International Journal of Computer Integrated Manufacturing
Vol. 25, Nos. 4–5, April–May 2012, 432–443

Harmonising software engineering and systems engineering cost estimation

Gan Wang[a], Ricardo Valerdi[b]*, Garry J. Roedler[c], Aaron Ankrum[a] and John E. Gaffney Jr.[c]

[a]*BAE Systems, Reston, VA, USA;* [b]*Engineering Systems Division, Massachusetts Institute of Technology, Cambridge, MA, USA;* [c]*Lockheed Martin, Philadelphia, PA, USA*

(*Received 30 September 2010; final version received 6 November 2010*)

Complex systems are developed with the help of numerous engineering disciplines. In software-intensive systems, software and systems engineers play a major role in design, development, test and deployment. Since these roles are often very tightly coupled, it is difficult to determine which life cycle functions can be performed jointly versus separately. This makes it particularly difficult for resource estimation and project planning. To resolve this issue, we examine the gaps and overlaps between software engineering and systems engineering cost models with intent to harmonise the estimates for project estimation. In particular, we evaluate the central assumptions of the constructive cost model II and constructive systems engineering cost model and propose an approach to identify gaps and overlaps between them. Furthermore, we provide guidelines on how to reconcile and resolve the identified gaps and overlaps. The ultimate purpose of this work is to develop effective techniques for accurately estimating the combined software and systems engineering effort for software-intensive systems.

Keywords: cost estimation; software engineering; COCOMO II; systems engineering; COSYSMO

1. Introduction

At the dawn of the Industrial Revolution in the late eighteenth century, Adam Smith advocated that a free market economy is more productive and more beneficial to society (Smith 1776). In his magnum opus *The Wealth of Nations*, Smith suggests the basic idea that specialisation of technical tasks increases productivity, and specialisation is further needed as the size of the market increases. There are clear parallels to complex software-intensive systems: the specialisation of systems engineers and software engineers is increasingly important as organisations wish to increase productivity in large marketplaces in light of increased technical complexity and numerous organisational layers.

1.1. Motivation

Smith tells of a parable about a pin-maker that highlights the societal benefits of specialisation and the synergy between specialists. What is interesting here is that the idea behind the division of labour is two-sided. Most of us only remember the benefits associated with increased productivity as a result of specialisation. However, there are alienating effects to too much specialisation. People who are only good at one thing eventually become obsolete. This is where the software engineering (SW) – systems engineering (SE) interface becomes more interesting. Both functional disciplines need to know something about the other in order for both to be effective.

Boehm (2007) claims that 'you can't do good software engineering by neglecting systems engineering.' The inverse is also true: you cannot do good SE without good software engineering. Together, these ideas lead to a mutually beneficial arrangement that results in a win–win scenario where both disciplines contribute to each other's success. It would be rational to assume that systems and software engineering would be more in harmony, yet there are many disconnects in practice.

It is ironic that while collaboration is so obviously beneficial, it can be so amazingly difficult. One possible reason is that organisations often only consider the high-level joint objective of software engineering and SE: to build a system that meets the cost, schedule and performance targets. The lower-level objective would expose potential turf wars that take place between the two technical functions for control of the product design and implementation process. Nevertheless, as systems become increasingly complex, the needs for the harmonisation of software engineering and SE become more critical. Efforts to integrate SW and SE standards (Singh 1995, Roedler 2008) and processes (Boehm 2000, 2006) have provided useful guidance. Studies have been keyed

*Corresponding author. Email: rvalerdi@mit.edu

ISSN 0951-192X print/ISSN 1362-3052 online
© 2012 Taylor & Francis
http://dx.doi.org/10.1080/0951192X.2010.542182
http://www.tandfonline.com

to integrating SE and software engineering in terms of standard activities in developing software-intensive systems (Boehm 2006, Pyster and Turner 2008). Similarly, as cost estimating models have evolved and matured, there has been a natural migration of attention towards the harmonisation of cost models. In particular, there have been increasing needs and, therefore, growing interests in unifying the cost models to enable seamless integration of systems and software estimates.

1.2. *Constructive cost model and constructive systems engineering cost model*

When assembling a total engineering estimate from the functional estimates for a system, one must go through the exercise of reconciling gaps and overlaps between these estimates of the elemental parts (e.g. SE and SW) to ensure a single, coherent bid that is competitive and reliable. These gaps and overlaps are a consequence of independent assumptions made with each of the functional estimates by using the respective model, which may result in double coverage of certain task areas while leaving other tasks under represented. The over- and under-representations are more significant in the boundary areas that models interface or hand over tasks, such as preliminary and detailed design, integration and test. To assess this, we consider two cost models with specific focus on software engineering and SE. The constructive cost model (COCOMO) is aimed at estimating the amount of software engineering effort needed to develop a system with certain technical characteristics (Boehm *et al.* 2000). The constructive systems engineering cost model (COSYSMO) is aimed at estimating the amount of SE effort needed to support the development of a system (Valerdi 2008). Both COCOMO and COSYSMO estimate design activities. One question is that: where does the software design activity stop and the SE design activity end? Another question is that: do both models together cover the full set of engineering activities needed to deliver a successful product? Similarly, which part of the integration and testing effort is considered by which model?

These scope overlap issues and potential gaps are more profound than providing these models correct size and cost driver counts and settings and exercising good practice when generating an estimate. The estimating scope (resource or labour estimate provided by the model/tool) is ultimately determined by the calibrations used. This means that the correct estimating scope has to be decided when collecting historical data (or obtaining data from historical repositories) and generating model calibrations.

1.3. *Research methodology*

The need from practical application of these cost models is to establish a consistent guideline for determining the model scopes and harmonising the functional estimates to be able to effectively generate a coherent product bid. Commercial vendors of cost estimating models have long attempted to address the same issue. Both PRICE (TruePlanning® by PRICE Systems, http://www.price-systems.com/products/price_trueplanning.asp) and SEER (SEER® by Galorath, http://www.galorath.com/index.php) have organised their respective tools in integrated suites, providing built-in interfaces between different functional models. Recent academic effort (Valerdi and Lane 2004) has focused on the key issues related to unifying three categories of cost models, between software and systems, and system-of-systems. There has also been a series of workshops and analyses performed to date to investigate the harmonisation of these cost models. These include the following:

- University of Southern California (USC) COSYSMO Workshop, March 2008 (Los Angeles, CA, USA) – this workshop scoped the problem, prioritised the needs for harmonisation and identified that operational guidance may be as significant to addressing the harmonisation as the modelling constructs and driver definitions.
- Practical Systems and Software Measurement (PSM) Workshop on Harmonising COCOMO and COSYSMO, July 2008 (Mystic, CT, USA) – this workshop analysed the typical work breakdown structure (WBS) elements against functional responsibilities, assessed the element coverage against COCOMO and COSYSMO and identified potential gaps and overlaps.
- BAE Systems and Lockheed Martin Internal Projects on COSYSMO – these projects examined the application of various changes to determine the effectiveness in implementation.

The efforts to address the harmonisation have been focused on the following six considerations:

- Overlap/gaps of tasks per typical work elements, work products and combined activities – to determine where the systems and software engineering cost estimation models may both be counting the same effort (resulting in double counting) or where neither model accounts for relevant effort.
- Analysis of cost drivers – to determine whether the models account for common drivers when they are relevant to both systems and software engineering.

- Commonality of terminology, constructs, life cycle phases and units – to ensure common interpretation, ability to scope/define the estimation and ease to communicate data requirements and results.
- Consideration of common size drivers – to provide the ability to improve the utility of the estimation models and the ability to use a common set of size drivers.
- Base assumptions of the models – to ensure that the models are built on a consistent set of valid assumptions.
- Compatibility issues from any findings or recommendations – to ensure that any recommended changes would not have adverse impact on the model usage for addressing their independent areas of estimation.

The primary focus of this article is to address the analysis of potential, collective over- and under-representation of these models and to provide practical recommendations on how to reconcile and resolve the identified gaps and overlaps. It describes the results of initial work in harmonising COCOMO II for SW (Boehm *et al*. 2000) and COSYSMO for SE (Valerdi 2008) to ensure that cost estimates adequately cover both functions. This analysis is done in four parts. First, we identify the overlaps between the activities covered by the two models as illustrated in Figure 1(a). It is important to understand what activities are being covered by both models, and whether the scope of their coverage may lead to an overestimation of project effort. Overlaps may also emerge during the operational use of the models from incorrect assumptions or interpretations by the users of the models. For example, the user of COSYSMO may assume that effort for the recursive levels that include the software are to be covered by COSYSMO application or the users of both models assume that 'Development Test and Evaluation (DT&E)' is covered by the model they are using.

Second, we identify the gaps between the two models as shown in Figure 1(b). The purpose is to understand what is being missed by both models that may lead to an underestimate in the project's engineering effort. The outcomes of both models are driven by assumptions made in the development of the models, variations in practice of SW and SE, and the degree of alignment of integrated process models. Gaps may also emerge during the operational use of the models from incorrect assumptions by the users of the models. For example, the user of COCOMO II may assume that certain activities like 'integration and test,' 'quality management' or 'development for reusability' are fully covered by COSYSMO.

Third, we will provide a short summary of the results of the analysis of the other five areas of consideration. These are also important to the effective use of the systems and software cost estimation models in a concurrent/integrated manner. However, a future article will go further into the details of these considerations.

The final section of the article provides next steps that are required to harmonise the relationships between COCOMO II and COSYSMO. These include standard phase/stage alignment for both models, per definitions used in ISO/IEC 15288 (2002) and 12207 (2008), a means to adequately account for recursion (at level of hands-off to SW), added guidance to COSYSMO drivers to account for time/storage constraints and requirements/architecture volatility, the ability to use COSYSMO size drivers in COCOMO for early estimates, and adding documented list of assumptions to COSYSMO.

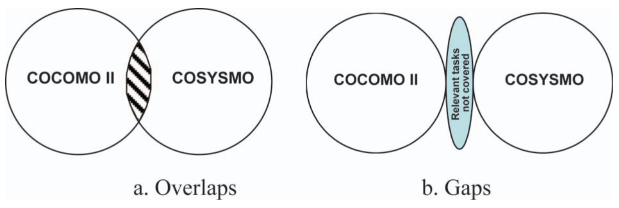

Figure 1. Overlaps and gaps between SW and SE models.

2. Software engineering and SE harmonisation

Fundamentally, any attempt to harmonise functional models starts with developing a comprehensive understanding of what is considered to be in the realm of SE vs. software engineering. This requires analysing the engineering activities based on some reference model and allocating them to SE or SW. It is followed by analysing how each estimating model accounts for these activities.

To accomplish this analysis, we conducted several roundtable workshops between different groups of over 20 stakeholders across the software engineering, SE and cost estimation communities, including those at the International COCOMO Forum at the University of Southern California and the annual PSM Users' Group meeting. The objective of these workshops was to achieve agreement among stakeholders on the responsibility assignment as well as the model coverage, as described in the following section.

For this purpose, we define a generic, reference WBS that represents the engineering scope of a system development project by a contractor. This WBS is defined based on MIL-HDBK-881A (2005) and the

ANSI/EIA 632 (1999) standard, by tailoring a reference structure from MIL-HDBK-881 A while considering the 33 technical activities defined in ANSI/EIA 632. To ensure a rigorous foundation, further work is in progress to vet the structure against ISO/IEC 15288, Systems Engineering – System Life Cycle Processes, and ISO/IEC 12207, Software Engineering Process Standards. This structure is oriented around engineering and delivery of a Prime Mission Product (PMP), which includes the required design, build, integration, verification and validation activities, as well as technical management of the project. The top-level breakdown of this structure is listed in Figure 2. The defined structure provides the breakdown to the third level, specifying major engineering activities for each of the system components or configuration items (CIs). The complete breakdown of the structure is listed in the left half of Appendix.

In addition, we define a generic organisation breakdown structure (OBS) with five standard 'functions'. They are SE, software engineering (SW), hardware engineering (HW), supportability engineering (SP) and project engineering management (PEM). HW in this context includes the traditional electrical and mechanical engineering functions. The PEM 'function' is comprised of all engineering management activities at the project level, which include the overall project planning, monitor and control activities which are the responsibility of the project manager and typically funded by the project management overhead budget.

The result is a WBS vs. OBS cross-reference matrix, as listed in the Appendix, with the work breakdown up and down on the left and the organisational breakdown on the top across the columns. Using this framework, we conduct an exercise of task assignment to functions. At the same time, we assess the cost model coverage of the tasks. We use COSYSMO as the default tool for providing the SE estimates and COCOMO II for the SW estimates. For the sake of completeness, a generic hardware model (think of, e.g. PRICE-H or SEER-H) is assumed for the HW estimates. The objective of this exercise is twofold. The first is to assign the functional responsibility or 'ownership' to each of the leaf level elements defined in the WBS, in the sense that a functional model should be responsible for estimating its scope. The second aspect is to assess the current coverage of the models (COSYSMO and COCOMO II), in the sense that the model defined as is today already provides the coverage of the work element in consideration, based on our understating of each of the model definitions.

The exercise involves placing a symbol 'X' in a cell intersecting a task and a function as shown in the cross-reference matrix in Appendix. For example, we put an 'X' in the cell connecting WBS element 2.1. – Systems Engineering Management to the systems function (SE) to indicate that this task is the responsibility of or is owned by the SE function. Therefore, the task scope should be estimated by the SE model or COSYSMO in this context. Similarly, we assign the SW function to WBS element 1.3.2.2., the design task for PMP Application Software by placing an 'X' to the cell intersecting the two.

A word about ownership: When we conduct the assignment exercise, we emphasise functional ownership. In other words, it is immaterial who (a person or job title) performs the actual work. It matters who (which function) owns the task. Projects often employ resources across functional lines to perform certain tasks, due to constraints such as availability of resources, scheduling requirements or cost considerations. However, the task is still owned by the function assigned. It typically is represented by the lead role of the integrated product team (IPT). This is a necessary measure to ensure consistency between different organisations or different projects in the same organisation. This guideline in no way should dictate how a project plans its activities. However, cost estimation provides the baseline for project planning. Clear and consistent policies like this in estimating ensure accurate accounting for project activities and enable effective task planning and resource management.

The next step is to determine the current model coverage. If we believe a COSYSMO estimate, as defined today, already covers the task, we put the symbol 'Y', instead of 'X', in the interconnecting cell. Similarly, we use the symbol 'S' for COCOMO II estimated tasks.

The result of this exercise is the cross-reference matrix, marked by symbols 'X', 'Y' and 'S', as in Appendix. The areas of gaps and overlaps in estimate scopes are identified by the number and the type of symbols marked for each task. To ensure clarification, we understand the estimating scope of both models as below.

1.0 – System/Project
- 1.1 – Integrated Project Management (IPM)
- 1.2 – Systems Engineering
- 1.3 – Prime Mission Product (PMP)
- 1.4 – Platform Integration
- 1.5 – System Test & Evaluation (ST&E)
- 1.6 – Training
- 1.7 – Data Management
- 1.8 – Peculiar Support Equipment
- 1.9 – Common Support Equipment
- 1.10 – Operational / Site Activation
- 1.11 – Industrial Facilities

Figure 2. Generic, contract WBS.

COCOMO II (Boehm *et al.* 2000), developed at the USC, is a model that estimates the cost, effort, and schedule when planning a new software development activity. The model creates software estimates by using source lines of code (SLOC). COCOMO II is the latest major extension to the original COCOMO (COCOMO 81) model published in 1981. It has been used prevalently over the years across the industry and government contracting environment in a variety of applications supporting decision-making related to software development. The COSYSMO is a model that can help people reason about the economic implications of SE on projects (Valerdi 2008). Similar to COCOMO II, it was also developed at USC as a research project with the help of BAE Systems, Boeing, General Dynamics, L-3 Communications, Lockheed Martin, Northrop Grumman, Raytheon, and SAIC. COSYSMO follows a parametric modelling approach used to estimate the quantity of SE labour, in terms of person months, required for the conceptualisation, design, test and deployment of large-scale software and hardware projects. User objectives include the ability to make proposal estimates, investment decisions, budget planning, project tracking, tradeoffs, risk management, strategy planning and process improvement measurement.

COCOMO II is designed to estimate the software effort associated with the analysis of software requirements and the design, implementation and test of software. COSYSMO estimates the system engineering effort associated with the development of the software system concept, overall software system design, implementation and test. More generally, COSYSMO estimates the system engineering effort for any system. Table 1 lists the major differences of both models in respective methods applied to developing estimates. The COCOMO II estimate of the software effort will surely account for the additional effort required by any additional testing of the software system; at the same time, the COSYSMO effort will account for additional test development and management since the systems engineers are required to perform additional validation and verification of the system. Either model can account for this effort based on how users wish to allocate the testing activity. Each organisation's unique relationship between these two functional disciplines needs to be reconciled when using COSYSMO and COCOMO II together. Our approach for accomplishing this is through examination of work scope of each discipline as defined by the WBS vs. OBS construct and intends to be organisation-neutral that can be adopted or adapted by specific organisational implementations.

3. Analysis of model scopes

The WBS vs. OBS cross-reference matrix as shown in Appendix provides an overview of the estimating scopes of the respective functional estimates. To ensure clarity, we list below the four types of associations between tasks and functions represented by the following four different letters:

- 'X' – indicates the ownership of the task by a function
- 'Y' – the task scope is estimated by the current COSYSMO definition
- 'S' – the task scope is estimated by the current COCOMO II definition
- 'U' – uncertain/undetermined or the task scope is not consistently covered by either model, which means it sometimes is estimated by COSYSMO; other times not

It is important to note that, in conducting this exercise, we attempt only to cover the nominal or '80 percentile' practices. In working with stakeholders, we stress that there will be exceptions to the assignments

Table 1. Differences between COCOMO II and COSYSMO.

	COCOMO II	COSYSMO
Estimates	Software development	Systems engineering
Estimates size via	Thousands of software lines of code (KSLOC), function points, or application points	Requirements, interfaces, algorithms and operational scenarios
Life cycle phases	MBASE/RUP phases: (1) Inception, (2) elaboration, (3) construction and (4) transition	Phases are a hybrid of ISO/IEC 15288 stages: (1) conceptualise, (2) develop, (3) operation, test, and evaluation, (4) transition to operation, (5) operate maintain or enhance and (6) replace or dismantle.
Form of the model	One size factor, five scale factors and 18 effort multipliers	Four size factors, one scale factor and 14 effort multipliers
Represents diseconomy of scale through	Five scale factors	One exponential factor for size

made and it could be different from the last program one has worked on. We recognise and accept these exceptions. A second and more subtle point is that, in making these assignments, we do not simply try to replicate the exact way someone may have planned a project today. However, we bias towards best practices in project planning. At the minimum, we achieve a consistent understanding in the way we believe how a project should be planned and executed. This consistency is the key in communicating estimates between stakeholders.

Using this matrix, the analysis is conducted as follows. We go down the WBS hierarchy and examine all the leaf elements. For a leaf element, there are three nominal outcomes. If there are no interconnecting cells, then it indicates a gap. It implies that no function is taking the ownership of the task and, therefore, is responsible for estimating it. If there is a single interconnecting cell, it represents a unique association between a task or WBS element and a function. It indicates no scope gaps or overlaps between the functions for that task. If a task has association with more than one function, indicated by more than one interconnecting cell, there are potential overlaps. In particular, we pay special attention to 'Y' and 'S' types of interconnections. If they appear on the same row or for the same task, then there is a potential overlap between COCOMO II and COSYSMO. On the other hand, if either appears for a WBS element, then this is potentially an under-lapped area.

Examination of Appendix reveals that all leaf elements are assigned to at least one function. This indicates there is a complete coverage of the engineering scope, at least in theory, by all functional organisations. It also shows that, while there are many singular associations between task and function, there are also many potential gaps and overlaps indicated by multiple interconnecting cells or absence of 'Y' and 'S' for a task.

A couple of patterns are worth noting at this level. First, the multiplicities under WBS elements 1.3. – PMP, 1.8. – Peculiar Support Equipment and 1.9. – Common Support Equipment do not necessarily indicate overlaps. This is due to the fact there can be multiple subsystems or components under the prime system. At the subsystem level, each CI is assumed to be assigned to a single functional IPT, which is responsible for all relevant development tasks as well as its estimated budget. For example, a custom circuit assembly can be designed by the hardware IPT and a Commercial-off-the-shelf (COTS) component is acquired and integrated by the system IPT at the same time. These CIs are traditionally estimated separately by different cost models. We do not consider these elements to be overlapped, and so will exclude them from the following discussion. Second, SP is identified as a separate function. However, we recognise that in many companies it is part of the SE organisation. In this context, we keep the two separate and attempt to address any gaps and overlaps between the two to provide more definitive guidance.

Next, we examine the identified gaps and overlaps in more detail. In the same discussion, we also provide our recommendations or a guideline for resolution and reconciliation of these gaps and overlaps.

3.1. *Potential gaps*

A summary of the identified gaps is listed in Table 2. Under WBS 1.1. – Integrated Project Management, there are four engineering tasks covered by neither COCOMO II nor COSYSMO. These tasks are in the project management overhead. They do not contribute directly to developing the system but are necessary for managing and executing the project. The first three tasks are owned by PEM and the last by Supportability. Specifically, while COSYSMO covers the SE management related (WBS 1.2.1.); it does not currently cover the technical management or the project engineer's effort at the project level. Nor does it cover process, quality management or IT/infrastructure effort, all under the PMO budget line. The Dismantle and Disposal tasks include those efforts to define a disposal strategy for the system and develop the plan for retirement. It is generally provided by SP but not estimated by COSYSMO. However, these overhead tasks are intrinsically no different from tasks such as system test and evaluation (ST&E), which are not part of the mission product but are necessary for successful completion and delivery of the PMP. The resolution we recommend to bridge these gaps is to include these overhead tasks in the COSYSMO scope by including the corresponding efforts in the calibration data, so that the estimate will provide the coverage for these tasks areas.

Under WBS 1.3.3. – PMP System Software, there are four elements that are of systems responsibility, but not currently estimated by COSYSMO. This is due to the fact that system software is generally at the subsystem level in between the application software and hardware of the system. The COSYSMO size drivers (requirements, interfaces, algorithms and scenarios) are at the system level only and generally do not concern implementation at the component level. We recommend keeping this scope outside of the system-level estimate. To bridge the gap for the total engineering estimate, the PMP System Software is brought in as a discrete estimate, which is developed independently as a separate system component at the next level system abstraction. However, the corresponding effort at the system level should not be zero. For seamless integration, the related system requirements can be classified as '*Adopted*' or '*Managed*', depending upon the level of system testing, for the system-level estimate in the COSYSMO reuse model (Wang *et al.* 2008).

Similarly, any subsystems or CIs below WBS 1.3.3. that are assigned under systems-led IPTs (e.g. typically COTS-based components) are not covered by COSYSMO. The recommended strategy is to treat the corresponding CIs as subcontracted 'systems' at the next level system abstraction and develop discrete estimates using relevant models (including COSYSMO and COCOMO II). Once again, apply the appropriate reuse model, '*Adopted*' or '*Managed*' category, for the system-level estimate in COSYSMO.

WBS 1.4.3. – Initial Spares and Repair Parts are traditionally supportability responsibility. COSYSMO currently does not cover its effort. However, we believe that the task is general enough that its planning effort should be covered by COSYSMO. Similarly, the two elements, 1.5.6. and 1.5.7., under ST&E are support in nature relative to DT&E and Operational test and evaluation (OT&E) activities. Nevertheless, they are considered as systems responsibility. They, too, are not included in the current COSYSMO scope. The recommended resolution is to include these support scopes in COSYSMO by including the associated efforts in the calibration data. In addition, for the case of WBS 1.5.7., ensure the ST&E Test Facility requirements are also included in the system requirements for driver counting purpose so that the scope is explicitly estimated.

Elements for Training and Data Management, under WBS 1.6. and WBS 1.7., are traditionally within SP responsibility. Neither COCOMO II nor COSYSMO estimate the scope. We recommend keeping these efforts outside of both the systems and the software scopes and, when appropriate, developing discrete estimates using other methods, e.g. based on metrics such as training class hours, number of documents or technical publication volumes. Similarly, elements under WBS 1.10. and 1.11. are construction or facility maintenance in nature. We again recommend that they be kept outside of the either model scope and estimated using other methods.

3.2. Potential overlaps

There are three major areas of potential overlaps identified between COSYSMO and COCOMO II. They are summarised in Table 3. Note that the other multiple associations under WBS elements 1.3., 1.8. and 1.9. are due to multiple system components and thus not considered as overlaps, as discussed before.

WBS 1.2.4. represents the system level design activities. It is part of the COSYSMO estimation. However, COCOMO II also estimates the design effort. For software-intensive systems, the COCOMO II estimate can span across the entire system design activities. As the result, double coverage may result between the two models. This is one of the most common problems encountered – to know where one model stops and the other picks up or where the handover point is between the models. COSYSMO, intrinsically a systems model, provides complete coverage of the system-level design activities regardless of the system type. COCOMO II, estimating Computer Software Configuration Items (CSCIs) based on the component lines of code count, should confine its coverage at the same level. A possible handover point can be determined based on requirements. Since a COSYSMO estimate corresponds to the 'system requirements' at the system sell-off level, COCOMO II estimate should start with derived and decomposed requirements at the CSCI level. In other words, while COSYSMO is responsible for all system requirements and its design and testing at the system level, COCOMO II should be accountable for the detail design effort corresponding to the CSCI-level requirements, derived and flown down from the system level.

Similarly, the DT&E activities, represented by WBS 1.5.3., should also be divided along the boundary of system level vs. subsystem or CSCI level. We recommend that the COCOMO II only take on the responsibility of verifying responsible CSCI-level requirements, while leaving the complete system-level requirement verification tasks to COSYSMO.

Specialty Engineering under WBS 1.2.8. represents the engineering domains that are not typical of the main engineering effort but required to address special system implementation issues. Examples of this category include tasks such as electromagnetic interference, electrical grounding, environmental engineering, certification and accreditation, security, information assurance, and compliance to local, state and federal guidelines and codes. COCOMO II typically does not estimate this scope. However, while this scope is generally considered part of SE scope, certain efforts may require special electrical or mechanical expertise or skills and sophisticated hardware models could provide coverage. For consistency, the recommended strategy is for systems to take over the entire responsibility and, thus, have a COSYSMO estimate provide the coverage. In practical implementation, the systems IPT would be responsible for managing the task and budget estimate, but may employ the required expertise from other functions, e.g. electrical and mechanical engineering, as required. When taking this approach, care must be given to exclude the same scope from the corresponding hardware estimates.

3.3. Uncertainties

There are several areas of 'uncertainty', represented by the symbol 'U' in the cross-reference matrix. The uncertainty indicates that there has not been an explicit policy in COSYSMO regarding these effort categories. As a consequence, the implementation has been inconsistent. The tasks assessed in this category are listed in Table 4.

Most of these areas involve discrepancy report (DR) work off within the PMP or the Peculiar and Common Support Equipments. DR maintenance mainly includes the resolution of DRs that are relatively small in scope and effort and that are local to the associated CI. In practical system development projects, the DR resolution is a natural part of development and responsible project managers plan them proactively. However, by definition, these tasks are at the component or CI level and, therefore, should be considered by the responsible CI-level models. We recommend all DR maintenance efforts be excluded from the COSYSMO estimation scope, except in response to DRs resulting from system-level IATC (integration, assembly, test and check-out), developmental test and evaluation and OT&E. There is a subtlety, however, in this area. While the effort for resolving DRs is outside the systems scope, the effort in identifying DRs generally starts at the system level and, therefore, should be included in the COSYSMO estimates.

The next area of uncertainty is WBS 1.5.5 – ST&E Mock-ups/Prototypes/Simulations and Test Equipment. These are special system or subsystem mock-ups, engineering test equipment or System Integration Labs in support of Design Solution Verification, DT&E or OT&E activities. However, the element is not explicitly covered by COSYSMO estimates as none of its drivers represents the system at this level. Since the development of test equipment can be a significant endeavour of its own, we recommend exclusion of this scope from the COSYSMO estimate. Instead, we propose treatment of the equipment as a separate 'system' and development of a discrete estimate independent from that of the prime system by applying appropriate models and methods at the next level of system abstraction. If the test system is software in nature, e.g. it is cost estimated using COCOMO II based on the corresponding SLOC count.

The last area of uncertainty is WBS 1.10.2. – Contractor Technical Support. This effort involves the services provided by the contractor, typically onsite, during or immediately after system activation and final turnover. The effort is not consistently estimated by COSYSMO as it may be construed as SP scope. For the sake of consistency, we recommend classifying this task as part of the systems scope and including it in the COSYSMO estimate. In fact, the requirements for such support should be identified in counting size or cost drivers.

4. Summary of other harmonisation considerations

This section provides a brief summary of the results of analysis for five other areas of consideration for the harmonisation of the systems and software cost estimation models. An analysis of cost drivers was performed to determine whether the models account for common drivers when they are relevant to both systems and software engineering. Most of the drivers have mappings between the models, albeit different in granularity or handling. The potential concerns have been covered in the gaps or recommendations.

The harmonisation efforts exposed the commonality of terminology, constructs, life cycle phases, and units to ensure common interpretation, ability to scope/define the estimation, and ease to communicate data requirements and results. This review indicated that additional commonality could improve concurrent usage, but is not essential to the harmonisation.

The potential use of common size drivers was examined to determine whether it would improve the utility of the estimation models and the ability to use a common set of size drivers. This analysis showed that the use of common size drivers is not essential to harmonisation, but may add utility to COCOMO and to COSYSMO, especially for early life cycle estimation, when only needs and high-level functional requirements are known.

There was an attempt to examine the base assumptions of the models to ensure that the models were built on a consistent set of valid assumptions. It was not possible to complete this analysis, since the assumptions for COSYSMO have not been formally documented.

Finally, an early effort is under way in exploring a holistic, unified model by combing systems and software drivers to create a total engineering estimate (Wang *et al.* 2009). This effort approaches the same problem from a different angle, attempting to directly estimate the total engineering scope instead of building the estimate as a sum of smaller parts as assumed in this article.

Special attention was also given to all of the evolving recommendations to determine whether there were any compatibility issues from the findings or recommendations. This was intended to ensure that any recommended changes would not cause adverse impact on the model usage for addressing their independent areas of estimation. There was no apparent compatibility issues (backward compatibility or with other models in COCOMO Suite) identified during the analysis.

There have been ongoing efforts within the ISO/IEC/IEEE communities to harmonise systems and software life cycle processes (Roedler 2008). The results of these efforts may have an impact on the integration of the cost estimation models. As such, there has been an ongoing exchange of information between the teams working these efforts to ensure consistency and mutual benefit from the analysis and actions. A review of the current plans in the standards harmonisation indicated low risk of impacts that would cause inconsistency.

5. Conclusion and summary of recommendations

This article summarised the effort in harmonising the systems and software estimates provided COCOMO and COSYSMO, respectively. The analysis is conducted in a cross-reference framework between an engineering WBS representing a contract project scope and a generic organisational breakdown structure representing five top-level functions. Assignments of task ownerships to functions are identified. The estimation coverage of these tasks is also assessed based on the current model definitions of COCOMO II and COSYSMO. The result is a list of potential gaps and overlaps between the two models, as well as uncertainties as the result of inconsistent application of these models. The list is analysed element by element in terms of cause and recommendations are provided for the resolution of the identified gaps and overlaps.

Some of the key recommendations include the following:

- Use system-level requirements vs. subsystem or CSCI level requirements as point of handover between models
- For those tasks that are recommended to be inclusive in either systems or software scope, ensure to aggregate the related historical actual effort correctly in the calibration data during data collection
- For those tasks that are recommended to be exclusive of either systems or software scope, ensure the effort is accounted for correctly using the appropriate methods, e.g. through discrete estimates
- For those tasks that are estimated outside of the systems scope, say by discrete estimates, ensure the corresponding system requirements are treated appropriately by applying the reuse model in COSYSMO

Additional recommendations based on the other areas of consideration include:

- Standard phase alignment for both models, per definitions used in ISO/IEC 15288 and 12207
- Establish means to adequately account for recursion (at level of hands-off to SW), needed to resolve gaps
- Establish operational guidance to minimise variation in usage
- Add Guidance to COSYSMO drivers to:
  - Account for constraints (e.g. time and storage) as requirements in the size.
  - Describe volatility covered in requirements/architecture understanding
- Look into ability to use COSYSMO size drivers in COCOMO for early estimates
- Add documented list of assumptions to COSYSMO

Work is in progress to document these guidelines in the COSYSMO 2.0. Practitioners' Guide. We emphasise that this effort to harmonise the systems and software estimation is still work in progress. Long-term goal is to provide recommended guidelines for integrating all functional models including hardware estimation. In the meantime, this work establishes an approach and provides a set of recommendations or strategies for practitioners of these models to enable their effort in developing integrated, total engineering estimation.

Acknowledgements

Many of the results presented have benefitted from inputs of subject matter experts who participated in the COSYSMO Workshop at the University of Southern California in March 2008 in Los Angeles, CA, and the Practical Software & Systems Measurement workshop in July 2008 in Mystic, CT.

References

ANSI/EIA, 1999. *ANSI/EIA-632-1988 processes for engineering a system*. New York, NY: American National Standards Institute.

Boehm, B.W., *et al.*, 2000. *Software cost estimation with COCOMO II*. Upper Saddle River, NJ: Prentice Hall.

Boehm, B.W., 2000. Unifying software engineering and systems engineering. *IEEE Computer*, 33 (3), 114–116.

Boehm, B.W., 2006. Some future trends and implications for system and software engineering processes. *Systems Engineering*, 9 (1), 1–19.

Boehm, B.W., 2005. Revisiting software engineering economics. *Keynote Address, IEEE Equity Conference*. Amsterdam, The Netherlands.

MIL-HDBK, 2005. *MIL-HDBK-881A Work breakdown structures for defense materiel items*. Washington, DC: Department of Defense.

ISO/IEC, 2002. *ISO/IEC 15288:2002(E) Systems engineering – system life cycle processes*. Geneva, Switzerland: International Organization for Standardization.

ISO/IEC, 2008. *ISO/IEC 12207:2008 Systems and software engineering – software life cycle processes*. Geneva, Switzerland: International Organization for Standardization.

OUSD (AT&L)/SSE-USC/CSSE. *Workshop on integrating systems and software engineering with the incremental commitment model,* Washington: DC, 15–17 July 2008.

Pyster, A. and Turner, R., 2008. A framework for integrating systems and software engineering. *NDIA Systems Engineering Conference*. San Diego, California.

Roedler, G., 2008. Is an integrated set of systems and software standards possible? *Systems and Software Technology Conference*. Salt Lake City, Utah.

Singh, R., 1995. Harmonization of software engineering and system engineering standards, *2nd IEEE Software Engineering Standards Symposium*. Montreal, Canada.

Smith, A., 1776. *An inquiry into the nature and causes of the wealth of nations*. London, UK: William Clowes and Sons.

Workshop on Integrating Systems and Software Engineering, 2007. *22nd International annual forum on COCOMO and system/software cost modeling*, Los Angeles, CA.

Valerdi, R. and Lane, J., 2004. Steps toward model unification for software, systems engineering and systems of systems. *19th forum on COCOMO and software cost modeling*, Los Angeles, CA.

Valerdi, R., 2008. *The constructive systems engineering cost model (COSYSMO): quantifying the costs of systems engineering effort in complex systems*. Saarbrücken, Germany: VDM Verlag.

Wang, G., *et al*., 2008. COSYSMO reuse extension. *18th INCOSE International Symposium*. Utrecht, The Netherlands.

Wang, G., *et al*., 2009. Towards a holistic, total engineering cost model. *19th INCOSE International Symposium*. Singapore.

Appendix

WBS	Description	Systems (COSYSMO)	Software (COCOMO II)	Hardware	Support ability	PEM
1.0.	*System/project*					
1.1.	Integrated project management (IPM)					
1.1.1.	Technical management					X
1.1.2.	Technical reviews					Y
1.1.3.	Change management					Y
1.1.4.	Technical process and quality management					X
1.1.5.	Acquisition and supply management (subcontract and technical oversight)					Y
1.1.6.	Information technology and infrastructure					X
1.1.7.	Dismantle and disposal				X	
1.2.	*Systems engineering*					
1.2.1.	Systems engineering management	Y				
1.2.2.	ConOps and stakeholder analysis	Y				
1.2.3.	Requirement analysis and management	Y				
1.2.4.	PMP design	Y	S			
1.2.5.	Modelling and simulation	Y				
1.2.6.	Logistics engineering				Y	
1.2.7.	Reliability, maintainability, safety (RMS) engineering	Y				
1.2.8.	Specialty engineering	Y		X		
1.3.	*PMP*					
1.3.1.	Subsystem/CI 1…n (specify names)					
1.3.1.1.	IPT engineering management	X		X		
1.3.1.2.	Design	X		X		
1.3.1.3.	Design analysis and verification	X		X		
1.3.1.4.	Construction/acquisition	X		X		
1.3.1.5.	IATC	X		X		
1.3.1.6.	DR maintenance	U		X		
1.3.2.	PMP application software					
1.3.2.1.	IPT engineering management		S			
1.3.2.2.	Design		S			
1.3.2.3.	Construction/acquisition		S			
1.3.2.4.	IATC		S			
1.3.2.5.	DR maintenance		S			
1.3.3.	PMP system software					
1.3.3.1.	IPT engineering management	X				
1.3.3.2.	Design	X				
1.3.3.3.	Construction/acquisition	X				
1.3.3.4.	IATC	X				
1.3.3.5.	DR maintenance	U				
1.3.4.	PMP IATC	Y				
1.3.5.	Operations/production support	Y				
1.4.	*Platform integration*					
1.4.1.	External interface and technical liaison coordination	Y				
1.4.2.	Transition to use	Y				
1.4.3.	Initial spares and repair parts				X	
1.5.	*ST&E*					
1.5.1.	ST&E management	Y				
1.5.2.	Design solution verification	Y				
1.5.3.	DT&E	Y	S			
1.5.4.	OT&E	Y				
1.5.5.	ST&E mock-ups/prototypes/simulations and test equipment	U	S	X		
1.5.6.	ST&E test and evaluation support	X				
1.5.7.	ST&E test facilities	X				
1.6.	*Training*					
1.6.1.	Equipment				X	
1.6.2.	Services				X	
1.6.3.	Facilities				X	

(*continued*)

(*Continued*).

WBS	Description	Systems (COSYSMO)	Software (COCOMO II)	Hardware	Support ability	PEM
1.7.	*Data management*					
1.7.1.	Technical publications				X	
1.7.2.	Engineering data				X	
1.7.3.	Management data				X	
1.7.4.	Support data				X	
1.7.5.	Data repository				X	
1.8.	*Peculiar support equipment*					
1.8.1.	Peculiar test and measurement equipment					
1.8.1.1.	IPT engineering management	Y	S	X		
1.8.1.2.	Design	Y	S	X		
1.8.1.3.	Design analysis and verification	Y		X		
1.8.1.4.	Construction/acquisition	Y	S	X		
1.8.1.5.	IATC	Y	S	X		
1.8.1.6.	DR maintenance	U	S	X		
1.8.2.	Support and handling equipment					
1.8.2.1.	IPT engineering management	Y	S	X		
1.8.2.2.	Design	Y	S	X		
1.8.2.3.	Design analysis and verification	Y		X		
1.8.2.4.	Construction/acquisition	Y	S	X		
1.8.2.5.	IATC	Y	S	X		
1.8.2.6.	DR maintenance	U	S	X		
1.9.	*Common support equipment*					
1.9.1.	Common test and measurement equipment					
1.9.1.1.	IPT engineering management	Y	S	X		
1.9.1.2.	Design	Y	S	X		
1.9.1.3.	Design analysis and verification	Y		X		
1.9.1.4.	Construction/acquisition	Y	S	X		
1.9.1.5.	IATC	Y	S	X		
1.9.1.6.	DR maintenance	U	S	X		
1.9.2.	Support and handling equipment					
1.9.2.1.	IPT engineering management	Y	S	X		
1.9.2.2.	Design	Y	S	X		
1.9.2.3.	Design analysis and verification	Y		X		
1.9.2.4.	Construction/acquisition	Y	S	X		
1.9.2.5.	IATC	Y	S	X		
1.9.2.6.	DR maintenance	U	S	X		
1.10.	*Operational/site activation*					
1.10.1.	System assembly, installation and checkout (onsite)	Y				
1.10.2.	Contractor technical support				U	
1.10.3.	Site construction				X	
1.10.4.	Site conversion/upgrade				X	
1.11.	*Industrial facilities*					
1.11.1.	Construction				X	
1.11.2.	Acquisition/modernisation				X	
1.11.3.	Maintenance				X	