The CBAM

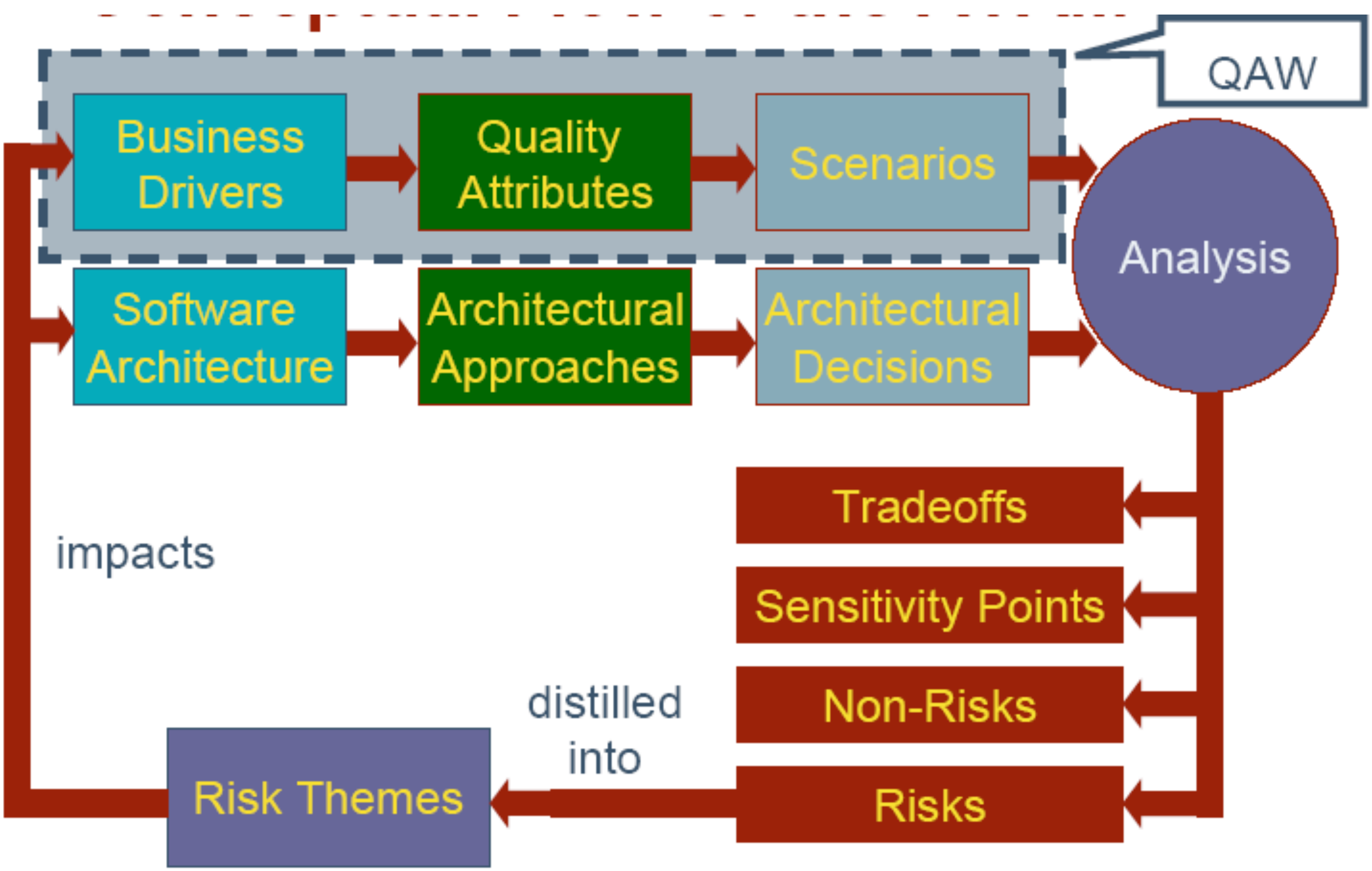
Economic Analysis of Architectures

主讲教师： 王灿
Email: wcan@zju.edu.cn
TA: 李奇平 liqiping1991@gmail.com
Course FTP: ftp://sa:sa@10.214.51.13

ATAM

- A risk-mitigation process used early in the software development life cycle
  - A means of evaluating technical tradeoffs
    - Architectural tradeoffs occur when a decision affects multiple attributes at once
  - How well the architecture has been designed with respect to quality attributes

Conceptual Flow of the ATAM



What is CBAM?

- Cost Benefit Analysis Method
- Provides assessments of the technical and economic issues and architectural decisions
- Starts where ATAM ends

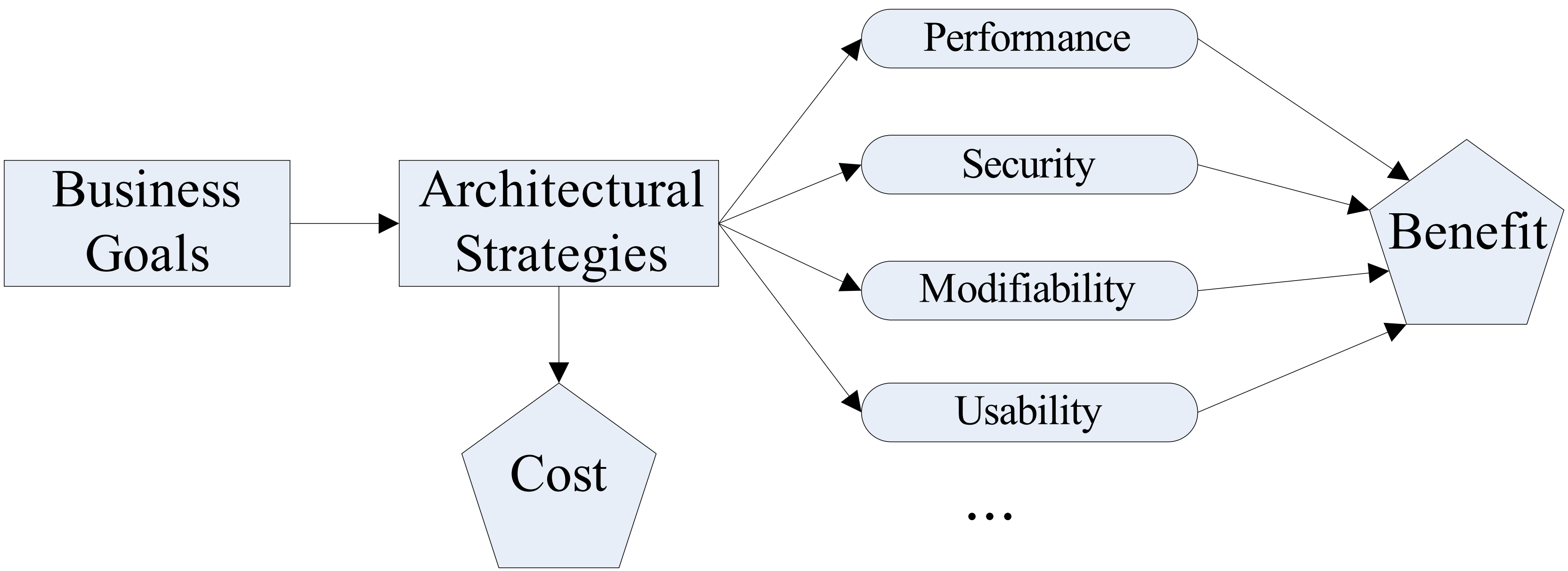
Motivations

- The biggest tradeoffs in large systems usually have to do with economics

- We need a process that helps us choose among architectural options, given that the resources for building and maintaining a system are limited
  - Maximize gains and minimize risks (financially)

Decision-Making Context (1)

- Goal: to maximize the difference between the benefit derived and the cost of implementation



Decision-Making Context (2)

- The budget for a project is always limited.
  - Every architectural choice is competing with every other one for inclusion
  - CBAM allows the stakeholders to judge whether the costs for different decisions are worthwhile or not
- CBAM elicits the costs and benefits associated with the architectural decisions

Decision-Making Context (3)

- Architectural strategies (collections of architectural tactics) affect the quality attributes of the system, which in turn provides the stakeholders with some benefit (utility)
  - E.g. to improve system performance, one can choose to add more servers or have developing team to come up with better algorithms
- These strategies also have cost and take time to implement
  - Cost in purchasing and deploying new servers, or in design, implement, test and deploy new modules.
- VFC (Value of Cost, *previously known as ROI, Return on Investment*) - the ratio of benefit to cost

The Basis for CBAM---Utility

- We consider how scenarios differ in the values of their projected responses and then assign utility to those values
  - Side effects of strategies must also be taken into account (other quality attributes that are affected other than the one intended)
  - Calculating all of these utilities together is necessary for the VFC

Utility: Variations of Scenarios

- The CBAM uses scenarios, similar to ATAM, to represent specific quality attributes
- To create a utility-response curve, the CBAM uses a ***set*** of scenarios that vary the values of the responses (scenario = stimulus, environment, and response)

Variations of Scenarios---an example

- You submit an order in an e-commerce website by clicking on "*Proceed to checkout*" button, and the response page is displayed in:

0.5 second

2 seconds

5 seconds

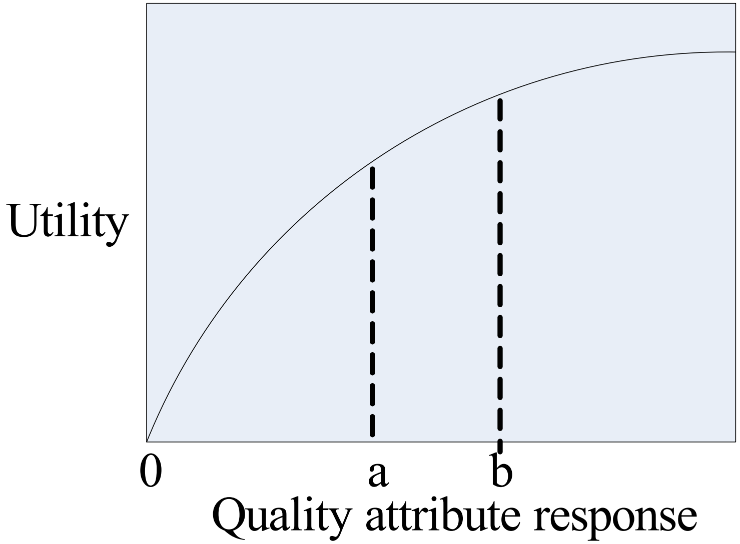
10 seconds

15 seconds

30 seconds

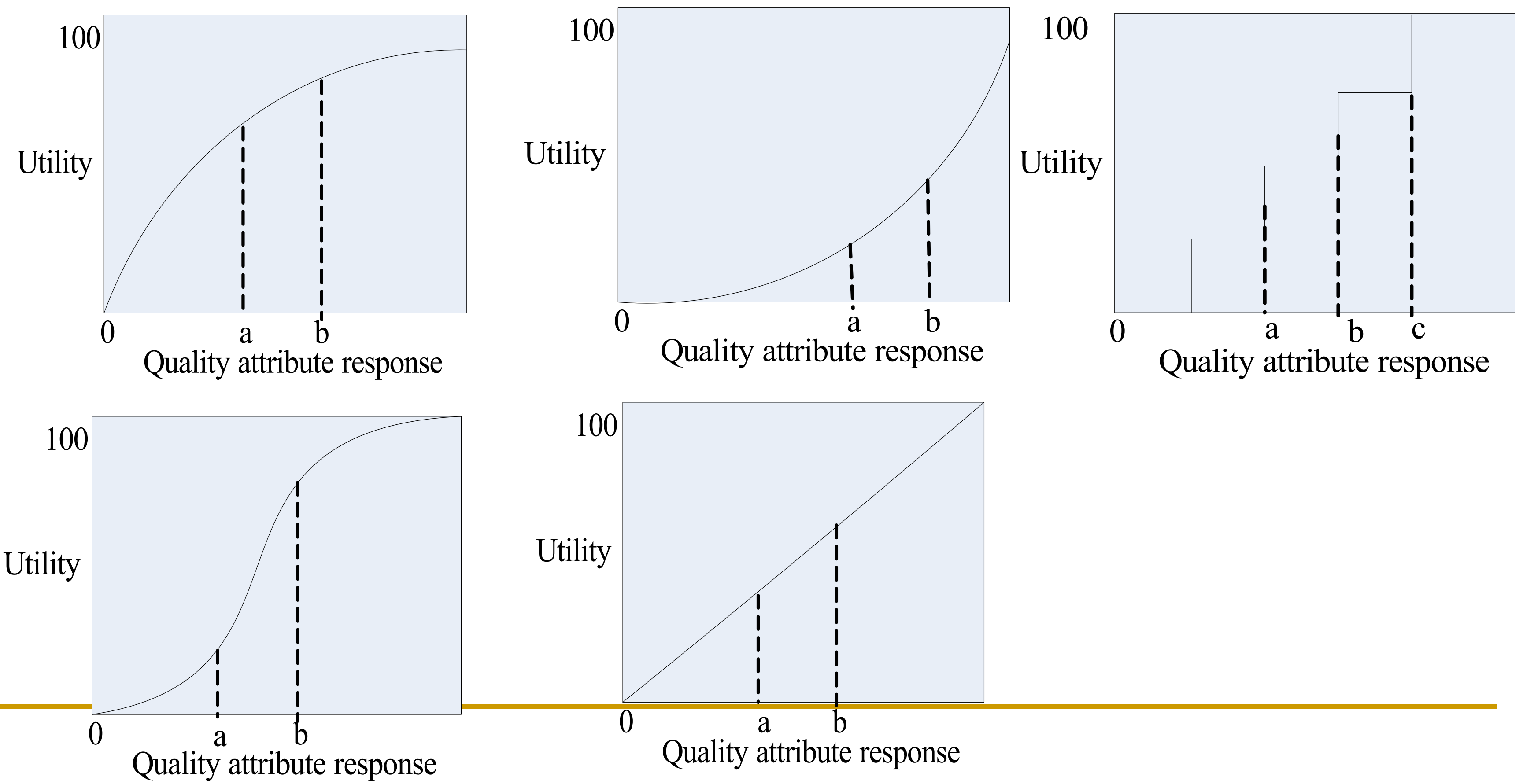
Utility: Utility-Response Curves (1)

- Utility is shown as a function of response value
- Determine the following points for a UR curve
  - Best case: e.g. value 100
  - Worst case: e.g. value 0
  - Current case
    - Value proposed by stakeholders
  - Desired case
    - Value proposed by stakeholders



Utility: Utility-Response Curves (2)

- UR curves may take various forms.



Utility: Priorities of Scenarios

- Two-step voting process to establish relative importance of a scenario
  - Stakeholders vote to establish an order to scenarios
  - Stakeholders then assign a weight of 1 to the highest-rated and a fraction to each additional one
- For the additional scenarios added in the future, the stakeholders make sure the weights are correct through consensus.

Utility: Architectural Strategies (1)

- One responsibility of the architect is to move from current to expected (or best-case) on the curve
- This is done with architectural strategies, for each strategy we can derive:
  - The ***expected value*** of the response in each scenario (utility found on curve)
  - The effect of the architectural strategy on other attributes of interest
  - A cost estimate for implementing the strategy

Utility: Architectural Strategies (2)

- Also need to determine utility of tradeoffs/side effects
  - The expected utility for a particular attribute may be negative
  - At worst case, we need to develop a whole new curve for this side effect
- Calculate the overall utility of an architectural strategy across scenarios
  - Summing the utility associated with each one, weighted by the importance of the scenario

$$B_i = \sum_j (b_{i,j} \times w_j)$$

Calculating VFC

- The VFC value for each architectural strategy is the ratio of the total benefit, B_i, to the cost, C_i, of implementing it. (C_i can be derived from the cost models) $$VFC_i = \frac{B_i}{C_i}$$
  - VFC VS. slope of the utility-responsive curve
- Using VFC score, architectural strategies can be rank ordered to determine optimal order for implementation

CBAM Steps (1)

- Step 1: Collate scenarios
  - Prioritize these scenarios based on satisfying the business goals of the system and choose the top one-third for further study
- Step 2: Refine scenarios
  - Elicit the worst-case, current, desired, and best-case quality attribute response level for each scenario

Scenario	Worst Case	Current	Desired	Best Case
Scenario #17: Response to user input	12 seconds	1.5 seconds	0.5 seconds	0.1 seconds

- Step 3: Prioritize scenarios
  - Prioritize scenarios using the two-step voting process; eliminate half of the scenarios

CBAM Steps (2)

- Step 4: Assign utility
  - Determine the utility for each quality attribute response level (worst-case, current, desired, best-case) for the scenarios from step 3

Scenario	Worst Case	Current	Desired	Best Case
Scenario #17: Response to user input	12 seconds	1.5 seconds	0.5 seconds	0.1 seconds
	Utility 5	Utility 50	Utility 80	Utility 85

- Step 5: Map architectural strategies to scenarios and determine their expected quality attribute response levels
- Step 6: Determine the utility of the "expected" quality attribute response levels by interpolation

$$y = y_a + (y_b - y_a)\,\frac{(x - x_a)}{(x_b - x_a)}$$

CBAM Steps (3)

- Step 7: Calculate the total benefit obtained from an architectural strategy

$$\sum((\exp ected - current) \times votes)$$

- Step 8: Choose architectural strategies based on VFC subject to cost and schedule constraints
  - Determine the cost and schedule implications of each architectural strategy; choose the top ones until the budget or schedule is exhausted
- Step 9: Confirm results with intuition
  - Check whether the architectural strategies satisfy the organization's business goals

Case Study: The NASA ECS Project

- The Earth Observing System is a constellation of NASA satellites that gathers data 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
  - ECS's mission is to process the data into higher-form information and make it available to scientists in searchable form
  - Performance
  - Availability
  - Modifiability
- The ECS project manager had a limited annual budget to maintain and enhance his current system

Step 1: Collate 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.
6	**There is no good method of tracking ECS Guest failed/canceled orders without much manual intervention (e.g., spreadsheets).**
7	Users need more information on why their orders for data failed.
8	Because of limitations, there is a need to artificially limit the size and number of orders.
9	**Small orders result in too many notifications to users.**
10	**The system should process a 50-GB user request in one day, and a 1-TB user request in one week.**

Step 2: Refine Scenarios

	Response Goals			
Scenario	**Worst**	**Current**	**Desired**	**Best**
1	10% hung	5% hung	1% hung	0% hung
2	> 5% lost	< 1% lost	0% lost	0% lost
3	10% fail	5% fail	1% fail	0% fail
4	10% hung	5% hung	1% hung	0% hung
5	10% lost	< 1% lost	0% lost	0% lost
6	**50% need help**	**25% need help**	**0% need help**	**0% need help**
7	10% get information	50% get information	100% get information	100% get information
8	50% limited	30% limited	0% limited	0% limited
9	**1/granule**	**1/granule**	**1/100 granules**	**1/1,000 granules**
10	**< 50% meet goal**	**60% meet goal**	**80% meet goal**	**> 90% meet goal**

Step 3: Prioritize Scenarios

		Response Goals			
Scenario	**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
6	**10**	50% need help	25% need help	0% need help	0% need help
7	**5**	10% get information	50% get information	100% get information	100% get informatio n
8	**5**	50% limited	30% limited	0% limited	0% limited
9	**10**	1/granule	1/granule	1/100 granules	1/1,000 granules
10	**5**	< 50% meet goal	60% meet goal	80% meet goal	> 90% meet goal

Step 4: Assign Utility

		Utility Scores			
Scenario	**Votes**	**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**
6	**10**	**0**	**80**	**100**	**100**
7	5	**10**	**70**	**100**	**100**
8	5	**0**	**20**	**100**	**100**
9	**10**	**50**	**50**	**80**	**90**
10	**5**	**0**	**70**	**90**	**100**

Step 5: Map Architectural Strategies to Scenarios & Determine Their Expected Quality Attribute Response Levels

Strategy	**Name**	**Description**	**Scenarios Affected**	**Current Response**	**Expected Response**
1	Order persistence on submission	Store an order as soon as it arrives in the system.	3 5 6	5% fail <1% lost 25% need help	2% Fail 0% lost 0% need help
2	…	……	…	…….	…….
3	Order bundling	Combine multiple small orders into one large order.	9 10	1 per granul 60% meet goal	1 per 100 55% meet goal
…… totally 10 scenarios, see P320 Table 12.5 for more details					

Step 6: Determine the Utility of the "Expected" Quality Attribute Response Levels by Interpolation

Strategy	Strategy	Scenarios Affected	Current Utility	Expected Utility
1	Order persistence on submission	**3** **5** **6**	**70** **70** **80**	**90** **100** **100**
2	Order chunking	**8**	**20**	**60**
3	Order bundling	**9** **10**	**50** **70**	**80** **65**
4	Order segmentation	**4**	**80**	**90**
		…….		

Step 7: Calculate the Total Benefit Obtained From an Architectural Strategy

Strategy	Scenario Affected	Scenario Weight	Raw Architectural Strategy Benefit	Normalized Architectural Strategy Benefit	Total Architectural Strategy Benefit
1	3	15	20	300	
1	5	15	30	450	
1	6	10	20	200	950
2	8	5	40	200	200
3	9	10	30	300	
3	10	5	-5	-25	275

Step 8: Choose Architectural Strategies Based on VFC Value Subject to Cost Constraints

Strategy	Cost	Total Strategy Benefit	Strategy VFC	Strategy Rank
1	1200	950	0.79	1
2	400	200	0.5	3
3	400	275	0.69	2
4	200	100	0.5	3
5	400	120	0.3	7
6	200	50	0.25	8
7	200	70	0.35	6
8	300	150	0.5	3
9	1000	225	0.22	10
10	100	25	0.25	8