



# A Value-Based Software Product Model

Dr. Ray Madachy  
USC Center for Software Engineering  
Cost Xpert Group

18th International Forum on COCOMO and Software Cost Modeling  
October 22, 2003



# Agenda

- Model Background
- Product Model Overview
- Analysis Results
- Summary



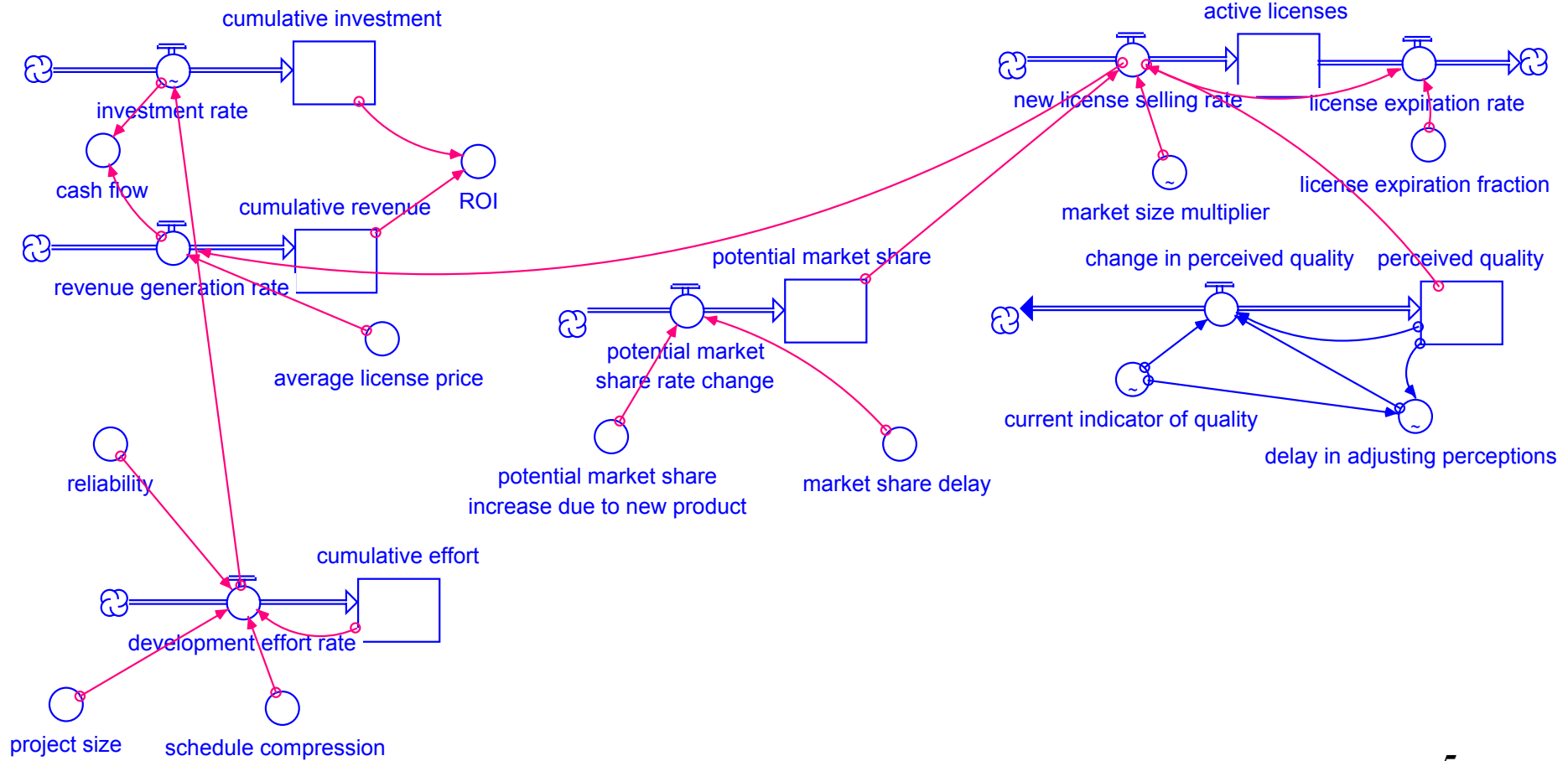
# Model Background

- Purpose: Support software business decision-making by experimenting with product strategies and development practices.
- Overview: System dynamics model relates the interactions between product development investments, software quality practices, market share, license retention, pricing and revenue generation for a commercial software enterprise.

# Model Assumptions

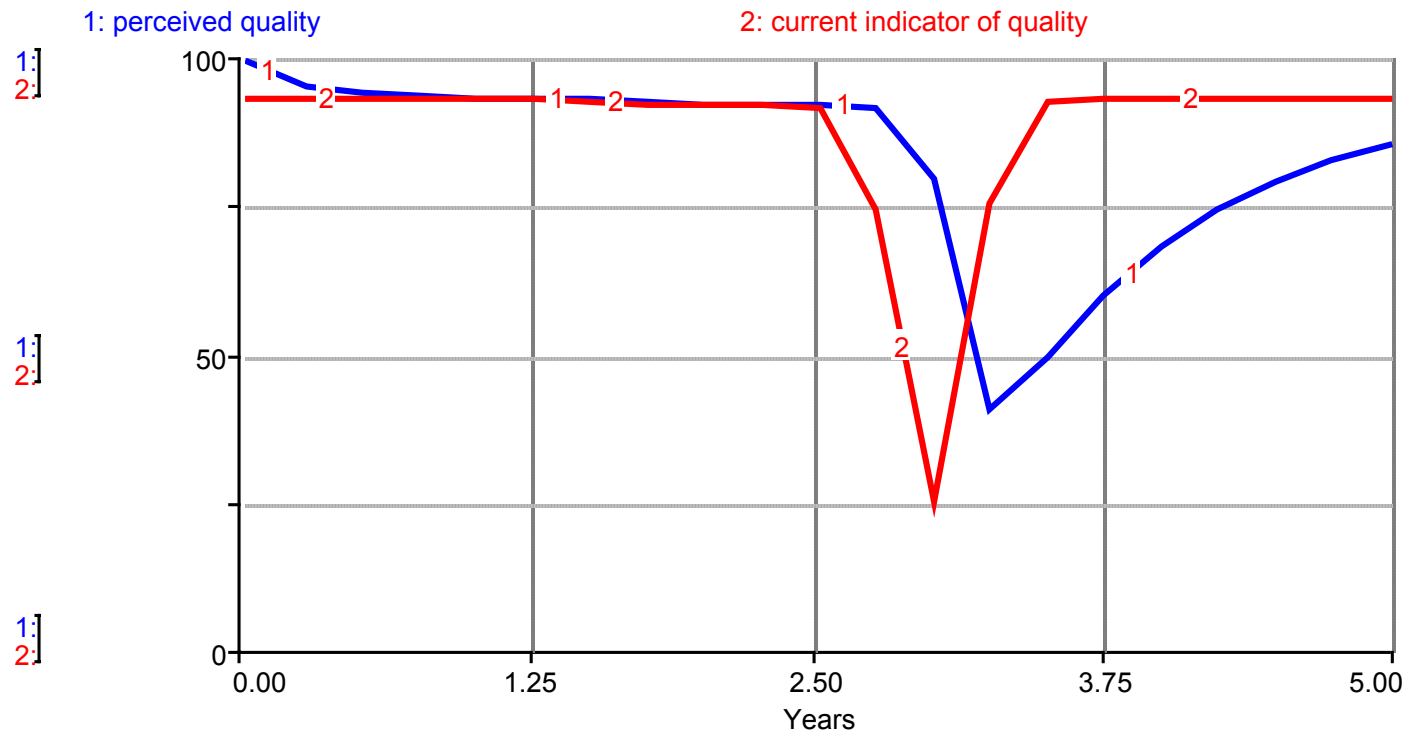
- COCOMO Reliability cost driver is a proxy for all quality practices
- Resulting quality will modulate the actual sales relative to the highest potential
- Parameterizations:
  - Initial total market size = \$64M annual revenue
    - vendor has 15% of market
    - overall market doubles in 5 years
  - A new 80 KSLOC product release can potentially increase market share by 15%-30% (varied in model runs)

# Model Diagram



# Perception of Quality

- Quality reputation quickly lost and takes much longer to regain
- Modeled as asymmetrical information smoothing via negative feedback loop
- The perception modulates sales and resultant market share.

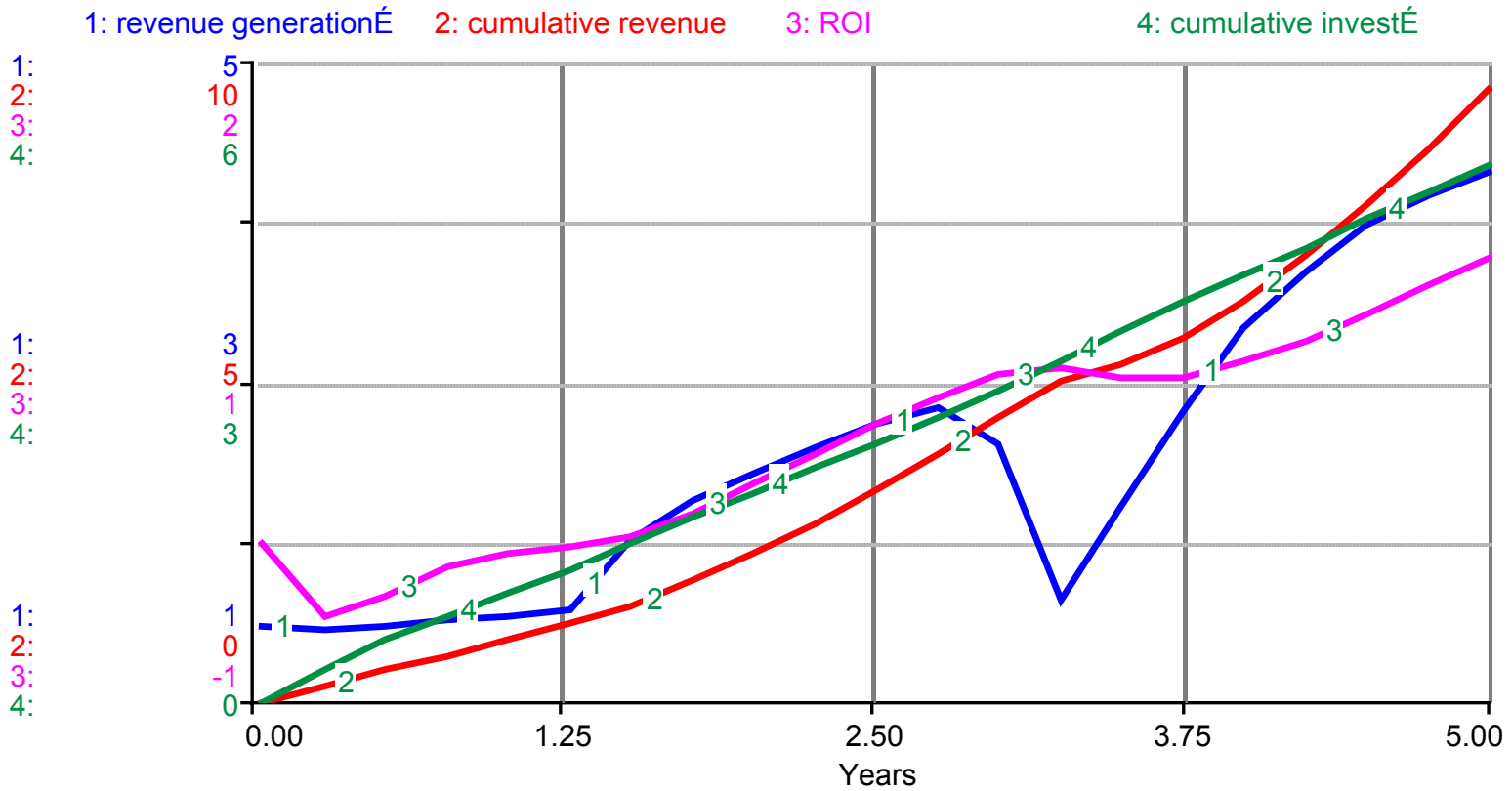




# Sales Impact of Quality

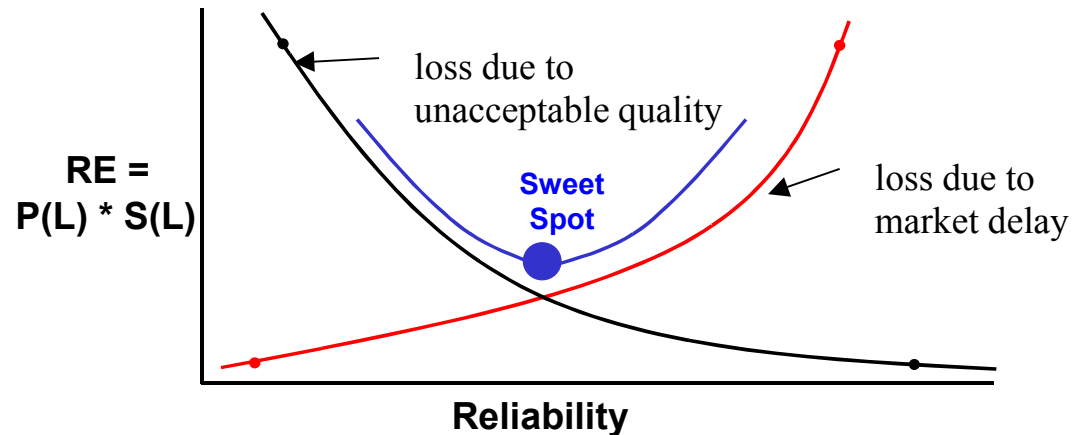
<b>Reliability Rating</b>	<b>Percent of Potential Sales Captured Relative to Highest Reliability</b>
Low	50%
Nominal	80%
High	95%
Very High	100%

# Sample Run Output



# Determining How Much Reliability is Enough

- Use risk exposure framework to find process optimum
- Vary Reliability across runs
- Assess risk consequences of opposing trends: market delays and bad quality losses
- Sum the costs
- Calculate resulting net revenue

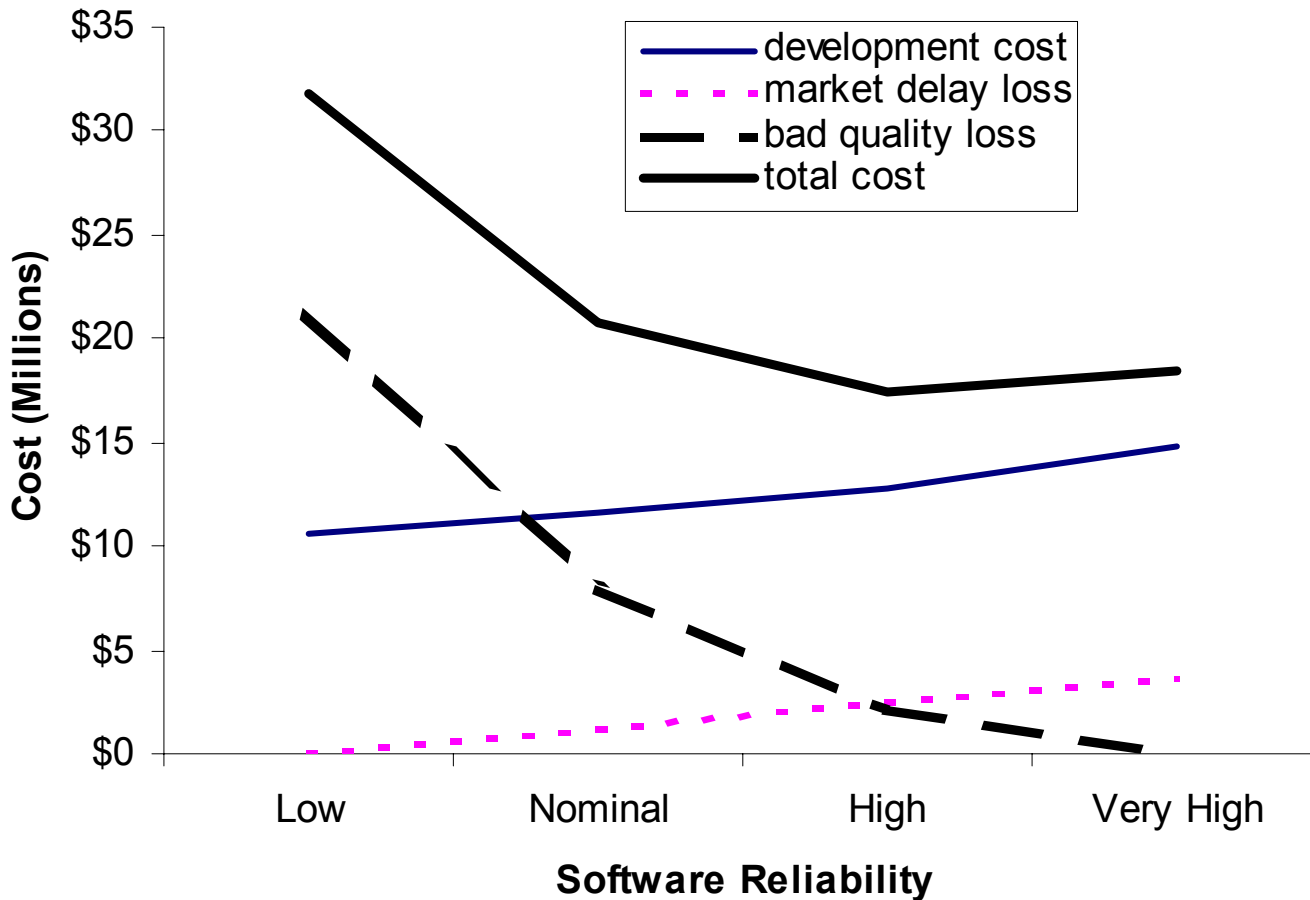


# Sample Experiment Results

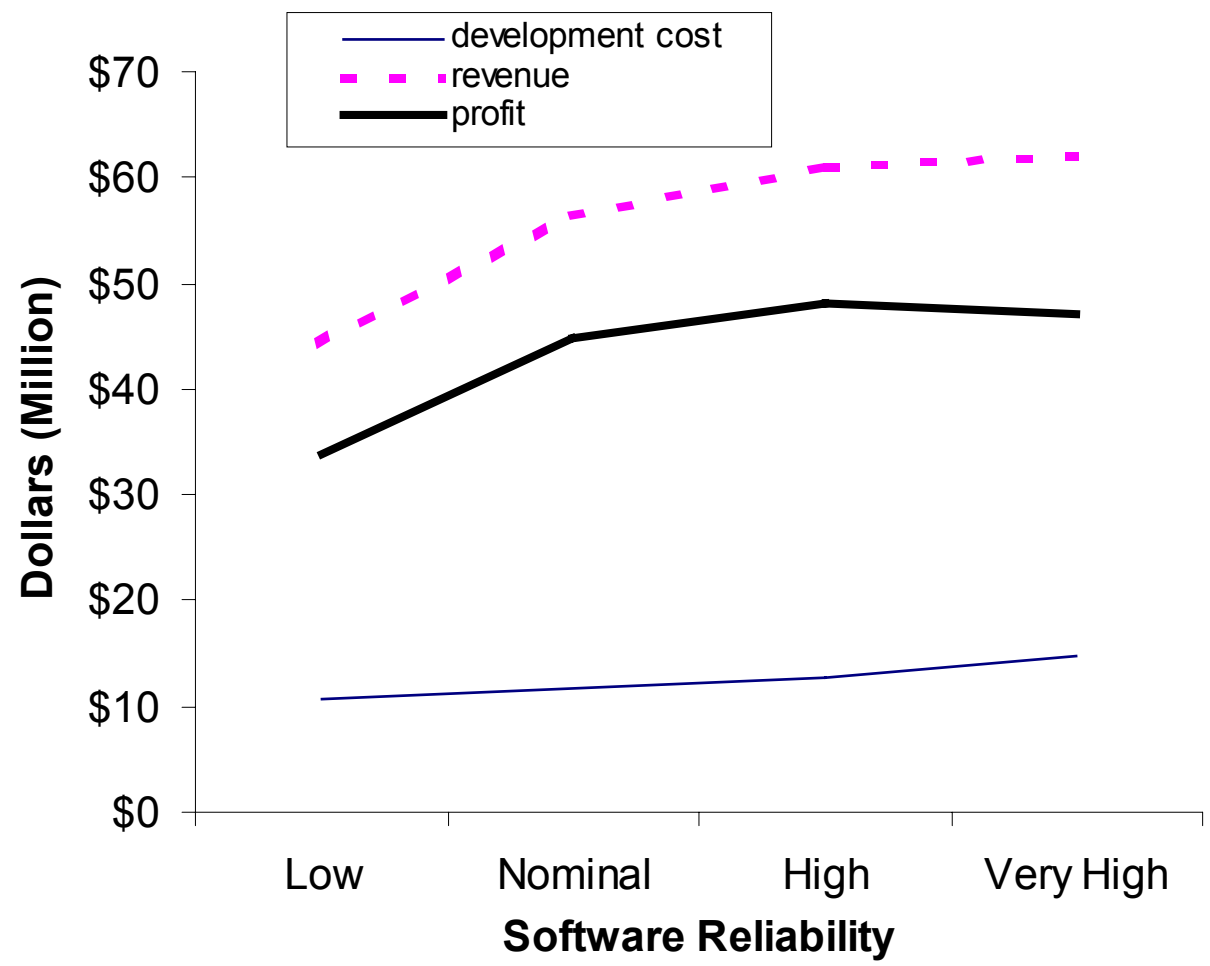
- 80 KSLOC, schedule 75% of nominal, 3 year revenue timeframe

	<b>Reliability Rating</b>				
	<b>Low</b>	<b>Nominal</b>	<b>High</b>	<b>Very High</b>	
Effort (Person-months)	479	521	573	656	
Schedule (Months)	17.5	17.9	18.5	19.3	
Cost (\$M)	10.6	11.6	12.8	14.8	
Revenue (\$M)	\$44.4	\$56.4	\$61.1	\$62.0	<b>(1)</b>
Maximum Potential Revenue with Same Timing (if highest quality at same schedule)	\$65.6	\$64.4	\$63.2	\$62.0	<b>(2)</b>
Market Delay Cost (\$M)	\$0.0	\$1.2	\$2.4	\$3.6	<b>65.6-(2)</b>
Bad Quality Loss (\$M)	\$21.2	\$8.0	\$2.1	\$0.0	<b>(2) – (1)</b>
Total Cost (\$M)	\$31.8	\$20.8	\$17.3	\$18.4	

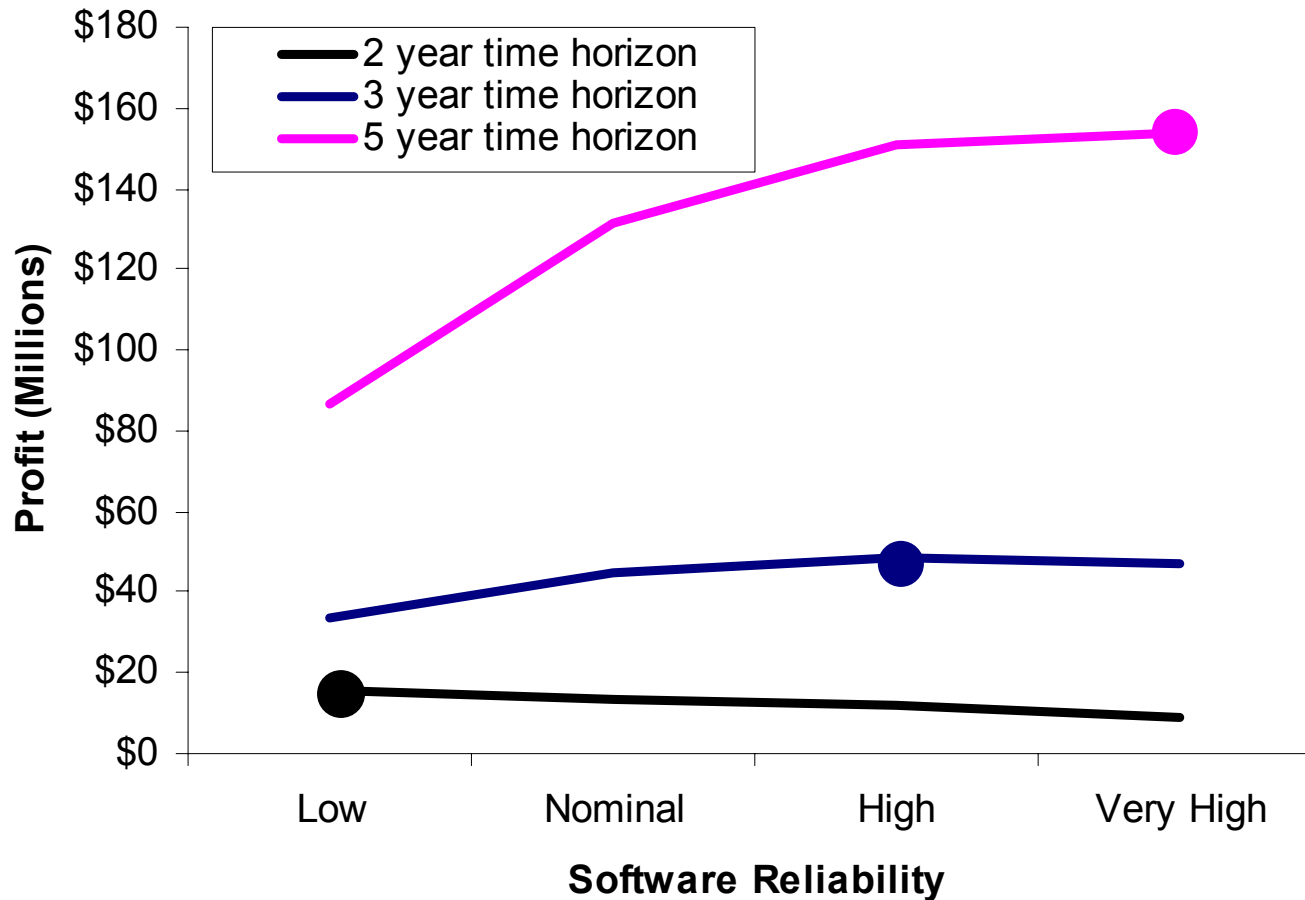
# Resultant Reliability Sweet Spot



# Profit Maximization View



# Sweet Spot Depends on Time Horizon



# Other Considerations

- Pricing scheme impacts
- Varying market assumptions
- Impact of new releases that increase (or decrease) quality
- Feedback from growing user base to incorporate new features

# Summary

- Decision-making can be improved with information gained from simulation experiments
- Risk exposure is a convenient framework for software decision analysis.
- Commercial process sweet spots with respect to reliability are a balance between market delay losses and quality losses
- Business policies operate within a multi-attribute decision space
- Quality impacts the bottom line