Harmonizing Systems and Software Cost Estimation

Gan Wang
BAE Systems
Reston, VA
gan.wang@baesystems.com

Ricardo Valerdi
MIT
Cambridge, MA
rvalerdi@mit.edu

Garry J. Roedler
Lockheed Martin
Philadelphia, PA
garry.j.roedler@lmco.com

Aaron Ankrum
BAE Systems
Reston, VA
aaron.ankrum@baesystems.com

John E. Gaffney, Jr.
Lockheed Martin
Rockville, MD
j.gaffney@lmco.com

Abstract. The objective of this paper is to examine the gaps and overlaps between software engineering and systems engineering cost models with intent to harmonize the estimates in engineering estimation. In particular, we evaluate the central assumptions of the COSYSMO and COCOMO II models and propose an approach to identify gaps and overlaps between them. 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 systems and software engineering effort for software-intensive systems.

Introduction

At the dawn of the Industrial Revolution in the late 18th 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 specialization of technical tasks increases productivity and specialization is further needed as the size of the market increases. There are clear parallels to large aerospace/defense systems: the specialization of systems engineers and software engineers is increasingly important as organizations wish to increase productivity in large marketplaces in light of increased technical complexity and numerous organizational layers.

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

Barry (2007) claims that “you can’t do good software engineering by neglecting systems engineering”. The inverse is also true: you can’t do good systems engineering 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 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 organizations often only consider the high-level joint objective of software engineering and systems engineering: 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 functions for control of the technical design and implementation process. Nevertheless, as systems become increasingly complex, the needs for the harmonization of software engineering and systems engineering become more critical. Efforts to integrate SW and SE standards (Roedler 2008; Singh 1995) and processes (Boehm 2000; 2006) have provided useful guidance. Studies have been keyed to integrating systems engineering 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 harmonization 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.

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. For instance, both COSYSMO and COCOMO estimate design activities. One question is where the SE model stops and the SW model starts. Another question is where both models together cover the full set of engineering activities. Similarly, which part of the integration and testing effort is considered by which model?

In fact, the scope overlap and gaps concerns 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 labor 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.

The need from practical application of these cost models is to establish a consistent guideline for determining the model scopes and harmonizing the functional estimates to be able to effectively generate a coherent system bid.

Commercial vendors of cost estimating models have long attempted to address the same issue. Both PRICE (by PRICE Systems) and SEER (by Galorath, Inc.) have organized 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 harmonization of these cost models. These include:

- USC COSYSMO Workshop, March 2008 – this workshop scoped the problem, prioritized the needs for harmonization, and identified that operational guidance may be as significant to addressing the harmonization as the modeling constructs and driver definitions.
- PSM Workshop on Harmonizing COSYSMO and COCOMO, July 2008 – this workshop analyzed the typical Work Breakdown Structure (WBS) elements against functional responsibilities, assessed the element coverage against COSYSMO and COCOMO, 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 harmonization 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 paper 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 harmonizing COCOMO II (Boehm, et al 2000) for SW and COSYSMO (Valerdi 2008) for SE 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 1a. 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” is covered by the model they are using.

Second, we identify the gaps between the two models as shown in Figure 1b. 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.

a. Overlaps b. Gaps

Figure 1 – Overlaps and Gaps between SW and SE Models

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 paper will go further into the details of these considerations.

The final section of the paper provides next steps that are required to harmonize the relationships between COCOMO II and COSYSMO. These include standard phase/stage alignment for both models (per definitions used in ISO/IEC 15288 and 12207), 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.

Analysis Approach

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

For this purpose, we define a generic, reference work breakdown structure (WBS) that represents the engineering scope of a system development project by a contractor. This WBS is defined based on MIL-HDBK-881A and the EIA/ANSI 632 standard, by tailoring a reference structure from MIL-HDBK-881 A while considering the thirty-three technical activities defined in 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 A.

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 work breakdown structure (WBS)

In addition, we define a generic organization breakdown structure (OBS) with five standard "functions". They are systems engineering (SE), software engineering (SW), hardware engineering (HW), supportability engineering (SP), and project engineering management (PEM). Hardware engineering in this context includes the traditional electrical and mechanical engineering functions. The project engineering management "function" is comprised of all engineering management activities at the project level, which include the overall project planning, monitor and control activities responsible for by 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 A, with the work breakdown up and down on the left and the organizational 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, for example, PRICE-H or SEER-H) is assumed for the HW estimates. The objective of this exercise is two fold. 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 A. 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 emphasize 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 organizations or different projects in the same organization. This guideline in no way should dictate how a project should plan 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 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 A. 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.

COCOMO II or COnstructive COst MOdel II (Boehm, et al 2000), developed at the University of Southern California (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 variety of applications supporting decision making related to software development. The Constructive Systems Engineering Cost Model (COSYSMO) is a model that can help people reason about the economic implications of systems engineering 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 modeling approach used to estimate the quantity of systems engineering labor, in terms of person months, required for the conceptualization, 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 organization'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 organization-neutral that can be adopted or adapted by specific organizational implementations.

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) Conceptualize, (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	1 size factor, 5 scale factors, and 18 effort multipliers	4 size factors, 1 scale factor, 14 effort multipliers
Represents diseconomy of scale through	Five scale factors	One exponential factor for size

We have conducted several roundtable workshops between different groups of the stakeholders in the community, including those at the International COCOMO Forum at the University of Southern California and the annual Practical Systems and Software Measurement (PSM) User Group meeting. These workshops achieved agreement among stakeholders on the responsibility assignment as well as the model coverage, as described in the following section.

Analysis of the Model Scopes

The WBS vs. OBS cross-reference matrix as shown in Appendix A 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 "eighty-percentile" practices. In working with stakeholders, we stress that there will be exceptions to the assignments made and it could be different from the last program one has worked on. We recognize 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 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 COSYSMO and COCOMO II. On the other hand, if either appears for a WBS element, then this is potentially an under-lapped area.

A quick glance at the Appendix A 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 organizations. 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 – Prime Mission Product (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 configuration item (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 COTS-based 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. Secondly, Supportability Engineering is identified as a separate function. However, we recognize that in many companies, it is part of the systems engineering organization. In this context, we keep the two separate and attempt to address any gaps and overlaps between the two to provide more definitive guidance.

Now, let's 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.

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 COSYSMO nor COCOMO II. 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 systems engineering 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 configuration items 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 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 System Test & Evaluation (ST&E) are support in nature relative to DT&E and 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.

Table 2 – Potential model gaps and recommended resolutions

WBS	Element	Potential Gaps	Recommendations and Remedies
1.1	*Integrated Project Management (IPM)*		
1.1.1	Technical Management	Proejct technical management overhead under PMO budget line	COSYSMO to provide coverage of the scope by including the actual effort in the calibration data
1.1.4	Technical Process and Quality Management	Process and quality management traditionally project management overhead under PMO budget	
1.1.6	Information Technology & Infrastructure	IT for project infrastructure under PMO	
1.1.7	Dismantle and Disposal	System retirement and disposal strategy and planning under supportability engineering	
1.3	*Prime Mission Product (PMP)*		
1.3.3	PMP System Software		
1.3.3.1	IPT Engineering Management	System software are typically COTS products acquired and integrated, assigned as the systems responsibiltiy. It is at the subsystem or component level and, therefore, not estimated by COSYSMO	Accounted for as a discrete estimate outside of the system scope and to be excluded from COSYSMO estimate. When added into the total engineering estimate, the assocated system requirements should be considered as the "Adopted" reuse category in COSYSMO for the system-level estimate.
1.3.3.2	Design		
1.3.3.3	Construction/Acquisition		
1.3.3.4	Integration, Assembly, Test & Checkout (IATC)		
1.4	*Platform Integration*		
1.4.3	Initial Spares & Repair Parts	Traditionally supporability engineering responsibility	COSYSM to provide coverage of the scope by including the actual effort in the calibration data
1.5	*System Test & Evaluation (ST&E)*		
1.5.6	ST&E Test & Evaluation Support	Suporting effort to DT&E and OT&E but not direct part of these activities	COSYSMO to provide coverage of the scope by including the actual effort in the calibration data and by considering the facility requirements at the system level
1.5.7	ST&E Test Facilities	Test facility	
1.6	*Training*		
1.6.1	Equipment	Traditionally supportability engineering responsibility	To keep the scope outside of the systems and software, and develop discreate estimates using other methods
1.6.2	Services		
1.6.3	Facilities		
1.7	*Data Management*		
1.7.1	Technical Publications	Traditionally supportability engineering responsibility	To keep the scope outside of the systems and software, and develop discreate estimates using other methods
1.7.2	Engineering Data		
1.7.3	Management Data		
1.7.4	Support Data		
1.7.5	Data Repository		
1.10	*Operational / Site Activation*		
1.10.3	Site Construction	Traditionally supportability engineering responsibility	To keep the scope outside of the systems and software, and develop discreate estimates using other methods
1.10.4	Site Conversion / Upgrade		
1.11	*Industrial Facilities*		
1.11.1	Construction	Traditionally supportability engineering responsibility	To keep the scope outside of the systems and software, and develop discreate estimates using other methods
1.11.2	Acquisition / Modernization		
1.11.3	Maintenance		

Elements for Training and Data Management, under WBS 1.6 and WBS 1.7, are traditionally within supportability engineering responsibility. Neither COSYSMO nor COCOMO II estimates 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 they be kept outside of the either model scope and estimated using other methods.

Potential Overlaps:

There are three major areas of potential overlaps identified between COSYSMO and COCOMO II. They are summarized 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 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 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.

Table 3 - Potential model overlaps and recommended resolutions

WBS	*Element*	*Potential Overlaps*	*Recommendations and Remedies*
1.2	Systems Engineering		
1.2.4	Prime Mission Product (PMP) Design	COSYSMO provides coverage; COCOMO II may provide coverage if it is software-intensive system	COSYSMO provides complete coverage for system-level design activiites; COCOMO II excludes the system activities but assumes the complete detail design activities at the CSCI level
1.2.8	Specialty Engineering	Scope may be coverage by COSYSMO as well as the hardware model depending on the nature of speciality	COSYSMO estimate provides complete coverage of the scope
1.5	*System Test & Evaluation (ST&E)*		
1.5.3	Development Test & Evaluation (DT&E)	COSYSMO provides coverage; COCOMO II may provide coverage if it is software-intensive system	COSYSMO provides complete coverage for the system-level DT&E activities; COCOMO II provides complete coverage for component testing at the CSCI level

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 as systems engineering 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 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.

Uncertainty:

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 discrepancy reports 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 discrepancy reports resulting from system-level IATC (integration, assembly, test, and check-out), developmental test and evaluation, and operational test and evaluation. There is a subtlety, however, in this area. While, the effort for resolving DRs are outside the systems scope, the effort in identifying DRs generally start at the system level and, therefore, should be included in the COSYSMO estimates.

Table 4 - Areas of uncertainty or inconsistent estimation scope

WBS	Description	*Potential Inconsistencies*	*Recommendations and Remedies*
1.3	*Prime Mission Product (PMP)*		
1.3.1	Subsystem / Configuration Item (CI) 1…n (Specify Names)	DR work-off are typically at the subsystem or CI level. The associated efforts are part of the CI development and should be included in the estimation effort of the responsible models	To exclude DR maintenance effort from the COSYSMO and to include the corresponding scope in the COCOMO II estimates by ensuring the calibration data includes the corresponding effort from the historical data
1.3.1.6	Discrepancy Report (DR) Maintenance		
1.3.3	PMP System Software		
1.3.3.5	Discrepancy Report (DR) Maintenance		
1.8	*Peculiar Support Equipment*		
1.8.1	Peculiar Test & Measurement Equipment		
1.8.1.6	Discrepancy Report (DR) Maintenance		
1.8.2	Support & Handling Equipment		
1.8.2.6	Discrepancy Report (DR) Maintenance		
1.9	*Common Support Equipment*		
1.9.1	Common Test & Measurement Equipment		
1.9.1.6	Discrepancy Report (DR) Maintenance		
1.9.2	Support & Handling Equipment		
1.9.2.6	Discrepancy Report (DR) Maintenance		
1.5	*System Test & Evaluation (ST&E)*		
1.5.5	ST&E Mock-ups / Prototypes / Simulations & Test Equipment	The effort is not consistently estimated by COSYSMO as it counts no drivers at this level. However, the correspondiing effort may be included in the calibration data	To exclude the development from COSYSMO system estimate. Instead, generate discrete estimate for this element as a separate "system"
1.10	*Operational / Site Activation*		
1.10.2	Contractor Technical Support	The effort is not consistently estimated by COSYSMO as it may construde as supportability engineering scope	To include the scope in the COSYSMO estimate

The next area of uncertainty is WBS 1.5.5 – ST&E Mock-ups / Prototypes / Simulations & Test Equipment. These are special system or subsystem mock-ups, engineering test equipment, or System Integration Labs (SILs) 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 endeavor of its own, we recommend exclusion of this scope from the COSYSMO estimate. Instead, treat the equipment as a separate "system" and develop discrete estimate independently 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, for example, estimate it 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 supportability engineering 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.

Summary of Other Harmonization Considerations

This section provides a brief summary of the results of analysis for five other areas of consideration for the harmonization 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 harmonization efforts reviewed 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 harmonization.

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 harmonization, 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. There has been a recommendation to the COSYSMO Users Group to document the assumptions from the dissertation and historical notes on the development of the COSYSMO model.

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 essentially approaches the same problem from a different angle, attempting to directly estimate the total engineering scope instead of building the whole from parts as assumed in this paper.

A 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 arena to harmonize 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 harmonization indicated low risk of impacts that would cause inconsistency.

Conclusion and Summary of Recommendations

This paper summarized the effort in harmonizing the systems and software estimates provided COSYSMO and COCOMO, respectively. The analysis is conducted in a cross-reference framework between an engineering work breakdown structure representing a contract project scope and a generic organizational 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 COSYSMO and COCOMO II. 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 analyzed element by element in terms of cause and recommendations are provided for the resolution of the identified gaps and overlaps.

Some of key recommendations include:

- 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 minimize variation in usage
- Add Guidance to COSYSMO Drivers to:
    - Account for constraints (e.g. Time & 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 emphasize that this effort to harmonize 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.

Appendix A

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 & Supply Management (Subcontract & Technical Oversight)					Y
1.1.6	Information Technology & Infrastructure					X
1.1.7	Dismantle and Disposal				X	
1.2	*Systems Engineering*					
1.2.1	Systems Engineering Management	Y				
1.2.2	ConOps & Stakeholder Analysis	Y				
1.2.3	Requirement Analysis & Management	Y				
1.2.4	Prime Mission Product (PMP) Design	Y	S			
1.2.5	Modeling & 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	*Prime Mission Product (PMP)*					
1.3.1	Subsystem / Configuration Item (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	Integration, Assembly, Test & Checkout (IATC)	X		X		
1.3.1.6	Discrepancy Report (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	Integration, Assembly, Test & Checkout (IATC)		S			
1.3.2.5	Discrepancy Report (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	Integration, Assembly, Test & Checkout (IATC)	X				
1.3.3.5	Discrepancy Report (DR) Maintenance	U				
1.3.4	PMP Integration, Assembly, Test & Checkout (IATC)	Y				
1.3.5	Operations/Production Support	Y				
1.4	*Platform Integration*					
1.4.1	External Interface & Technical Liaison Coordination	Y				
1.4.2	Transition to Use	Y				
1.4.3	Initial Spares & Repair Parts				X	
1.5	*System Test & Evaluation (ST&E)*					
1.5.1	ST&E Management	Y				
1.5.2	Design Solution Verification	Y				
1.5.3	Development Test & Evaluation (DT&E)	Y	S			
1.5.4	Operational Test & Evaluation (OT&E)	Y				
1.5.5	ST&E Mock-ups / Prototypes / Simulations & Test Equipment	U	S	X		
1.5.6	ST&E Test & 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	
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 & 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	Integration, Assembly, Test & Checkout (IATC)	Y	S	X		
1.8.1.6	Discrepancy Report (DR) Maintenance	U	S	X		
1.8.2	Support & 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	Integration, Assembly, Test & Checkout (IATC)	Y	S	X		
1.8.2.6	Discrepancy Report (DR) Maintenance	U	S	X		
1.9	*Common Support Equipment*					
1.9.1	Common Test & 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	Integration, Assembly, Test & Checkout (IATC)	Y	S	X		
1.9.1.6	Discrepancy Report (DR) Maintenance	U	S	X		
1.9.2	Support & 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	Integration, Assembly, Test & Checkout (IATC)	Y	S	X		
1.9.2.6	Discrepancy Report (DR) Maintenance	U	S	X		
1.10	*Operational / Site Activation*					
1.10.1	System Assembly, Installation & Checkout (On-Site)	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 / Modernization				X	
1.11.3	Maintenance				X	

References

Boehm, B. W., Abts, C., Brown, A. W., Chulani, S., Clark, B., Horowitz, E., Madachy, R., Reifer, D. J. and Steece, B. (2000). *Software Cost Estimation With COCOMO II*, Upper Saddle River, NJ: Prentice Hall.

Boehm, B. W., Unifying software engineering and systems engineering, *IEEE Computer*, Vol. 33, No. 3, pp. 114-116, March 2000.

Boehm, B. W. (2006). Some Future Trends and Implications for System and Software Engineering Processes, *Systems Engineering*, 9(1), pp. 1-19.

Boehm, B. W. (2007). Revisiting Software Engineering Economics, Keynote Address, IEEE Equity.

OUSD(AT&L)/SSE-USC/CSSE *Workshop on Integrating Systems and Software Engineering with the Incremental Commitment Model*, Washington, DC, July 15-17, 2008

Pyster, A., Turner, R., A Framework for Integrating Systems and Software Engineering, *NDIA Systems Engineering Conference*, October 2008.

Roedler, G. (2008). "Is An Integrated Set of Systems and Software Standards Possible?", *Systems and Software Technology Conference*, 2008.

SEER® by Galorath, http://www.galorath.com/index.php

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

Smith, A. (1776). *An Inquiry Into the Nature and Causes of the Wealth of Nations*, London: William Clowes and Sons.

TruePlanning® by PRICE Systems, http://www.pricesystems.com/products/price_trueplanning.asp

Workshop on Integrating Systems and Software Engineering, *22nd International Annual Forum on COCOMO and System/Software Cost Modeling*, Los Angeles, CA, October 31—November 2, 2007

Valerdi, R., Lane, J., Steps Toward Model Unification for Software, Systems Engineering and Systems of Systems, *19th Forum on COCOMO and Software Cost Modeling*, Los Angeles, CA, October 2004.

Valerdi, R., *The Constructive Systems Engineering Cost Model (COSYSMO): Quantifying the Costs of Systems Engineering Effort in Complex Systems*, VDM Verlag, 2008.

Wang, G., Valerdi, R., Ankrum, A., Millar, C., Roedler, G., COSYSMO Reuse Extension, *18th INCOSE International Symposium*, June 2008.

Wang, G., Shernoff, A., Turnidge, J., Valerdi, R., Towards a Holistic, Total Engineering Cost Model, *19th INCOSE International Symposium*, July 2009.

Acknowledgements

Many of the results presented have benefitted from inputs of subject matter experts who participated in the Practical Software & Systems Measurement workshop in July 2008 in Mystic, Connecticut.

Biographies

Gan Wang, Ph.D., is an EI&S Engineering Fellow at BAE Systems. He has been engaged in the research and development of systems engineering processes, engineering cost modeling and life cycle cost estimation, decision support methods, and system-of-systems engineering and management methodologies. Prior to joining BAE Systems, Dr. Wang has spent many years developing real-time geospatial data visualization applications for mission planning and rehearsal, battlefield management, command and control (C2), flight simulation, and aircrew physiological training systems. He also developed control systems and aircraft simulation models for various man-in-the-loop flight training systems. He has over twenty years of experience in systems engineering, software development, research and development, and engineering management involving complex, software-intensive systems.

Ricardo Valerdi, Ph.D., is a Research Associate with the Systems Engineering Advancement Research Initiative and the Lean Advancement Initiative at MIT. He received his doctoral degree in systems engineering from USC in 2005, where he developed the COSYSMO model for systems engineering cost estimation. He is also a Senior Member of the Technical Staff at the Aerospace Corporation. Prior to his doctoral studies, Dr. Valerdi was a systems engineer at Motorola Radio Network Solutions Group. He serves on the INCOSE Board of Directors as Treasurer.

Garry Roedler is the Senior Manager of Systems Engineering (SE) at the Lockheed Martin Engineering Process Improvement Center. He is responsible for the development/selection of SE processes, implementation assets, training, and tools for the corporation towards an integrated set of SE enablers to aid program performance. This role also has responsibility for managing the corporate councils for SE, Test & Evaluation (T&E) and Specialty Engineering. Previously, he was the Engineering Process Integration Manager for LM Integrated Systems & Solutions, responsible for strategic planning of technology needs, process technology development and infusion, and process maintenance and improvement of engineering processes. Prior to that, he chaired the Systems Integration Process Review Board for LM Management and Data Systems (M&DS), focusing on process improvement and achievement/sustainment of Level 5 CMM/CMMI objectives, including a world first to achieve Level 5 ratings in the SE-CMM.

Garry has over 27 years experience in engineering, measurement, and teaching and holds degrees in mathematics education and mechanical engineering from Temple University. Other work includes leadership roles in various technical and standards organizations, including: US Head of Delegation and Task Group leader for ISO/IEC JTC1/SC7 Working Group 7 (software and systems engineering process standards), Practical Software and Systems Measurement (PSM) Technical Steering Group; International Council On Systems Engineering (INCOSE) Corporate Advisory Board, Technical Board and Committees; INCOSE Delaware Valley Chapter co-founder; and the IEEE Standards Association. Garry has worked on the author teams of several currently used standards, including ISO/IEC 15288, Systems Life Cycle Processes (Project Editor); ISO/IEC 15939, Measurement (Co-editor); IEEE-1220, Application and Management of the Systems Engineering Process, ISO/IEC 16085, Risk Management; IEEE 1540, Software Risk

Management. Recent collaborative products he has supported or led include development of a systems engineering cost estimation model (COSYSMO) from USC, Technical Measurement Guide from PSM/INCOSE, and the SE Leading Indicator Guide from LAI/INCOSE.

Aaron Ankrum is a Senior Principal Enterprise Architect at BAE Systems. He is currently supporting a FoS Enterprise Architecture program and has been engaged in the development of cost estimating and decision support methodologies for enterprise and capability-based engineering. Prior to this effort, Mr. Ankrum supported two other FoS efforts, providing enterprise architecture expertise. Prior to joining BAE Systems, Mr. Ankrum has spent many years developing real-time systems and executing on systems engineering efforts. He has over 18 years of experience in software development and software-intensive systems engineering and integration efforts. He also has experience in business development, corporate audit, software configuration management and engineering leadership. He also has Masters of Arts in Systems Engineering from Virginia Polytechnic and State University.

John Gaffney is a Lockheed Martin Fellow and is a staff member of the Systems & Software Resource Center (SSRC), Corporate Engineering and Technology, Lockheed Martin Corporation. He provides support on measurement and quantitative management to projects across Lockheed Martin. He has developed and continues to create new techniques and tools for defect analysis/estimation and reliability, and cost/effort estimation and related risk estimation. He was the lead coauthor of the Software Measurement Guidebook (Thomson Computer Press, 1995). John developed the SWEEP, STEER, and DIRC defect models/tools, and has contributed to the evolution of the COSYSMO systems engineering labor estimation tool capability, and developed the COSYSMOR systems engineering tool ("COSYSMO Risk/ Reuse"), which is being rolled out across Lockheed Martin and which is being provided to other organizations, gratis. COSYSMOR provides an ability to estimate the range of effort values and their probabilities. He also has developed a tool, COCOMOR ("COCOMO Risk/ Reuse"), for software engineering estimation that is analogous to COSYSMOR. He has written and presented many papers.

Before joining Lockheed Martin, John worked at IBM and the Systems and Software Consortium (formerly Software Productivity Consortium). He also worked at the National Weather Service and was a Visiting Professor of Computer Science at Polytechnic University (Brooklyn, NY), both while on leave from IBM. Later, he taught evening courses in the Johns Hopkins graduate computer science program. John holds an A.B in Engineering from Harvard, an M.S. in Electrical Engineering from Stevens Institute of Technology, and completed the course work for a Ph.D. in Statistics at American University. He is Registered Professional Engineer (P.E.), in Electrical Engineering in the District of Columbia.