CSER 2020

Integrating the Constructive Systems Engineering Cost Model with the System Modeling Language

James E. Dalton II

The MITRE Corporation, 202 Burlington Road, Burlington, MA 01730, USA

ORCID 0000-0002-8564-124X

Abstract

This paper presents a method for utilizing the Constructive Systems Engineering Cost Model (COSYSMO) using inputs from a model that conforms to the Systems Modeling Language (SysML). The motivation for this integration is to enhance the usefulness of system models while also streamlining and simplifying the application of COSYSMO. Program cost components must be addressed as an early acquisition effort and updated as the program matures. Currently the application of COSYSMO is an individual task that takes place outside the SysML environment and no tool is freely available that incorporates COSYSMO as part of a System Modeling Language environment. SysML is a very intuitive place to store COSYSMO inputs. The language is built for these sorts of extensions, and a SysML model should represent a single source of authoritative truth which can include cost information. The goal of this work is ultimately to provide a freely available tool and further the adoption of COSYSMO or , at a minimum, show a repeatable method that others could build upon.

Keywords: COSYSMO; Cost Modeling; SysML; MBE; MBSE

1. Introduction

The Constructive Systems Engineering Cost Model (COSYSMO) is a parametric equation intended for use as a systems engineering cost estimator. COSYSMO was initially proposed by Dr. Barry W. Boehm of the University of Southern California and further developed by Ricardo Valerdi in his 2005 PhD dissertation and in his book in 2008 [1]. The original work was based on eight different cost models shown in Table 1. Since then, several improvements and extensions have been made. Most notable are Design With Re-use, Wang, 2008 [2] and Design For Re-use, Wang et al, 2014 [3] which takes into account that when designing *with* reused elements there will be a cost savings and when designing *for* reuse there will be additional effort. Another addition is Requirements Volatility, Peña, 2012 [4] which identified that there is a cost impact that is proportional to the amount of requirements changes. When applying COSYSMO to System of Systems, Lane et al., 2011 [5]

illustrated the equation's effectiveness. The amalgamation of these efforts culminates into COSYSMO 3.0 An Extended, Unified Cost Estimating Model For Systems Engineering, Alstad, 2018 [6] integrates these works into a complete cost model.

Table 1. Cost Models

Model Name	Owner/Developer	Domain
COCOMO II	USC	Software
PRICE-H	PRICE Systems, LLC	Hardware
PRICE-S	PRICE Systems, LLC	Software
SE Resource Forecasting Tool	Raytheon	Hardware
SEER-H	Galorith, Inc	Hardware
SEER-SEM	Galorith, Inc	Software
SSCM	The Aerospace Corporation	Hardware
USCM8	Los Angeles Air Force Base	Hardware

The current methods for applying COSYSMO are limited to just a few options. There is a Microsoft Excel worksheet, with instructions, that is freely available as a download from MIT [7], a web-based version from the Naval Postgraduate School [8] and a commercial product from SoftStar [9]. SoftStar's website specifically states that its offering is based on the COSYSMO 2.0 model and, while not explicitly stated, it is believed that the others are based on this model as well. While these are all valuable tools, they share one major drawback; they are stand-alone tools that require manual entry. As a program matures, the size drivers (requirements, interfaces, constraints and use-cases) become better defined and grow in number. Each iteration of the cost estimate will require the growing number of size drivers to be captured, counted and graded for the estimate update. It could be that the tedious nature of this work has been a factor in the slow adoption of COSYSMO. However, with the increased focus on digital engineering and the use of SysML within both the DoD and industry, there is a path to dramatically reduce the effort of a COSYSMO implementation: integrating COSYSMO 3.0 with a SysML model.

Section 2 provides a formal definition for the COSYSMO equation, though the reader should refer to Alstad 2018 for a complete definition. Section 3 provides an overview of the method used to integrate COSYSMO into a SysML environment. Section 4 summarizes the rational and the potential impacts of the integration. Section 5 provides lessons learned, and potential goals for any related future efforts.

2. The COSYSMO Equation

The COSYSMO parametric equation allows engineers to estimate the level of effort, in person-hours, it will take to staff systems engineering resources for a given program. This level of effort is estimated using the following equation,

$$Effort = A\left(size_{adjusted}\right)^{E} \prod_{j=1}^{13} EM_j \qquad (1)$$

A is referred to as the *calibration constant* and assigned a value of 26.33. While this value can be adjusted for a particular organization given their level of experience and bureaucracy, through Delphi assessments Jim Alstad defined the generalized value. The variable $size_{adjusted}$ refers to several unique types of components within a program which are additive. These are: number of requirements, interfaces, algorithms (parametrics) and use cases. The variable *E* refers to an exponential value that account for the *diseconomy of scale* for the system being analysed. Finally, the product of fifteen effort multipliers, EM, are incorporated in order to capture subjective yet critical dimensions to cost estimation. The following sub-sections discuss $size_{adjusted}$, E and

EM_j in greater detail.

2.1 $size_{adjusted}$

Each of the $size_{adjusted}$ variables (requirements, interfaces, algorithms and use cases), denoted by *k*, is assigned a tag value of easy (e), nominal (n) or difficult (d), corresponding to its level of understanding, implementation or complexity. The total adjusted size is express as the summation of all tagged weightings, where Φ = the count of *k* and ω = the weight of *k*.

$$size_{adjusted} = \sum(\omega_{e,k}\Phi_{e.k} + \omega_{n,k}\Phi_{n.k} + \omega_{d,k}\Phi_{d.k}) \quad (2)$$

The acceptable value weight factors for each *k* are listed below in table 2,

Table 2. Weight factors for *adjusted_size*

adjusted size components	Easy	Nominal	Difficult
Requirements	0.51	1.00	4.47
Interfaces	1.08	2.71	5.92
Algorithms	1.92	3.81	9.75
Use-Cases	6.37	13.57	26.34

2.2 Diseconomy of Scale

The exponent E represents the diseconomy of scale in program size $E = E_{Base} + SF_{RR} + SF_{PC} + SF_{RV}$ Where, E_{BASE} = constant = 1.052, SF = Scale Factor, RR =Risk Resolution, PC = Process Capability, RV = Requirements Volatility.

The acceptable values for each variable within *E* are captured below in table 3.

Table 3. Values for the exponent E

E (exponent) base = 1.0522	Very Low	Low	Nominal	High	Very High	Extra High
Risk Resolution	0.0602	0.0482	0.0361	0.0241	0.0120	N/A
Process Capability	0.0536	0.0429	0.0322	0.0214	0.0107	N/A
Requirements Volatility	N/A	0.0095	0.0189	0.0284	0.0379	N/A

2.3 Effort Multipliers

Effort Multipliers are, by definition, multiplicative. There are currently thirteen effort multipliers that span across 5 dimensions, Understanding, Complexity, Operation, People, and Environment. Like the weight factors and diseconomy of scale variables, EM values are applied a series of acceptable scores. The acceptable values for EM are listed in table 4.

Table 4. Values for the Effort Multipliers

Effort Multiplier	Very Low	Low	Nominal	High	Very High	Extra High

Requirements understanding	1.71	1.31	1.00	0.76	0.59	N/A
Architecture understanding	1.54	1.24	1.00	0.81	0.65	N/A
Stakeholder team cohesion	1.55	1.25	1.00	0.80	0.64	N/A
Level of service requirements	0.61	0.78	1.00	1.28	1.63	N/A
Technology risk	0.63	0.79	1.00	1.26	1.59	N/A
Number of recursive levels in the design	0.72	0.85	1.00	1.18	1.39	N/A
Design for Re-Use	N/A	0.88	1.00	1.13	1.28	1.45
Number & diversity of platforms	N/A	N/A	1.00	1.24	1.53	1.90
Migration complexity	N/A	N/A	1.00	1.25	1.57	1.96
Personnel/team capability	1.45	1.20	1.00	0.83	0.69	N/A
Personnel experience/continuity	1.36	1.17	1.00	0.86	0.74	N/A
Multi-site coordination	1.52	1.23	1.00	0.81	0.66	0.54
Tool Support	1.26	1.12	1.00	0.80	0.80	N/A

3. Integration Method

The effort to bring COSYSMO and SysML together was achieved by developing both a profile and a plugin for Cameo Enterprise Architecture, a packaged product provided by NoMagic that includes the base tool (MagicDraw), and additional add-ons. The profile is used to add additional information to relevant model elements. The plugin adds a menu item to the Cameo User Interface (UI) toolbar which allows an engineer to assign or modify COSYSMO parameters, culminating in the calculation of system level of effort. It is important to note that this integration method, i.e. the profile and plugin could be transferred to other SysML-conformant tools such as IBMs Rational Rhapsody. The profile defines stereotypes and related tag values (easy, nominal, difficult) for the size drivers (requirements, interfaces, algorithms and use-cases). This is the primary mechanism through which a systems engineer embeds COSYSMO inputs within the model. For example: every requirement should have a COSYSMO-requirement stereotype with one of the three possible tag values set as shown in Fig. 1.

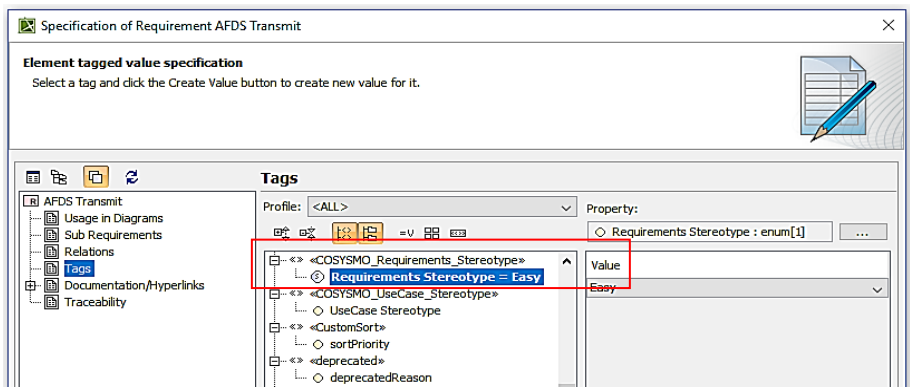

Fig. 1, Stereotype example

The plugin is primarily interactable through a Graphic User Interface (GUI). It allows the systems engineer to perform several functions:

- Select the grading for each Effort Multiplier and display their metrics
- Assign the appropriate values for the weight factors of the effort multipliers shown in tables 2 and 3
- Collect the size drivers from the stereotypes and assign the appropriate values from table 1

The GUI then generates a report in .TXT format that captures the following:

- Number of each size driver category
- Effort Multiplier choices made within the GUI
- Exponent choices made within the GUI
- Final values of all COSYSMO variables
- Calculated results

This GUI is launched by the user from within the existing cameo user interface. Fig. 1 shows the main GUI window, which contains the questions for the Effort Multipliers and components for the exponent.

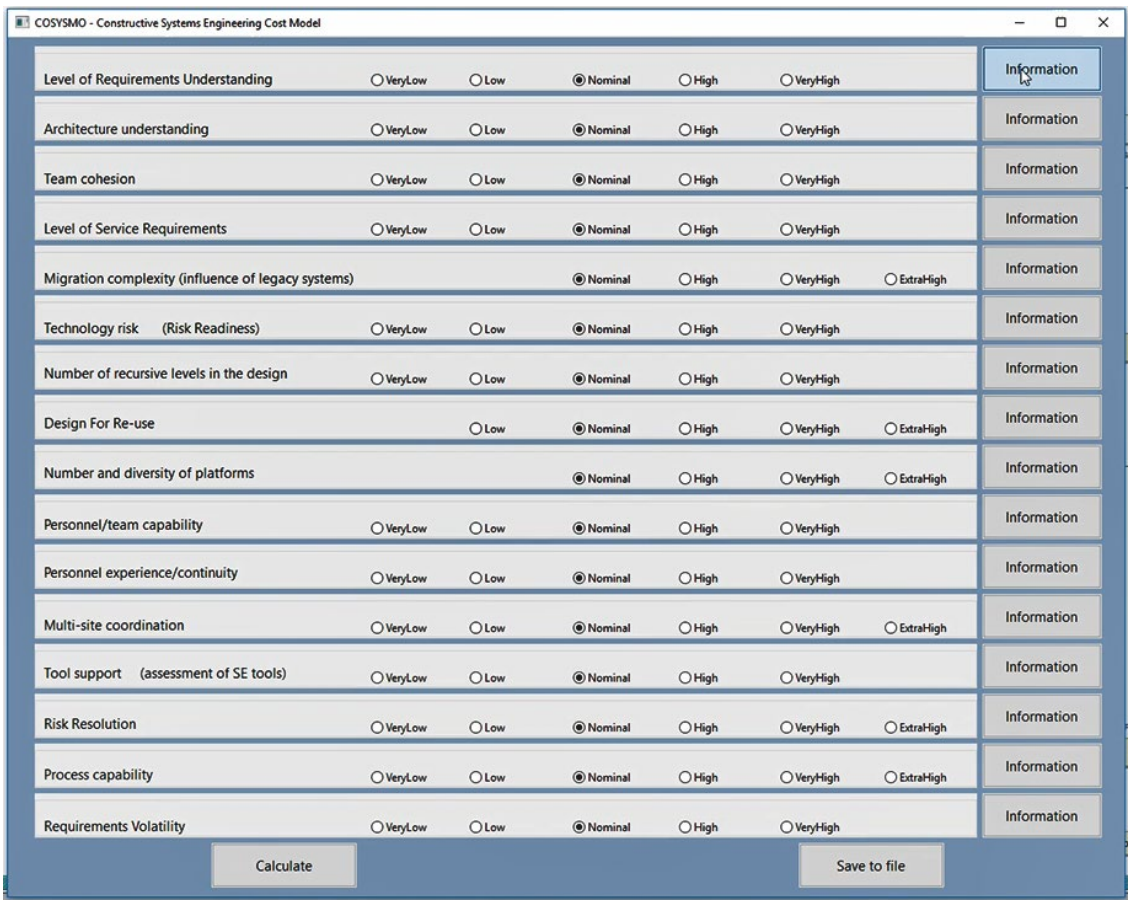

Fig. 2, GUI with Effort Multiplier and exponent grading

Clicking the “Information” button next to each assessment displays the metric definitions for the appropriate assessment as shown in Fig. 2.

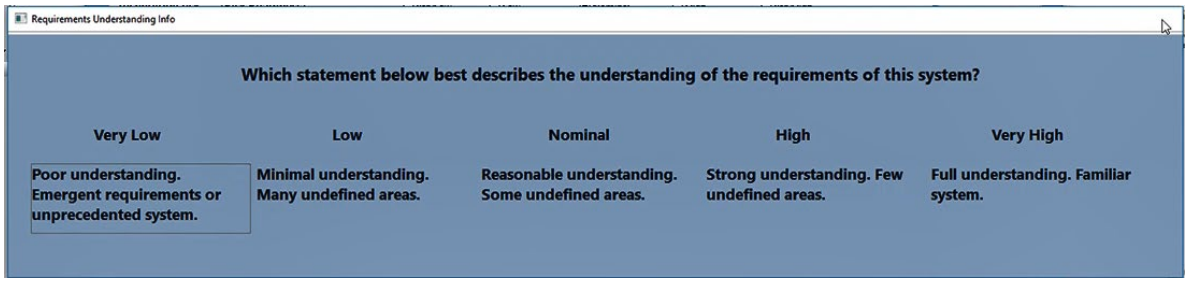

Fig. 3, Sample of metric display

This integration method was inspired by the work of Dennis Edwards from the Naval Postgraduate School [10] where the model is exported as an xml file to be parsed. In this effort a temp directory is created, the xml file [from the model export] is written to that directory and the file is then parsed to count the size driver elements. These counts are set as variable values within the java code and then the file and the temp directory are deleted. Each selection from the questionnaire sets values for each question variable (effort multiplier). A sample of the text file written from the GUI is shown in Fig 4.

```
************************************************
*********   Extracted from the model   **********
************************************************
Number of requirements     Easy =....4
                          Nominal =..0
                         Difficult = 2
------------------------------------------------
          interfaces      Easy =....3
                          Nominal =..0
                         Difficult = 2
------------------------------------------------
         algorithms       Easy =....0
                          Nominal =..0
                         Difficult = 1
------------------------------------------------
           use cases      Easy =....1
                          Nominal =..0
                         Difficult = 1
************************************************
The chosen ratings for this project estimate are:
************************************************
Requirements Understanding -------- Low
Architecture Understanding -------- Nominal
Team Cohesion  -------------------- Nominal
Level Of Service Requirements ----- Nominal
Migration Complexity -------------- Nominal
Technology Risk ------------------- Nominal
Number Of Recursive Levels -------- Nominal
Design For Reuse ------------------ Nominal
Number and Diversity of Platforms - Very High
Personnel - Team Capability ------- Nominal
Personnel Experience - Continuity - Nominal
Multi-Site Coordination ----------- Nominal
Tool Support ---------------------- Nominal
Risk Resolution ------------------- Nominal
Process Capability ---------------- Nominal
Requirements Volatility ----------- Nominal

The effort breakdown for this project is estimated to be:

PH = A * (size)^E * (EMj)
Calibration Constant = ....26.33
Size = ..................... 68.88
Exponent E = ............... 1.1394
Effort Multiplier = ....... 2.0043

Person Hours = 26.33 * (68.88)^1.1394 *(2.0043) = 6,557.48

PERSON HOURS = 6,557.48
*****************************************
PERSON MONTHS = 37.26
*****************************************
STE = 3.28
*****************************************
```

Figure 4: Sample output .txt file

4. Discussion and Impacts

All input variables to COSYSMO are easily implemented using existing SysML constructs. Additionally, beyond the main goal of COSYSMO, which is to estimate the Systems Engineering cost of a program, there is another important element that should not be overlooked; quantifying changes to any of the COSYSMO variables. One example would be changes to personnel on a project. Imagine a program that is in the design stage of the life cycle. The program manager is notified that two of the lead engineers are going to be reassigned to a very important new project. Currently all the program manager can do is argue from an intuitive sense that this will impact the program negatively with respect to cost and schedule. This is problematic because, though it is understood to be true, it is still only qualitative. However, the program utilizing COSYSMO can run a new cost estimate reflecting the changes to both "Personnel–Team Capability" and "Personnel Experience – Continuity" and "Team Cohesion", remember that the last run choices were saved to a file . With only a few clicks of a mouse the program manager can quantify the impacts of the changes in terms of time and money.

5. Future Work

Though this is a step in the right direction in terms of integrating multi-disciplinary data, it does not fully equip us with a method for realizing the wide adoption of COSYSMO. There are two additional items to be addressed. First, is the fact that COSYSMO will be less likely to be accurate when applied pre-milestone A. In the early stages of the acquisition process the requirements are not fully developed. The same is true for the other size drivers. As such, the estimate will not be able to provide a realistic accounting of the size components. One possible solution is to perform a study on the average growth of the size drivers between Concept Development and milestone A. Using this data we may be able to adjust the calibration factor to reflect anticipated growth when COSYSMO is applied to early acquisition. Second, the equation produces a point estimate. In general, cost analysts define a range for program cost estimates. Further work on COSYSMO would benefit from an effort to use the variable values as the peaks in individual PERT distributions [11] to then run Monte Carlo simulation. This could produce confidence intervals where the cost range would be represented as the 80% to 90% range with the final 10% accounting for the suggested reserve.

Beyond the application of using COSYSMO for estimating Systems Engineering cost, some very interesting work has been done toward using COSYSMO as a proxy for Total Program Cost by both Eric Honour (2013) [12] and Cole and Woodward (2014) [13]. Extending the plugin described in this paper to include an estimate for the total program cost and then implement a trial in conjunction with a program applying traditional cost estimating techniques could prove very informative and further the body of knowledge these authors have defined.

References

[1] R. Valerdi, The Constructive Systems Engineering Cost Model, Saarbrucken: VDM Verlag Dr. Muller Aktiengesellschaft & Co., 2008.

[2] G. V. R. A. A. M. C. &. R. G. J. Wang, "COSYSMO reuse extension," in *18th Annual International Symposium of the International Council on Systems Engineering*, Utrecht, 2008.

[3] G. J. R. M. P. R. V. Gan Wang, "A Generalized Systems Engineering Reuse Framework and its Cost Estimating Relationship," in *24th Annual INCOSE International Symposium*, Las Vegas, 2014.

[4] M. Pena, *Quantifying the Impact of Requirements Volatility on Systems Engineering Effort,* Phd Dissertation, 2012.

[5] J. A. Lane and R. Valderi, "System Interoperability Influence on System of Systems Engineering Effort," in *Annual Conference on Systems Engineering Research*, Redondo Beach, 2011.

[6] D. J. Alstad, "COSYSMO 3.0: An Extended, Unified Cost Estimating Model For Systems Engineering," in *6th Annual SERC Doctoral Students Forum*, Washington, 2018.

[7] M. Massachusettes Institute of Technology, "Downloads COSYSMO," WordPress & the Atahualpa Theme by BytesForAll., [Online]. Available: http://cosysmo.mit.edu/downloads/. [Accessed 5 5 2019].

[8] R. Madachy, "Expert COSYSMO - Systems Engineering Cost Model and Risk Advisor," [Online]. Available: http://csse.usc.edu/tools/ExpertCOSYSMO.php. [Accessed 4 4 2019].

[9] S. Systems, "COCOMO and COSYSMO Estimation Tools," [Online]. Available: http://www.softstarsystems.com/COSYSMO.htm. [Accessed 5 5 2019].

[10] D. J. Edwards, "Exploring the integration of COSYSMO with a model-based systems engineering methodology in early trade space analytics and decisions," June 2016. [Online]. Available: https://apps.dtic.mil/dtic/tr/fulltext/u2/1026556.pdf. [Accessed 3 8 2018].

[11] "RiskAmp/beta-pert," Structured Data LLC, [Online]. Available: https://www.riskamp.com/beta-pert. [Accessed 9 4 2019].

[12] E. C. Honour, "http://www.hcode.com/seroi/," 26 3 2013. [Online]. Available: http://www.hcode.com/seroi/documents/SE-ROI%20Thesis-distrib.pdf. [Accessed 4 6 2018].

[13] Reggie Cole, Kevin Woodward, "System Cost Modeling Using Proxy Estimation and COSYSMO," Lockheed Martin, 2014.