Critical Cost Issues for Ground Systems

Stephen A. Book

Chief Technical Officer
MCR Federal, LLC
390 No. Sepulveda Blvd.
El Segundo, CA 90245
(310) 640-0005

Prepared for
Breakout Session 10E: Ground System Costs

Ground Systems Architecture Workshop

The Aerospace Corporation
El Segundo, CA
31 March 2004

Contents

- Software: Facts and Fictions
- COTS: It’s Not Spelled “F-R-E-E”
- GFE: It’s Not Free Either
- Impact of Correlation on Probable Cost
- Summary

Contents

- Software: Facts and Fictions
- COTS: It's Not Spelled "F-R-E-E"
- GFE: It's Not Free Either
- Impact of Correlation on Probable Cost
- Summary

Software Costing Concerns

- **Standard Cost-Estimating Paradigm for Hardware is not Applicable to Software**
  - Software Requirements Cannot Be Fully Captured in Any Finite List: True List of Requirements Is Virtually Infinite
  - Software Development Is Uniquely Personnel-intensive: Even Within Same Company or Workgroup, Productivity May Vary As Much As 100 to 1 Among Programmers
  - Programming is the Easy Part – Figuring Out a Software Solution to the Technical Problem is What's Difficult
- **There Are No "Technical" Characteristics Such As Weight, Power, etc., that Play the Role of Cost Driver**
  - Primary "Measurable" Cost Driver is Number of Lines of Code, which is Notoriously Difficult to Estimate
  - Naval Center for Cost Analysis Found Average Lines-of-Code Growth of 63% for Software Projects of Various Types (http://www.ncca.navy.mil/software/handbook/software.htm)

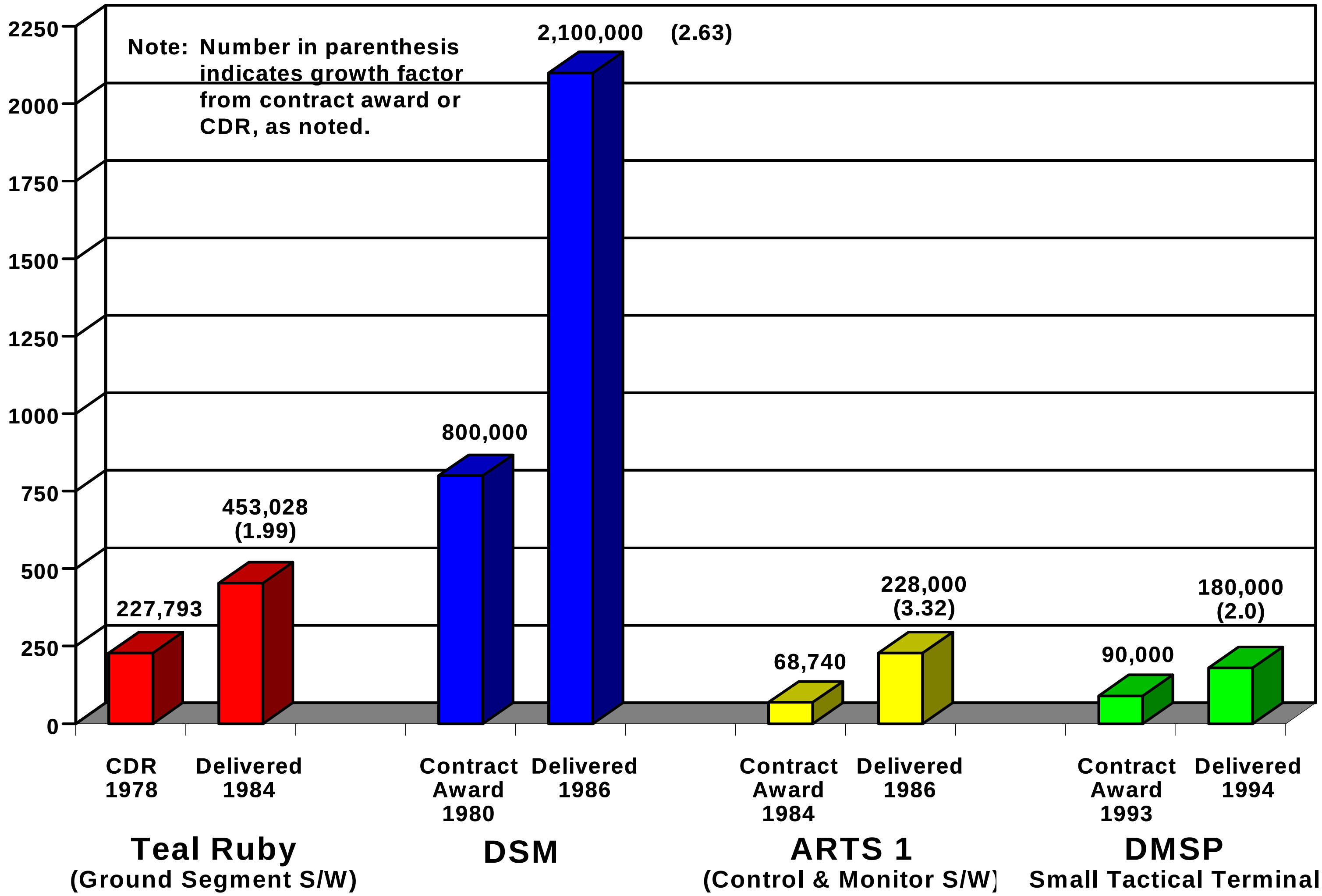
Software Code-Growth Experience

DATA SOURCE	MISSION PURPOSE	ESTIMATED SLOC*	ACTUAL SLOC*	GROWTH FACTOR
AF Space Projects:	C^2	618,000	709,000	1.15
	C^2	23,599	25,814	1.09
	C^2	14,000	70,143	5.01
	Testing	41,800	46,303	1.11
	Software Tools	45,000	45,000	1.00
	C^2	39,294	119,400	3.04
	C^2	22,000	30,000	1.36
	Signal Processing	15,500	26,513	1.71
	C^2	100,000	122,000	1.22
	Mission Planning	532,000	877,129	1.65
Navy Projects:	C^2	206,650	394,309	1.91
	C^2	74,000	82,930	1.12
	C^2	213,800	261,800	1.22
	C^2	153,000	185,000	1.21
	C^2	83,900	108,850	1.30
	C^2	1,246,272	1,272,200	1.02

Reference: Naval Center for Cost Analysis, "Software Development Estimating Handbook, Phase One," 1998.

*** Source Lines of Code**

Lines-of-Code Estimating Risk for Satellite Ground Stations



THE AEROSPACE CORPORATION

Contents

- **Software: Facts and Fictions**
- **COTS: It's Not Spelled "F-R-E-E"**
- **GFE: It's Not Free Either**
- **Impact of Correlation on Probable Cost**
- **Summary**

COTS Software

- COTS is an Attractive Addition to a Ground-System Cost Estimate
  - It Looks Inexpensive
  - It's Politically Correct
  - It's a "New Way of Doing Business"
- But, in Order to <u>Really</u> Incorporate COTS Software into the System …
  - The COTS Software Has to be Thoroughly Tested for Situations in which It May Act Erratically or "Crash" the System
  - Integration ("Glue") Code Has to be Written and Tested
  - Non-COTS Portion of System Often Has to be Designed Suboptimally to Accommodate COTS

Title Chart of U.K. MoD COTS Software Study Briefing

Space Projects

SPS SPECIALIST PROCUREMENT SERVICES

SPS/CF smart solutions for SMART problems

Jim Armstrong **Bob Anderton**

David Frankis **David Saddleton**

John Taylor **Dave Thombs**

Can You Afford COTS Software?

(The MOD SOUP Study)

Potential COTS Software Users Must Validate These Characteristics*

To adequately assess the safety of SOUP, it must be validated for each of the following types of potential cause of software related hazard (called ***evidential requirements***):

- **i.** Normal functionality – coverage of requirements for functionality;
- **ii.** Exceptional functionality – error signalling and handling;
- **iii.** Architectural build configuration – version control of installations;
- **iv.** Set-up configuration – procedures for initialisation and start-up;
- **v.** Algorithmic sufficiency – type-safety and accuracy;
- **vi.** Timing;
- **vii.** Memory Usage – predictability of storage use;
- **viii.** Availability;
- **ix.** Functional independence – isolation of critical from non-critical functions;
- **x.** Soundness – extra safety requirements due to chosen implementation strategy;
- **xi.** Interface security – protection from misuse;
- **xii.** Robustness – continued service under stressful conditions;
- **xiii.** Vicelessness – safe service under stressful conditions.

***Reference: D. Frankis and J. Armstrong, "Software Reuse in Safety-Critical Applications, Summary Final Report," U.K. Ministry of Defence, Defence Procurement Agency, Crown Copyright 2001, pages 5-6.**

U.K. MoD-Recognized Practices in Testing Software Characteristics*

i. **Common practice:** the method is generally accepted and in common use for safety assessment.

ii. **Uncommon practice:** the method is mature and well tested but not often used.

iii. **New practice:** the method is beginning to be established, but the likely long term attitude of assessors is as yet unclear.

iv. **Speculative practice:** the method is the subject of academic research only and assessors are very unlikely to require it in the short term; in the longer term it may become part of best practice.

***Reference Cited, pages 6-7.**

U.K. MoD-Recommended Testing*

EVIDENTIAL REQUIREMENT	Black Box Assessment	Code Assessment	Open Box Assessment
a. Normal Functionality	**Common Practices:** Scenario-based Testing Examine User Manuals **New Practices:** Field Trials Lab Simulation	N/A	N/A
b. Exceptional Functionality	**Common Practices:** Stress Testing Scenario-based Testing Domain Testing Error Guessing Examine User Manuals **New Practices:** Field Trials Lab Simulation **Speculative Practices:** Random Testing	**Common Practices:** Code Walkthrough **New Practices:** Language Subset Analysis	**Common Practices:** Domain Testing Stress Testing **New Practices:** Fault Injection **Uncommon Practices:** Assertion Testing
c. Build Configuration	N/A	**Common Practices:** Code Walkthrough	N.B. *A bespoke build may be feasible.*
d. Set-up Configuration	**Common Practices:** User Manuals **New Practices:** Field Trials **Speculative Practices:** Accelerated Life Testing	**Common Practices:** Code Walkthrough	**Common Practices:** Coverage Matrix Testing

*Reference Cited, pages 7-8.

EVIDENTIAL REQUIREMENT	Black Box Assessment	Code Assessment	Open Box Assessment
e. Algorithmic Sufficiency	**Common Practices:** Stress Testing Domain Testing Error Guessing **New Practices:** Statistical Testing **Speculative Practices:** Random Testing Accelerated Life Testing	**Common Practices:** Code Walkthrough **New Practices:** Language Subset Analysis **Uncommon Practices:** Control Flow Analysis Data Flow Analysis Semantic Analysis Translation **Speculative Practices:** S/W Fault Tree Analysis Partial Correctness Proof Termination Proof Refinement Proof Retrospective Specification	**Common Practices:** Stress Testing Domain Testing Coverage Testing **Uncommon Practices:** Assertion Testing Symbolic Execution **Speculative Practices:** Exhaustive Testing
f. Timing	**Common Practices:** Stress Testing Scenario-based Testing **New Practices:** Field Trials Lab. Simulation	**Common Practices:** Code Walkthrough **Uncommon Practices:** SCA Control Flow Analysis	**Common Practices:** Stress Testing Coverage Testing **Speculative Practices:** Exhaustive Testing
g. Memory Usage	**Common Practices:** Stress Testing Error Guessing	**Common Practices:** Code Walkthrough **New Practices:** Language Subset Analysis	**Common Practices:** Stress Testing **Speculative Practices:** Exhaustive Testing

SPECIALIST PROCUREMENT SERVICES

EVIDENTIAL REQUIREMENT	Black Box Assessment	Code Assessment	Open Box Assessment
h. Availability	**Common Practices:** Safety Target Setting **New Practices:** Statistical Testing **Speculative Practices:** Accelerated Life Testing	N/A	N/A
i. Functional Independence	N/A	**Common Practices:** Code Walkthrough **Uncommon Practices:** Data Flow Analysis Source Code Architecture	**Common Practices:** Coverage Testing **Uncommon Practices:** Assertion Testing **Speculative Practices:** Exhaustive Testing
j. Soundness	N/A	**Common Practice:** Code Walkthrough Complexity Measurement **New Practices:** Language Subset Analysis **Uncommon Practices:** Translation Control Flow Analysis Data Flow Analysis Semantic Analysis **Speculative Practices:** S/W Fault Tree Analysis Refinement Proof Retrospective Specification	**Common Practice:** Domain Testing Coverage Testing **Uncommon Practices:** Assertion Testing **Speculative Practices:** Exhaustive Testing

EVIDENTIAL REQUIREMENT	Black Box Assessment	Code Assessment	Open Box Assessment
k. Interface Security	**Common Practices:** Stress Testing Domain Testing Error Guessing **New Practices:** Lab. Simulation **Speculative Practices:** Random Testing	**Common Practices:** Code Walkthrough	**Uncommon Practices:** Assertion Testing
l. Robustness	**Common Practices:** Stress Testing Domain Testing Error Guessing **Speculative Practices:** Random Testing	**Common Practices:** Code Walkthrough Complexity Measurement **New Practices:** Language Subset Analysis **Uncommon Practices:** Control Flow Analysis Data Flow Analysis **Speculative Practices:** S/W Fault Tree Analysis Termination Proof	**Common Practices:** Coverage Testing **New Practices:** Fault Injection **Uncommon Practices:** Assertion Testing **Speculative Practices:** Exhaustive Testing
m. Vicelessness	**Common Practices:** Scenario-based Testing Error Guessing **New Practices:** Statistical Testing Lab. Simulation	**Common Practices:** Code Walkthrough **New Practices:** Lang. Subset. Analysis	**Common Practices:** Stress Testing **Uncommon Practices:** Assertion Testing Fault Injection **Speculative Practices:** Exhaustive Testing

Cost Estimates of COTS Software Testing Activities* (Chart 1 of 2)

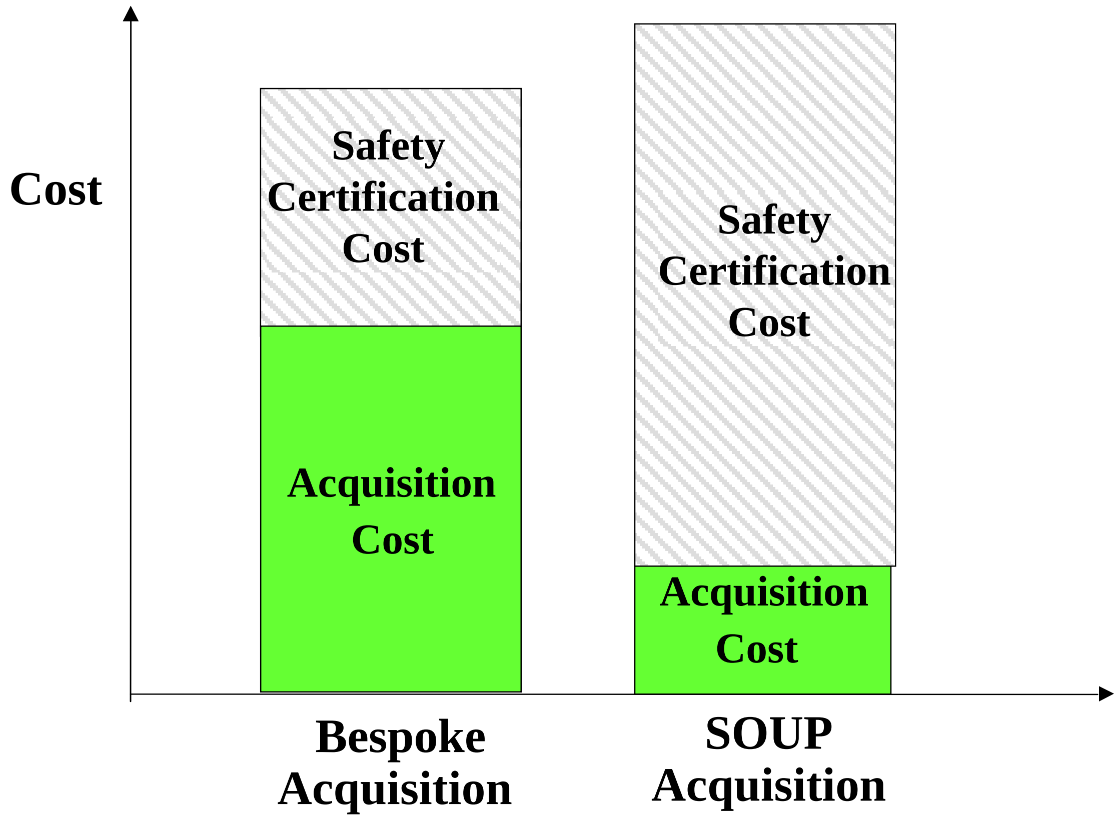
Method	*Elementary component*	*Size (man-days)*	*Limit on no of components*	*Number of components*	*Cost*	*Quality of estimate*
Scenario based testing	One scenario	10	Number of operational procedures	100 – 1000	Medium	C
Field trials	One trial	1000	Availability of facilities	10 – 100	Medium	C
Lab simulation	One run	10		1000	Medium	C
Examine User Manuals	–	100 (total)	-	-	Low	C
Random Testing	One run	.01	Input space size	$10^6 – 10^8$	Medium to high	C
Error guessing	One run	10		100	Medium	C
Accelerated life testing	One run	10	Number of setup states	100	Medium	C

*Reference Cited, pages 19-20.

Cost Estimates of COTS Software Testing Activities* (Chart 2 of 2)

Method	*Elementary component*	*Size (man-days)*	*Limit on no of components*	*Number of components*	*Cost*	*Quality of estimate*
Statistical testing	Define runs (once); One run	10; 0.1	Weighted size of input space	$10^6 – 10^8$	High	C
Safety target setting	Once	10 – 100	-	1	Low	B
Domain testing	One domain	1	No of equivalence partitions in critical functions	$10^3 – 10^6$	Medium	C
Stress testing	One run	10	Number of hazards/ways of stressing the system	10-100	Low	C

UK MoD's Notional Comparison of Costs of Bespoke* Software vs. SOUP**



* i.e., Fully Understood
**COTS

Contents

- Software: Facts and Fictions
- Reality Checks: Are They Useful?
- COTS: It's Not Spelled "F-R-E-E"
- **GFE: It's Not Free Either**
- Inter-element Correlation: Its Impact on Probable Cost
- Comparing Competing Proposals on the Basis of Risk
- Summary

GFE: A Great Way to Reduce Ground-System Cost Estimates

- … but Not Necessarily Ground-System Costs
- GFE = Government-Furnished Equipment
- GFE is a Popular “Code Word” that Contractors (and Government Project Managers) Use to Lower the Proposed Cost of a Program
  - It is Advertised to Do the Job
  - It is Low-Cost or Sometimes Even No-Cost
  - GFE is Usually Free to the Proposer, so It Adds Zero to his Bid (and to the required budget)
- It Even Seems to Make Sense Sometimes
  – Ground-System Hardware (e.g., Computers, Antennas, Communications Capability) are Often Available as Unused Spares from Earlier or Partially Completed Programs
  – Ground-System Software (e.g., for Testing, Data Management, Communications) Has Often Been Written to do the Same or a Similar Task on Another Government Program

GFE: A Government Obligation

- GFE is a Trap Set for the Government
  - It's Free to Proposer, so It Doesn't Appear in his Bid
  - That Means the Government Assumes the Obligation to Deliver that Portion of System
  - Most Often, However, GFE Does Not Do Job Anticipated
- If Government Accepts the Proposal and then GFE Fails to Perform, then ...
  - ECPs (Engineering Change Proposals) are Written to Task the Contractor to Develop Substitute Hardware and/or Software
  - Government Incurs Additional Costs Beyond the Bid Amount (even if there is no overrun on what was bid!)
  - Situation is Typically Written Off as an Increase in Government Requirements (but it really isn't), so the Additional Cost is Deemed Justified
  - What Really Happened, Though, Was that the Contract Shifted a Portion of the Program's Risk to the Government

Contents

- Software: Facts and Fictions
- COTS: It's Not Spelled "F-R-E-E"
- GFE: It's Not Free Either
- Impact of Correlation on Probable Cost
- Summary

Ground-System Costs are Correlated among Themselves

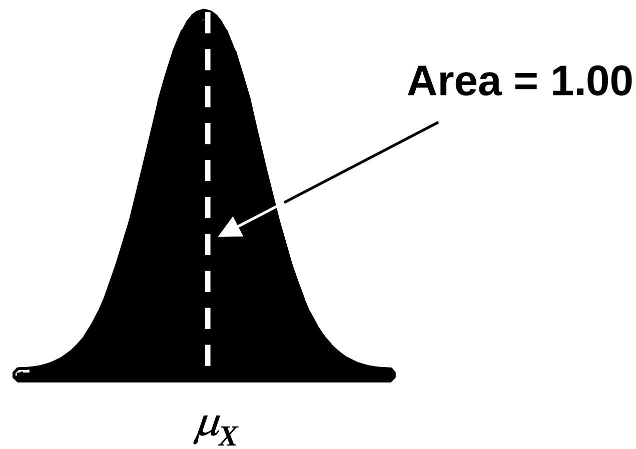
- **Resolving One WBS Element's Risk Issues by Spending More Money on It Often Involves Increasing Cost of Several Other Elements as Well**
  - For Example, Technical Risks in Radar Subsystem Will Tend to Induce Weight (and therefore) Cost Growth in Power, Platform, Software, and Other Subsystems
  - Schedule Slippage Due to Problems in One WBS Element Lead to Cost Growth in Other Elements ("Standing Army Effect")
- **Numerical Values of Inter-WBS-Element Correlations are Difficult to Estimate, but That's Another Story**

Ground-System Costs are Correlated with Satellite Costs

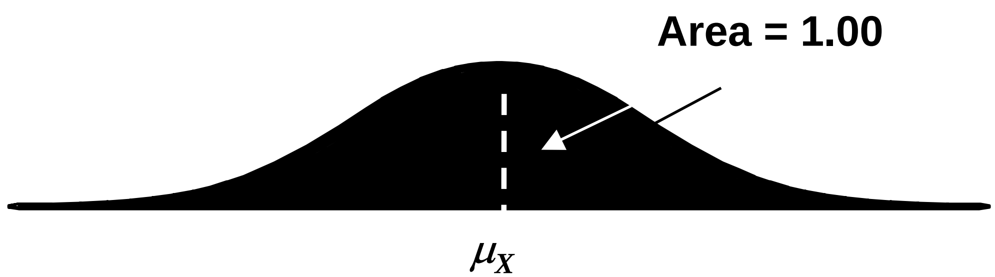
- **Difficulties in Meeting Original Specifications for Space Vehicle's Bus and Payload Tend to Induce Requirements Changes (and new costs) in Ground System**
  - Resolving Technical Risks in Spaceborne Observing System Often Involves Adjusting Analysis and Data Base Software on Ground
  - Schedule Slippage in Space Vehicle Production or Launch Forces Delays in Ground-System Testing and Final Checkout
  - Satellite Hardware Problems Discovered Late in Program Often Have to Be Circumvented by Making Expensive Last-minute Fixes to Ground-System Software
- **As We Will Soon See, Inter-Element Correlation Tends to Increase the Spread of the Total-Cost Probability Distribution**

Variance (σ^2) of Cost Distribution Measures Cost Uncertainty

- σ_X^2 Small Means Less Uncertainty



- σ_X^2 Large Means More Uncertainty



Correlation Affects the Variance

- $X_1, \; X_2, \; \ldots, \; X_n$ **are Costs of WBS Elements (Random Variables)**
- **Total Cost** = $\sum_{k=1}^{n} X_k = X_1 + X_2 + \ldots + X_n$
- **Mean of Total Cost** = $E\left(\sum_{k=1}^{n} X_k\right) = \sum_{k=1}^{n} E(X_k) = \sum_{k=1}^{n} \mu_k$
- **Variance of Total Cost** = $Var\left(\sum_{k=1}^{n} X_k\right)$

$$= \sum_{k=1}^{n} \sigma_k^2 + 2\sum_{j=2}^{n} \sum_{i=1}^{j-1} \rho_{ij} \sigma_i \sigma_j$$

Does Correlation Matter?

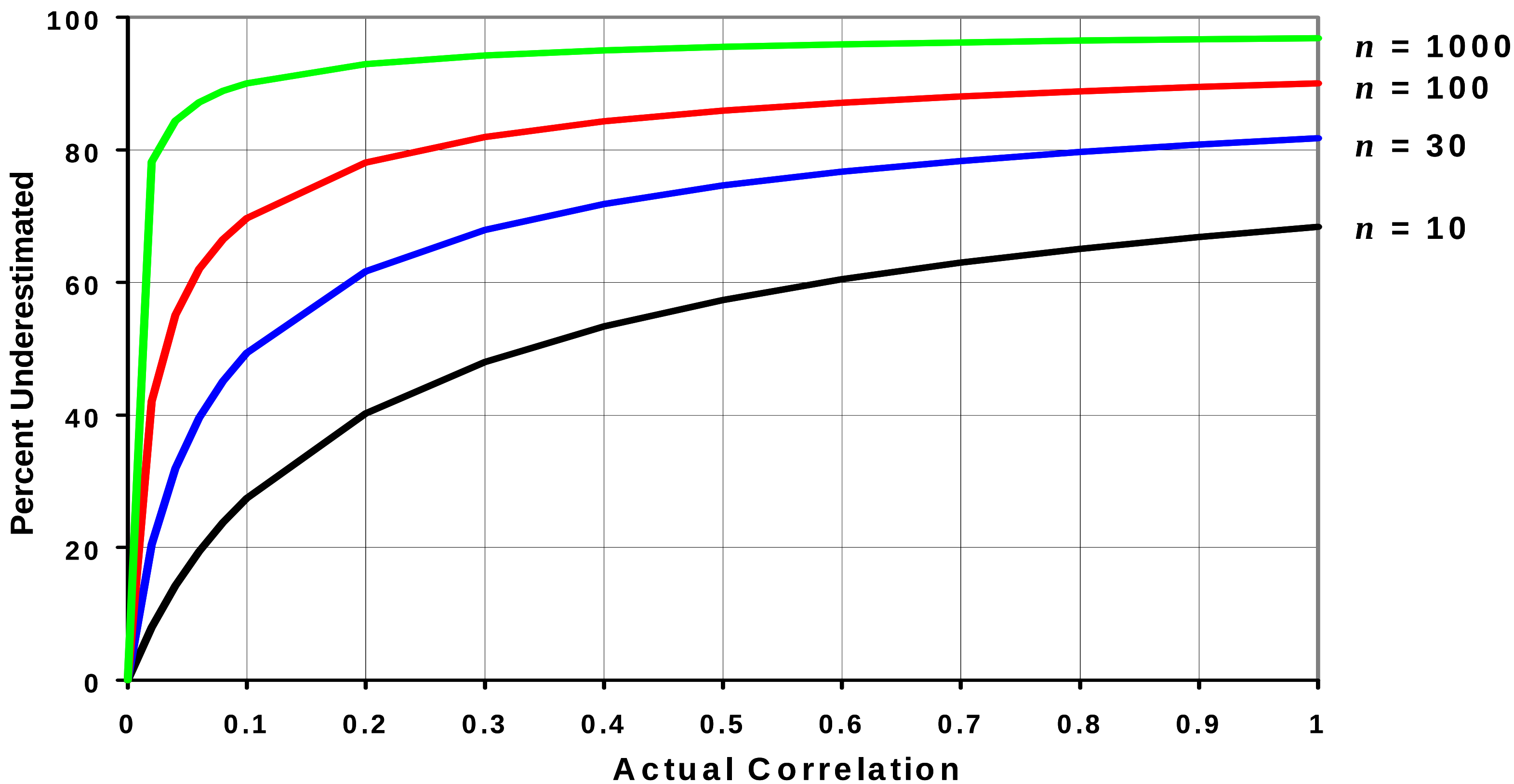
- **Suppose for Simplicity**
  - There are *n* Cost Elements $C_1, \; C_2, \; \ldots, \; C_n$
  - Each $Var(C_i) = \sigma^2$
  - Each $Corr(C_i, C_j) = \rho < 1$
  - Total Cost $C = \sum_{k=1}^{n} C_i$

- $$Var(C) = \sum_{k=1}^{n} Var(C_i) + 2\rho \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \sqrt{Var(C_i)\ Var(C_j)}$$
  $$= n\sigma^2 + n(n-1)\rho\sigma^2$$
  $$= n\sigma^2(1 + (n-1)\rho)$$

Correlation	0	ρ	1
$Var(C)$	$n\sigma^2$	$n\sigma^2(1+(n-1)\rho)$	$n^2\sigma^2$

Yes, Correlation Matters

- **Total-Cost Sigma is Underestimated When Inter-Element Correlations are Assumed to be Zero**
- **The Graph Shows the Percent Underestimation When Correlation Assumed to be 0 Instead of ρ**



Contents

- Software: Facts and Fictions
- COTS: It's Not Spelled "F-R-E-E"
- GFE: It's Not Free Either
- Impact of Correlation on Probable Cost
- **Summary**

Summary

- Software Cost Overruns and Schedule Slips are Almost Routine due to …
  - Inability to Define Requirements Precisely Up Front
  - Tendency to Underestimate Lines of Code Needed to Implement Software Solution
  - Ineffectiveness of Lines of Code (or anything else) as a Software Cost Driver
- The COTS Conundrum
  - Lack of Insight into Details of COTS Software Necessitates Very Thorough Testing
  - Integration and Testing Costs May Outweigh Acquisition Savings
- The GFE Trap
  - Proposal Obligates Government to Delivering Portions of System
  - GFE Often Fails to Meet Expectations, Necessitating ECOs and Actual, if not Official, Cost Overruns
- Correlations Between Risks
  - Correlations Increase Uncertainty in Total Cost
  - Ignoring Correlation Narrows Cost Distribution Unrealistically