COCOMO 2.0:
Recent Developments

Barry Boehm, USC
COCOMO/SCM Forum 9
October 6, 1994
Outline

• COCOMO 2.0 refresher
  – Objectives
  – Coverage of future software practices
  – Model highlights

• Model refinements

• Usage experience
  – Cost-benefit rationale

• Long-range plan
COCOMO 2.0 Objectives

- Retain COCOMO internal, external openness
- Develop a 1990-2000’s software cost model
  - Addressing full range of market sectors
  - Addressing new processes and practices
- Develop database, tool support for continuous model improvement
- Develop analytic framework for economic analysis
  - E.g., software product line return-on-investment analysis
- Integrate with process, architecture research
The future of the software practices marketplace

User programming
(55M performers in US in year 2005)

- Application generators (0.6M)
- Application composition (0.7M)
- System integration (0.7M)

Infrastructure (0.75M)
COCOMO 2.0 Coverage of Future SW Practices Sectors

- User Programming: No need for cost model
- Applications Composition: Use object counts or object points
  - Count (weight) screens, reports, 3GL routines
- System Integration; development of applications generators and infrastructure software
  - Prototyping: Applications composition model
  - Early design: Function Points and 4-6 cost drivers
  - Post-architecture: Source Statements or Function Points and 14-20 cost drivers
  - Stronger reuse/reengineering model
Baseline Object Point Estimation Procedure

Step 1: Assess Object-Counts: estimate the number of screens, reports, and 3GL components that will comprise this application. Assume the standard definitions of these objects in your ICASE environment.

Step 2: Classify each object instance into simple, medium and difficult complexity levels depending on values of characteristic dimensions. Use the following scheme:

<table>
<thead>
<tr>
<th>Number of Views contained</th>
<th>For Screens</th>
<th>For Reports</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td># and source of data tables</td>
<td># and source of data tables</td>
</tr>
<tr>
<td>Total &lt; 4 (&lt; 2 srvr &lt; 3 clnt)</td>
<td>Total &lt; 8 (&lt; 2 srvr 2-3 clnt)</td>
<td>Total 8+ (&gt; 3 srvr &gt; 3 clnt)</td>
</tr>
<tr>
<td>Total &lt; 8 (&gt; 3 srvr 3-5 clnt)</td>
<td>Total 8+ (&gt; 3 srvr &gt; 5 clnt)</td>
<td></td>
</tr>
<tr>
<td>&lt; 3</td>
<td>simple</td>
<td>simple</td>
</tr>
<tr>
<td>3 - 7</td>
<td>simple</td>
<td>medium</td>
</tr>
<tr>
<td>&gt; 8</td>
<td>medium</td>
<td>difficult</td>
</tr>
</tbody>
</table>

Step 3: Weigh the number in each cell using the following scheme. The weights reflect the relative effort required to implement an instance of that complexity level:

<table>
<thead>
<tr>
<th>Object Type</th>
<th>Complexity-Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Simple</td>
</tr>
<tr>
<td>Screen</td>
<td>1</td>
</tr>
<tr>
<td>Report</td>
<td>2</td>
</tr>
<tr>
<td>3GL Component</td>
<td>10</td>
</tr>
</tbody>
</table>

Step 4: Determine Object-Points: add all the weighted object instances to get one number, the Object-Point count.

Step 5: Estimate percentage of reuse you expect to be achieved in this project. Compute the New Object Points to be developed, \( NOP = (Object-Points) \times (100 - \% \text{reuse}) / 100 \).

Step 6: Determine a productivity rate, \( PROD = NOP / \text{person-month} \), from the following scheme

<table>
<thead>
<tr>
<th>Developers' experience and capability</th>
<th>Very Low</th>
<th>Low</th>
<th>Nominal</th>
<th>High</th>
<th>Very High</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICASE maturity and capability</td>
<td>Very Low</td>
<td>Low</td>
<td>Nominal</td>
<td>High</td>
<td>Very High</td>
</tr>
<tr>
<td></td>
<td>PROD</td>
<td>4</td>
<td>7</td>
<td>13</td>
<td>25</td>
</tr>
</tbody>
</table>

Step 7: Compute the estimated person-months: \( PM = NOP / PROD \).
Nonlinear Reuse Effects

Data on 2954 NASA modules [Selby, 1988]

Relative cost

0.046

0.25

0.5

0.75

1.0

Amount Modified

0.55

0.70

1.0

Usual Linear Assumption
## Prototype of Software Understanding Rating/Increment Table

<table>
<thead>
<tr>
<th></th>
<th>Very Low</th>
<th>Low</th>
<th>Nom</th>
<th>High</th>
<th>Very High</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Structure</strong></td>
<td>Low-cohesion, High-coupling spaghetti code</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>Strong modularity, information hiding, data/control structure</td>
</tr>
<tr>
<td><strong>Application Clarity</strong></td>
<td>No match between program and application world-views</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>Clear match between program and application world-views</td>
</tr>
<tr>
<td><strong>Self-Descriptiveness</strong></td>
<td>Obscure code; Documentation missing, obscure, or obsolete</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>Self-descriptive code; Documentation up-to-date, well-organized, with design rationale</td>
</tr>
<tr>
<td><strong>SU Increment to AAF</strong></td>
<td>50</td>
<td>40</td>
<td>30</td>
<td>20</td>
<td>10</td>
</tr>
</tbody>
</table>
### Table 1: Comparison of COCOMO, Ada COCOMO, and COCOMO 2.0

<table>
<thead>
<tr>
<th></th>
<th>COCOMO</th>
<th>Ada COCOMO</th>
<th>COCOMO 2.0: Stage 1</th>
<th>COCOMO 2.0: Stage 2</th>
<th>COCOMO 2.0: Stage 3</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Size</strong></td>
<td>Delivered Source Instructions (DSI) or Source Lines Of Code (SLOC)</td>
<td>DSI or SLOC</td>
<td>Object Points</td>
<td>Function Points (FP) and Language</td>
<td>FP and Language or SLOC</td>
</tr>
<tr>
<td><strong>Breakage</strong></td>
<td>Requirements Volatility rating: (RVOL)</td>
<td>RVOL rating</td>
<td>Implicit in model</td>
<td>Breakage %: BRAK</td>
<td>BRAK</td>
</tr>
<tr>
<td><strong>Maintenance</strong></td>
<td>Annual Change Traffic (ACT) = %added + %modified</td>
<td>ACT</td>
<td>Object Point ACT</td>
<td>Reuse model</td>
<td>Reuse model</td>
</tr>
<tr>
<td><strong>Scale (b) in</strong> M&lt;sub&gt;NOM&lt;/sub&gt; = a(Size)&lt;sup&gt;b&lt;/sup&gt;</td>
<td>Organic: 1.05, Semidetached: 1.12, Embedded: 1.20</td>
<td>Embedded: 1.04 - 1.24 depending on degree of: * early risk elimination * solid architecture * stable requirements * Ada process maturity</td>
<td>1.02 - 1.26 depending on the degree of: * precededness * conformity * early architecture, risk resolution * team cohesion * process maturity (SEI)</td>
<td>1.02 - 1.26 depending on the degree of: * precededness * conformity * early architecture, risk resolution * team cohesion * process maturity (SEI)</td>
<td></td>
</tr>
<tr>
<td><strong>Product Cost Drivers</strong></td>
<td>RELY, DATA, CPLX</td>
<td>RELY&lt;sup&gt;<em>&lt;/sup&gt;, DATA, CPLX&lt;sup&gt;</em>&lt;/sup&gt;, RUSE</td>
<td>None</td>
<td>RCPX&lt;sup&gt;↑↑&lt;/sup&gt;, RUSE&lt;sup&gt;↑↑&lt;/sup&gt;</td>
<td>RELY, DATA, DOCU&lt;sup&gt;↑↑&lt;/sup&gt; CPLX&lt;sup&gt;↑&lt;/sup&gt;, RUSE&lt;sup&gt;↑↑&lt;/sup&gt;</td>
</tr>
<tr>
<td><strong>Platform Cost Drivers</strong></td>
<td>TIME, STOR, VIRT, TURN</td>
<td>TIME, STOR, VMVH, VMVT, TURN</td>
<td>None</td>
<td>Platform difficulty: PDIF&lt;sup&gt;↑↑&lt;/sup&gt;</td>
<td>TIME, STOR, PVOL(=VIRT)</td>
</tr>
<tr>
<td><strong>Personnel Cost Drivers</strong></td>
<td>ACAP, AEXP, PCAP, VEXP, LEXP</td>
<td>ACAP&lt;sup&gt;<em>&lt;/sup&gt;, AEXP, PCAP&lt;sup&gt;</em>&lt;/sup&gt;, VEXP, LEXP&lt;sup&gt;*&lt;/sup&gt;</td>
<td>None</td>
<td>Personnel capability and experience: PERS&lt;sup&gt;↑↑&lt;/sup&gt;, PREX&lt;sup&gt;↑↑&lt;/sup&gt;</td>
<td>ACAP&lt;sup&gt;<em>&lt;/sup&gt;, AEXP&lt;sup&gt;↑&lt;/sup&gt;, PCAP&lt;sup&gt;</em>&lt;/sup&gt;, PEXP&lt;sup&gt;↑↑&lt;/sup&gt;, LTEX&lt;sup&gt;↑↑&lt;/sup&gt;, PCON&lt;sup&gt;↑↑&lt;/sup&gt;</td>
</tr>
<tr>
<td><strong>Project Cost Drivers</strong></td>
<td>MODP, TOOL, SCED</td>
<td>MODP&lt;sup&gt;<em>&lt;/sup&gt;, TOOL&lt;sup&gt;</em>&lt;/sup&gt;, SCED, SECU</td>
<td>None</td>
<td>SCED, FCIL&lt;sup&gt;↑&lt;/sup&gt;</td>
<td>TOOL&lt;sup&gt;↑&lt;/sup&gt;, SCED, SITE&lt;sup&gt;↑&lt;/sup&gt;</td>
</tr>
</tbody>
</table>

* Different multipliers.  
† Different rating scale.
Outline

- COCOMO 2.0 refresher
  - Objectives
  - Coverage of future software practices
  - Model highlights

=>

- Model refinements

- Usage experience
  - Cost-benefit rationale

- Long-range plan
Model Refinement

- Added cost drivers
- Revised scale driver model
- Integrated reuse/maintenance model
- Initial schedule estimation model
- Output ranges
Added Cost Drivers

- DOCU -- Required Documentation
- PCON -- Personnel Continuity
- SITE -- Multisite development
<table>
<thead>
<tr>
<th>RELY</th>
<th>Low</th>
<th>Nominal</th>
<th>High</th>
<th>Very High</th>
<th>Extra High</th>
</tr>
</thead>
<tbody>
<tr>
<td>slight inconvenience</td>
<td>low, easily recoverable losses</td>
<td>moderate, easily recoverable losses</td>
<td>high financial loss</td>
<td>risk to human life</td>
<td></td>
</tr>
<tr>
<td>DATA</td>
<td>DB bytes/Pgm SLOC &lt; 10</td>
<td>10 ≤ D/P &lt; 100</td>
<td>100 ≤ D/P &lt; 1000</td>
<td>D/P ≥ 1000</td>
<td></td>
</tr>
<tr>
<td>CPLX</td>
<td>see Table 8</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RUSE</td>
<td>none</td>
<td>across project</td>
<td>across program</td>
<td>across product line</td>
<td>across multiple product lines</td>
</tr>
<tr>
<td>DOCU</td>
<td>Many life-cycle needs uncovered</td>
<td>Some life-cycle needs uncovered.</td>
<td>Right-sized to life-cycle needs</td>
<td>Excessive for life-cycle needs</td>
<td>Very excessive for life-cycle needs</td>
</tr>
<tr>
<td>TIME</td>
<td>≤ 50% use of available execution time</td>
<td>70%</td>
<td>85%</td>
<td>95%</td>
<td></td>
</tr>
<tr>
<td>STOR</td>
<td>≤ 50% use of available storage</td>
<td>70%</td>
<td>85%</td>
<td>95%</td>
<td></td>
</tr>
<tr>
<td>PVOL</td>
<td>major change every 12 mo.; minor change every 1 mo.</td>
<td>major: 6 mo.; minor: 2 wk.</td>
<td>major: 2 mo.; minor: 1 wk.</td>
<td>major: 2 wk.; minor: 2 days</td>
<td></td>
</tr>
<tr>
<td>PCON</td>
<td>48% / year</td>
<td>24% / year</td>
<td>12% / year</td>
<td>6% / year</td>
<td>3% / year</td>
</tr>
<tr>
<td>AEXP</td>
<td>≤ 2 months</td>
<td>6 months</td>
<td>1 year</td>
<td>3 years</td>
<td>6 years</td>
</tr>
<tr>
<td>PEXP</td>
<td>≤ 2 months</td>
<td>6 months</td>
<td>1 year</td>
<td>3 years</td>
<td>6 year</td>
</tr>
<tr>
<td>LTEX</td>
<td>≤ 2 months</td>
<td>6 months</td>
<td>1 year</td>
<td>3 years</td>
<td>6 year</td>
</tr>
<tr>
<td>TOOL</td>
<td>edit, code, debug</td>
<td>simple, frontend, backend CASE, little integration</td>
<td>basic lifecycle tools, moderately integrated</td>
<td>strong, mature lifecycle tools, moderately integrated</td>
<td>strong, mature, proactive lifecycle tools, well integrated with processes, methods, reuse</td>
</tr>
<tr>
<td>SITE: Collocation</td>
<td>International</td>
<td>Multi-city and Multi-company</td>
<td>Multi-city or Multi-company</td>
<td>Same city or metro. area</td>
<td>Same building or complex</td>
</tr>
<tr>
<td>SITE: Communications</td>
<td>Some phone, mail</td>
<td>Individual phone, FAX</td>
<td>Narrowband email</td>
<td>Wideband electronic communication.</td>
<td>Wideband elect. comm., occasional video conf.</td>
</tr>
<tr>
<td>SCED</td>
<td>75% of nominal</td>
<td>85%</td>
<td>100%</td>
<td>130%</td>
<td>160%</td>
</tr>
</tbody>
</table>
Table 13: COCOMO 2.0 Effort Multipliers

<table>
<thead>
<tr>
<th>Cost Driver</th>
<th>Rating</th>
<th>Productivity Range</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Very Low</td>
<td>Low</td>
</tr>
<tr>
<td>RELY</td>
<td>0.75</td>
<td>0.88</td>
</tr>
<tr>
<td>DATA</td>
<td>0.94</td>
<td>1.00</td>
</tr>
<tr>
<td>CPLX</td>
<td>0.75</td>
<td>0.88</td>
</tr>
<tr>
<td>RUSE</td>
<td>0.89</td>
<td>1.00</td>
</tr>
<tr>
<td>DOCU</td>
<td>0.85</td>
<td>0.93</td>
</tr>
<tr>
<td>TIME</td>
<td></td>
<td>1.00</td>
</tr>
<tr>
<td>STOR</td>
<td></td>
<td>1.00</td>
</tr>
<tr>
<td>PVOL</td>
<td></td>
<td>0.87</td>
</tr>
<tr>
<td>ACAP</td>
<td>1.50</td>
<td>1.22</td>
</tr>
<tr>
<td>PCAP</td>
<td>1.37</td>
<td>1.16</td>
</tr>
<tr>
<td>PCON</td>
<td>1.26</td>
<td>1.11</td>
</tr>
<tr>
<td>AEXP</td>
<td>1.23</td>
<td>1.10</td>
</tr>
<tr>
<td>PEXP</td>
<td>1.26</td>
<td>1.12</td>
</tr>
<tr>
<td>LTEX</td>
<td>1.24</td>
<td>1.11</td>
</tr>
<tr>
<td>TOOL</td>
<td>1.20</td>
<td>1.10</td>
</tr>
<tr>
<td>SITE</td>
<td>1.24</td>
<td>1.10</td>
</tr>
<tr>
<td>SCED</td>
<td>1.23</td>
<td>1.08</td>
</tr>
</tbody>
</table>

The corresponding combined baseline effort multipliers are given in Table 15. The resulting seven cost drivers are easier to estimate in early stages of software development than the 17 Stage...
Revised Scale Driver Model

- Requirements Volatility factor eliminated
  - Double counting w.r.t Breakage (BRAK)
  - Scale effects reflected via BRAK contribution to size

- New scale equation
  \[ B = 1.01 + 0.01 \times \sum \text{(scale factors)} \]
Integrated Reuse and Maintenance Model

- Replace Annual Change Traffic (ACT) by reuse drivers
  - Incorporates nonlinear small-modification effects
  - Applies scaling equation to modification effects
  - Makes reuse, maintenance models consistent
Other Model Refinements

- Initial schedule estimation model

\[ TDEV = 3.0 \times PM(0.33 + 0.2 \times (B - 1.01)) \]

- Output Ranges
  - One standard deviation optimistic and pessimistic estimates
  - Reflect sources of uncertainty in model inputs (see "funnel chart"):

<table>
<thead>
<tr>
<th>Stage</th>
<th>Optimistic Estimate</th>
<th>Pessimistic Estimate</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.50E</td>
<td>2.0E</td>
</tr>
<tr>
<td>2</td>
<td>0.67E</td>
<td>1.5E</td>
</tr>
<tr>
<td>3</td>
<td>0.80E</td>
<td>1.25E</td>
</tr>
</tbody>
</table>
Software Costing and Sizing
Accuracy vs. Phase
Outline

• COCOMO 2.0 refresher
  – Objectives
  – Coverage of future software practices
  – Model highlights

• Model refinements

⇒ • Usage experience
  – Cost-benefit rationale

• Long-range plan
COCOMO 2.0 Affiliate Usage Experience

Ray Madachy

COCOMO 2.0 Kickoff Meeting
USC Center for Software Engineering
University of Southern California
July 26, 1994
AFFILIATE CONSIDERATIONS

• Relevance to process improvement efforts

• Amadeus® tool learning

• Tool integration with current data collection and cost estimation processes

• Incomplete cost data and inconsistent data granularity - may need to strengthen cost collection procedures

• Data confidentiality, security and legal considerations - check constraints as soon as possible and resolve
PROCESS IMPROVEMENT

• COCOMO 2.0 Affiliate activities are compatible with process improvement via metrics definition and collection, and specific improvement of the cost estimation process.

• To support continuous process improvement:
  - establish baselines by measuring products and processes
  - provide periodic status reports and graphs for feedback and improvement
  - perform comparative analysis, monitor progress relative to baselines and support project tracking
  - analyze and summarize data for cost and schedule estimation
  - periodically update baselines

• Monthly COCOMO data collection and analysis is an implementation of continuous improvement.
COST DRIVER RATING

- Need to rate cost drivers in a consistent and objective fashion within an organization.

- Cost driver ratings profile:
  - graphical depiction of historical ratings to be used as a reference baseline
  - used in conjunction with estimating tools to gauge new projects against past ones objectively
COST DRIVER RATINGS PROFILE

**RELY - required software reliability**
- Very Low: slight inconvenience, low, easily recoverable losses. SLOCS < 10.
- Low: low, easily recoverable losses, 10 ≤ D/P < 100.
- Nominal: easily recoverable losses, 100 ≤ D/P < 1000.
- High: high risk to financial losses, D/P ≥ 1000.
- Very High: recovery possible, life savings.
- Extra High: human life.

**DATA - data base size**
- DB bytes/Prog: 10 ≤ D/P < 100.
- SLOCS: < 10

**CPLX - product complexity**
- See attached table.
PHASED IMPLEMENTATION APPROACH

- **Phase 1** - code analysis for pilot projects, and implemention of import, entry and report templates for project tracking metrics. Generate initial baselines for pilot projects.

- **Phase 2** - build up data repositories and strengthen analysis both within and across projects, providing process improvement insight.

- **Phase 3** - division-wide adoption and standardization, use of networking across repositories. This will provide a stable framework for continuous process improvement across all projects.
Cost-Benefit Rationale

• General COCOMO 2.0 benefits
• Additional COCOMO 2.0 Affiliate benefits
• COCOMO 2.0 Affiliate costs
• Bottom Line
General COCOMO 2.0 Benefits

- Reduced risk of inaccurate estimates
  - Stronger base for planning and control
- More relevant, credible sensitivity analyses
- More accurate technology investment analyses
  - Tools, components, methods, processes
- Acceleration of continuous process improvement
  - Means toward achieving SEI CMM Level 4
- Means toward compliance with Government contract data reporting requirements
Added COCOMO 2.0 Affiliate Benefits

- Early access to general COCOMO 2.0 benefits

- Higher level of general COCOMO 2.0 benefits via specially-calibrated versions

- Comparative benchmarking of project and organizational productivity

- Amadeus, USC COCOMO 2.0, future tools
COCOMO 2.0 Affiliate Costs
(Rough Estimates)

- $15K/year or portion of $25K general USC or UCI affiliate membership
- 1 person-month/site startup effort
- 1 person-day/project initialization effort
- 30 minutes/periodic project data update
  - These vary as a function of existing data collection procedures
- 2 trips/person/year, synchronized with USC Research Review and COCOMO/CSM Forum
CUSTOMIZATION EXAMPLES

- Form template for manual input of COCOMO 2.0 general project data took about 30 minutes.

- Form template and agent to measure source code required less than 30 minutes per project.

- Templates for importing data from metrics spreadsheet took about one hour.

- Templates only need to be developed once and are reused thereafter.

- Templates are platform independent.
ANOTHER PILOT PROJECT EXAMPLE

• **Before**: it took 4-5 weeks for project personnel to provide the SEPG with the size of the source code.
  - what was measured did not conform to the standard counting rules consistent with COCOMO sizing and cost estimation procedures.

• **After**: Amadeus® could be invoked to measure the same code and provide results within seconds.
  - not only are we confident that the measured size conforms to our cost model definition, additional measures are provided to increase visibility.
COST SAVINGS

- Amadeus® saves costs by replacing current manual processes
  - most of the cost of implementing Amadeus® is in planning and the development of custom templates and agents

- Batch mode also an option versus manual invocation

- Break-even point of using automated procedures should occur relatively early
EVALUATION SUMMARY

- Tool automation increases process visibility by providing consistent metrics not previously available.

- Requires less effort over the long run compared to manual procedures.

- Can leverage current processes if carefully planned.

- Amadeus® can serve as a framework for integrating metrics and enable cross-metric analyses.

- Customization requires minimal amount of nominal technical skills.

- Still have open issues for some projects.
**Bottom Line**

- Difficult to compute ROI

- Varies as function of existing data collection and analysis efforts
  - If strong, added effort should be small
  - If weak, added effort should be reasonable

- “If you think measurement is expensive, try ignorance”
Outline

- COCOMO 2.0 refresher
  - Objectives
  - Coverage of future software practices
  - Model highlights

- Model refinements

- Usage experience
  - Cost-benefit rationale

=> • Long-range plan
COCOMO 2.0 Long Range Plan

1995  1996  1997  1998-
Project effort,
schedule IOC Upgrades ------
Phase, activity
distribution IOC Upgrades ------
Cost/quality
model IOC Upgrades -------
Dynamics model,
Sizing model, ...

IOC: Initial Operational Capability
Upgrades: Accuracy, functionality, tailored versions