Case Example 5:

Software Methodologies

The example here illustrates the differences in schedules, costs, and quality for three of these methodologies: Agile, Waterfall, and the combination of Team Software Process (TSP) and Personal Software Process (PSP) both of which were developed by Watts Humphrey of the Software Engineering Institute (SEI). TSP/PSP are normally used together.

For a detailed analysis of methodologies, see Quantitative Comparison of 60 Software Development Methodologies, Capers Jones, published in July of 2017 by CRC Press.

Example 5: How Software Risk Master (SRM) Evaluates Software Methodologies						
	Java Language for all 3 Cases					
	1000 function points for all 3 Cases					
	$7.500 per month for all 3 Cases					
	SRM can evaluate 60 methodologies including hybrid in 2017					
	New methodologies added when data becomes available					
	SRM also evalutes 5 levels of capability maturity model integrated (CMMI)					
	2017 is the 30th anniversary of IFPUG function point metrics					
		Waterfall	**Agile**	**TSP/PSP**	**TSP = team software process**	
		CMMI 1	**CMMI 0**	**CMMI 5**	**PSP = personal software process**	
Project Risks						
Cancellation		**19.98%**	**14.19%**	**11.59%**	**Risks vary among methodologies**	
Negative ROI		**25.31%**	**18.00%**	**14.68%**		
Cost overrun		**21.98%**	**15.90%**	**12.75%**		
Schedule slip		**26.64%**	**19.30%**	**15.45%**		
Unhappy customers		**15.98%**	**12.30%**	**9.27%**		
Litigation		**8.79%**	**6.26%**	**5.10%**		
Technical debt/high COQ		**22.46%**	**16.00%**	**13.03%**		
Cyber attacks		**13.69%**	**9.75%**	**7.94%**		
Financial Risk		**29.48%**	**21.00%**	**17.10%**		
High waranty repairs		**20.71%**	**14.75%**	**12.01%**		

Poor maintainability		**15.44%**	**11.00%**	**8.96%**		
RISK AVERAGE		**20.04%**	**14.40%**	**11.63%**	**Quality strong methodologies have lower risks**	
Total Defects in Application		**6,000**	**4,800**	**2,700**	**Agile, waterfall are not "quality strong" methodologies**	
					TSP/PSP are "quality strong" methodologies	
Pre-Test Defect Removal %		**45.00%**	**69.75%**	**81.00%**		
Defects Removed		2,700	3,348	2,187		
Defects Remaining		**3,300**	**1,452**	**513**		
Joint Application Design (JAD)		No	Yes	Maybe		
Scrum sessions		No	Yes	Maybe		
Informal reviews		Yes	Yes	No		
Quality function deployment (QFD)		No	No	Yes		
Six Sigma for software		No	No	Maybe		
Requirements inspection		No	No	Yes	**DRE goes up with inspections**	
Design inspection		No	No	Yes		
Code inspection		No	No	Yes		
Test material inspection		No	Maybe	Yes		
Static analysis		No	Maybe	Yes	**DRE goes up with static analysis**	

Test Defect Removal %		**70.00%**	**81.90%**	**87.50%**		
Defects Removed		2,310	1,189	449		
Defects Remaining		**990**	**263**	**64**		
Unit test		Yes	Yes	Yes		
Function test		Yes	Yes	Yes		
Regression test		Yes	Yes	Yes		
Performance test		Yes	Yes	Yes		
Component test		No	No	Yes		
System test		Yes	Yes	Yes		
Acceptance/Beta test		Yes	Yes	Yes		
Bad fix injection %		**9%**	**5%**	**4%**	**Bad-fix injection is low with quality-strong methodologies**	
Bad fixes (new bugs in repairs)		**89**	**13**	**3**		
Defects detected but not repaired						
prior to delivery to customers		**289**	**107**	**22**	**Unrepaired defects are low with quality-strong methodologies**	
Cumulative Defect Removal %		**78.69%**	**92.30%**	**96.79%**	**All projects should top 96% defect removal efficiency (DRE)**	
					DRE developed by IBM circa 1973	
Total Defects Removed		**4,721**	**4,430**	**2,613**		

Total Defects Delivered		**1,079**	**276**	**67**		
High-Severity Defects Delivered		**270**	**55**	**11**		
Security Flaws Delivered		**36**	**7**	**1**		
Average monthly cost		$7,500	$7,500	$7,500		
OVERALL PROJECT						
Development Schedule (months)		15.85	11.82	12.02		
Staff (technical + management)		10	7	7		
Development Effort (staff months)		158	84	86		
Development Costs		$1,188,670	$633,043	$644,070		
DEVELOPMENT ACTIVITES						
Requirements Effort (staff months)		15.85	7.17	8.59		
Design effort (staff months)		31.70	13.50	11.16		
Coding effort (staff months)		31.70	21.95	20.61		
Testing effort (staff months)		45.96	25.32	27.48		
Documentation effort (staff month)		12.68	6.75	6.87		
Management effort (staff months)		19.81	9.28	11.16		
TOTAL EFFORT (Staff months)		**157.70**	**83.98**	**85.88**		

Function points per mont		**6.34**	**11.91**	**11.64**		
Total Cost of Development		**$1,188,670**	**$633,043**	**$644,070**		
Total Cost of Maintenance		**$2,315,681**	**$600,688**	**$144,284**	**Mainteance is cheaper with quality-strong methodologies**	
Total Cost of Enhancement		**$439,808**	**$234,226**	**$238,306**		
TOTAL COST OF OWNERSHIP (TCO)		**$3,944,159**	**$1,467,957**	**$1,026,660**	**TCO is cheaper with quality-strong methodologies**	
			END OF EXAMPLE			
Table 1: Methodologies Supported by Software Risk Master (SRM)						
	Best quality (Quality-strong)					
1	Robotic development with 99% standard parts					
2	Reuse-oriented (85% reusable materials)					
3	Animated, 3D, full color design development					
4	Pattern-based development					
5	Virtual reality global development					
6	T-VEC development					
7	IntegraNova development					
8	Kaizen development					
9	Container development (65% reuse)					

10	Model-driven development			
	Good Quality (Quality strong)			
11	Clean room development			
12	Team software process (TSP) + PSP			
13	Feature driven (FDD)			
14	Personal software process (PSP)			
15	Specifications by Example			
16	CMMI development			
17	Micro service development			
18	Evolutionary Development (EVO)			
19	Rational Unfied Process (RUP) from IBM			
20	Prototypes - disposable			
21	Open-source development			
22	Object Oriented (OO) development			
23	Global 24 hour development			
24	Disciplined agile delivery (DAD)			
25	Product Line engineering			
26	Service-Oriented modeling			
27	Mashup development			
	Average quality			
28	Prototypes - evolutionary			
28	Information engineering (IE)			
29	Crystal development			
30	Extreme programming (XP)			
31	Pair programming development			
32	Lean development			
33	Microsoft solutions			
34	Spiral development			
35	GIT development			
36	Legacy renovation			
37	Legacy replacement development			
38	Iterative development			
39	Test-driven development (TDD)			
40	CASE development			

41	Hybrid (agile + waterfall)				
42	Agile + scrum				
43	Legacy repair development				
44	Structured development				
45	Continuous development				
46	Dynamic system development method (DSDM)				
	Poor quality				
47	DevOps development				
48	Legacy data mining				
49	Prince 2 development				
50	Merise development				
51	Agile/Scrum				
52	Rapid application development (RAD)				
53	Reverse engineering				
54	V-Model development				
55	Reengineering				
56	Cowboy development				
57	ERP modification development				
58	Waterfall development				
59	COTS Modifications				
60	Anti patterns				