Extensions of COSYSMO to Represent Reuse

Ricardo Valerdi
Massachusetts Institute of Technology
Garry Roedler
Lockheed Martin
John Gaffney
Lockheed Martin
John Rieff
Raytheon

ABSTRACT

As the maturity of COSYSMO increases, users continue to identify areas in which the model can be improved. Recent emphasis has been placed on the clarification of counting rules for the COSYSMO size drivers. These drivers represent various attributes of the total size of the task of the systems engineering effort estimated by COSYSMO; in terms of person months. The intent of these rules is to ensure consistent interpretation and use of the size input parameters that include: requirements, interfaces, algorithms, and operational scenarios. Experience in applying these rules has exposed a limitation of the current version of the model; there was no way of including the affect of reusing system components in the calculation of systems engineering effort.

This has resulted in inaccurate estimates of systems engineering effort for systems that incorporated significant reuse, as in the case of programs with a high degree of COTS integration. As a result, a method was needed to account for the fact that not all of the requirements that drive systems engineering effort are new. Specifically, some of the requirements for a new system may be "reused" from a prior system. Further, some of the new system's requirements may be "modified" from a prior system. Moreover, the evolution of system requirements over the system life cycle may result in "deleted" requirements from the initial configuration baseline.

On the surface, the notion of reuse in COSYSMO may appear as a necessity-is-the-mother-of-invention activity but in reality it was an inevitable feature. One reason is that most software cost estimation models – especially COCOMO II – go into great detail in addressing aspects of software reuse. The other is that reuse is more prevalent among defense contractors that aim for higher productivity gains as they avoid pursuing designs from scratch.

For these reasons, this paper provides (1) an approach for handling reuse in systems engineering in terms of the number of systems requirements in COSYSMO, (2) a discussion on the potential cost drivers that could be influenced by reuse, and (3) strategies in which this approach can be extended to include the three other size drivers in the model.

Proposed Approach

The approach taken to represent the requirements size is analogous to that used to represent software code size in those frequent instances in which there are several categories of code, including new, modified, deleted and reused. The development of a new system with these four categories may include the deletion of code from a prior system or previous version or build. The cost of such deletion must be accounted for. In the case of software, the code size is often represented as "ESLOC," or "Equivalent New Code." ESLOC is computed as the weighted sum of the new, the reused, the modified, and the deleted code. Simlarly, we have chosen to use "ERequirements" in COSYSMO, which is equal to the weighted sum of the new requirements count, the reused requirements count, the modified requirements count, and the deleted requirements count.

The counts of these four types of requirements are converted into one count, called "Equivalent New," symbolized as E_{Ti} , for each of the four categories of requirements, i=1,2,3,4. Now, define $E_{TT,}$ E_{TT}=Total Equivalent New Requirements, where $E_{TT}=\sum E_{Ti}$, for i=1 through 4. COSYSMO with reuse representation capability uses E_{TT} as "S," the overall requirements size driver, in the formula for systems engineering labor hours.

Equivalent New Requirements Formula Derivation

E_T stands for E_{Ti} in this derivation; it applies for any value of i, i=1,2,3, or 4, corresponding to the four categories of requirements. Please refer to the following definitions. In each of them, the subscript "i" is assumed, but suppressed for the sake of ease of expression:

N_N=count of new requirements;
N_M=count of modified requirements;
N_R=count of reused requirements;
N_D=count of deleted requirements;
N_T=total count of requirements.

Where:

$N_T=N_N+N_M+N_R+N_D$.

Note that N_{Ti}=the weighted sum of easy, nominal, and difficult requirements of the i^{th} category.

We can relate E_T to N_N, N_M, N_R, and N_D as now described. Let c_M=the unit effort or cost for a modified requirement relative to that for a new requirement (=labor hours per modified requirement divided by labor hours per new requirement). Similarly, let c_R= the unit effort or cost for a reused requirement relative to that for a new requirement. Also, let c_D=the unit effort or cost for a deleted requirement. We expect that $c_R \leq 1$. It is possible under some circumstances that $c_M>1$ or $c_D>1$, as work may have been devoted to implementing a requirement and than at some point during the development process, circumstances and/or the customer might dictate the need to either delete a requirement or modify it. Because of these uncertainties, mathematically, it is assumed in the remainder of this document only that $0<c_M, 0<c_R$, and $0<c_D$.

Now, define $E_T=N_N+(c_M*N_M)+(c_R*N_R)+(c_D*N_D)$; E_T is analogous to "ESLOC" in the case of software, the count of "Equivalent New Lines of Code."

Further, let p_N=proportion of N_T that is new, or $N_N=p_N*N_T$. Similarly, for the modified requirements, $N_M=p_M* N_T$, and for the reused requirements, $N_R=p_R*N_T$, and for the deleted requirements, $N_D=p_D*N_T$. Further, recognize that $p_N+p_M+p_R+p_D=1$, and thus, we can write $p_N=1-p_M-p_R-p_D$.

Now, we combine the formula given above for E_T and the relationships between N_M and N_T and N_R and N_T and N_D and N_T into an expression for E_T in terms of N_T, p_M, p_R, p_D, c_M, c_R, and c_D.

$E_T =(p_N*N_T)+(c_M*p_M*N_T)+(c_R*p_R*N_T)+(c_D*p_D*N_T)$. Now, we use the fact that $p_N=1-p_M-p_R-p_D$ and combine terms to obtain the formula for E_T, the "Equivalent New" requirements count:

$$\mathbf{E_T=[1-(p_M*(1-c_M))-(p_R*(1-c_R))-(p_D*(1-c_D))]*N_T}$$

Note that the factor $[1-(p_M*(1-c_M))-(p_R*(1-c_R))-(p_D*(1-c_D))]$ is usually ≤ 1, BUT it may be >1 in some circumstances (see earlier description about the values of c_M and c_D). and thus, USUALLY, but not necessarily, $E_T \leq N_T$.

We recognize that N_{Ti} is the weighted sum of the "easy," "nominal," and "difficult" requirements for each of the four categories of requirements in COSYSMO, "system requirements," "system interfaces," "algorithms," and "operational scenarios."

As stated above, there is an expression of this form for each of the four categories of requirements, E_{Ti}, for i=1,2,3,4. E_{TT}=Total Equivalent New Requirements, where $E_{TT}=\sum E_{Ti}$, for i=1 through 4. COSYSMO with reuse requirements capability would use E_{TT} as "S," the overall requirements size driver in the formula for systems engineering labor hours.

Operationally, when employing the reused requirements capability in COSYSMO, the user would enter the counts for the three levels of difficulty "easy," "nominal," and "difficult" for each of the four categories of requirements. Also, he would enter the values of p_M, c_M, p_R, c_R, p_D, and c_D for each of the four categories of requirements. Note that the values of p_{Mi}, c_{Mi}, p_{Ri} , c_{Ri} , p_{Di}, and c_{Di} are assumed to apply to the "easy," "nominal," and "difficult" counts for each category of requirement, "i." The user would assign some

"average" value for each of the parameters p_{Mi}, etc. that are taken to apply to all three requirements levels, e.g., "easy." As can be seen from the expression for E_T (E_{Ti}) above, when there are no "modified," "reused" or "deleted" requirements, then $E_{Ti}=N_{Ti,}$ the weighted sum of the "easy," "normal," and "difficult" requirements as in Academic COSYSMO. In the case of the (future) COSYSMO with reuse capability and with a "risk" capabilty, the user would enter the counts for the "easy," "nominal," and "difficult" requirements for the "low," "likely," and "high" probability values. Then the weighted sum of the "easy," "nominal," and "difficult" counts for each of these three is multiplied by the quantity:
$[1-(p_{Mi}*(1-c_{Mi}))-(p_{Ri}*(1-c_{Ri}))-(p_{Di}*(1-c_{Di}))]$, which is probably ≤ 1 in most cases, but could be >1 under some circumstances as indicated above..

References

Boehm, B. Abts, C., Brown, A. W., Chulani, S., Clark, B., Horowitz, E., Madachy, R., Reifer, D., Steece, B., *Software Cost Estimation with COCOMO II,* Prentice-Hall, 2000.

Conte, S. D., Dunsmore, H. E., Shen, V. Y., *Software Engineering Metrics and Models*, Benjamin-Cummings Publishing Company, 1986.

Gaffney, J., Cruickshank, R., et al. *The Software Measurement Guidebook,* Thomson Computer Press, 1995.

Jones, C., *Estimating Software Costs,* McGraw-Hill, 1998.

Stutzke, R., *Estimating Software-Intensive Systems*, Addison-Wesley, 2005.