COCOMO III Workshop

Brad Clark, Moderator
USC Center for Systems and Software Engineering
29th International Forum on COCOMO and Systems/Software Cost Modeling
October 23, 2014
COCOMOIII Collaboration

• CSSE Website based on UCC site
  http://csse.usc.edu/ucc_wp/

• Sign-up for site
  – Email notification list for status, updates, decisions, news
  – Email for newsletter
  – Download content according to access level

• Website forum to discuss issues
  – Start a topic
  – Add discussion to an existing topic
COCOMO III Charter

• Follow a “practical” approach to updating the COCOMO II model

• Objectives
  – Adjust “Nominal” ratings for Scale/Cost (S/C) drivers, if appropriate
  – Review S/C drivers
    • Combine highly correlated drivers
    • Review drivers for relevance
    • Add new S/C drivers for estimating today’s development practices (data permitting)
  – Retire older data & collect new data
  – Account for productivity decline in incremental development
COCOMO III Charter

• Objectives (continued)
  – Reduce subjectivity in S/C driver ratings with better selection description
  – Provide a Rosetta stone to map back to previous COCOMO models
  – Explore “reasonableness” of incorporation of other COCOMO-suite models (additional S/C drivers)
COCOMO III Charter

• Objectives (continued)
  – Investigate “alternative” size inputs
    • Theoretically, any consistently applied sizing method should work with S/C drivers
    • Only multiplicative and scale driver constants need to be calibrated
  – Review relevance of the Adaptation Adjustment Multiplier
    • AAF = (0.4 x DM) + (0.3 x CM) + (0.3 x IM)
      – % Design Modified (DM)
      – % Code & Unit Test Modified (CM)
      – % Integration & Test Modified (IM)
    • Software Understanding (SU)
    • Assessment and Assimilation (AA)
    • Programmer Unfamiliarity (UNFM)
COCOMO III Charter

• Objectives (continued)
  – Strategy for maintaining past COCOMO models
  – Address Maintenance / Sustainment costs
  – Develop a set of ~10 Use Cases for COCOMO III
    • Useful in making choices regarding models, etc.
    • Used by Novice Users or Advanced Users?
  – Time-boxing releases
  – Solve for one unknown given other inputs
  – Good for the next ~14 years
  – Needs to useful for the commercial world
  – Needs to be locally calibratable
  – Multi-Driver rating interactions
    • Fences to prevent invalid inputs
COCOMO III Charter

• Objectives (continued)
  – New model should be “consistent” with previous model, e.g. no surprises, no big radical change, upwardly compatible
  – Support incremental or spiral development
  – Produce distributions of estimates
Workshop Purpose

- Consider incorporating a Segmentation Strategy
- Discuss additional model forms
- Review current set of cost and scale drivers
- Collect enough feedback to create a data collection form for COCOMO III
Segmentation Strategy
COCOMO II 2000 Effort Prediction Accuracy

COCOMO II model accuracy improved when segmented by Organization

<table>
<thead>
<tr>
<th>Prediction Accuracy¹</th>
<th>Before Stratification by Organization</th>
<th>After Stratification by Organization</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRED(20)</td>
<td>63%</td>
<td>70%</td>
</tr>
<tr>
<td>PRED(25)</td>
<td>68%</td>
<td>76%</td>
</tr>
<tr>
<td>PRED(30)</td>
<td>75%</td>
<td>80%</td>
</tr>
</tbody>
</table>

Note:
1. PRED(X) = Y% means that Y% of the predicted values fall within X% of the actual values
Globbing Study – Oct 2007

• Create “pre-sets” for Scale/Cost driver ratings for different application domains
  – By selecting an application domain, driver ratings are selected based on the average ratings from data in the domain
  – Allows estimators to generate estimates quickly
  – Permits estimators to create a knowledge base on the basis of application domain characteristics

• Create calibration groups within the data
  – Hypothesis: projects in the database are so diverse that “local” calibration within a group should improve model accuracy and precision
Globbing Study – Driver Selection

- Business-driven drivers (these are independent of the application)
  - Analyst capability
  - Programmer capability
  - Personnel continuity
  - Application experience
  - Platform experience
  - Language and tool experience
  - Use of software tools
  - Multi-site development
  - Required development schedule
  - All Scale Drivers

- Application-driven drivers
  - Required software reliability
  - Database size
  - Product complexity
  - Developed for Reuse (maybe not)
  - Documentation match to life-cycle needs
  - Execution time constraint
  - Main storage constraint
  - Platform volatility
Globbing Study – Pre-set Rating Example

Globbing around application-type and size has been successfully demonstrated in other cost models.

<table>
<thead>
<tr>
<th></th>
<th>Glob-1</th>
<th>Glob-2</th>
<th>Glob-3</th>
<th>Glob-4</th>
</tr>
</thead>
<tbody>
<tr>
<td>REL Y</td>
<td>VH</td>
<td>H</td>
<td>H</td>
<td>N</td>
</tr>
<tr>
<td>DATA</td>
<td>N</td>
<td>H</td>
<td>L</td>
<td>N</td>
</tr>
<tr>
<td>CPLX</td>
<td>H</td>
<td>N</td>
<td>H</td>
<td>L</td>
</tr>
<tr>
<td>RUSE</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>DOCU</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>TIME</td>
<td>VH</td>
<td>H</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>STOR</td>
<td>VH</td>
<td>H</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>PVOL</td>
<td>H</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
</tbody>
</table>
Software Classification Method

- Super-Domain.
- Within the Super-Domain, are Application Domains

Super-Domains:

- Application Domains

Support:

- Information Systems

Engineering:

- Real-Time

Software Configuration Item

- Signal Processing
- Vehicle Payload
- Vehicle Control
- Real Time Embedded
- Microcode & Firmware
- Command & Control
- Communication
- System Software
- Scientific & Simulation
- Process Control
- Test, Mea, Diag, Equip.

Software Tools

Training

Mission Planning

Custom AIS Software

Enterprise Service Sys

Enterprise Info Sys

29th International Forum on Open Source, Systems/Software Cost Modeling
Real-Time (RT) Super-Domain

**Definition**
Real-Time is the most constrained type of software. These are specific solutions limited by system characteristics such as memory size, performance, or battery life. These projects take the most time and effort due to constraints, e.g.,
- May have guaranteed task execution requirements, i.e., missed deadline means catastrophic result
- May have to be compact and efficient due to limited storage capacity and high throughput requirements
- Could have very high reliability requirements (life critical, manned mission)
- Might have tightly coupled interfaces
- Program code may be imprinted on hardware devices
- May process sensor inputs and directs actuator outputs.
- Sometimes executed on special-purpose processors

<table>
<thead>
<tr>
<th>Relevant Application Domains</th>
<th>Microcode &amp; Firmware</th>
<th>Signal Processing</th>
<th>Vehicle Control</th>
<th>Vehicle Payload</th>
</tr>
</thead>
<tbody>
<tr>
<td>Other Real-Time Embedded</td>
<td>Command &amp; Control</td>
<td></td>
<td>Communication</td>
<td></td>
</tr>
</tbody>
</table>
# Engineering (ENG) Super-Domain

| Definition | Engineering software operates under less severe constraints than real-time software. This software may take the outputs of real-time software and further process them to provide human consumable information or automated control of devices. Or the software may perform transformation and aggregation / distribution of data. These projects take more time and effort due to multiple factors, e.g.,
|            | • May have a fast response time requirement
|            | • May have more storage capacity
|            | • Might need to be highly reliable but not life critical
|            | • May have multiple interfaces with other systems
|            | • May implement complex algorithms, models or protocols
|            | • Program code can be modified and uploaded
|            | • Executes on general purpose processors that may be embedded in special purpose hardware |

<table>
<thead>
<tr>
<th>Relevant Application Domains</th>
<th>System</th>
<th>Process Control</th>
<th>Scientific and Simulation</th>
<th>Test, Measurement, Diagnostic and Evaluation</th>
</tr>
</thead>
</table>
Support (SUP) Super-Domain

**Definition**
Support software assists with operator training and software testing. This software has few constraints, e.g.,
- Has to have an acceptable response time most of the time
- Less limited by storage or throughput
- Less stringent reliability requirement
- Software restarts are acceptable
- Fewer interfaces
- Relatively low complexity algorithms, models or protocols
- Program code can be modified and uploaded
- Executes on general purpose processors on general purpose computer boards

**Relevant Application Domains**

<table>
<thead>
<tr>
<th>Training</th>
<th>Software Tools</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Automated Information Systems (AIS)

Super-Domain

<table>
<thead>
<tr>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Automated information system software provides information processing services to humans or software applications. These applications allow the designated authority to exercise control and have access to typical business / intelligence processes and other types of information access. These systems also includes software that facilitates the interface and control among multiple COTS / GOTS software applications. This software has few constraints, e.g.,</td>
</tr>
<tr>
<td>• Must have acceptable response time</td>
</tr>
<tr>
<td>• Fewer storage or throughput constraints</td>
</tr>
<tr>
<td>• Must be reliable enough to prevent data loss</td>
</tr>
<tr>
<td>• May consist of a single COTS / GOTS solution or multiple products coordinated with customer software</td>
</tr>
<tr>
<td>• Algorithms, models and protocols are well understood</td>
</tr>
<tr>
<td>• Code may not be available for modification</td>
</tr>
<tr>
<td>• Software restarts are acceptable</td>
</tr>
<tr>
<td>• Executes on commercial processing hardware</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Relevant Application Domains</th>
<th>Mission Planning</th>
<th>Enterprise Service Systems</th>
<th>Custom AIS Software</th>
<th>Enterprise Information Systems</th>
</tr>
</thead>
</table>

23 Oct 2014  29th International Forum on COCOMO® and Systems/Software Cost Modeling
Workshop Discussion

- Identify the interaction between “applicable” COCOMO II Scale/Cost Drivers and each Super-Domain

<table>
<thead>
<tr>
<th></th>
<th>RT</th>
<th>ENG</th>
<th>SUP</th>
<th>AIS</th>
</tr>
</thead>
<tbody>
<tr>
<td>PREC</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FLEX</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RESL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TEAM</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PMAT</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RELY</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CPLX</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RUSE</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATA</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>…</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Model Form
COCOMO II Models

- Three models comprise COCOMO II
  - Application Composition
    - Object Points (screens, reports and third-generation language modules)
  - Early Design: Size + 7 predictor variables
    \[
    PM = A \cdot \text{Size}^E \times \prod_{i=1}^{7} EM_i + PM_{Auto}
    \]
  - Post-Architecture: Size +22 predictor variables
    \[
    PM = A \cdot \text{Size}^E \times \prod_{i=1}^{17} EM_i + PM_{Auto}
    \]
    \[
    E = B + 0.01 \times \sum_{j=1}^{5} SF_j
    \]
    \[
    PM_{Auto} = \frac{\text{Adapted SLOC} \times \left(\frac{AT}{100}\right)}{\text{ATPROD}}
    \]
Model Calibration

\[ PM = A \times \text{Size}^E \times \prod EM \]

where \( E = B + 0.01 \times \sum SF \)

A: Multiplicative constant
B: Exponent constant
EM: Effort Multiplier for each Cost Driver
SF: Scale Factor for each Scale Driver

How would a segmentation strategy impact calibration?
• Use pre-sets only?
• Different A and B values?
• Different A, B, EM and SF values?

Should a value for E be derived for each segment?
# Globbing Results

In the Globbing study discussed earlier, COCOMO II constants, A and B, were created for each Glob

<table>
<thead>
<tr>
<th>Size Bucket</th>
<th>Glob-1</th>
<th>Glob-2</th>
<th>Glob-3</th>
<th>Glob-4</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>2-25 KSLOC</strong></td>
<td>n</td>
<td>6</td>
<td>10</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>A</td>
<td>3.53</td>
<td>4.27</td>
<td>2.88</td>
</tr>
<tr>
<td></td>
<td>B</td>
<td>0.87</td>
<td>0.75</td>
<td>0.92</td>
</tr>
<tr>
<td></td>
<td>PRED(25)</td>
<td>83%</td>
<td>80%</td>
<td>100%</td>
</tr>
<tr>
<td></td>
<td>PRED(20)</td>
<td>67%</td>
<td>70%</td>
<td>100%</td>
</tr>
<tr>
<td><strong>25-100 KSLOC</strong></td>
<td>n</td>
<td>13</td>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td>A</td>
<td>7.14</td>
<td>5.42</td>
<td>6.44</td>
</tr>
<tr>
<td></td>
<td>B</td>
<td>0.69</td>
<td>0.76</td>
<td>0.72</td>
</tr>
<tr>
<td></td>
<td>PRED(25)</td>
<td>69%</td>
<td>80%</td>
<td>73%</td>
</tr>
<tr>
<td></td>
<td>PRED(20)</td>
<td>69%</td>
<td>80%</td>
<td>73%</td>
</tr>
<tr>
<td><strong>100+ KSLOC</strong></td>
<td>n</td>
<td>18</td>
<td>8</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>A</td>
<td>4.58</td>
<td>9.52</td>
<td>1.20</td>
</tr>
<tr>
<td></td>
<td>B</td>
<td>0.87</td>
<td>0.73</td>
<td>1.04</td>
</tr>
<tr>
<td></td>
<td>PRED(25)</td>
<td>72%</td>
<td>75%</td>
<td>50%</td>
</tr>
<tr>
<td></td>
<td>PRED(20)</td>
<td>56%</td>
<td>75%</td>
<td>50%</td>
</tr>
</tbody>
</table>
Size Input

\[ PM = A \times Size^E \times \prod EM \]

Size represents the amount of work, i.e. the problem scope

Percentage adjustments to effort-size relationship

Should the size input be limited to only source lines of code? Would the model still work with other well defined and consistently measured size types?
- Requirements
- Unadjusted Function Points
- Use Case Points
- Story Points
- COSYSMO size types
SLOC Sizing Model

Size = \left(1 + \frac{REVL}{100}\right) \times (\text{New KSLOC} + \text{Equivalent KSLOC})

Equivalent KSLOC = \text{Adapted KSLOC} \times \left(1 - \frac{AT}{100}\right) \times \text{AAM}

\begin{align*}
\text{where AAM} &= \begin{cases} 
\frac{AA + AAF \times (1 + [0.02 \times SU \times UNFM])}{100}, & \text{for } AAF \leq 50 \\
\frac{AA + AAF + (SU \times UNFM)}{100}, & \text{for } AAF > 50
\end{cases} \\
AAF &= \left(0.4 \times DM\right) + \left(0.3 \times CM\right) + \left(0.3 \times IM\right)
\end{align*}

Are all of these factors still relevant?  
How are different code types treated (modified, reused, auto-generated)?
Cost & Scale Driver Review
Scale/Cost Driver Interactions

- Pairwise correlation analysis
- Multi-collinearity analysis
- Possible opportunity for consolidation
- COSYSMO parameter harmonization?
Adjusting Scale/Cost Driver Nominal

• Ratings for drivers shifting away from nominal
• See following charts
COCOMO II Data: Process Maturity Trends
Use of Software Tools Rating Trends

Figure 5. Linear Regression of TOOL on Year (p<0.0001)
Platform Experience Rating Trends

Figure 7. Linear regression of PLEX on Year (p<0.0001)
Application Experience Rating Trends
Language & Tool Experience Rating Trends

Figure 9. Linear regression of LTEx on Year (p<0.0001)
Storage Constraint Rating Trends

Figure 10. Linear regression of STOR on Year
Execution Time Constraint Rating Trends

Figure 11. Linear regression of TIME on Year
Shift Definition for PMAT Nominal

<table>
<thead>
<tr>
<th>Scale Factors</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>PMAT</td>
<td>SW-CMM</td>
<td>SW-CMM</td>
<td>SW-CMM</td>
<td>SW-CMM</td>
<td>SW-CMM</td>
<td>SW-CMM</td>
</tr>
<tr>
<td></td>
<td>Level 1</td>
<td>Level 1</td>
<td>Level 2</td>
<td>Level 3</td>
<td>Level 4</td>
<td>Level 5</td>
</tr>
<tr>
<td></td>
<td>Lower</td>
<td>Upper</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The estimated Equivalent Process Maturity Level (EPML) or SW-CMM Level 1 Lower

Solution:
- Shift Nominal label over to the High label
- Shift the definition of High over to Nominal

What problems would this shift cause?
New Scale/Cost Drivers

- Parameters from other COCOMO Suite models
  - CORADMO
  - COQUALMO
  - COINCOMO
  - COPLIMO

- Review drivers for relevance

- Add new S/C drivers for estimating today’s development practices (data permitting)
  - Process Methodology: indicate the type of processes that are planned for the development, e.g. plan-driven, rapid development, architected agile, formal methods, COTS-Intensive, etc

  - See other model parameters handout
COCOMO Suite Parameters

• CORADMO (For smaller/shorter projects)
  – Parameters
    • Very High-level Languages
    • Development Process Reengineering
    • Collaboration Efficiency
    • Prepositioning Assets

• COQUALMO (Impact of rework)
  – Parameter
    • Disciplined Methods
  – Collect defect injection & removal data
COCOMO Suite Parameters

• COPLIMO (relative cost of reuse)
  – Parameters
    • Product-specific fraction of software product’s size (PFRAC)
    • Adapted-software fraction (AFRAC)
    • Reused-software fraction (RFRAC)
    • Relative cost of writing for Reuse (RCWR)

• COINCOMO (Incremental development)
  – Collect data by increments, e.g.
    • Start & end dates
    • Increment size
    • Increment effort
New Driver Candidates

• End-user expertise
  – The degree of usability for an effective end-user experience, e.g. how simple does it have to be