



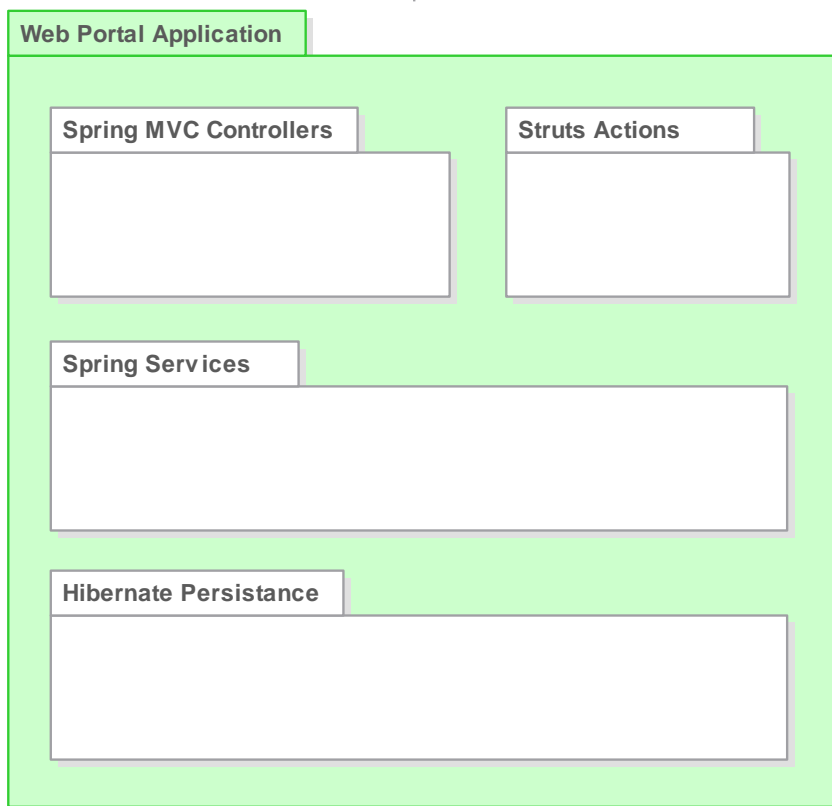
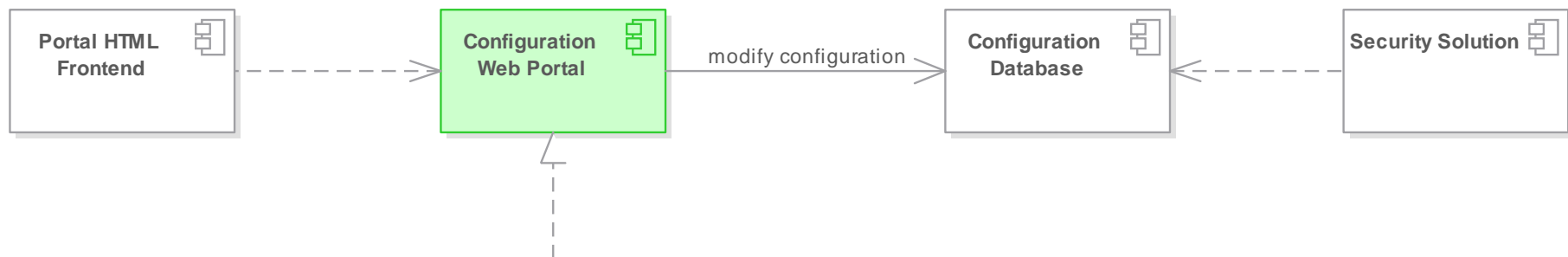
# COST-BENEFIT ANALYSIS IN TECHNICAL DEBT REDUCTION

Andriy Shapochka

April 2015

# Our Agenda





Portal Size (LOC)	55K+
File Number	800+
Class Number	860+

**Context:** Development team works on a Java Web Portal which is a part of larger network security solution

# Client's Concerns



Development team productivity degradation



More effort required to achieve the production quality

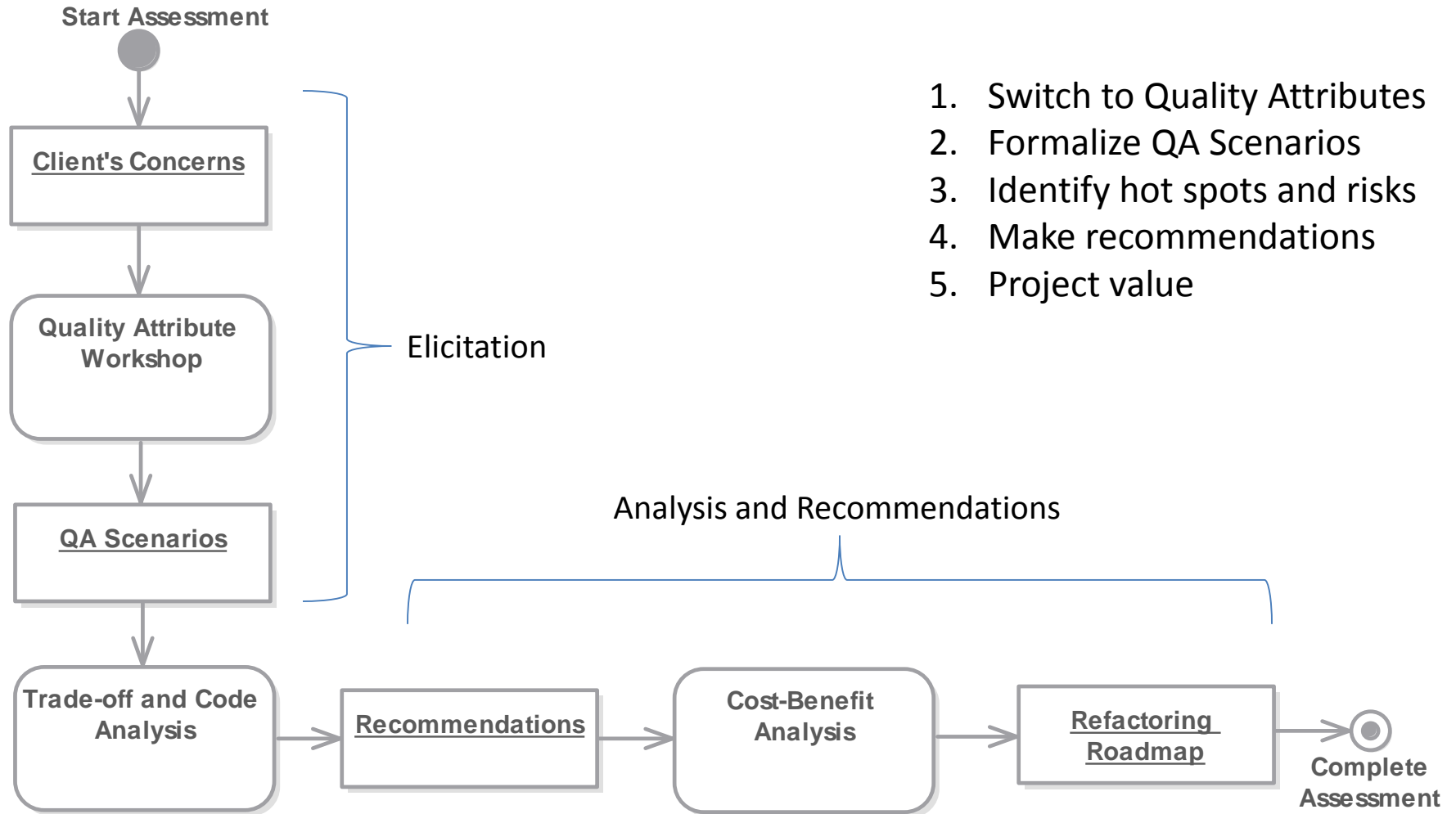


Concerns about design and implementation quality

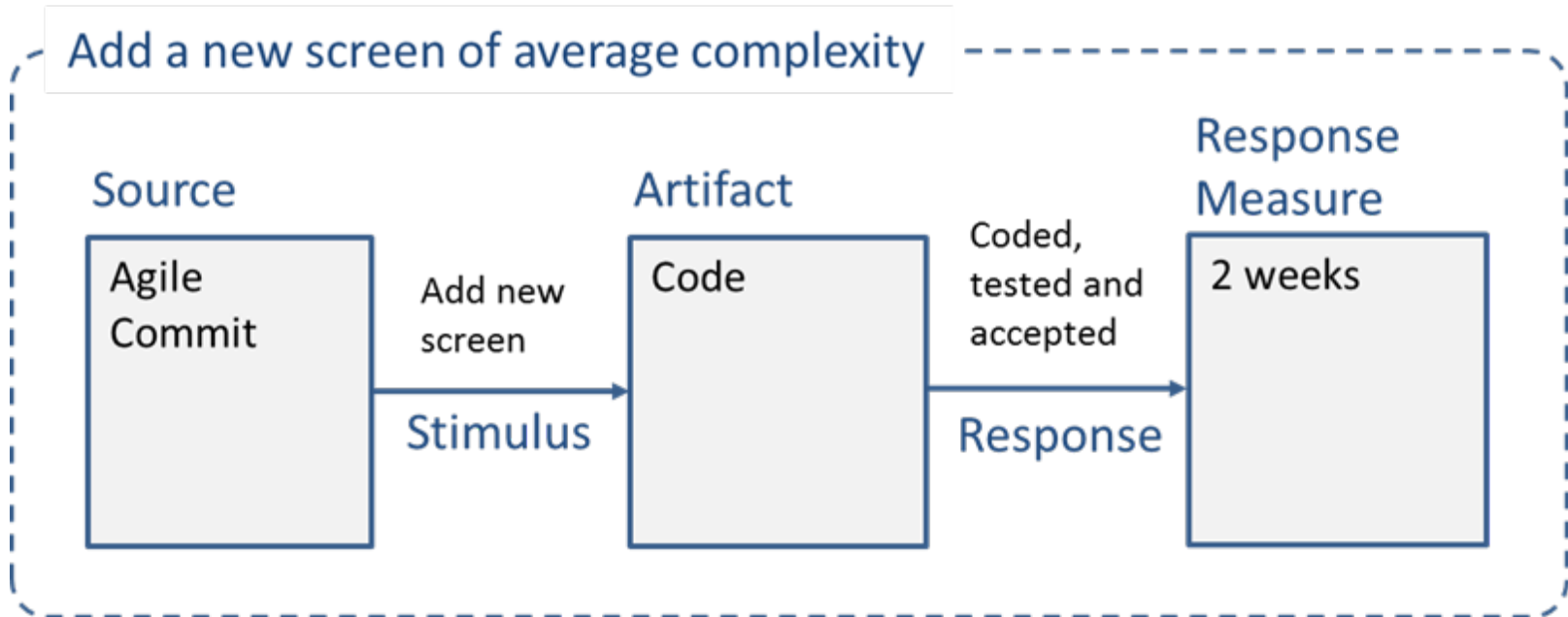



**Assessment Challenge:** Find and efficiently address technical issues causing the business problems

# Approach: Architectural Assessment



# QAW: Representative Scenario



ID	Type	Area	Architecture Driver Description	Importance	Penalty	Worst	Current	Desired	Best	Worst	Current	Desired	Best
						Response	Response	Response	Response	Benefit	Benefit	Benefit	Benefit
QAS1	Quality attribute	Testability	Implement integration test for customer notification functionality within 1 day of development effort	50	0	2	1	1	0.5	50	50	75	100
QAS2	Quality attribute	Testability	Implement integration test and they should be stable (avoid random failures) for 2+ iterations	25	0	0	1	2	4	0	50	75	100
QAS3	Quality attribute	Testability	Cover non-legacy code in Portal UI project with 90%+ by unit tests	30	0	0	52	90	100	0	30	90	100
QAS4	Quality attribute	Testability	Implement JavaScript unit test for UI code (customer notification screen) within 0.5-1 day	75	0	30	0.5	0.5	0.5	0	0	90	100
QAS5	Quality attribute	Modifiability	Minor functional feature in Portal UI should be added within 2.5 days of development + testing + bug fixes (in newly added code)	80	0	4	4	3	2	50	50	100	100
QAS6	Quality attribute	Modifiability	Add new screen of average complexity to Portal should take up to 1 week (dev+review+tasting+fixes)	80	0	4	2	1	1	30	30	90	100
QAS7	Quality attribute	Modifiability	Remove redundant check permission methods from Action within 2 days of development effort	10	0	3	3	2	1	50	50	75	100
QAS8	Quality attribute	Modifiability	Add ability to search by email in two different pages in Portal within 3-4 days	80	0	9	9	4	3	40	40	80	100
QAS9	Quality attribute	Modifiability	UI: Implement jTable or another way (with angular) without pagination with styling on all brandings within 2 days of development effort	80	0	4	4	2	2	50	50	100	100
QAS10	Quality attribute	Modifiability	UI: Implementing Notifications Counter for Branded Portals 1 day of developers effort	80	0	3	3	1	1	50	50	100	100
QAS11	Quality attribute	Modifiability	Implement template for email and check it's look in all supported clients 1,5 day of developer effort	25	0	5	5	1.5	1	40	40	90	100
QAS12	Quality attribute	Modifiability	Introduce mod_csrf project that will cover CSRF attacks within 1 week of configuration and deployment to production	80	50	8	8	1	1	0	0	100	100
QAS13	Quality attribute	Modifiability	Implement fix for LDAP defects which occurred because of newest jquery version 5h developers effort within 2 days	50	0	10	10	1	1	10	10	100	100
QAS14	Quality attribute	Reusability	Implement overlapping IP subnets validation within 2 days of development effort	50	0	4	4	2	2	50	50	100	100
QAS15	Quality attribute	Reusability	Ability to paste comma separated list of proxies to the proxy filed. In scoup of this add UI validation for proxy input field 1day of developers effort	60	0	2	2	1	1	50	50	100	100
QAS16	 Quality attribute	Testability	Set up environment exactly as it is in Release Ticket before start regression testing within 0.5 days	70	0	0.5	0.5	0.1	0.1	50	50	100	100

# Code Analysis

Size: Packages, Classes,  
LOC, Bytecode Instructions

Violations: Design,  
Implementation, Style

Technical Debt  
Measurement

Excessive Structural  
Complexity

- Fat packages and classes
- Cyclomatic complexity of methods
- Tangles (cyclic dependencies) of packages and classes

Dependency Graphs and  
Dependency Structure  
Matrices

Actual Code Layering  
Analysis

**structure101**

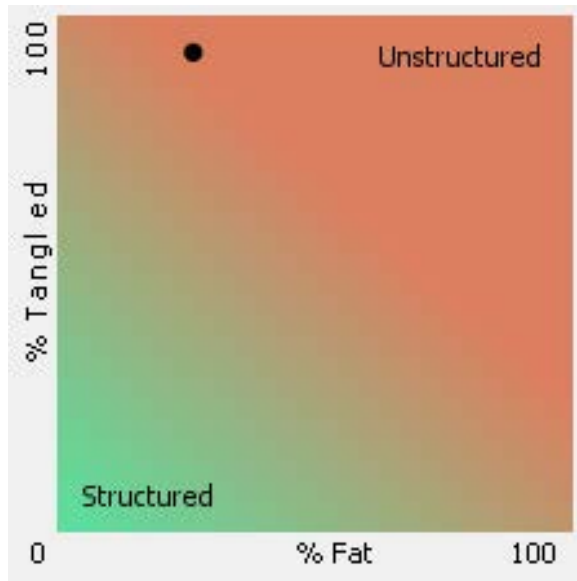
**sonarqube**





# Code Analysis Example

\* Structure 101



Item	Tangled	Fat	Size	XS
Package1	41%	2	584,810	240,549
Package2	13%	15	458,578	58,149
Package3	6%	89	335,635	20,734
Package4	12%	9	89,540	10,572
Package5	23%	26	19,513	4,566
Package6	20%	8	19,724	3,854
Package7	14%	3	26,071	3,687
Package8	43%	4	7,381	3,163
Package9	38%	10	7,024	2,681
Package10	11%	5	23,784	2,571

Average Excessive Complexity: 65%

Metric (and scope)	Threshold	#Offenders	Offenses (%)	XS contribution
Tangled (design)	0	29 of 124	23%	95%
Fat (design)	120	0 of 124	0%	0%
Fat (leaf package)	120	1 of 325	0%	0%
Fat (class)	120	21 of 2,832	1%	3%
Fat (method)	15	29 of 20,617	0%	1%
<b>Total</b>				<b>100%</b>

## Recommended Approach

Defined technical debt elimination process for the development team

Identified specific classes and packages for refactoring

Pointed legacy technologies to be wiped out from the project and recommended version upgrades

Recommended refactoring options to remove cyclic dependencies and improve layering of the code

Advised approaches to the test refactoring

And more...

# CBAM: Value for Cost Formula

$$VFC_{decision} = \frac{Value\%_{decision}}{Weight_{cost} \times Cost\%_{decision} + Weight_{risk} \times Risk\%_{decision}}$$

$$Value\%_{decision} = 100\% \times \frac{Weight_{benefit} \times \sum Weight_{driver} \times Benefit_{decision} + Weight_{penalty} \times \sum Weight_{driver} \times Penalty_{driver}}{Weight_{benefit} \times \sum Weight_{driver} \times Best\ Case\ Benefit_{driver} + Weight_{penalty} \times \sum Weight_{driver} \times Penalty_{driver}}$$

$$Benefit_{decision} = Benefit_{lower\ response} + (Response_{decision} - Response_{lower}) \times \frac{Benefit_{higher\ response} - Benefit_{lower\ response}}{Response_{higher} - Response_{lower}}$$

# CBAM: Steps

Step	
1	Define appropriate weights for Value for Cost formula
2	Build a list of architecture drivers classified by type and area, including unique driver id, description, and level of importance from 1 to 100
3	Select a smaller set of measurable key architecture drivers to use in the Cost Benefit analysis and comparison of the existing and recommended architectural decisions. Determine penalties for not being implemented (1-100), worst, current, desired, and best case responses as numerical values and map to benefit values (1-100)
4	Form the list of the current and candidate architectural decisions addressing the selected key drivers and evaluate the cost and risk values associated with them (1-100)
5	In the Decision Tradeoff Matrix: Define columns as actual or expected architectural decision responses and corresponding benefit values Define rows as key architecture drivers
6	Fill response cells with the actual or expected response values in the Decision Tradeoff Matrix
7	Calculate the actual or expected benefit values in the corresponding cells with INTERPOLATE() in the Decision Tradeoff Matrix
8	Analyze and compare the resulting total Values for Cost for each decision and review the decision benefit by architecture driver radar chart
9	Identify the potentially most appropriate architectural decision based on the highest Value for Cost number and validate it with further qualitative analysis and intuition

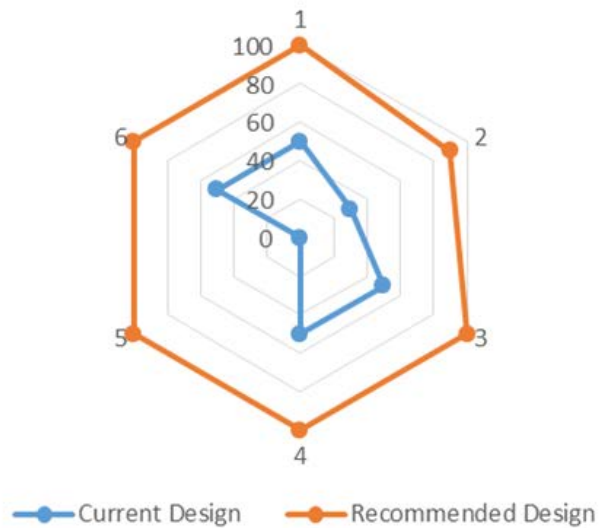
# CBAM: Design Trade-off Matrix

Key Driver	Imp	Pn	BBen	Wgt	D1 Response	D1 Benefit	D2 Response	D2 Benefit	Driver Description
<b>QAS5</b>	80	0	100	0.17	4	50	3	100	Minor functional feature in Portal UI should be added within 2.5 days of development + testing + bug fixes (in newly added code)
<b>QAS6</b>	80	0	100	0.17	2	30	1	90	Add new screen of average complexity to Portal should take up to 1 week (dev+review+tasting+fixes)
<b>QAS9</b>	80	0	100	0.17	4	50	2	100	UI: Implement jTable or another way (with angular) without pagination with styling on all brandings within 2 days of development effort
<b>QAS10</b>	80	0	100	0.17	3	50	1	100	UI: Implementing Notifications Counter for Branded Portals 1 day of developers effort
<b>QAS12</b>	80	50	100	0.17	8	0	1	100	Introduce mod_csrf project that will cover CSRF attacks within 1 week of configuration and deployment to production
<b>QAS16</b>	70	0	100	0.14	0.5	50	0.1	100	Set up environment exactly as it is in Release Ticket before start regression testing within 0.5 days
<b>Total</b>	<b>470</b>	<b>8.5</b>	<b>100</b>			<b>42.9</b>		<b>98.4</b>	

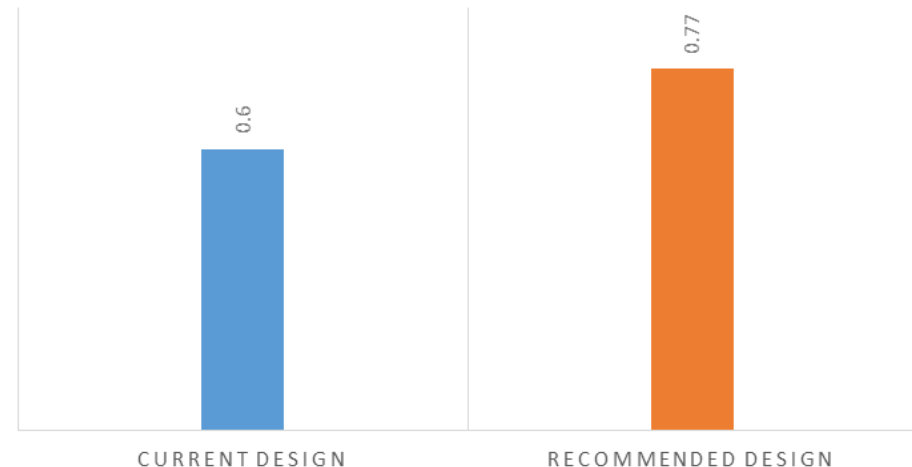
# CBAM: Value for Cost

Decision	All Costs	Value	VFC	Description
D1	71.5	42.9	0.6	Current Design
D2	128.4	98.4	0.77	Recommended Design

Design Benefit by Key Driver



VALUE FOR COST BY DECISION



# CBAM: Q&A

---

**Q:** How to assign importance values to the specific drivers?

**A:** Stakeholders can vote with some number of points for each of the drivers and the votes get added up. Or the priorities are translated to importance values. Or based on some objective factors.

---

**Q:** How to select the key drivers for Cost Benefit Analysis

**A:** Based on their importance values or on the consensus

---

**Q:** How to assign penalties?

**A:** Based on the potential impact on business goals of the driver not being implemented

---

**Q:** How to assign costs to the decisions?

**A:** Estimate costs based on the effort costs, licensing costs, TCO and similar, then recalculate to % taking highest acceptable cost as 100%

---

**Q:** How to find the Risk values for the decisions

**A:** They can be based on the formula  $\text{Risk Exposure} = \text{Risk Probability} * \text{Cost if it occurs}$  and then mapped to %

---

**Q:** Which architectural decisions to include into Analysis

**A:** Those which address maximum number of the selected key architecture drivers and make sense qualitatively (for example, if a specific technology cannot be accepted from the business perspective a decision based on it should not be included even if it formally covers the key drivers)

---

# Thank You!

## SoftServe Headquarters

One Congress Plaza,  
111 Congress Avenue, Suite 2700 Austin, TX  
78701  
Tel: 512.516.8880

## Contacts

Andriy Shapochka,  
Principal Architect @ SoftServe Inc.  
[ashapoch@softserveinc.com](mailto:ashapoch@softserveinc.com)







- Leading global Product and Application Development partner founded in 1993
- 3,500+ employees across North America, Eastern and Western Europe
- Thousands of successful outsourcing projects!

## Clients include:



SaaS/Cloud Solutions . Mobility Solutions . UX/UI  
BI/Analytics/Big Data . Software Architecture . Security