Outline

- Model Development History and Support
- Problem Context
- COTS Software Integration Cost Sources
- COCOTS vs. COCOMO Cost Sources
- COTS Assessment
- COTS Tailoring
- COTS Glue Code Development and Test
- COTS Volatility Effects on Application Development Cost
- Total COTS Integration Cost Estimate
- Prospective COCOTS Follow-ons
- Conclusions

Model Development History and Support

- USAF/ESC Effort
  - March 1996 through June 1997
    - Initial Glue Code Model Definition, Experimental Calibration
- FAA Effort
  - Phase 1 (July to October, 1997)
    - Glue Code Model Redefinition, Experimental Calibration
  - Phase 2 (October 1997 to July 1998)
    - Glue Code Model Refinement
    - Assessment, Tailoring, and Volatility Models Defined
  - Phase 3 (July 1998 to December 1998)
    - Further Data Collection & Model Refinement, Calibration
    - Goal: calibrated model available by end 1998
- ONR Effort
  - January 1998 through 1999
    - Further Refinement of Models; Data Collection & Calibration
    - Determination of How Best to Associate COCOTS with COCOMO II

Coordination of FAA and ONR Data Collection Being Pursued with Help of DoD
Problem Context: What is (and Isn’t) COTS?
- terms from recent Ground System Architectures Workshop

- COTS: Commercial Off-the-Shelf
- GOTS: Government Off-the-Shelf
- HOTS: Hot Off-the-Shelf
- NOTS: Not Off-the-Shelf
- ROTS: Research Off-the-Shelf

Problem Context: COTS Phenomena, Pitfalls and Practices

- You have no control over a COTS product’s functionality or performance.
- Most COTS products are not designed to interoperate with each other.
- You have no control over a COTS product’s evolution
- COTS vendor behavior varies widely
**Problem Context: Modeling**

COTS and Custom Applications Components

New COCOTS Modeling Problem

COTS Infrastructure

COCOMO II: PVOL, PEXP

COTS Tools

LTEX, TOOL

Cost Modeling Currently Addressed

*Initial COCOTS Focus: Software Development; Operations & Maintenance to be addressed later*

---

**COTS Software Integration Cost Sources**

1. COTS Assessment
   - *Initial Filtering*
   - Final Selection

2. COTS Tailoring

3. COTS Application Glue Code Development and (System) Test

4. COTS Volatility Effects on Application Development Cost

*Initial COCOTS Focus: Software Development; Operations & Maintenance to be addressed later*
COCOMO Cost Sources
(No COTS in System)

Application Code Development Integration and Test Without The Use of COTS in System

1) COTS Assessment  
2) COTS Tailoring
3) COTS/Application Glue Code Development and (System) Test
4) Increased Application Effort due to COTS Volatility

COCOMO vs. COCOTS Cost Sources  
(COTS in System)
Objectory Management Checkpoints

<table>
<thead>
<tr>
<th>Inception</th>
<th>Elaboration</th>
<th>Construction</th>
<th>Transition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Iteration 1</td>
<td>Iteration 2</td>
<td>Iteration 3</td>
<td>Iteration 4</td>
</tr>
</tbody>
</table>

**Major Milestones**
- LCO
- LCA
- IOC
- Full Release

Strategic focus on global concerns of the entire software project

**Minor Milestones**

Tactical focus on local concerns of current iteration

**Status Assessments**

Periodic synchronization of stakeholder expectations

---

Objectory Information Set Evolution

<table>
<thead>
<tr>
<th>Engineering Stage</th>
<th>Manufacturing Stage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inception</td>
<td>Elaboration</td>
</tr>
<tr>
<td>Feasibility Iterations</td>
<td>Architecture Iterations</td>
</tr>
<tr>
<td>LCO</td>
<td>LCA</td>
</tr>
</tbody>
</table>

RATIONAL
COTS Integration Cost Sources:

1) Assessment

Initial Filtering Effort

Total Effort = \(\sum (\# \text{COTS Candidates}) \times \left( \text{Average Filtering Effort} \right) \)

Final Selection Effort

Total Effort = \(\sum (\# \text{COTS Candidates}) \times \left( \text{Average Assessment Effort for Attribute in Given Domain} \right) \)

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

<table>
<thead>
<tr>
<th>Correctness</th>
<th>Understandability</th>
<th>Portability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy</td>
<td>Documentation</td>
<td></td>
</tr>
<tr>
<td>Correctness</td>
<td>Simplicity</td>
<td></td>
</tr>
<tr>
<td>Availability/Robustness</td>
<td>Ease of use</td>
<td></td>
</tr>
<tr>
<td>Fault tolerance</td>
<td>Version Compatibility</td>
<td>Downward compatibility</td>
</tr>
<tr>
<td>Redundancy</td>
<td>Upward compatibility</td>
<td>Maturity</td>
</tr>
<tr>
<td>Robustness</td>
<td>Inter-component Compatibility</td>
<td>Vendor Maturity</td>
</tr>
<tr>
<td>Safety</td>
<td>Compatibility with other components</td>
<td></td>
</tr>
<tr>
<td>Security</td>
<td>Interoperability</td>
<td>Vendor Support</td>
</tr>
<tr>
<td>Security (Access related)</td>
<td>Flexibility</td>
<td></td>
</tr>
<tr>
<td>Security (Sbateage related)</td>
<td>Extendability</td>
<td></td>
</tr>
<tr>
<td>Product Performance</td>
<td>Installation-Upgrade Ease</td>
<td>User Training</td>
</tr>
<tr>
<td>Execution performance</td>
<td>Installation Ease</td>
<td>User Training</td>
</tr>
<tr>
<td>Error Handling</td>
<td>Upgrade/Refresh ease</td>
<td>Vendor Concessions</td>
</tr>
<tr>
<td>Performance</td>
<td>Willingness to escrow source code</td>
<td></td>
</tr>
<tr>
<td>Response time</td>
<td>Willingness to make modifications</td>
<td></td>
</tr>
<tr>
<td>Throughput</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Center for Software Engineering
COTS Integration Cost Sources:

2) Tailoring

\[ \text{Total Effort} = \sum \left( \frac{\text{# COTS Candidates Tailored at Complexity Level}}{\text{Tailoring Complexity Levels}} \right) \left( \frac{\text{Average Effort at Tailoring Complexity Level in Domain}}{i} \right) \]

-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.

<table>
<thead>
<tr>
<th>Tailoring Activities &amp; Tasks</th>
<th>Very Low (point value = 1)</th>
<th>Low (point value = 2)</th>
<th>Standard (point value = 3)</th>
<th>High (point value = 4)</th>
<th>Very High (point value = 5)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter Specification</td>
<td>10 to 50 parameters to be initialized</td>
<td>51 to 100 parameters to be initialized</td>
<td>101 to 300 parameters to be initialized</td>
<td>301 to 500 parameters to be initialized</td>
<td>501 or more parameters to be initialized</td>
</tr>
<tr>
<td>Script Writing</td>
<td>Automated or standard templates used</td>
<td>Automated or standard templates used</td>
<td>Automated or standard templates used</td>
<td>Automated or standard templates used</td>
<td>Manual or custom designed</td>
</tr>
<tr>
<td>I/O Report &amp; GUI</td>
<td>Automated or standard templates used</td>
<td>Automated or standard templates used</td>
<td>Automated or standard templates used</td>
<td>Automated or standard templates used</td>
<td>Manually written or custom designed</td>
</tr>
<tr>
<td>Security/Access</td>
<td>1 security level; 1 to 20 user profiles; 1 input screen/user</td>
<td>2 security levels; 21 to 50 user profiles; 2 input screens/user</td>
<td>3 security levels; 51 to 75 user profiles; 3 input screens/user</td>
<td>4 security levels; 76 to 100 user profiles; 4 input screens/user</td>
<td>5 or more security levels; 101 or more user profiles; 5 or more input screens/user</td>
</tr>
<tr>
<td>Availability of COTS Tailoring Tools</td>
<td>No tech available</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>Tech available</td>
</tr>
</tbody>
</table>

Total Point Score = 
COTS Integration Cost Sources:
3) Glue Code Development and Test

Total Effort = A • [(size)(1+breakage)]^B • Π (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

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) ACPS - COTS Supplier Product Support
9) APCPD - 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:**

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

**Detailed Model with COCOMO II Parameters:**

\[ \text{Total Effort} = (\text{Application Effort}) \times \left( \left( 1 + \frac{\text{BRAK COTS}}{1 + \text{BRAK}} \right)^{1.01 + \sum} \right) \times (\text{EAF}) \]

- BRAK COTS: % application code breakage due to COTS volatility
- BRAK: % application code breakage otherwise
- \( \sum \): 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

<table>
<thead>
<tr>
<th>Scale Factor</th>
<th>Very Low</th>
<th>Low</th>
<th>Nominal</th>
<th>High</th>
<th>Very High</th>
<th>Extra High</th>
</tr>
</thead>
<tbody>
<tr>
<td>Precededness</td>
<td>thoroughly unprecedented</td>
<td>largely unprecedented</td>
<td>somewhat unprecedented</td>
<td>generally familiar</td>
<td>largely familiar</td>
<td>thoroughly familiar</td>
</tr>
<tr>
<td>Development Flexibility</td>
<td>rigorous</td>
<td>occasional relaxation</td>
<td>some relaxation</td>
<td>general conformity</td>
<td>some conformity</td>
<td>general goals</td>
</tr>
<tr>
<td>Architecture/Risk Resolution</td>
<td>little (20%)</td>
<td>some (40%)</td>
<td>often (60%)</td>
<td>generally (75%)</td>
<td>mostly (90%)</td>
<td>full (100%)</td>
</tr>
<tr>
<td>Team Cohesion</td>
<td>some difficult interactions</td>
<td>basically cooperative interactions</td>
<td>largely cooperative</td>
<td>highly cooperative</td>
<td>seamless interactions</td>
<td>N/A</td>
</tr>
<tr>
<td>Process Maturity</td>
<td>CMM Level 1</td>
<td>CMM Level 2</td>
<td>CMM Level 3</td>
<td>CMM Level 4</td>
<td>CMM Level 5</td>
<td>N/A</td>
</tr>
</tbody>
</table>

* 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)

**Prospective COCOTS Follow-ons**

- Extensive data collection and conditioning
- Recalibration and iteration of the model within current structure
- Experimental usage and refinement, including exploration of other cost drivers and model forms
- Modeling of schedule estimation and activity distribution
- Integration with COCOMO II estimation model
- More extensive model implementation
- Modeling other COTS related costs
  - Licenses, training, maintenance, hardware
Modeling Other COTS Related Costs

• Largely a \( \text{unit cost} \) * (# units) framework
  – Unit costs vary by quantity, platform, time

• Need to consider time-phasing of acquisition, implementation, operations & maintenance

• Biggest challenge will be complex, dynamic COTS price structures

Extended COCOTS Model

\[
\text{Cost (t)} = \text{Cost [SW development]} + \text{Cost [SW maintenance]}(t) + \text{Cost [SW COTS integration]} + \text{Cost [SW COTS integ. maint.]}(t) + \text{Cost [SW COTS]}(t) + \text{Cost [HW COTS]}(t)
\]

-- COCOMO II, others
-- COCOMO II, others
-- COCOTS
-- COCOTS
-- (see chart following)
-- (see chart following)
Estimating Cost of Software COTS

\[
\text{Cost [SW COTS]}(t) = \text{Cost [acquisition office]}(t) + \text{Cost [licenses]}(t) + \text{Cost [implementation]}(t) + \text{Cost [Op. & Mnt.]}(t)
\]

Estimating Cost of Hardware COTS

\[
\text{Cost [HW COTS]}(t) = \text{Cost [acquisition office]}(t) + \sum_{i} \left( \text{Cost [acquisition]}(t) + \text{Cost [implementation]}(t) + \text{Cost [O&M]}(t) \right)
\]

for

\[ i = \text{processors, storage, workstation, communications} \]
Conclusions

• COCOTS provides solid framework for estimating software COTS integration cost
  – needs further data, calibration, iteration
  – current spreadsheet model could be used experimentally
• COCOTS can be extended to cover other COTS related costs
  – biggest challenge will be complex, dynamic COTS price structures

Proposal:

• Go with single project-level set of ratings rather than separate ratings per component

• Replace current APVOL definition (#releases/COTS component) by % BRAK due to COTS volatility
  [In glue code:
  * in application SW]
• Replace “COTS/NDI” by “COTS”

<table>
<thead>
<tr>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>Less data to collect</td>
<td>Harder for users to average ratings</td>
</tr>
<tr>
<td>Avoids formidable</td>
<td>Need data entry aggregation</td>
</tr>
<tr>
<td>rating-aggregation problems</td>
<td>guidelines for multi-component entries</td>
</tr>
<tr>
<td>Provides approach for model</td>
<td>FAA buy-in to current approach</td>
</tr>
<tr>
<td>#4: added App Develop effort</td>
<td></td>
</tr>
<tr>
<td>NDI handled by COCOMO II</td>
<td></td>
</tr>
</tbody>
</table>
Rating-Aggregation Problems

• Can’t just average ratings
  - relative # interactions of COTS
  - relative interaction complexity
  - volatility effects
  - breakage per release
  - aggregation of release updates

• No simple formulas for aggregating those effects

<table>
<thead>
<tr>
<th>#COTS</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td># interactions with APP, COTS</td>
<td>2.1 2.1 8.4</td>
</tr>
</tbody>
</table>

Disposition:

• Go with project-level BRAK GLUE parameter
  - avoid aggregation difficulties
  - includes effects of application volatility
  - compatible with BRAK COTS approach for added applications effort

• Leave other cost drivers at component level
  - easier user data/rating entry
  - start with simple averaging of ratings

• Replace “COTS/NDI” by “COTS”

Center for Software Engineering
BRAK GLUE Data Definition:

Added breakage in Glue App code due to COTS volatility
  • Relative to breakage with no volatility
  • Judgement based function of several factors
    - # releases during development for each COTS component
    - strategy for batching releases during development
    - number and complexity of interaction among COTS, applications components

Model:

\[
\begin{align*}
\text{△ App Effort} &= \text{App Effort} \cdot \left( 1 + \frac{\text{BRAK} + \text{BRAK COTS}}{100} \right)^{1.01 + \Sigma} \cdot (\text{EAF})_{\text{COTS}} \cdot (\text{EAF})_{\text{APP}} \\
\text{△ App Effort} &= \text{App Effort} \cdot \left( 1 + \frac{\text{BRAK}}{100} \right)^{1.01 + \Sigma} \cdot (\text{EAF})_{\text{APP}} \\
\text{△ App Effort} &= \text{App Effort} \cdot \left( 1 + \frac{\text{Bc}}{1 + \text{B}} \right)^{1.01 + \Sigma} \cdot (\text{EAF})_{\text{COTS}}
\end{align*}
\]