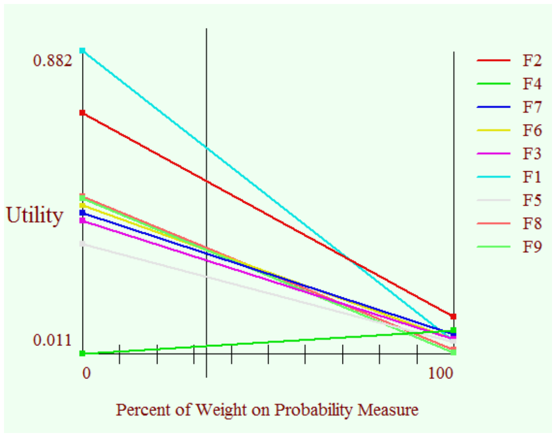


Figure 8. Sensitivity Analysis for “Quality Risk Probability” Weights

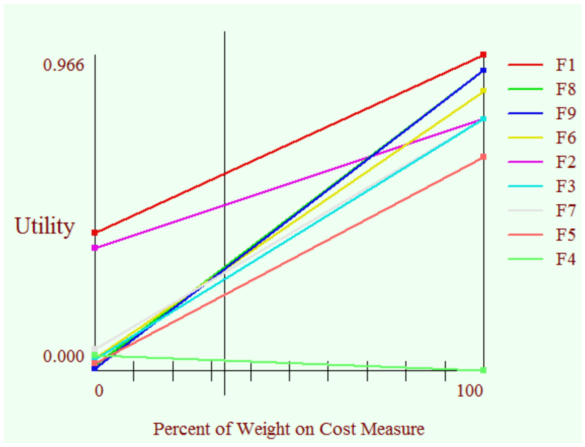
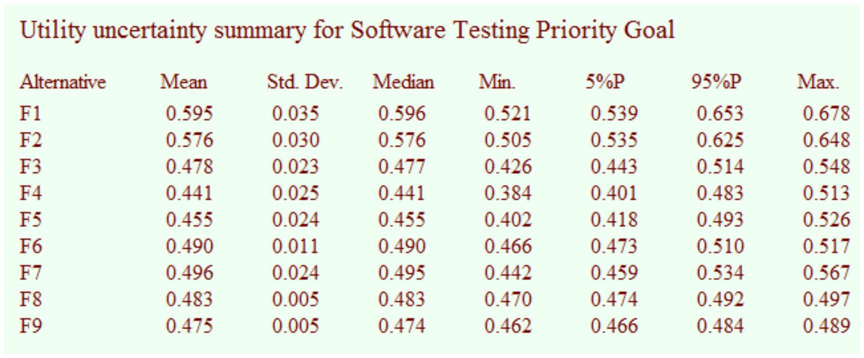


Figure 9. Sensitivity Analysis for "Testing Cost" Weights

6. With Uncertainty

It is easier and more meaningful if we could give the inputs "Business Importance", "Quality Risk Probability" and "Cost" probability distribution rather than point estimations, especially when the project information is not available and incomplete. This could also provide more useful decision support information for decision makers. The below is one example of estimation with uncertainty for the testing priority. We could see that although F1 has the largest standard deviation, however, its minimum value is 0.521 that still dominates F6, 7, 8, 9 since their maximum values are all less than F1's minimum value.



Utility uncertainty summary for Software Testing Priority Goal

Alternative	Mean	Std. Dev.	Median	Min.	5%P	95%P	Max.
F1	0.595	0.035	0.596	0.521	0.539	0.653	0.678
F2	0.576	0.030	0.576	0.505	0.535	0.625	0.648
F3	0.478	0.023	0.477	0.426	0.443	0.514	0.548
F4	0.441	0.025	0.441	0.384	0.401	0.483	0.513
F5	0.455	0.024	0.455	0.402	0.418	0.493	0.526
F6	0.490	0.011	0.490	0.466	0.473	0.510	0.517
F7	0.496	0.024	0.495	0.442	0.459	0.534	0.567
F8	0.483	0.005	0.483	0.470	0.474	0.492	0.497
F9	0.475	0.005	0.474	0.462	0.466	0.484	0.489

Figure 10. Priority Uncertainty

7. Discussion and Conclusion

- Weights assessment plays an important role in the multi-objective decision making process, no matter for multiplicative or addictive value function form. In our case study, we use direct assessment for weights of benefits and penalty (2:1) for calculating total business importance, also for weights of Business Importance, Testing Cost and Quality Risk Probability (1:1:1). For calculating Quality Risk Probability, we use AHP method for risk items weights assessment [3]. AHP has been proved that it could reduce the subjective biases for weights assessment, but has a more complex and costly procedure when compared to direct assessment, so this is also a trade-off between cost and accuracy. Weights assessment should be elicited based on thoroughly understanding of all attribute measures' value functions and their relative importance.

- Extension from the multiplicative value function to additive one also shows the similar feature testing priorities result. No matter the value function is multiplicative or additive, as long as they reflect reasonably the similar SCSs' win condition' preferences, and they are supposed to generate the similar priority results. Both dynamic prioritizations could make the ROI of testing investment reach the peak at the early stage of testing, which is especially effective when the time to market is limited. This extension of value function is supported by Value-Based Utility Theory.

Reference:

1. Boehm, B., Value-Based Software Engineering. *ACM Software Engineering Notes*, **2003; 28(2).**

2. Ramler, R., S. Biffl, and P. Grunbacher, Value-Based Management of Software Testing, *Value-Based Software Engineering*. 2005, Springer. pp. 226-244.

3. Li, Q., et al., Bridge the Gap between Software Test Process and Business Value: A Case Study. *In Proceedings of International Conference on Software Process*:2009, pp 212-223

4. E.Wiegers, K., First Things First: Prioritizing Requirements. Software Development, 1999; 7(10): pp. 24-30.

5. Craig W. Kirkwood, *Strategic Decision Making: Multiobjective Decision Analysis with Spreadsheets*, 1996, Duxbury Press