UNIVERSITY OF SOUTHERN CALIFORNIA

COCOTS Software Integration Cost Model: Insights & Status

Dr. Barry Boehm (USC)

Mr. Chris Abts (USC)

Dr. Betsy Bailey (Software Metrics)

USC Center for Software Engineering

GSAW99 Conference

The Aerospace Corporation

El Segundo, CA

March 4, 1999

UNIVERSITY OF SOUTHERN CALIFORNIA

USC Center for Software Engineering

Points of Contact at USC-CSE in Los Angeles

Mr. Chris Abts (primary graduate researcher)……..……. ……(213) 740-6470
Ms. Ladonna Pierce (CSE Office Administrator)…..……..……(213) 740-5703
Dr. Barry W. Boehm (CSE Director)……..….(213) 740-8163
USC Center for Software Engineering FAX line……..……..……(213) 740-4927
COCOTS E-Mail……..……*cots-info@sunset.usc.edu*
World Wide Web page……..……..……*http://sunset.usc.edu/COCOTS/cocots.html*

Additional Contact at Software Metrics, Inc. in Virginia (near D.C.)

Dr. Elizabeth (Betsy) Bailey……..……..(703) 754-0115
FAX line……..…….(703) 754-0115
E-Mail……..bkbailey@erols.com

COTS Definition

- "Commercial Off the Shelf" Software
- Commercial Software Products
  - sold, leased, licensed at advertised prices
- Source Code Unavailable
  - generally an application program interface (API)
  - frequently tailoring options
- Usually periodic releases with feature growth, obsolescence

Rationale for Using COTS Products

- Significant change in s/w development practice over past 20 years:
  - building systems with pre-existing software to keep development & maintenance costs as low as possible
  - One such source: COTS
- Rationale for COTS based systems:
  - involve less development time by taking advantage of existing, market proven, vendor supported products, thus lowering overall development costs

UNIVERSITY OF SOUTHERN CALIFORNIA

COTS Advantages and Disadvantages

Advantages

- Available now;earlier payback
- Avoids expensive development & maintenance
- Predictable license costs & performance
- Rich in functionality
- Broadly used, mature technology
- Frequent upgrades often anticipate organization's needs
- Dedicated support organization
- Hardware/software independence
- Tracks technology trends

Disadvantages

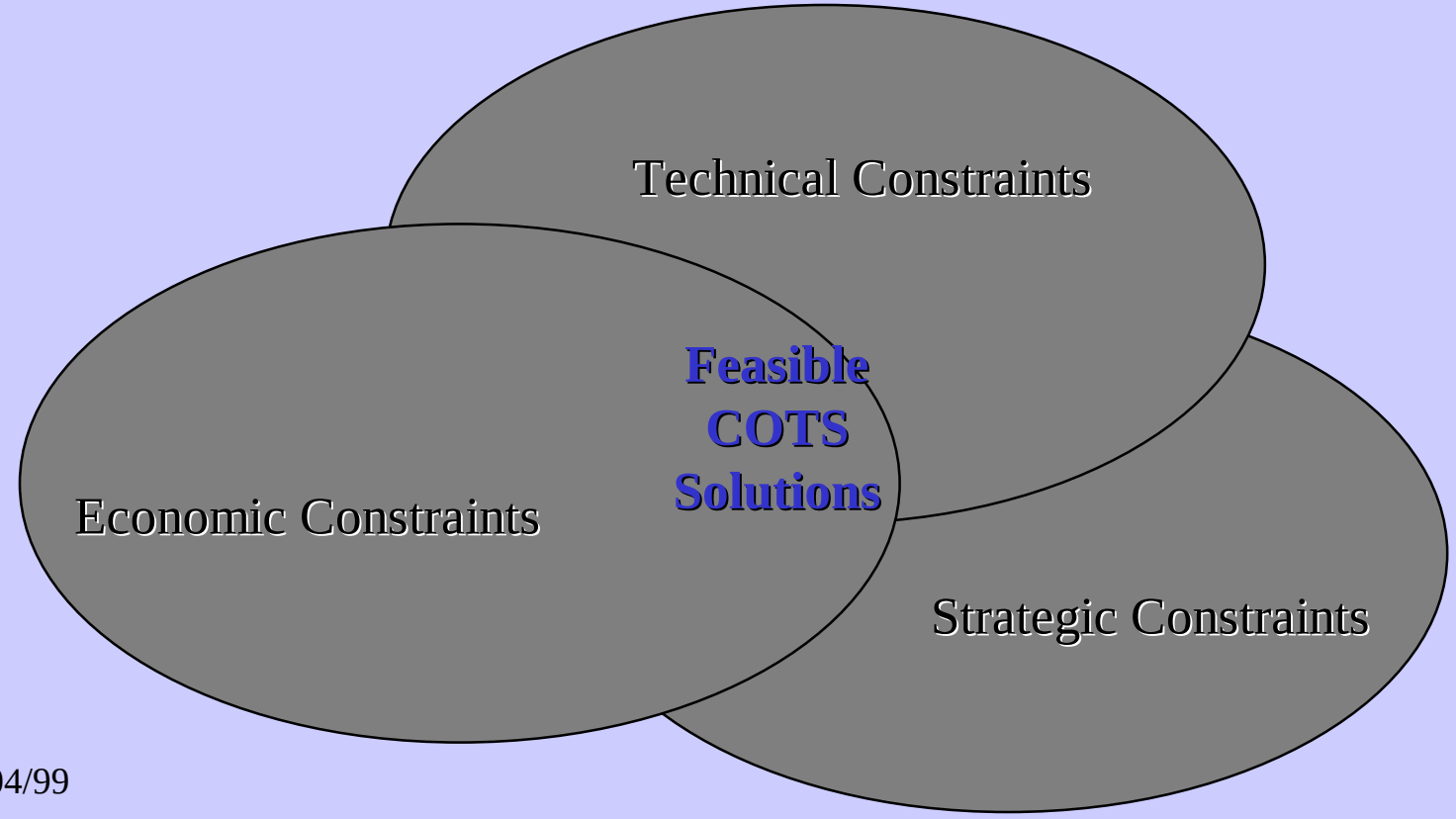
- Licensing and intellectual property procurement delays
- Up front license fees
- Recurring maintenance fees
- Reliability often unknown/ inadequate; scale often difficult to change
- Unnecessary features compromise usability, performance
- Functionality, efficiency constraints
- No control over upgrades/maintenance
- Dependency on vendor
- Efficiency sacrifices
- Integration not always trivial; incompatibilities among vendors
- Synchronizing multiple-vendor upgrades

Caveat to Using COTS Products

- Two main characteristics of COTS:
  - source code not available to developer
  - evolution not under control of developer
- Results in trade-off:
  - development time can be reduced, but often at cost of increased s/w component integration work
- Unique risks associated with COTS:
  - cost of licensing and redistribution rights, royalties, effort needed to understand the COTS software, pre-integration assessment and evaluation, post-integration certification of compliance with mission critical or safety critical requirements, indemnification against faults or damage caused by vendor supplied components, and costs incurred due to incompatibilities with other needed software and/or hardware

When are COTS Products the "Right" Solution?

- When they lie at the intersection of the three determinants of feasibility, ***and do so demonstrably better than could original code*:**



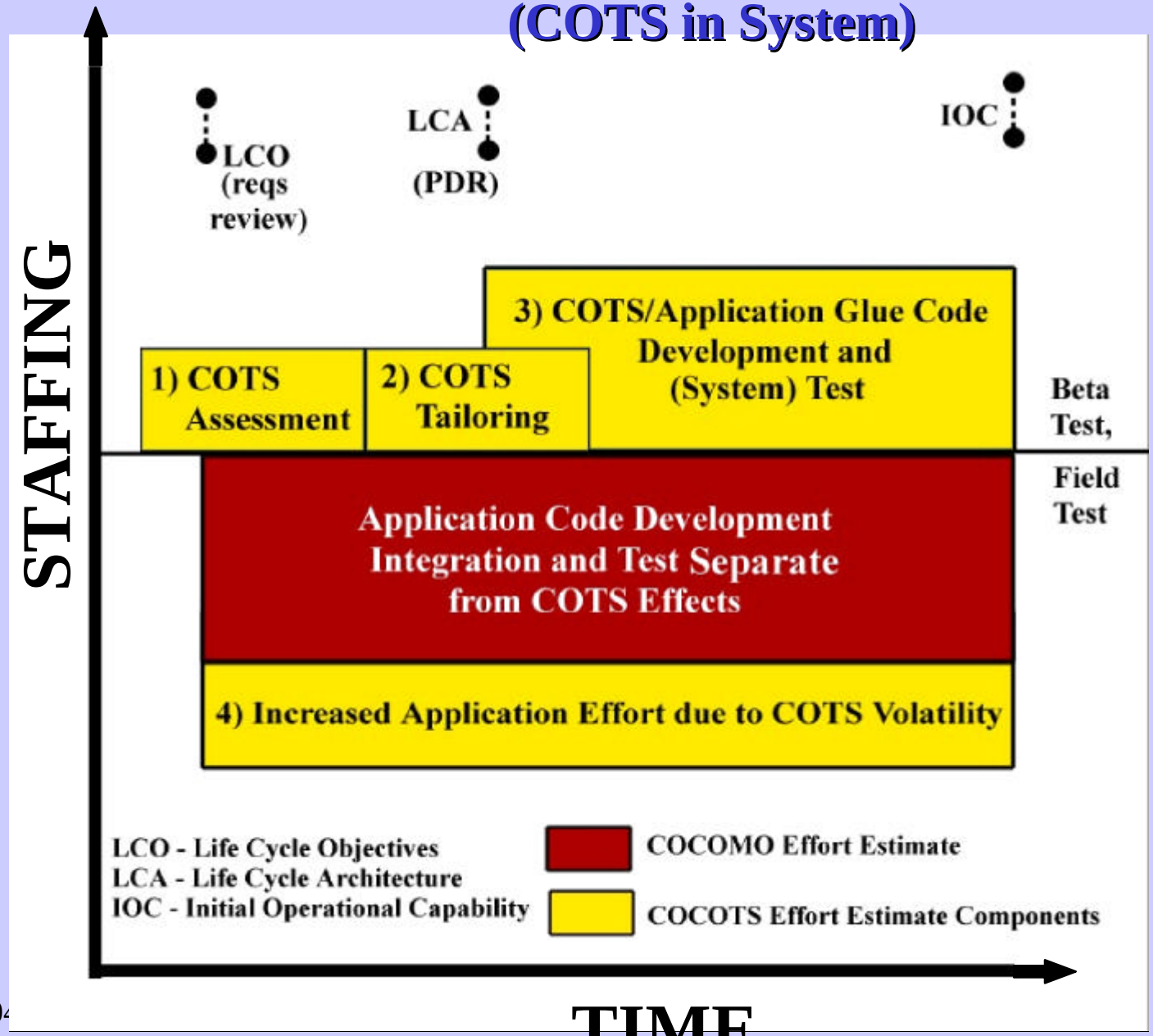
Constraints on COTS Solution Feasibility

- **Technical**
  - ability to supply the desired functionality at the required level of reliability
- **Economic**
  - ability to be incorporated and maintained in the new system within the available budget and schedule
- **Strategic**
  - ability to meet needs of the system operating environment--including technical, political, and legal considerations--now, and as environment is expected to evolve in the future

COTS Integration Sources of Effort

- COTS Assessment (pre- and post- commitment)
  - Of functionality, performance, interoperability, etc.
- COTS Tailoring and Tuning
  - Effects of platform, other COTS products
- Glue Code Development
  - Similar to other COCOMO II estimation
- Application Volatility Due to COTS
  - COTS volatility, shortfalls, learning curve
- Added Application V&V Effort
  - COTS option and stress testing
  - Debugging complications, incorrect fixes

COCOMO vs. COCOTS Cost Sources
(COTS in System)



COTS Integration Cost Sources:
1) Assessment

Initial Filtering Effort

$$\text{Total Effort} = (\#\ \text{COTS Candidates})\left(\frac{\text{Average Filtering Effort}}{\text{Candidate}}\right)$$

Final Selection Effort

$$\text{Total Effort} = \sum_{\substack{\text{Assessment} \\ \text{Attributes}}} (\#\ \text{COTS Candidates})\left(\frac{\text{Average Assessment Effort for Attribute in Given Domain}}{\text{Candidate}}\right)_i$$

- *List of attributes refined in collaboration with Dr. Elizabeth Bailey*
- *Effort/candidate is project-dependent, within domain guidelines*

COTS Integration Cost Sources:
1) Assessment - Assessment Attributes

Correctness		**Understandability**		**Portability**
Accuracy		Documentation quality		Portability
Correctness		Simplicity		
		Testability		**Functionality**
Availability/Robustness				Functionality
Availability		**Ease of use**		
Fail safe		Usability/Human Factors		**Price**
Fail soft				Initial purchase/lease
Fault tolerance		**Version Compatibility**		Recurring costs
Input error tolerance		Downward compatibility		
Redundancy		Upward compatibility		**Maturity**
Reliability				Product Maturity
Robustness		**Inter-component Compatibility**		Vendor Maturity
Safety		Compatibility with other components		
		Interoperability		**Vendor Support**
Security				Response time for critical problems
Security (Access related)		**Flexibility**		Support
Security (sabotage related)		Extendability		Warranty
		Flexibility		
Product Performance				**User Training**
Execution performance		**Installation/Upgrade Ease**		User training
Information/data capacity		Installation Ease		
Precision		Upgrade/Refresh ease		**Vendor Concessions**
Memory performance				Willingness to escrow source code
Response time				Willingness to make modifications
Throughput				

COTS Integration Cost Sources: 2) Tailoring

$$\text{Total Effort} = \sum_{\substack{\text{Tailoring} \\ \text{Complexity} \\ \text{Levels}}} \left(\begin{array}{c}\#\text{ COTS Candidates} \\ \text{Tailored at} \\ \text{Complexity Level}\end{array}\right)_i \left(\begin{array}{c}\text{Average Effort at Tailoring} \\ \text{Complexity Level in Domain}\end{array}\right)_i$$

- *Five tailoring effort complexity levels:*
  - *Very Low, Low, Nominal, High, Very High*
- *Differentiated based on number tailored parameters, difficulty of needed scripts, API iterations, etc.*

COTS Integration Cost Sources:
2) Tailoring - Dimensions of Tailoring Difficulty

	Individual Activity & Aid Complexity Ratings					
Tailoring Activities & Aids	**Very Low** (point value = 1)	**Low** (point value = 2)	**Nominal** (point value = 3)	**High** (point value = 4)	**Very High** (point value = 5)	**Corre-sponding Points**
Parameter Specification	Zero to 50 parms to be initialized.	51 to 100 parms to be initialized.	101 to 500 parms to be initialized.	501 to 1000 parms to be initialized.	1001 or more parms to be initialized.	-------
Script Writing	Menu driven; 1 to 5 line scripts; 1 to 5 scripts needed.	Menu driven; 6 to 10 line scripts; 6 to 15 scripts needed.	Hand written; 11 to 25 line scripts; 16 to 30 scripts needed.	Hand written; 26 to 50 line scripts; 31 to 50 scripts needed.	Hand written; 51 or more line scripts; 51 or more scripts needed.	-------
I/O Report & GUI Screen Specification & Layout	Automated or standard templates used; 1 to 5 reports/screens needed.	Automated or standard templates used; 6 to 15 reports/screens needed.	Automated or standard templates used; 16 to 25 reports/screens needed.	Hand written or custom designed; 26 to 50 reports/screens needed.	Hand written or custom designed; 51 or more reports/screens needed.	-------
Security/Access Protocol Initialization & Set-up	1 security level; 1 to 20 user profiles; 1 input screen/user.	2 security levels 21 to 50 user profiles; 2 input screens/user.	3 security levels 51 to 75 user profiles; 3 input screens/user.	4 security levels 76 to 100 user profiles; 4 input screens/user.	5 or more security levels 101 or more user profiles; 5 or more input screens/user.	-------
Availability of COTS Tailoring Tools	No tools available.	**N/A**	**N/A**	**N/A**	Tools are available.	-------

Total Point Score = _____

COTS Integration Cost Sources: 3) Glue Code Development and Test

$$\text{Total Effort} = A \cdot [(\text{size})(1+\text{breakage})]^{B} \cdot \square \quad (\text{effort multipliers})$$

- **A** - a linear scaling constant
- **Size** - of the glue code in SLOC or FP
- **Breakage** - of the glue code due to change in requirements and/or COTS volatility
- **Effort Multipliers** - 13 parameters, each with settings ranging **VL** to **VH**
- **B** - an architectural scale factor with settings **VL** to **VH**

UNIVERSITY OF SOUTHERN CALIFORNIA

COTS Integration Cost Sources:
3) Glue Code Development and Test - Glue Code Cost Drivers

Personnel Drivers

1) ACIEP - COTS Integrator Experience with Product
2) ACIPC - COTS Integrator Personnel Capability
3) AXCIP - Integrator Experience with COTS Integration Processes
4) APCON - Integrator Personnel Continuity

COTS Component Drivers

5) ACPMT - COTS Product Maturity
6) ACSEW - COTS Supplier Product Extension Willingness
7) APCPX - COTS Product Interface Complexity
8) ACPPS - COTS Supplier Product Support
9) ACPTD - COTS Supplier Provided Training and Documentation

Application/System Drivers

10) ACREL - Constraints on Application System/Subsystem Reliability
11) AACPX - Application Interface Complexity
12) ACPER - Constraints on COTS Technical Performance
13) ASPRT - Application System Portability

Nonlinear Scale Factor

1) AAREN - Application Architectural Engineering

COTS Integration Cost Sources:
4) Increased Application Effort Due to COTS Volatility

Approximate Model:

$$\text{Total Effort} = (\text{Application Effort}) \cdot \left[\frac{\text{BRAK COTS}}{100}\right] \cdot (\text{EAF})_{\text{COTS}}$$

Detailed Model with COCOMO II Parameters:

$$\text{Total Effort} = (\text{Application Effort}) \cdot \left[\left(1+\frac{\text{BRAK COTS}}{1+\text{BRAK}}\right)^{1.01+\square} - 1\right] \cdot (\text{EAF})_{\text{COTS}}$$

BRAK COTS: % application code breakage due to COTS volatility
BRAK : % application code breakage otherwise
□ : COCOMO II scale factor
EAF : Effort Adjustment Factor (product of effort multipliers)

COTS Integration Cost Sources:
4) Increased Application Effort Due to COTS Volatility
- COCOMO II Scale Factors

Scale Factor	Very Low	Low	Nominal	High	Very High	Extra High
Precedentedness	thoroughly unprecedented	Largely unprecedented	somewhat unprecedented	generally familiar	largely familiar	thoroughly familiar
Development Flexibility	rigorous	Occasional Relaxation	some relaxation	general conformity	some conformity	general goals
Architecture/Risk Resolution	little (20%)	some (40%)	often (60%)	generally (75%)	mostly (90%)	full (100%)
Team Cohesion	some difficult interactions	Basically cooperative interactions	largely cooperative	highly cooperative	seamless interactions	N/A
Process Maturity	Chaos	CMM Level 1	CMM Level 2	CMM Level 3	CMM Level 4	CMM Level 5

* percentage of module interfaces specified, percentage of significant risks eliminated.

Total COTS Integration Cost Estimate

Total Integration Effort (in Person-Months) =

Assessment Effort + Tailoring Effort + Glue Code Effort + Volatility Effort

where

Assessment Effort = Filtering Effort + Final Selection Effort

Total integration Cost =

(Total Integration Effort) • ($$/Person-Month)

Two Models, Differing Fidelity in Development

(Parallels COCOMO II modeling)

Early Design COCOTS model

-roll up of parameters in Assessment, Glue code submodels into fewer, more aggregated factors; inclusion of only the approximate Volatility model.

-less fidelity but requires fewer data points to calibrate.

-intended for more "what if" kind of estimating, earlier in the development process.

Post-architecture COCOTS model

-the full model as presented in preceding charts

UNIVERSITY OF SOUTHERN CALIFORNIA

Calibration Data Collection Status

- **6 Student Digital Library Projects**
  - **– 8 more by end Spring ‘99 semester**
- **12 Industrial Projects**
  - **– FAA & aerospace contractors**
  - **– 8+ additional projects anticipated by mid ‘99**
  - **– will allow calibration of Early Design version**
- **Other Sources Being Explored**
  - **– NASA, DoD, Commercial**
  - **– USC-CSE Affiliates, GSAW & ICSE conferences**

UNIVERSITY OF SOUTHERN CALIFORNIA

Experiences with Student Data

Raw Project Data

ACTIVITY	Group 1 Edgar Cpr	Group 2 Med. Mscrp	Group 3 Tech Rpts	Group 4 LAPIS	Group 5 CNTV Archv	Group 6 Hancock PH	Total Pers-hrs by Activity	% Total Pers-hrs by Activity
General Activity								
Determine Requirements:	16.00	49.50	86.50	26.50	5.50	38.50	222.50	4.99
Prepare, update plans :	107.00	142.00	209.50	39.00	83.50	134.75	715.75	16.06
Design product :	99.00	3.00	103.50	63.50	13.00	96.00	378.00	8.48
Code product :	161.00	20.50	190.00	168.00	67.50	115.00	722.00	16.20
Participate in formal design/code reviews:	14.00	8.00	21.00	21.00	22.50	24.00	110.50	2.48
Integrate and test :	70.00	94.50	85.50	6.50	13.00	29.50	299.00	6.71
Fix defects found in testing:	60.00	27.50	61.00	2.00	15.00	71.00	236.50	5.31
COTS Related Activity								
Understand and qualify COTS:	**2.00**	6.00	98.50	10.00	61.00	19.50	197.00	4.42
Design COTS glue code :	0.00	0.00	7.50	0.00	0.30	9.00	16.80	0.38
Code COTS glue code :	0.00	0.00	4.00	0.00	16.80	30.50	51.30	1.15
Fix defects found in COTS testing:	5.00	0.00	2.50	1.00	1.50	4.00	14.00	0.31
Administrative Activity								
Management:	8.50	34.00	33.50	13.50	10.00	25.00	124.50	2.79
Documentation :	52.50	449.50	38.00	59.50	68.00	126.00	793.50	17.81
Other:	114.00	239.00	31.50	8.00	100.00	82.50	575.00	12.90
TOTAL WEEKLY Person-Hours	709.00	1073.50	972.50	418.50	477.60	805.25	4456.35	99.99

Table VIII.1- Effort hours by activity for graduate software engineering class projects incorporating COTS products.

Key: Group 1 - EDGAR Corporate Data
Group 2 - Medieval Manuscripts
Group 3 - Technical Reports
Group 4 - Latin American Pamphlets
Group 5 - CNTV Moving Image Archive
Group 6 - Hancock Photo Archive

UNIVERSITY OF SOUTHERN CALIFORNIA

Experiences with Student Data

COTS Assessment Effort Distribution

Groups 3 & 5 (search engines)

Gross Attributes	Activities I	II	III	
1. Functionality	20%		10%	30%
2. Performance	10%			10%
3. Dependability	10%	10%		20%
4. Usability	10%		10%	20%
5. Adaptability		10%		10%
6. Operability			10%	10%
7. Cost				
	50%	20%	30%	100%

Distribution of assessment effort by activity and attribute.

I: nominal exercise - use COTS as intended by vendor
II: off-nominal exercise - adapt COTS to new use
III: reading and research

Experiences with Student Data

Glue Code Submodel Calibration

Project	A	Size (SLOC)	B	xEAFs	Estimate (P-hr)	Actual (P-hr)	Relative Error
3	0.009	500	1.04	1.82	10.50	11.50	-9%
5	0.009	400	1.12	2.25	16.62	17.10	-3%
6	0.009	218	1.16	10.42	48.38	39.50	22%

A = .009 => 111 SLOC/P-hr

UNIVERSITY OF
SOUTHERN CALIFORNIA

Suggested COTS Classes

- **database**
- **network management**
- **GUI builders**
- **operating systems**
- **report generators**
- **device drivers**
- **compilers**
- **decision support systems**
- **other???**

Immediate COCOTS Follow-ons

- Modeling of schedule estimation and activity distribution
- Integration with COCOMO II estimation model
- More extensive tool implementation

In Conclusion: COCOTS' Most Important Aspect

- COCOTS is completely open. Regardless of whatever estimates it provides, the descriptions of the elements that have gone into the model help highlight the most important factors that should be of concern to managers and developers of software systems using COTS software components.

- It's the essence of a "constructive" cost model:
  - one that helps an estimator better *understand* the complexities of a given software job to be done
  - by being open permits the estimator to know exactly *why* a model gives the estimate it does