Resolving Issues in
The COSYSMO 3.0 Model

Jim Alstad, USC-CSSE
USC Center for Systems and Software Engineering
2015 Annual Research Review
April 15, 2015
Minutes (plus a few changes)
Agenda

Agenda:
• Introduction to COSATMO & COSYSMO
• Discussion of Model Features
• Discussion of COSYSMO 1.0 Effort Multipliers (?)
The Problem

- How much will the total system cost?
- Is one phase being optimized while increasing total cost?
- Is the system affordable?
- Does the acquisition comply with the Better Buying Power initiatives (DoD)?
COSATMO Objective

• Context:
  – Current and future trends create challenges for full-system cost estimation
    • Emergent requirements, rapid change, net-centric systems of systems, COTS, clouds, apps, widgets, high assurance with agility, multi-mission systems
  – Current development practices can minimize cost of one phase, such as development, while raising full-system cost

• The COSATMO project is developing a modern full-system cost model (first space systems, then other DoD domains)
  – “Constructive SATellite cost MOdel”
  – Current estimating models focus on one aspect, such as system engineering
  – COSATMO will enable:
    • System-level trades to be handled within a single model
    • Easy customer evaluation of full-system cost
    • Modern technologies to be covered
COSATMO as a Research Umbrella

• General direction:
  – Develop a full-coverage satellite system cost estimating model
  – Generalize that to additional applications

Specific current research initiatives:
  – COSYSMO 3.0
  – COCOMO III

Research vehicles:
  – My thesis
  – Other theses
  – Other research
Software Cost Models

- COCOMO 81 1981
- COCOMO II 2000
- DBA COCOMO 2004
- COINCOMO 2004,2012

Other Independent Estimation Models

- COCOTS 2000
- COSYSMO 2005
- COSoSIMO 2007

Software Extensions

- COQUALMO 1998
- iDAVE 2004
- iDAVE 2004
- COPLIMO 2003
- COPSEMO 1998
- CORADMO 1999,2012

Legend:
- Model has been calibrated with historical project data and expert (Delphi) data
- Model is derived from COCOMO II
- Model has been calibrated with expert (Delphi) data

Dates indicate the time that the first paper was published for the model
History of COSYSMO Models

COSYSMO 1.0
Valerdi, 2005
- Identifies form of model
- Identifies basic cost drivers
- Identifies Size measure

With Reuse
Fortune, 2009
- Adds weights to Size elements, reducing net Size in the presence of reuse

Req’ts Volatile
Pena, 2012
- Adds scale factor based on requirements volatility

For Reuse
Wang et al, 2014
- Adds weights to Size elements, reducing net Size when artifacts are only partially completed

Sys of Sys
Lane et al, 2014
- Adds effort multiplier when in the presence of system-of-systems

COSYSMO 3.0
Alstad, 20xx
- Combines features of previous models
COSYSMO 3.0 Directions
(Adapted from ARR slides [8])

Harmonize existing COSYSMO family models:

- Several factors affecting the COSYSMO cost model have been shown to be valuable in increasing estimation accuracy (terminology from [5]):
  - Reuse (simple model--SEWR) [3]
  - Reuse (with SEFR) [1]
  - Requirements volatility (SERV) [4]

The rating scales for these could be integrated into a comprehensive COSYSMO model.

Enhancement planned for inclusion:

- System-of-system considerations are hypothesized to affect system engineering costs:
  - Interoperability considerations [6]
COSYSMO 3.0 Directions

Part 2

Enhancements under discussion:

- Explore a model for total development cost based primarily on the COSYSMO parameters [7]
Harmonized COSYSMO 3.0 Top-Level Model

\[ PM_{C3} = A_{C3} \cdot (Size_{C3})^{E_{C3}} \cdot \prod_{j=1}^{14+} EM_{C3,j} \]

Elements of the Harmonized COSYSMO 3.0 model:

- **Calibration parameter A**
- **Interoperability**
- **Size model**
  - eReq submodel
  - Partial development submodel
- **Exponent (E) model**
  - SF submodel
  - REVL submodel
- **Effort multipliers EM**
  - 14 unchanged EMs
  - SEFR
  - Interoperability
- **Multi-subproject model**
Non-Controversial Model Features

These features have been proposed (Boehm, “Suggested Data-Driven Reorganization”), and were once believed to be non-controversial (see “Discussed” slides below when marked with “*”):

1. Use COSYSMO 1.0 as a starting point
   - Don’t change anything except as noted below
   - Its effort multipliers are grouped into*: 
     • Understanding Factors
     • Complexity Factors
     • Operations Factors
     • People Factors
     • Environment Factors

2. Drop the Documentation Match to Life Cycle Needs effort multiplier (EM) from Complexity Factors

3. Regroup the Personnel Experience/Continuity EM under People Factors
Non-Controversial Model Features

Non-controversial (con’t):

4. Handle partial development in some fashion
   – I.e., developing artifacts through only part of the life cycle
   – This is the key part of SEWR and can be part of SEFR
     • From the Generalized Reuse Framework [1]
   – However, how to handle this is controversial

5. Address interoperability per [6]*
   – I.e., model two approaches, then drop one
   – As EM put it under Operations Factors

6. Add CONOPS Understanding as an EM (under Operations Factors)*

7. Add Requirements Volatility as a scale factor (per [9])*

8. Move Process Capability to be a scale factor*

9. Add Architecture and Risk Resolution as a scale factor*

10. Don’t add Schedule Compression
Non-Controversial Model Features

Non-controversial (con’t):
11. Multi-subsystem development*
Model Features Discussed (1/2)

A. Names of EM Groups - Open

B. Handling of RV, beyond scale factor
   i. Add size impact, depending on timing of volatility
   ii. Agree to set Requirements Understanding to Nominal when using scale factor
      i. Gan’s view: needs more thought; usability of model & tools needs to be considered
      ii. Garry: If bad RU drives RV, count just RV. But if other issues are driving RV, then count both. (Put this in guidance.) - Resolution

C. Put CONOPS U under Understanding Factors. - Resolution

D. On I/operability, make sure definitions are consistent w/ [6], without bringing in non-SoS concerns. - Jim to work. Also, Gan to think about this further. - Gan to work

E. For Process Capability, consider it both ways (EM, scale factor) - Resolution
F. “Risk and Opportunity Resolution” - *Resolution*

G. On Multi-Subsystems, Gan concerns:
   i. Isn’t this the same as the hierarchy for any system; if so, why is a special equation needed? The equations seems to be saying that syseng costs from lower-level systems are to be included with top-level “system of interest”) syseng costs, thereby mudding the waters. - Gan’s concern open
1. Can EMRs and EM values from previous models be used without further calibration?
Model Features for Discussion (2/4)

2. How is the “partial” aspect of partial development to be included:
   - As a modification to size?
   - As an EM?
Model Features for Discussion (3/4)

3. How is the extra cost due to SEFR to be included:
   – As a modification to size, perhaps bundled with the partial development approach?
   – As an EM?
4. How to handle the impact-of-a-step-is-too-large problem (stated in [11])
   – Through a geometric mean approach, per [11]?
   – Through enhanced calibration approaches?
Bibliography (1/2)


9. "Quantifying the Impact of Requirements Volatility on Systems Engineering Effort” (presentation), Mauricio Peña, Ricardo Valerdi, October 18, 2012 (COCOMO Forum)


COSYSMO 3.0 Directions
(Adapted from ARR slides [8])

Harmonize existing COSYSMO family models:

• Several factors affecting the COSYSMO cost model have been shown to be valuable in increasing estimation accuracy (terminology from [5]):
  – Reuse (simple model--SEWR) [3]
  – Reuse (with SEFR) [1]
  – Requirements volatility (SERV) [4]

The rating scales for these could be integrated into a comprehensive COSYSMO model.

Enhancement planned for inclusion:

• System-of-system considerations are hypothesized to affect system engineering costs:
  – Interoperability considerations [6]
Enhancements under discussion:

• Explore a model for total development cost based primarily on the COSYSMO parameters (Cole)
• Reduce the number of Effort Multipliers (Roedler)
Elements of the Harmonized COSYSMO 3.0 model:

- **Calibration parameter** $A$
- Interoperability
- Size model
  - eReq submodel
  - Partial development submodel
- Exponent (E) model
  - SF submodel
  - REVL submodel
- Effort multipliers EM
  - 14 unchanged EMs
  - SEFR
  - Interoperability
- Multi-subproject model
Harmonized COSYSMO 3.0 Interoperability Model

• Lane & Valerdi [6] propose that interoperability be considered a cost influence in the COSYSMO family

• Motivation: if a system is part of a system-of-systems, then that context is reflected in interoperability requirements on the system

• Two ways this influence could be manifested are proposed:
  – Method 1: Add a new effort multiplier
  – Method 2: Adjust the easy/medium/difficult rating scale for system interfaces (part of the Size model)

• Both Methods are shown in this presentation; presumably only one would be retained in COSYSMO 3.0.
Harmonized COSYSMO 3.0 Size Model

\[ \text{Size}_{c3} = \sum_{\text{Prods}} \text{eReq}(\text{Type}(\text{Prod}), \text{Difficulty}(\text{Prod})) \cdot \text{PartialDevFactor}(\text{Phase}_{\text{Start}}(\text{Prod}), \text{Phase}_{\text{End}}(\text{Prod})) \]

- Prod is one of the four system engineering products that determines size in COSYSMO family (per [2]):
  - System requirement
  - System interface
  - System algorithm
  - Operational scenario
- For simplicity in model explanation, each individual Prod is considered separately
- There are two submodels:
  - Equivalent nominal requirements (“eReq”)
  - Partial development
Size Model –
eReq Submodel (1/2)

• The eReq submodel is unchanged from [2].
  – Though terminology is a little different
  – Also, see next slide

• The submodel computes the size of a Prod, in units of eReq ("equivalent nominal requirements")

• Each Prod is evaluated as being easy, nominal, or difficult.

• Each Prod is looked up in this size table to get its number of eReq:

<table>
<thead>
<tr>
<th>Prod Type</th>
<th>Easy</th>
<th>Nominal</th>
<th>Difficult</th>
</tr>
</thead>
<tbody>
<tr>
<td>System Requirement</td>
<td>0.5</td>
<td>1.0</td>
<td>5.0</td>
</tr>
<tr>
<td>System Interface</td>
<td>1.1</td>
<td>2.8</td>
<td>6.3</td>
</tr>
<tr>
<td>System Algorithm</td>
<td>2.2</td>
<td>4.1</td>
<td>11.5</td>
</tr>
<tr>
<td>Operational Scenario</td>
<td>6.2</td>
<td>14.4</td>
<td>30.0</td>
</tr>
</tbody>
</table>
Adjustment for interoperability (Method 2):

- [6] proposes (in its Table 3) that the table that defines the easy/medium/hard rating scale for a system interface (from [2]) be adjusted by adding a new row (the last row in this table):

<table>
<thead>
<tr>
<th>Easy</th>
<th>Medium</th>
<th>Difficult</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simple messages and protocols</td>
<td>Moderate communication complexity</td>
<td>Complex protocol(s)</td>
</tr>
<tr>
<td>Uncoupled</td>
<td>Loosely coupled</td>
<td>Tightly coupled</td>
</tr>
<tr>
<td>Strong consensus among stakeholders</td>
<td>Moderate consensus among stakeholders</td>
<td>Low consensus among stakeholders</td>
</tr>
<tr>
<td>Well behaved</td>
<td>Predictable behavior</td>
<td>Emergent behavior</td>
</tr>
<tr>
<td>Domain or enterprise standards employed</td>
<td>Functional standards employed</td>
<td>Isolated or connected systems with few or no standards</td>
</tr>
</tbody>
</table>
Partial Development Submodel (1/2)

- The following new partial development submodel is proposed
  - It is a generalization of the Generalized Reuse Framework model [1] (Generalized\(^2\) Reuse Framework?)
  - An alternate submodel is discussed below

\[
\text{PartialDevFactor}(\text{Phase}_{\text{Start}}(\text{Prod}), \text{Phase}_{\text{End}}(\text{Prod}))
\]

- A Partial Development Factor from 0.0 to 1.0 is assigned based on the starting and ending phase of development on Prod
  - The phases: Requirements and Architecture, Detailed Design, Implementation, Integration & Test, Operational Test & Evaluation
  - A Prod may enter late, typically because it’s being reused
  - A Prod may exit early, typically because this is an IR&D project

- \(\text{PartialDevFactor} = 1.0\) for a complete development life cycle
Size Model – Partial Development Submodel (2/2)

- \( \text{PartialDevFactor}(\text{Phase}_{\text{Start}}, \text{Phase}_{\text{End}}) \) might be approximated by:

\[
\sum_{p=\text{Phase}_{\text{Start}}}^{\text{Phase}_{\text{End}}} \text{Fraction of development cost for phase } p
\]

- But there might be some overhead for early entry or early exit

Alternate \( \text{PartialDevFactor} \) submodel:
- Use results from Generalized Reuse Model [1]
  - Restricted to Prods that either start at the beginning or finish at the end
- Would be employed if data didn’t validate the proposed model
Harmonized COSYSMO 3.0

Exponent Model

• Exponent model is unchanged from Peña [4, 9]

\[ E_{C3} = E_{COSYSMO} + SF_{RV} \]

• \( E_{COSYSMO} = 1.06 \) [2]
• \( SF_{RV} \) per next slide
Exponent – SF Submodel

\[ SF_{RV} = \left( C \cdot \left( \frac{REVL}{100} \right) \cdot w_{vl} \right) \]

- Where
  - \( REVL \) = The % of the baseline requirements that is expected to change over the system lifecycle – see next slide
  - \( C \) = Scale factor constant = 0.05 (calibration parameter)
  - \( w_{vl} \) = aggregate lifecycle phase volatility weighting factor

- And:
  - \( w_{vl} = (w_{concep} \Theta_{concep} + w_{dev} \Theta_{dev} + w_{test} \Theta_{test} + w_{tran\_ops} \Theta_{tran\_ops}) \)

- \( w_i \) = weighting factor for each life cycle phase\(^1\) (determined by experts [4])
- \( \Theta_i \) = % of total requirements changes per life cycle phase (taken from data from 25 projects [4])
- \( l \) = life cycle phases

\(^1\)Life Cycle Phases: Conceptualize, Development, Operational Test and Evaluation, and Transition to Operation
Exponent – REVL Submodel – Ranking

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Very Low</th>
<th>Low</th>
<th>Moderate</th>
<th>High</th>
<th>Very High</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>System requirements baselined and agreed to by key stakeholders</td>
<td>Fully 1</td>
<td>Mostly 2</td>
<td>Generally 3</td>
<td>Somewhat 4</td>
<td>No Agreement 5</td>
<td>26%</td>
</tr>
<tr>
<td>Level of uncertainty in key customer requirements, mission objectives, and stakeholder needs</td>
<td>Very Low 1</td>
<td>Low 2</td>
<td>Moderate 3</td>
<td>High 4</td>
<td>Very High 5</td>
<td>22%</td>
</tr>
<tr>
<td>Number of co-dependent systems with influence on system requirements</td>
<td>Very Low 1</td>
<td>Low 2</td>
<td>Moderate 3</td>
<td>High 4</td>
<td>Very High 5</td>
<td>16%</td>
</tr>
<tr>
<td>Strength of your organization’s requirements development process and level of change control rigor</td>
<td>Very High 1</td>
<td>High 2</td>
<td>Moderate 3</td>
<td>Low 4</td>
<td>Very Low 5</td>
<td>8%</td>
</tr>
<tr>
<td>Precededness of the system, use of mature technology</td>
<td>Very High 1</td>
<td>High 2</td>
<td>Moderate 3</td>
<td>Low 4</td>
<td>Very Low 5</td>
<td>9%</td>
</tr>
<tr>
<td>Stability of stakeholders’ organizations (developer, customer)</td>
<td>Very High 1</td>
<td>High 2</td>
<td>Moderate 3</td>
<td>Low 4</td>
<td>Very Low 5</td>
<td>14%</td>
</tr>
<tr>
<td>Experience level of the systems engineering team in requirements analysis and development</td>
<td>Very High 1</td>
<td>High 2</td>
<td>Moderate 3</td>
<td>Low 4</td>
<td>Very Low 5</td>
<td>6%</td>
</tr>
</tbody>
</table>

Have been modified a little in Harmonized COSYSMO 3.0.

Developed based on surveys of experienced S/W and Systems Engineers (N =38) [4]
Exponent –

REVL – Ranking to Percentage

- Based on data from 25 projects [4]
Harmonized COSYSMO 3.0 Effort Multiplier Model (1/3)

- 14 effort multipliers unchanged from COSYSMO 1.0 (Table 16 of [2]):

<table>
<thead>
<tr>
<th>Driver Name</th>
<th>Data Item</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requirements understanding</td>
<td>Subjective assessment of the system requirements</td>
</tr>
<tr>
<td>Architecture understanding</td>
<td>Subjective assessment of the system architecture</td>
</tr>
<tr>
<td>Level of service requirements</td>
<td>Subjective difficulty of satisfying the key performance parameters</td>
</tr>
<tr>
<td>Migration complexity</td>
<td>Influence of legacy system (if applicable)</td>
</tr>
<tr>
<td>Technology risk</td>
<td>Maturity, readiness, and obsolescence of technology</td>
</tr>
<tr>
<td>Documentation to match life cycle needs</td>
<td>Breadth and depth of required documentation</td>
</tr>
<tr>
<td># and Diversity of installations/ platforms</td>
<td>Sites, installations, operating environment, and diverse platforms</td>
</tr>
<tr>
<td># of Recursive levels in the design</td>
<td>Number of applicable levels of the Work Breakdown Structure</td>
</tr>
<tr>
<td>Stakeholder team cohesion</td>
<td>Subjective assessment of all stakeholders</td>
</tr>
<tr>
<td>Personnel/team capability</td>
<td>Subjective assessment of the team’s intellectual capability</td>
</tr>
<tr>
<td>Personnel experience/continuity</td>
<td>Subjective assessment of staff consistency</td>
</tr>
<tr>
<td>Process capability</td>
<td>CMMI level or equivalent rating</td>
</tr>
<tr>
<td>Multisite coordination</td>
<td>Location of stakeholders and coordination barriers</td>
</tr>
<tr>
<td>Tool support</td>
<td>Subjective assessment of SE tools</td>
</tr>
</tbody>
</table>

- Rating levels and rating scales unchanged
Harmonized COSYSMO 3.0 Effort Multiplier Model (2/3)

• A new, 15th effort multiplier is “System Engineering for Reuse (SEFR)”
  – I.e., is the project developing intermediate and final system engineering results to be reused on later projects?
    • Reuse for product line is one example
  – Inspired by [1]

• Assumes there is an added cost for SEFR

• Starting point for rating scale (as suggested by Boehm) is COCOMO II RUSE:
  – Low: Not for reuse
  – Nominal: Reused across project
  – High: Reused across program
  – Very High: Reused across product line
  – Extra High: Reused across multiple product lines
Adjustment for interoperability (Method 1):
• “Interoperability” might be a new, 16th effort multiplier
• Table 2 of [6] proposes this rating scale, depending on whether the project is for an existing system or a new system:

<table>
<thead>
<tr>
<th>Type of Development</th>
<th>Level</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Very Low</td>
<td>Low</td>
<td>Nominal</td>
<td>High</td>
<td>Very High</td>
</tr>
<tr>
<td>Existing systems</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(based upon LISI</td>
<td>Isolated</td>
<td>Connected</td>
<td>Functional</td>
<td>Domain</td>
<td>Enterprise</td>
</tr>
<tr>
<td>levels)</td>
<td></td>
<td></td>
<td>standards</td>
<td>standards</td>
<td>standards</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>employed</td>
<td>employed</td>
<td>employed</td>
</tr>
<tr>
<td>New system (s)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(based upon LCIM</td>
<td>System-specific</td>
<td>Documented</td>
<td>Aligned</td>
<td>Aligned</td>
<td>Harmonized</td>
</tr>
<tr>
<td>conceptual levels)</td>
<td>data</td>
<td>data</td>
<td>static data</td>
<td>dynamic data</td>
<td>data</td>
</tr>
</tbody>
</table>
Harmonized COSYSMO 3.0
Multi-Subproject Model

• Sometimes a project consists of multiple subprojects
  – Where the subprojects use significantly different effort multipliers.
  – However, scale factors should apply to the project as a whole.

• Example:
  – Part of a project is SEFR; the rest is not

• The equation below is adapted from Equation 2 of [10] and is based on the Multiple Modules model of COCOMO II
  – When applicable, it supersedes the Top-Level Model

\[ PM_{C3M} = A_{C3} \cdot (Total Size_{C3})^E_{C3} \cdot \sum_{s \in \text{Subprojects}} \left( \frac{\text{Subproject}_s \cdot Size_{C3}}{Total Size_{C3}} \cdot \prod_{j=1}^{14+} EM_{C3:s,j} \right) \]
The Solution

COSATMO assists acquirers and developers during these phases (highest payoff during early phases)

COSATMO estimates the cost for these phases
COSATMO Concept

- Focused on current and future satellite systems
  - Accommodating rapid change, evolutionary development, Net-Centric SoSs, Families of systems, DI2E SWASe’s
    - Software, Widgets, Assets, Services, etc.
  - Recognizes new draft DoDI 5000.02 process models
    - Hardware-intensive, DoD-unique SW-intensive, Incremental SW-intensive, Accelerated acquisition, 2 Hybrids (HW-, SW-dominant)
  - Supports affordability analyses (total cost of ownership):
    - Covers full life cycle: definition, development, production, operations, support, phaseout
    - Covers full system: satellite(s), ground systems, launch
    - Covers hardware, software, personnel costs

- Extensions to cover systems of systems, families of systems
- Several PhD dissertations involved (as with COSYSMO)
  - Incrementally developed based on priority, data availability
Approach

• Technical approach:
  – Develop a satellite system cost model
    • Divide overall system cost into segments. For each segment:
      – Identify an existing cost model (one or more) that covers it, or
      – Develop a new cost model for the segment
    • For any new cost models, follow the well-developed COCOMO-family methodology:
      – Identify cost drivers
      – Obtain expert opinion on impact of cost driver
      – Combine that statistically with cost data from actual systems
      – Iterate as needed
    – Generalize to other DoD systems

• The near-term activities, then, are:
  – Convene groups of experts to identify cost drivers and impacts
  – Identify sources of data
Near-Term Work Approach

• Developing a segment model typically consists of two topics (which are somewhat independent):
  1. Identifying cost drivers and determining which are most important (compare slides 9-11)
  2. Gathering actual, total segment costs for multiple systems, including actual values of cost driver
     – After 1 & 2 are complete, data can be analyzed and the segment cost model can be finalized

• Segments (see slides 7-8) that seem to have the highest benefit/cost ratio for near-term work on either or both topics:
  – Total engineering cost (all through EMD phase—slides 3, 7)
  – Operation & support
  – Other ground segments
Segments of Satellite System Cost

- Total satellite system cost [tied to slide 3 phases] =
  System engineering cost [EMD]
  + Satellite software cost [EMD]
  + Satellite vehicle hardware development [EMD] and production [Prod] cost
  + Launch cost [Deploy]
  + Initial ground software cost [EMD]
  + Initial ground custom equipment cost [EMD]
  + Initial ground facility (buildings, communications, computers, COTS software) cost [EMD]
  + Operation & support cost [Deploy, O&S]

- Updated at GSAW (Feb 2014)
- Model as sum of submodels is new structure in COCOMO family
COSATMO Segment Tentative Models

- System engineering: COSYSMO, perhaps with add-ons
- Satellite vehicle hardware development and production: Current Aerospace hardware cost model(s); exploring extensions of COSYSMO for hardware cost estimation
- Satellite vehicle, ground system software development: COCOMO II, COCOTS, perhaps with add-ons
- Launch model: similarity model, based on vehicle mass, size, orbit
- Ground system equipment, supplies: construction, unit-cost, services cost models
- Operation & support: labor-grade-based cost models, software maintenance models
Key Overall Satellite System Cost Drivers

• Most Important:
  – Complexity, Architecture Understanding, Mass, Payload TRL level/Technology Risk, and Requirements Understanding.

• Important:
  – Reliability, Pointing Accuracy, Number of Deployables, Number of Key Sponsors, Data Rate, and Security Requirements for Communications.

• Determined at COCOMO Forum (Oct 2013)
Ground System Segment Development (1/2)

- Determined at GSAW (Feb 2014)
- Ground system-wide cost drivers
  - Most important: Accreditation (information assurance, etc), Required security
  - Also important: # satellites*
- Initial software cost drivers
  - Required data throughput
  - Generally handled by COCOMO II, COCOTS, COPLIMO

*Indicates a size measure
Ground System Segment
Development (2/2)

• Ground custom equipment cost drivers
  – Most important: Amount of new development required, # of
custom equipment sites*, Required site availability &
reliability, Required site security
  – Also important: # driving requirements*

• Ground facility cost drivers
  – Most important: # facilities*, location of facilities (especially
US vs foreign), # ground RF terminals*
  – Also important: Facility “reuse”

• Operation and support cost drivers
  – Most important: # years of operation*, # FTE staff (with
labor mix)*
  – Also important: Size of software maintained*, Leased line
cost*, level of automation

*Indicates a size measure
Medium-Term Issues

Model-changing issues:
1. Use of small satellites vs more traditional satellites vs mixed
2. Ownership model (own vs leased services, etc)
3. Is support for multiple missions required?

Develop a phased cost model.

Is this a reasonable generalization to other domains:

• Total system cost =
  System engineering cost
  + Embedded software cost
  + Hardware development cost through first article
  + Deployment cost
  + Initial logistics software cost
  + Initial logistics custom equipment cost
  + Initial logistics facility cost
  + Operation & support cost?
Generalized Reuse Framework*
Top Level Part 1

• The Generalized (Systems Engineering) Reuse Framework extends the COSYSMO family of cost estimating models to account for the influence of reusing system engineering artifacts and developing them for such reuse

• Under this model, all system engineering effort falls under one of these types:
  – Development with Reuse
  – Development for Reuse

*Material in this section is taken from [1].
Development for Reuse produces artifacts intended for later reuse on projects. A completed DFR artifact may (intentionally) not be completely developed, so that it will be in one of these DFR states:

- Conceptualized for Reuse (e.g., Concept of Operations document)
- Designed for Reuse (e.g., component detailed design)
- Constructed for Reuse (e.g., integrated component)
- Validated for Reuse (e.g., validated component)
Generalized Reuse Framework: Development with Reuse

• Development with Reuse is project development, with reusable artifacts being brought into the product
  – A special case: zero reusable artifacts

• Each reusable artifact is included in one of these DWR states of maturity:
  – New (i.e., not reused)
  – Re-implemented (through requirements & architecture)
  – Adapted (through detailed design)
  – Adopted (through implementation)
  – Managed (through system verification & validation)
Generalized Reuse Framework: Top Level Part 2

• A system engineering project to be estimated will consist of these types of effort:
  – Development with Reuse; or
  – Development for Reuse; or
  – Both, with the DFR effort typically producing some artifacts for use in the DWR effort.

• A project’s estimated total system engineering effort, then, is estimated as:
  – Estimated DFR effort + estimated DWR effort

• DFR effort is estimated via an extended COSYSMO model
  – DWR effort, likewise
Generalized Reuse Framework: COSYSMO (1/2)

- COSYSMO [2] starts by computing the “size” of a system engineering project, in units of eReq (“equivalent nominal requirements”)
- These artifacts are considered in the size: system requirements, system interfaces, system-critical algorithms, and operational scenarios.
- Each artifact is evaluated as being easy, nominal, or difficult.
- Each artifact is looked up in this size table to get its number of eReq, and then these are summed to get the system size:

<table>
<thead>
<tr>
<th>Artifact Type</th>
<th>Easy</th>
<th>Nominal</th>
<th>Difficult</th>
</tr>
</thead>
<tbody>
<tr>
<td>System Req’ts</td>
<td>0.5</td>
<td>1.0</td>
<td>5.0</td>
</tr>
<tr>
<td>System Interfaces</td>
<td>1.1</td>
<td>2.8</td>
<td>6.3</td>
</tr>
<tr>
<td>System Algs</td>
<td>2.2</td>
<td>4.1</td>
<td>11.5</td>
</tr>
<tr>
<td>Op Scenarios</td>
<td>6.2</td>
<td>14.4</td>
<td>30.0</td>
</tr>
</tbody>
</table>
Generalized Reuse Framework

COSYSMO (2/2)

\[ Size_{COSYSMO} = \sum_{\text{artifacts}} \text{size(art type, art difficulty)} \]

- Size is raised to an exponent, representing diseconomy of scale, and then multiplied by factors for 14 effort multipliers and a calibration constant.
- This results in the following equation for a COSYSMO estimate of effort in person-months:

\[ PM_{COSYSMO} = A \cdot (Size_{COSYSMO})^E \cdot \prod_{j=1}^{14} EM_j \]
Generalized Reuse Framework: DFR Model Equations

- A DFR estimate adjusts each artifact’s size contribution by considering its DFR state according to this table:

<table>
<thead>
<tr>
<th>DFR State (Degree of Development)</th>
<th>DFR State Factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conceptualized for Reuse</td>
<td>36.98%</td>
</tr>
<tr>
<td>Designed for Reuse</td>
<td>58.02%</td>
</tr>
<tr>
<td>Constructed for Reuse</td>
<td>79.15%</td>
</tr>
<tr>
<td>Validated for Reuse</td>
<td>94.74%</td>
</tr>
</tbody>
</table>

\[
Size_{DFR} = \sum_{\text{artifacts}} \text{size(art type, art difficulty)} \cdot \text{DFRStateFactor(art state)}
\]

\[
PM_{DFR} = A_{DFR} \cdot (Size_{DFR})^{E_{DFR}} \cdot \prod_{j=1}^{14} EM_{DFRj}
\]
Generalized Reuse Framework: DWR Model Equations

• A DWR estimate adjusts each artifact’s size contribution by considering its DWR state according to this table:

<table>
<thead>
<tr>
<th>DWR State (Maturity)</th>
<th>DWR State Factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>New</td>
<td>100.00%</td>
</tr>
<tr>
<td>Re-Implemented</td>
<td>66.73%</td>
</tr>
<tr>
<td>Adapted</td>
<td>56.27%</td>
</tr>
<tr>
<td>Adopted</td>
<td>38.80%</td>
</tr>
<tr>
<td>Managed</td>
<td>21.70%</td>
</tr>
</tbody>
</table>

\[
\text{Size}_{DWR} = \sum_{\text{artifacts}} \text{size(art type, art difficulty)} \cdot \text{DWRStateFactor(art state)}
\]

\[
PM_{DWR} = A_{DWR} \cdot (\text{Size}_{DWR})^{E_{DWR}} \cdot \prod_{j=1}^{14} EM_{DWR_j}
\]
COSATMO/COSYSMO Generalized Reuse Framework Topic

• Can model be generalized/simplified by just looking at which phases of development an artifact needs to be put through? (Alstad)
  – I.e., just develop a per-phase cost model
    • Presumably separate parameters for DFR & DWR
  – Would need a common set of phases for DFR & DWR.
  – Would remove restrictions that DFR development always starts from scratch and that DWR development always goes to product completion.
Summary of 2013 Meetings

• 24 September at Aerospace
  – Presentations on satellite cost estimation
    • Notably, Lisa Colabella’s survey of cost data gathering for Operations & Support (see backup chart)

• 24 October at COCOMO Forum
  – Started official COSATMO modeling effort
  – Got 1st draft of most important cost drivers, list of experts

• 18 November at JPL
  – Presentations on their satellite cost models, including some operations modeling

• 18 December at SMC
  – Obtained pointers to some of their operation & support data
Summary of 2014 Meetings

• 26 February at Ground Systems Arch. Workshop
  – Obtained segments, cost drivers for ground systems

• 19 March at Annual SERC Technical Review
  – Presented status

• 9 April at BAE Systems
  – Private meeting on directions for COSYSMO 3.0

• 29 April at CSSE Annual Research Review
  – General coverage
  – Detailed discussion on directions for COSYSMO 3.0
  – Kick off COSYSMO 3.0 Working Group

• 29 July at CMU
  – SERC RT-113/RT-119 meeting
  – Overview of COSATMO
COCOMO Family of Cost Models

Software Cost Models

- COCOMO 81 1981
- COCOMO II 2000
  - DBA COCOMO 2004
  - COINCOMO 2004, 2012
- COQUALMO 1998
- iDAVE 2004
- COPLIMO 2003
- COPSEMO 1998
- CORADMO 1999, 2012

Other Independent Estimation Models

- COCOTS 2000
- COSYSMO 2005
  - COSoSIMO 2007

Software Extensions

- AGILE C II 2003
- COTIPMO 2011
- COPROMO 1998

Legend:
Model has been calibrated with historical project data and expert (Delphi) data
Model is derived from COCOMO II
Model has been calibrated with expert (Delphi) data

Dates indicate the time that the first paper was published for the model
My Tentative Research Objectives

- Provide improved cost estimation capabilities for the portions of and changing needs of space systems that are most needed and most currently tractable, including availability of calibration data. For example, SMC's main current concern is better estimation of post-deployment operations and sustainment costs.

- Develop a framework of cost estimation methods best suited for the various aspects of current and future space systems and other domains, such as the use of unit costing for production, acquisition, and consumables costs, and the use of activity-based costing for operations and sustainment labor costs.

- Prioritize the backlog of estimation models to be developed next.
USC-CSSE Modeling Methodology
- concurrency and feedback implied

- Determine Model Needs
- Analyze existing literature
- Perform Behavioral analyses
- Define relative significance, data, ratings
- Perform expert-judgment Delphi assessment, formulate a priori model
- Gather project data
- Determine Bayesian A-Posteriori model
- Gather more data; refine model
- Concurrency and feedback implied
Current and Future Estimation Challenges

- Emergent requirements
  - Cannot prespecify requirements, cost, schedule, EVMS
  - Need to estimate and track early concurrent engineering
- Rapid change
  - Long acquisition cycles breed obsolescence
  - Need better models for incremental development
- Net-centric systems of systems
  - Incomplete visibility and control of elements
- Model, COTS, service-based, Brownfield systems
  - New phenomenology, counting rules
- Major concerns with affordability
  - Multi-mission ground system challenges
Rapid Change Creates a Late Cone of Uncertainty
– Need evolutionary/incremental vs. one-shot development

Uncertainties in competition, technology, organizations, mission priorities

Phases and Milestones:
- Feasibility
- Plans and Rqts.
- Product Design
- Detail Design
- Devel. and Test
- Accepted Software

Relative Cost Range
- 4x
- 2x
- 1.5x
- 1.25x
- 0.8x
- 0.67x
- 0.5x
- 0.25x
Multi-Mission Ground Systems Costing

• **Product Line Engineering**
  – Identify multi-mission commonalities and variabilities
  – Identify fully, partially sharable commonalities
  – Develop plug-compatible interfaces for variabilities

• **Product Line Costing (COPLIMO) Parameters**
  – Fractions of system fully reusable, partially reusable and cost of developing them for reuse
  – Fraction of system variabilities and cost of development
  – System lifetime and rates of change

• **Product Line Life Cycle Challenges**
  – Layered services vs. functional hierarchy
  – Modularization around sources of change
  – Version control, CTS refresh, and change prioritization
  – Balancing agility, assurance, and affordability
Software Estimation: The Receding Horizon

IDPD: Incremental Development Productivity Decline
MBSSE: Model-Based Systems and Sw Engr.
COTS: Commercial Off-the-Shelf
SoS: Systems of Systems

Time, Domain Understanding

A
B
C
D

Estimation Error

Relative Productivity

Unprecedented
Preceded
Component-based
COTS
Agile
SoS. Apps, Widgets, IDPD, Clouds, Security, MBSSE