Early COCOTS

Betsy Clark / Chris Abts / Brad Clark / Ye Yang
17th International Forum on COCOMO and Software Cost Modeling
October 23, 2002
Greetings from Betsy
Outline

• A few words about COCOTS
• Objectives for Early COCOTS
• Sizing Inputs and Effort Multipliers
• Causal Analysis: Example
• Type of COTS-based Application
• Next Steps
What is COCOTTS?

• Constructive Cost Model for COTS-based applications
• Part of the COCOMO suite
• Estimates effort for
  – Assessment
  – Tailoring
  – Glue Code development
• Addresses initial development as well as maintenance
• Many of the cost drivers are important but are not typically known early
  – Example: Number of screens and reports
  – From trial uses of COCOTS, we found that people have a difficult time estimating these until they are well into prototyping or operational evaluations
What is Early COCOTS?

• 35,000 foot view to be used early in the lifecycle for:
  – rough order-of-magnitude (ROM) estimates
  – basic investment decisions (e.g., custom or COTS?)

• Simplified model
  – Uses information known early in the lifecycle is limited
Objectives for Early COCOTS -1

1. Provide a basis for estimating costs of COTS-Based systems early in the life-cycle
2. Cost drivers can be estimated or are known early on
3. Should handle COTS, NDI, and new code (the entire realm of possibility)
4. Constructive input parameters
   - Doing the rating exercise helps you do the job
Objectives for Early COCOTS -2

• Should help people understand consequences of basic decisions, e.g.,
  – What functionality should be custom and what should be COTS?
  – What is the impact of architecting for change (upfront vs downstream costs)?
Model Approach

- Model will estimate total life-cycle costs
- Lifecycle costs are estimated as a range of outputs, not a single point estimate
- Costs will be estimated at the system level, not at the level of individual components
- Cost will be aggregated across activities
Candidate Sizing Inputs

- Number of COTS products
- Number and criticality of COTS-provided user functions
- Degree of uncertainly about product choices
- Amount of developed software (equivalent SLOC)
Candidate Cost Drivers

- Product/marketplace maturity
- Integrator capability and experience
- Degree of mismatch between COTS capabilities and user needs
- Number of security levels and user profiles
- Number of operational configurations
- Complexity of integration
- Complexity of data conversion
- Operational constraints (reliability, availability, performance)
Causal Analysis

• For each candidate cost driver, we conducted a causal analysis, describing likely impacts on Development, Installation, and Maintenance

• Example causal analysis for Number of Products
### Development

**Causal Analysis on “Number of Products”**

- Increased number of product assessments
- More time spent creating an evolvable architecture
- More glue code
- More interfaces between products and between products and custom code
- More effort spent in analysis of dependencies (between products and hardware)
- More points of interaction for undesirable behavior
- Different vendor upgrade cycles. Decisions must be made about refresh even before initial deployment.
- Complication of managing multiple licenses
- Different product versions increases configuration management

### Installation

- Mixture of licensing types: platform, network seats, types of platforms supported

### Maintenance

- Different vendor upgrade cycles
- Number of interfaces and analysis of dependencies (other products and hardware)
- Understanding of product upgrade on system
- Evaluation of vendor patches / fixes; coordinating new system releases with the latest product upgrades / fixes; hardware requirements could be impacted
- Rewriting of glue code
COTS-Solution vs COTS-Intensive

• In the course of collecting and analyzing calibration data for COCOTS, we found two fundamentally different types of COTS-based systems
A Spectrum of COTS - Based Systems

**COTS-Solution Systems**

One substantial product (suite) tailored to provide significant system functionality

- Generic solutions; tightly coupled to end-user / business processes
- Vendor maintained
- Tailoring; parameterization focus

**COTS-Intensive Systems**

Multiple products from multiple suppliers integrated to collectively provide system functionality

- Probably more flexible in supporting end-user / business processes
- Project maintained
- Integration, engineering focus
  - Products / parts are “black boxes”
  - COTS, NDI, legacy
Next Steps

• Early COCOTS will be discussed at the workshop
• We will review cost drivers and ratings descriptions in the context of the two types of COTS-Based Systems
Workshop Objectives

• Provide feedback on
  – Basic requirements of an early model
  – Specific size and cost driver inputs

• Boehm has proposed a strawman rating scale for size and cost drivers

• Example:

<table>
<thead>
<tr>
<th>Multiplier</th>
<th>Very Low</th>
<th>Low</th>
<th>Nominal</th>
<th>High</th>
<th>Very High</th>
</tr>
</thead>
<tbody>
<tr>
<td>Complexity of Integration</td>
<td>Simple Adaptations</td>
<td>Straightforward glue code</td>
<td>Some critical-issue glue-code engineering</td>
<td>Complex, coupled glue-code engineering</td>
<td>Very complex, highly coupled glue-code engineering</td>
</tr>
</tbody>
</table>

Complexity of Integration

Multiplier | Very Low | Low | Nominal | High   | Very High |
-----------|----------|-----|---------|--------|-----------|
Simple Adaptations | Straightforward glue code | Some critical-issue glue-code engineering | Complex, coupled glue-code engineering | Very complex, highly coupled glue-code engineering |
## Relative Impact by COTS-Based System

<table>
<thead>
<tr>
<th>Cost Driver</th>
<th>COTS-Solution (single vendor)</th>
<th>Typically administrative, business critical</th>
<th>COTS-Intensive (multi-vendor)</th>
<th>Typically safety-critical</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of COTS products</td>
<td>N/A</td>
<td>⚫⚫⚫</td>
<td>⚫⚫⚫</td>
<td></td>
</tr>
<tr>
<td>Product/Marketplace Maturity</td>
<td>⚫⚫</td>
<td>⚫⚫⚫</td>
<td>⚫⚫⚫</td>
<td></td>
</tr>
<tr>
<td>Integrator Capability and Experience</td>
<td>⚫</td>
<td>⚫⚫⚫</td>
<td>⚫⚫⚫</td>
<td></td>
</tr>
<tr>
<td>Degree of Mismatch between COTS Capability and User Needs</td>
<td>⚫⚫⚫</td>
<td>⚫</td>
<td>❌</td>
<td></td>
</tr>
<tr>
<td>Number of Security Levels and User Profiles</td>
<td>⚫⚫⚫</td>
<td>⚫⚫⚫</td>
<td>⚫⚫⚫</td>
<td></td>
</tr>
<tr>
<td>Number of Operational Configurations</td>
<td>⚫</td>
<td>⚫⚫⚫</td>
<td>⚫⚫⚫</td>
<td></td>
</tr>
<tr>
<td>Complexity of Integration</td>
<td>⚫⚫</td>
<td>⚫⚫⚫</td>
<td>⚫⚫⚫</td>
<td></td>
</tr>
<tr>
<td>Complexity of Data Conversion</td>
<td>⚫⚫⚫</td>
<td>⚫</td>
<td>⚫</td>
<td></td>
</tr>
<tr>
<td>Operational Constraints</td>
<td>⚫⚫</td>
<td>⚫⚫⚫</td>
<td>⚫⚫⚫</td>
<td></td>
</tr>
</tbody>
</table>
For More Information

Betsy Clark
(703) 754-0115
Betsy@software-metrics.com

Chris Abts
(979) 862-8055
CAbts@cgsb.tamu.edu

Brad Clark
(703) 754-0115
Brad@software-metrics.com

Ye Yang
(213) 740-6470
yey@usc.edu