Estimating Systems Engineering Reuse

Jared Fortune[1], Ricardo Valerdi[2], Barry W. Boehm[3] and F. Stan Settles[4]

[1] University of Southern California, USA, fortune@usc.edu
[2] Massachusetts Institute of Technology, USA, rvalerdi@mit.edu
[3] University of Southern California, USA, boehm@usc.edu
[4] University of Southern California, USA, settles@usc.edu

Abstract
Systems engineering reuse is the utilization of previously developed systems engineering products or artifacts such as architectures, requirements, and test plans across different projects. Such reuse is intended as a means of reducing development cost, project schedule, or performance risk, by avoiding the repetition of some systems engineering activities. Although projects involving systems engineering reuse are becoming more frequent, models or tools for estimating the cost, benefit, and overall impact on a project as a result of reusing products or artifacts have not yet been adequately developed. This paper provides an overview of systems engineering reuse and recent developments with the Constructive Systems Engineering Cost Model (COSYSMO) to estimate the effect of reuse on systems engineering effort. The overview of systems engineering reuse includes a review of how reuse is handled in other domains and results from an industry survey. The recent developments in COSYSMO presents on-going research in the creation of a reuse extension for the model such as the identification of categories of systems engineering reuse, reuse extensions for the size drivers in the model, and a revised set of cost drivers.

Key words – systems engineering, cost, estimation, reuse, COSYSMO.

1 Introduction

Complex systems have reached the point where they can no longer be developed from a clean slate [1]. In an effort to address increasing complexity while maintaining manageability, systems engineers often leverage heritage components and other legacy capabilities as a means of reducing development schedule, system cost, or performance risk. This strategy effectively reduces the amount of new effort required to develop a system by "re-using" existing capabilities. The concept of re-use is not new, as the notion of "not re-inventing the wheel" can be traced back to ancient times and is even implicit in engineering disciplines, as the reuse of qualified parts and proven strategies is essential to good engineering practice [2].

When a reuse strategy is proposed, it is assumed that reusing an artifact is more likely to require fewer resources than the amount needed to develop the artifact. However, current systems engineering cost models do not account for systems engineering reuse in their estimates of expected effort to complete the systems engineering activities.

Despite the importance of systems engineering activities to the successful development of a complex system, up until recently, only limited methods were available to estimate the required amount of systems engineering effort. Typically, methods such as rules of thumb, heuristics, percentages of total effort, or analogies were utilized as rough estimates for the necessary systems engineering effort. To provide quantifiable justification for the amount of systems engineering effort expected for a system of interest, the Constructive Systems Engineering Cost Model (COSYSMO) was developed in 2005 at the University of Southern California Center for Systems and Software Engineering (USC-CSSE), presented below in Equation 1 [3]. The basic premise behind COSYSMO is that systems engineering effort, $\mathbf{PM_{NS}}$, can be estimated as a function of four size drivers and fourteen cost drivers. The impact of reuse can be captured through the four size drivers, Φ_x, shown as a sum with different weights, w_x, applied to a variety of conditions.

$$PM_{NS} = A \cdot \left(\sum_k (w_{e,k}\Phi_{e,k} + w_{n,k}\Phi_{n,k} + w_{d,k}\Phi_{d,k}) \right)^E \cdot \prod_{j=1}^{14} EM_j$$

Where,

$\mathbf{PM_{NS}}$ = effort in Person Months (Nominal Schedule)

A = calibration constant derived from historical project data

k = {REQ, IF, ALG, SCN}

$\mathbf{w_x}$ = weight for "easy", "nominal", or "difficult" size driver

Φ_x = quantity of "k" size driver

E = represents (dis)economies of scale

EM = effort multiplier for the j_{th} cost driver. The geometric product results in an overall effort adjustment factor to the nominal effort.

Equation 1 – *COSYSMO Operational Equation.*

Since that time, COSYSMO has been widely accepted in industry, government and academia. To date, several proprietary versions of COSYSMO have been developed by aerospace contractors, multiple commercial cost estimation software packages have incorporated the tool, and graduate courses at University of Southern California, Massachusetts Institute of Technology, University of California San Diego, and George Mason University have integrated the model into their graduate coursework on cost estimation.

Although COSYSMO has been extremely successful and useful to many systems engineers, the model is not without its limitations. Supporters noted that during some COSYSMO implementations, large errors were observed between the model estimates and actuals. Upon further investigation, it was discovered that organizations that experienced these errors had a significant amount of systems engineering reuse in their projects, which was not adequately handled by the existing version of the model [4].

The need for addressing systems engineering reuse in COSYSMO stems from the fact that the model assumes all systems are "built from scratch". In other words, the model assumes that all systems engineering activities and resulting artifacts will need to be completed as new, and no previous systems engineering activities (and their associated effort) are reused. Frequently, a system of interest is related to a previous system such that some systems engineering activities (and the results of those activities, artifacts) can be leveraged. As a result, this research addresses the question, when does systems engineering reuse save effort and when does it cost effort?

Given the broad adoption of COSYSMO, continual development and improvement of the model is both needed and justified. The identification of reuse by both practitioners and academic sponsors as a missing and necessary component of the model has motivated the development of a second, revised version of COSYSMO, known as COSYSMO 2.0. Before describing the approach for estimating the economic impact of reuse it is important to summarize the most relevant concepts from the literature on this topic.

2 Overview of Reuse

Reuse can be defined in many ways, but a common theme exists: reuse is to assemble a product from existing components and limit the creation of new components to ones that do not exist [5].

By any definition of the term, reuse is not a new concept. Early forms of reuse include the repetition of mathematical models and algorithms across problems to ensure correct calculations [6]. The construction and automobile industries rely heavily on the reuse of key components and parts [7]. Even the utilization of engineering specifications and standards, essential to any engineering practitioner, is a form of reuse. Fundamentally, reuse is the result of a natural human problem solving technique whereby people determine if a problem they are faced with has already been solved, if they have an existing solution to a similar problem that can be adapted, or if the problem is unprecedented and needs to be decomposed into a smaller set of sub-problems [8]. Reuse differs from the concept of re-engineering in that re-engineering occurs when an existing system is transformed into another system, whereas reuse occurs when an artifact is re-applied to a new system [9].

More refined definitions of reuse for systems engineering applications are: 1) the repeated use of an application in different places of the design of parts, manufacturing tools and processes, analysis, and particularly knowledge gained from experience; using the same object in different systems or at different times in the same system [10], 2) the use of systems artifacts and processes in the development of solutions to similar problems. [11].

The development of COSYSMO 2.0 is focused on systems engineering reuse and specifically, how reuse impacts the expected amount of systems engineering effort for a system. Systems engineering activities are mainly support-focused and do not produce physical products. Instead, systems engineering activities typically produce artifacts in support of complex systems such as architectures, requirements, test plans, analyses, and trade studies. The role of the systems engineer is quite different than other engineering disciplines such as hardware or software engineering because it involves both technical and managerial responsibilities such as coordination, life cycle ownership and design [12].

Since systems engineering artifacts are produced by a set of systems engineering activities, they are representative of the systems engineering effort required to produce them. By reusing an artifact, some amount of the activities associated with producing that artifact, and subsequently some amount of the systems engineering effort, should not be required during the development of a system. This is similar to the concept of the learning curve which describes reduced engineering effort as a result of repetition [13].

Ideally, reuse should result in a reduction to the amount of systems engineering effort required to complete a system; however, in some instances reuse can potentially require more effort than a new development. The need for estimating the expected reduction to systems engineering effort as a result of reuse, as well as, identifying when reuse can actually be more costly is what motivated the development of COSYSMO 2.0.

2.1 State of the Art

The COSYSMO model was developed with guidance and insight from dozens of systems engineering experts, and grounded in internationally accepted systems engineering standards [14]. Given COSYSMO 2.0 is intended to be an extension of the COSYSMO tool; a similar development methodology was followed, and relevant pieces of the systems engineering literature and systems engineering standards were reviewed. This review was intended to help inform and guide the development of COSYSMO 2.0 by understanding how to best account for systems engineering reuse. After conducting a search of systems engineering texts, journal articles, handbooks, and standards, a gap in the systems engineering literature was apparent. Systems engineering reuse was mentioned only in a very limited capacity and never with respect to the reuse of systems engineering artifacts. However, many systems engineering standards that discussed reuse gave reference to various software engineering standards. Knowing the similarities between systems and software engineering, it became apparent that a review of how the software engineering literature discussed reuse should be conducted, and the results of which could be applied to the systems engineering domain. These results are described below (a longer version of these observations is available in [15]) summarized as eight observations.

The idea of reusing software was first discussed publicly in 1969 by Bell Laboratories, when Bell proposed to make software development more "industrialized" instead of "craft-based" [16]. Over the past few decades, software reuse has been described as a means for enabling projects to achieve higher quality, increased productivity, shorter development schedules, reduced overruns, and improved leveraging of technical skills and knowledge [17,18]. To date, dozens of models have been developed to estimate a wide range of parameters associated with software reuse; extensive summaries of software reuse metrics and models can be found in [19,20,21,22].

The motivation most often stated in the literature for software reuse is a reduction in the cost of developing new products by avoiding redevelopment of capabilities and increasing productivity by incorporating components whose reliability has already been established [23,24]. For example, research has shown that the reuse of software code can result in fewer program faults and repeat mistakes can be avoided [25,26].

<u>Observation #1</u>: *Reuse is done for the purpose of economic benefit, intending to shorten schedule, reduce cost, and/or increase performance.*

Naturally, once an organization finds a product or artifact that performs well, they want to replicate that success. When successful, a software reuse program can result in cost savings between 10-35% [27]; however, reuse is not a "silver bullet". Organizations frequently predict that reuse will result in huge increases in productivity [28] or overstate their capabilities and overestimate the chances for reuse success [29]. Even a 10-20% modification of an artifact can negate any potential reuse benefits, therefore making it more efficient to start with a new artifact than a reused one [30].

<u>Observation #2</u>: *Reuse is not free, upfront investment is required to understand the technical opportunities and limitations.*

Reusable artifacts are product, process, or knowledge focused [12]. These artifacts can be requirements, designs, code, tests, test cases, architectures, documentation, interfaces, and plans [11,18]. Artifacts can also include certification processes, configuration management records, quality records, and verification data [31]. Cybulski identifies over one hundred reusable artifacts such as budgets, SWOT analyses, contracts, and prototypes [32]. In addition to the wide variety of reusable artifacts, the processes captured within and associated with the creation of artifacts are also essential for successful reuse. For example, Boehm states that software reuse itself needs to be process oriented; meaning the development of software with reused artifacts should be preconceived, repeatable, and well documented [33]. Reuse processes should be formal and institutionalized to capture reuse principles, produce quality results, and be repeatable [8,11].

<u>Observation #3</u>: *Products, processes, and knowledge are all reusable artifacts.*

According to Tracz, software reuse is not something that will just happen [28]. For reuse to be successful, it must be planned from the onset of the project, as the difficulty of implementing reuse becomes increasingly harder as a project progresses [34]. Because of this, reuse is most frequently successful when it is applied systematically, compared to a non-planned or ad hoc approach [35]. Ad hoc reuse is the idea that a development can be stopped at selected life cycle stages, potential reusable components can be reviewed for applicability, and reuse of those components can occur [2]. While ad hoc reuse is characterized by unplanned, short term solutions, systematic reuse is driven by a careful and well-coordinated planning process [18]. The IEEE software reuse standard defines systematic reuse as the practice of reuse according to a well-defined, repeatable process [36]; simply putting components or artifacts together is usually unsuccessful and frequently results in negative impacts to project schedule and total effort [37]. Systematic reuse is similar to product line engineering [38], which is the strategic and planned use of architectures and components across development efforts [22]. The success of systematic reuse over ad hoc reuse can be attributed to the fact that a systematic approach helps an engineer to assess the impact of reuse on the project beforehand and prepare for the potential issues [39].

<u>Observation #4</u>: *Reuse needs to be planned from the conceptualization phase of programs.*

One of the most commonly discussed aspects of reuse in the software engineering literature is that reuse is not only a technical problem, it is a psychological, sociological, or economic one [28]. From an economic perspective, a viable business case for reuse needs to exist before such a strategy should be pursued [40]. A business case should not focus purely on the potential economic benefits expected from reusing software or explain the quality of the artifact being reused, but rather the capabilities of a skilled workforce and knowledge of the system that the artifacts were derived from [7]. From a sociological standpoint, products are rarely built from scratch, personnel do not typically forget years of training and experience, and this knowledge usually exists somewhere in an engineering or corporate memory [11].

<u>Observation #5</u>*: Reuse is as much of an organizational issue as it is a technical one.*

Subsequently, knowledge and personal experience, when captured in artifacts, can be reused [41]; however, capturing such information is often challenging [42] because it can be laborious, time consuming, and difficult; not costless and instantaneous as commonly suggested [29]. Knowledge reuse includes having experienced personnel tell others where to find related information, how to solve a particular problem [43], or how to apply other project specific information [44]. However, as mentioned previously, capturing knowledge and making it available for reuse is a major challenge because much of it is tacit knowledge that is not written down. Reusable knowledge often exists only within a person [29] and an organization needs to have adequate processes in place to capture, store, recall, and apply that information.

<u>Observation #6</u>*: Reuse is knowledge that must be deliberately captured in order to be beneficial.*

A major evaluation criterion for reuse is domain compatibility [45]. Successful reuse requires an understanding of the technology domain of interest in order to recognize what should be reused and how to accomplish it successfully [46]. Analyzing a domain is: 1) the process of identifying, collecting, organizing, and representing the relevant information and 2) based upon the study of existing systems and their development histories [8]. The failure to perform systematic and rigorous domain analysis accounts for the failure of many reuse programs [47,39], as the potential for reuse cannot be judged by only looking at inputs and outputs of a system [48].

<u>Observation #7</u>*: The benefits of reuse are limited to closely related domains.*

Selby determined that even for reuse within a related domain, the benefits of reuse do not scale in a linear fashion [26]. As project size or complexity increases, the application of reuse cannot be expected to deliver benefits along a linear trend. For example, if the reuse of a specific artifact can result in an expected 5% effort savings for a small-scale project, it is very unlikely that the reuse the same artifact on a large-scale project will also reduce the expected effort by 5%. Therefore, reuse may provide greater benefits to smaller projects than larger projects.

<u>Observation #8</u>*: The benefits of reuse do not scale linearly*

The observations made during this review of the software engineering literature led to a survey on how systems engineering practitioners address reuse. Highlights of this survey are presented below.

2.2 State of the Practice

After conducting a review of the literature on the topic of reuse, eight observations were captured. A better grasp of the practical approaches to reuse can complement the theoretical observations extracted from the literature. To obtain this, a survey was created and distributed to industry representatives from the systems engineering domain familiar with COSYSMO development efforts. The results of this survey helped to guide the proposed reuse drivers for COSYSMO 2.0.

The eight observations from the literature were informative for developing questions for the survey by focusing on the most critical issues in industry with regard to reuse. Furthermore, we were interested in capturing industry perspectives and practices on reuse and determine where the challenges existed. To obtain the industry perspective on the subject, a brief ten-question survey was developed and distributed to affiliates of the USC Center for Systems and Software Engineering (USC-CSSE) as well as other interested parties [49]. The goal of the survey was to obtain more focused answers on reuse that would help support an approach for accounting for systems engineering reuse in COSYSMO 2.0.

Again, using observations from the literature review of how the software engineering domain handles reuse, the following key questions on the systems engineering reuse were formulated:
1) How do systems engineering organizations define reuse?
2) What systems engineering artifacts are typically reused?
3) When in the system life cycle does reuse occur?
4) What contributes to successful or unsuccessful reuse?

In addition to these questions, industry opinion was solicited on the issues of the scaling of reuse and the five proposed reuse categories (size driver extensions). Understanding how the benefit of reuse changes with system complexity is critical to the development of an estimation model; in particular, increases to system size and complexity were believed to have a significant impact on reuse and therefore must be accounted for in COSYSMO 2.0. Furthermore, previous research has already been conducted on the identification [50] and definition [51] of categories of systems engineering reuse. The survey was used as an opportunity to validate the definitions of the reuse categories.

In total, eleven responses were received from six different aerospace and engineering contractors. The results of the survey can be summarized into four main lessons for how industry handles systems engineering reuse. These results informed how COSYSMO 2.0 could best account for reuse (a more detailed presentation of these results is available in [52]).

<u>Result #1</u>*: Requirements reuse is only performed occasionally, but has the largest "benefit" associated with it.*

Requirements reuse was identified as being performed occasionally; less frequent than the reuse of any other the artifacts mentioned in the survey. This is believed to be due to the applicability of requirements being reused relative to other artifacts. For example, documentation is an artifact that is usually general enough to be applicable to multiple system development efforts. One subject matter expert stated that “hundreds of systems engineering documents are available for potential reuse, but they are often more of a convenience, such as a document template, than a means of achieving a significant reduction in systems engineering effort.” Another expert described the reuse of requirements was described as a potential “home run” for systems engineering reuse. If a requirement can be reused, most of the systems engineering artifacts associated with requirements can be reused as well. Requirements are not too specific of an artifact such that they cannot be applied to related systems and are not too generic of an artifact such that they would fail to provide the capabilities to result in a sizable reduction in systems engineering effort. Therefore, the challenge appears to reside in finding systems that you can reuse applicable requirements without substantial modification.

Result #2: *Reuse occurs more frequently early in the life cycle than later.*

Industry respondents indicated that systems engineering artifacts are reused on a more frequent basis during the first three life cycle phases: Conceptualization, Development, and Test and Evaluation as defined in ISO 15288 [52]. This conclusion makes sense given that the majority of the systems engineering effort occurs in these phases and by the time the system reaches the Transition to Operation phase, the system is all but complete and limited opportunities for reuse are available. One subject matter expert who stated that systems engineering reuse must be planned from the conceptualization phase because as the schedule progresses, it becomes more difficult to identify opportunities for reuse and reuse could have potentially unforeseen, negative consequences. For example, the reuse of an artifact later in the life cycle will force a systems engineer to re-validate and re-verify all the interfaces that could potentially be affected by the reuse. Another expert believed that since a major part of the systems engineering effort occurs during the Conceptualization phase and the Test and Evaluation phase, opportunities for systems engineering reuse will be more frequent in these two phases specifically.

Result #3: *Cost savings is the most promoted benefit for reuse, but benefits also exist in risk, schedule, and performance.*

Not surprisingly, cost savings was identified as the most promoted benefit for systems engineering reuse. Although the realization of cost savings is much more difficult than the promoted benefits, the motivation for reuse appears to be the opportunity to reduce the amount of resources required to complete a project. However, somewhat surprisingly, the other four benefits listed in the survey all had fairly equal results and were not listed with significantly less frequently than cost. It was unclear if responders inherently associated risk, performance, schedule, or quality benefits as a means of achieving cost savings or if these factors are equally promoted. Despite schedule being ranked as the second most promoted benefit for reuse, additional discussions with experts cited risk reduction as the other major benefit. Reusing an artifact with a proven history of success dramatically reduces the risk associated with a new system. Furthermore, a reduction in risk can manifest itself in performance, through an artifact delivering on its capability, or quality, by an artifact not failing.

Result #4: *The proposed five categories of reuse are reasonable in characterizing systems engineering reuse.*

The five categories of reuse described in the survey were New, Modified, Adopted, Managed, and Deleted. These categories and their ability to characterize systems engineering reuse are further explained in section 3. Experts were also asked about their opinion on how reuse benefits scale (increases or decreases) with system complexity. Most responders, citing an increasing number of system interfaces, believe that reuse benefits decrease non-linearly with system complexity, but there is little empirical evidence available to justify this conclusion.

Result #5: *Experienced personnel is a key factor for successful reuse.*

Based on the responses to the survey and the follow-up interviews, the most significant reason for the successful reuse of systems engineering artifacts is the utilization of personnel with experience on previous system that developed the artifact. The identification of personnel as a key factor to the success of reuse in the systems engineering domain mirrors the observations from the literature in the software engineering domain. Successful systems engineering reuse appears to require more than just reusable or proven artifacts, non-technical factors such as personnel knowledge has a critical role. Therefore, a strategy which only accounts for systems engineering reuse through a purely technical viewpoint is incomplete. In terms of the COSYSMO model, the survey results indicate that reuse should be addressed in both the size (technical) and cost (non-technical) drivers.

A second, but still important, reason cited for the successful systems engineering reuse is the utilization of artifacts with minimal to no modification. One subject matter expert responded to this question by saying that any modification of an artifact that exceeds approximately 20% will nullify any potential benefit from reuse. As supported by the survey results, reuse with modification can result in the same amount of effort than developing an artifact as new.

Comparatively, the most significant reason for the failure of systems engineering reuse is an artifact lacks a specific core capability. Even if an artifact is designed for reuse, it may be too generic to used for a specific application. For example, there could be a domain incompatibility, modification required, or the artifact delivers multiple capabilities satisfactorily, but no particular one well. This fact further supports the possible approach of addressing reuse in both the cost and size drivers. The potential misapplication of a reuse artifact needs to be accounted for in the model and the incorporation of an additional cost driver appears to provide the best capability to accomplish this.

The results presented and discussed in this chapter have directly influenced the development of COSYSMO 2.0. The considerations for successful systems engineering reuse (state of the art observations) can be related to the outcomes from the reuse survey (state of the practice results), all of which can be mapped into the proposed reuse drivers for COSYSMO 2.0.

3 Systems Engineering Reuse Estimation Methods

As mentioned previously, the existing COSYSMO model is based on the assumption that the systems engineering effort estimate is for the development of an entirely new system and all corresponding systems engineering activities are also completed as new.

During the development of COSYSMO, the accounting for reuse was deferred since, at the time, there was insufficient data available to calibrate the model with a reuse factor. After the completion of the COSYSMO tool, industry practitioners and USC-CSSE affiliates identified the lack of a reuse estimation capability as a potential limitation to the model. In 2006, continuing on with the reuse concept identified in the COSYSMO dissertation, a potential reuse strategy was presented at the COCOMO Forum [4]. This strategy applied a set of five reuse categories across each of the *Easy*, *Nominal*, and *Difficult* categories for the *Number of Requirements* size driver. The application of the reuse categories to the requirements size driver produced a revised *Number of Requirements*, called *Total Equivalent New Requirements*, which accounted for the number of new and reused requirements in the system. With the *Total Equivalent New Requirements* count, a COSYSMO estimate that incorporated a limited degree of reuse could be produced.

Although this methodology created a possible strategy for accounting for reuse in the model, it was the first attempt at such a capability and did not receive full buy-in with the industrial community as the accepted approach for reuse in COSYSMO. The significant contribution from this methodology was the categorization of non-new requirements as *modified*, *reused*, or *deleted*. This categorization approach led to additional reuse developments and ultimately helped to guide the COSYSMO 2.0 strategy.

Continuing with the methodology initially proposed, John Gaffney, working at Lockheed Martin Corporation, developed the COSYSMO-Risk/Reuse (COSYSMO-R) model in 2007 [53]. The motivation for COSYSMO-R was to extend the capabilities of COSYSMO in the areas of risk and reuse estimation. In addition to the lack of a reuse estimation capability, COSYSMO is also limited to single-point estimates of systems engineering effort. Due to the uncertainty associated with effort, schedule, and cost estimates, COSYSMO-R intended to account for risk and confidence factors. The details of the COSYSMO-R (Risk) model will not be discussed further as they are outside the scope of this paper. The COSYSMO-R (Reuse) model attempts to account for systems requirements that are not new by enabling a user to subdivide the requirements driver into reused, deleted, or modified categories, an approach similar to that described in [4].

Overall, the COSYSMO-R tool does provide a capability to estimate systems engineering reuse; however, two issues remain. First, although practitioners at the sponsoring company have an understanding of these categories and a systems engineering organization structured to divide activities into these categories, they are not accepted industry-wide. Some additional efforts have been made, described below, to agree on more appropriate and acceptable definitions of reuse across multiple organizations. Second, and more importantly, the weights associated with each of the categories are user defined and not validated with data from multiple sources, which could lead to discrepancies across systems or organizations. Additionally, COSYSMO-R does not account for the effect of reuse on the cost drivers of the COSYSMO model, which may be significant.

COSYSMO 2.0 will have to deliver capabilities beyond those of COSYSMO-R for the model to be adopted at COSYSMO supporters like Lockheed Martin Corporation. However, since the proposed COSYSMO 2.0 model will be validated with data from multiple organizations and across varying system domains, the model is expected to deliver better estimation power and enable more realistic cost comparisons.

In 2007, the discussions on four reuse categories continued. Up until this point, a major obstacle for the incorporation of reuse into the model was the lack of consensus on the definition of each reuse category. At a PSM Conference COSYSMO Working Group, preliminary definitions, described below, were formulated with the assistance of industry and academic stakeholders [50]. The four categories are very similar to those presented in first attempt at a reuse extension [4] as well as COSYSMO-R [53]. The four reuse categories were:

1) New: Items that are completely new
2) Adopted: Items that are incorporated unmodified
3) Modified: Items that are reused but are tailored
4) Deleted: Items that are removed from a system

With a methodology for accounting for reuse proposed and the reuse categories more clearly defined, the next area to be addressed was the derivation of the weights associated with each reuse category. At the 2007 COSYSMO Working Group meeting at the COCOMO Forum, a methodology for deriving the reuse weights was presented [54]. The methodology proposed correlating the thirty-three systems engineering activities defined by the EIA-632 standard [55], which COSYSMO is based upon, with the four reuse categories. This methodology would determine the potential for each systems engineering activity to be "binned" into one of the four categories. Continuing with this methodology, by using the results from a Delphi survey [56] on the distribution of systems engineering activities across the system life-cycle phases, the percentage of the total systems engineering effort that could be reduced via reuse is identified. The derived percentages (weights) could then be assigned to each of the corresponding reuse categories, and a single point estimate for the effort reduction associated with each reuse category could be obtained [57].

This proposed methodology for deriving the weights of the reuse categories showed promising results when a pilot test was conducted with historical data [58] from BAE Systems. Dr. Gan Wang utilized the approach outlined above and generated the first set of weights for the reuse categories based on data. However, the proposed four categories of reuse were observed as inadequate to cover certain instances of reuse representative in this particular data set. Based on this observation, a fifth reuse category was introduced for the BAE Systems data, which intended to provide the model (and ultimately, the user) with more detail and explanatory power. The proposed five reuse categories were:

1) New: Items that are completely new
2) Modified: Items that are inherited, but are tailored
3) Adopted: Items that are incorporated unmodified, also known as "black box" reuse
4) Managed: Items that are incorporated unmodified and untested
5) Deleted: Items that are removed from a system

Although the four reuse categories were previously agreed to by COSYSMO supporters, the results of the pilot test with the five reuse categories raised a debate between the two strategies. To reconcile the difference in the number of categories, it has been suggested that the two strategies could be combined to create a hybrid set of four categories with a fifth subcategory. Currently, the "adopted" and "managed" categories are similar, so having "managed" be a subcategory of "adopted" would leave the four main reuse categories intact, but still provide users of the tool with the detail of a fifth category. Because of the debate between the number of reuse categories, future COSYSMO 2.0 research will examine the explanatory power of both four and five reuse categories, and will present the results to the COSYSMO supporters prior to finalizing on a set of reuse categories for COSYSMO 2.0.

Up until this point, all of the development associated with a systems engineering reuse model and a reuse extension for COSYSMO was focused around revising the size drivers of the model to account for a reduction in the size of the systems engineering effort. More specifically, most of the discussion on the revision of the size drivers with the reuse categories was centered on the *Number of Requirements* driver. Since COSYSMO takes a total of eighteen drivers into consideration when calculating an estimate of systems engineering effort, only addressing reuse in one (to at most four) drivers and only within the size drivers may be insufficient.

In 2008, leveraging the observations from the literature review of the software engineering domain on reuse, several considerations for successful systems engineering reuse were proposed [15]. One of the major outcomes from this research was the identification of the need to account for the non-technical aspects of reuse. From a COSYSMO perspective, this was interpreted as examining the effect of reuse in both the size and the cost drivers.

3.1 COSYSMO 2.0

Currently, the COSYSMO 2.0 model is still under development, although a preliminary version of the Cost Estimating Relationship is presented in Equation 2 [51].

$$PM_{NS} = A \cdot \left[\sum_k \left(\sum_r w_r (w_{e,k}\Phi_{e,k} + w_{n,k}\Phi_{n,k} + w_{d,k}\Phi_{d,k}) \right) \right]^E \cdot \prod_{j=1}^{14} EM_j$$

Where,

PM$_{NS}$ = effort in Person Months (Nominal Schedule)

A = calibration constant derived from historical project data

k = {REQ, IF, ALG, SCN}

r = {New, Modified, Adopted, Deleted, Managed}

w$_r$ = weight for defined degrees of reuse

w$_x$ = weight for "easy", "nominal", or "difficult" size driver

Φ_x = quantity of "k" size driver

E = represents diseconomies of scale

EM = effort multiplier for the jth cost driver. The geometric product results in an overall effort adjustment factor to the nominal effort.

Equation 2 – *Preliminary COSYSMO 2.0 Operational Equation.*

To calculate the size of the systems engineering effort (the part of the equation to the left of the sum product of the effort multipliers), each of the four size drivers are sub-divided into *Easy*, *Nominal*, and *Difficult* ratings (same as with the original COSYSMO) as well as the five categories of reuse (New, Modified, Adopted, Deleted Managed).

After values for each of the size drivers are inputted into the model, the values are multiplied by the weights associated with each of the old (Easy, Nominal, Difficult) and additional new (New, Modified, Adopted, Deleted, Managed) ratings. The product of the weight and the value for each of ratings for each of the drivers are then summed to obtain the size of the system. The size of the system calculated by the COSYSMO 2.0 equation is different from the size calculated by the original COSYSMO model since it does account for systems engineering reuse. Therefore, the size calculated by the COSYSMO 2.0 equation is for an "equivalent size" of a new system with some reuse. By accounting for reuse, COSYSMO 2.0 will in most cases reduce the original COSYSMO estimate for system size by discounting for reuse and estimating the size of, and effort for, only the new aspects of the system.

By using this methodology, will COSYSMO 2.0 estimate the size of the system in a slightly different manner than the original COSYSMO, but will still result in an estimate for an effectively "new" system, which can then be multiplied by the sum product of the cost drivers. Although the methodology for calculating system size in COSYSMO 2.0 could result in up to 60 inputs (four size drivers * three ratings * five reuse categories) for the size driver values, this is not expected to be unmanageable because a user would only need to input values beyond those required in the original COSYSMO (four size drivers * three ratings) if the reuse of systems engineering artifacts was expected to occur. When systems engineering reuse is expected, only the reuse of the artifacts that relate to the size drivers (Requirements, Interfaces, Algorithms, Scenarios) need to be considered at this point in the estimation process. If an artifact (or artifacts) related to any of these size drivers was expected to be reused, then at that point a user of the tool would need to provide values for the fifteen (three ratings * five reuse categories) inputs associated with each impacted size driver. Since a user of the existing COSYSMO tool already has to define each size driver according to *Easy*, *Nominal*, and *Difficult* ratings, COSYSMO 2.0 would only require that a user take each of those ratings values and decide if the reuse of an artifact is expected to occur, and if so, at what level (category) of reuse.

The preliminary version of COSYSMO 2.0, as presented in Equation 2, describes a methodology for accounting for systems engineering reuse in the size drivers of the COSYSMO tool, but not in the cost drivers of the model. The potential need to account for reuse in the cost drivers was clearly identified in the software engineering literature reviewed as well as the results from the industrial survey. Therefore, the development of COSYSMO 2.0 will examine the effect of accounting for systems engineering reuse in only the size drivers as well as in both the size and cost drivers. The results of each of these strategies will be presented to the COSYSMO supporters for feedback, prior to finalizing the COSYSMO 2.0 tool.

When completed, COSYSMO 2.0 will provide capability, flexibility, and additional detail for users who desire to account for systems engineering reuse, but is not expected to overwhelm users who do not need to account for reuse.

3.2 Expected Results

This research is expected to contribute to the field of systems engineering in a number of ways. These contributions will have implications for researchers, practitioners, and educators. Practitioners will benefit from the incorporation of reuse into the COSYSMO tool because it improves their ability to estimate how reuse will affect the amount of expected systems engineering effort. This research will provide practitioners, researchers, and educators with definitions of systems engineering reuse as well as drivers that characterize the technical (product) and non-technical (people, processes, organizational) aspects of reuse. By validating COSYSMO 2.0, practitioners will be able to quantifiably account for systems engineering reuse in an estimate. Accounting for reuse in the model will assist in the justification of expected savings in (or increases to) systems engineering effort associated with a reuse strategy. This research will identify aspects of systems engineering reuse for additional exploration and examination. Researchers can use this review of the state of the art and state of the practice to identify potential academic and industry needs on the topic. Finally, COSYSMO 2.0 improvements will examine the effect of accounting for both the technical (size drivers) and non-technical (cost drivers) aspects of systems engineering reuse. Practitioners will benefit from this improvement to the COSYSMO tool by being presented with a more complete perspective on the considerations (technical and non-technical) associated with a reuse strategy and the resulting savings in (or increases to) systems engineering effort.

4 Conclusion

This paper provides a background on systems engineering reuse and identifies potential methodologies for accounting for the effect of reuse in estimating systems engineering effort. The primary methodology identified in the paper for estimating systems engineering reuse is through the development of a reuse extension to the already successful and widely accepted COSYSMO model. The reuse extensions described in this paper, which will be incorporated into the COSYSMO 2.0 model, include revisions to the four COSYSMO size drivers; in addition, future research will explore the estimation power of accounting for reuse in the fourteen cost drivers as well. The development of COSYSMO 2.0 will also explore the need for potential updates to the existing fourteen COSYSMO cost drivers. To help guide the development process and help ensure a model is produced that meets the needs of the community, feedback has been solicited from academic and industry practitioners. As a result of this research, the authors expect COSYSMO 2.0, when completed and calibrated with industry data, to provide better estimation power than the original COSYSMO.

4.1 Future Research

To support the development of COSYSMO 2.0, a call for participation has been issued to perform an industry calibration. We are seeking industry data in the form of labor actuals on various types of systems engineering projects that involved a significant amount of reuse. If you are interested in contributing to this research, please contact any of the authors.

4.2 Acknowledgements

The authors wish to thank the USC Center for Systems and Software Engineering Corporate Affiliates and the MIT Lean Advancement Initiative Consortium Members for their continued support of COSYSMO research, as well as the survey respondents who participated in this research.

5 References

[1] Valerdi, R., Rieff, J., Roedler, G. and Wheaton, M. (2007), “Lessons Learned from Industrial Validation of COSYSMO”, *17th INCOSE Symposium*, June 2007, San Diego, CA

[2] Prieto-Diaz, R. (1996), “Reuse as a New Paradigm for Software Development”, *International Workshop on Systematic Reuse*, Liverpool, England.

[3] Valerdi, R. (2005), “The Constructive Systems Engineering Cost Model”, PhD Dissertation, *University of Southern California*, Los Angeles, CA.

[4] Valerdi, R., Gaffney, J., Roedler, G. and Rieff, J. (2006), “Extensions to COSYSMO to Represent Reuse”, *21st International Forum on COCOMO and Software Cost Modeling*, Los Angeles, CA.

[5] Robertson, S. (1996). “Reuse Lifecycle: Essentials and Implementations”, *International Workshop on Systematic Reuse*, Liverpool, England.

[6] Prieto-Diaz, R. (1993), “Status Report: Software Reusability”, *IEEE Software*, Vol. 10, No. 3.

[7] de Judicibus, D. (1996), “Reuse: A Cultural Change”, *International Workshop on Systematic Reuse*, Liverpool, England.

[8] Mili, H., Mili, A., Yacoub, S. and Addy, E. (2002), “Reuse-Based Software Engineering”, *John Wiley & Sons*.

[9] Lam, W. and Loomes, M. (1998), “Re-engineering for Reuse: A Paradigm for Evolving Complex Reuse Artefacts”, *22nd International Computer Software and Application Conference*, Vienna, Austria.

[10] Allen, T. Moses, J., Hastings, D., Lloyd, S., Little, J., McGowan, D., Magee, C., Moavenzadeh, F., Nightingale, D., Roos, D. and Whitney, D. (2001), “Engineering Systems Division Terms and Definitions”, *Massachusetts Institute of Technology Engineering Systems Division*, Ver. 12.

[11] Whittle, B., Lam, W. and Kelly, T. (1996), “A Pragmatic Approach to Reuse Introduction in an Industrial Setting”, *Workshop on Systematic Reuse*, Liverpool, UK.

[12] Wright, T. (1936), Factors Affecting the Cost of Airplanes, Journal of Aeronautical Science, 3(4), 122-128.

[13] Sheard, S. A. (1996), Twelve Systems Engineering Roles, Software Productivity Consortium.

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

[15] Fortune, J. and Valerdi, R. (2008), “Considerations for Successful Reuse in Systems Engineering”, *AIAA Space 2008*, San Diego, CA.

[16] Isoda, S. (1996), “Software Reuse in Japan”, *Information and Software Technology*, Vol. 38, Issue 3.

[17] Basili, V., Romach, H., Bailey, J. and Joo, B. (1987), “Software Reuse: A Framework for Research”, *Tenth Minnowbrook Workshop on Software Performance Evaluation*, Blue Mountain Lake, NY.

[18] Lim, W. (1998), “Managing Software Reuse”, *Prentice Hall*.

[19] Frakes, W. and Fox, C. (1995), “Sixteen Questions About Software Reuse”, *Communications of the ACM*, Vol. 38, No. 6.

[20] Lim, W. (1996), “Reuse Economics: A Comparison of Seventeen Models and Directions for Future Research”, *Fourth International Conference on Software Reuse*, Orlando, FL.

[21] Poulin, J. (1997), “Measuring Software Reuse”, *Addison-Wesley*.

[22] Wiles, E. (1999), “Economic Models of Software Reuse: A Survey, Comparison, and Partial Validation”, PhD Dissertation, *University of Wales*, Aberystwyth, Ceredigion, UK.

[23] Bollinger, T. and Pfleeger, S. (1992), “Economics of Reuse: Issues and Alternatives”, *Information and Software Technology*, Vol. 32, No. 10.

[24] Poulin, J. and Caruso, J. (1993), “Determining the Value of a Corporate Reuse Program”, *1st International Software Metrics Symposium*, Baltimore, MD.

[25] Antelme, R., Moultrie, J. and Probert, D. (2000), “Engineering Reuse: A Framework for Improving Performance”, *IEEE International Conference on Management of Innovation and Technology*, Singapore.

[26] Selby, R. (2005), “Software Reuse in Large-Scale Systems”, *AIAA Space 2005*, Long Beach, CA.

[27] Stephens, R. (2004), “Measuring Enterprise Reuse in a Large Scale Corporate Environment”, *37th Southeastern Symposium on System Theory*, Fort Lauderdale, FL.

[28] Tracz, W. (1988), “Software Reuse Myths”, *Software Engineering Notes*, Vol. 13, No. 1.

[29] Szulanski, G. and Winter, S. (2002), “Getting It Right the Second Time”, *Harvard Business Review*, Vol. 80, Issue 1.

[30] Glass, R. (1999), “Reuse: What’s Wrong With This Picture?”, *IEEE Software*, Vol. 15, No. 2.

[31] Lougee, H. (2004), “Reuse and DO-178B Certified Software: Beginning with Reuse Basics”, *CrossTalk*, Vol. 17, No. 12.

[32] Cybulski, J., Neal, R., Kram, A. and Allen, J. (1998), “Reuse of Early Life Cycle Artifacts: Workproducts,

Methods, and Tools", *Annals of Software Engineering*, Vol. 5, No. 1.
[33] Boehm, B. W. (1999), "Managing Software Productivity and Reuse", *Computer*, Vol. 32, No. 9.
[34] Finkelstein, A. (1988), "Re-use of Formatted Requirements Specifications", *Software Engineering Journal*, Vol. 3, Issue 5.
[35] Dusink, L and van Katwijk, J. (1995), "Reuse Dimensions", *Symposium on Software Reliability*, Seattle Washington.
[36] IEEE. (1999), "IEEE 1517-1999 – Software Life Cycle-Reuse Processes", *IEEE*.
[37] Garlan, D., Allen, R. and Ockerbloom, J. (1995), "Architectural Mismatch: Why reuse is so hard", *IEEE Software*, Vol. 12, Issue 6.
[38] Beckert, M. (2000), "Organizational Characteristics for Successful Product Line Engineering", Masters Thesis. *Massachusetts Institute of Technology*, Cambridge, MA.
[39] Lam, W., McDermid, A. and Vickers, A. (1997), "Ten Steps Towards Systematic Requirements Reuse", *Requirements Engineering*, Vol. 2, No. 2.
[40] Reifer, D. (1997), "Practical Software Reuse", *John Wiley & Sons*.
[41] Basili, V. and Romach, H. (1991), "Support for Comprehensive Reuse", *Software Engineering Journal*, Vol. 6, Issue 5.
[42] Moore, M. (2001), "Software Reuse: Silver Bullet?", *IEEE Software*, Vol. 18, Issue 5.
[43] Malhotra, A. and Majchrzak, A. (2004), "Enabling Knowledge Creation in Far-Flung Teams: Best Practices for IT Support and Knowledge Sharing", *Journal of Knowledge Management*, Vol. 8, No. 4.
[44] Cooper, L., Majchrzak, A. and Faraj, S. (2005), "Learning from Project Experiences Using a Legacy-Based Approach", *38th Annual Hawaii International Conference on System Science*, Big Island, HI.
[45] Konito, J., Caldiera, G. and Basili, V. (1996), "Defining Factors, Goals, and Criteria for Reusable Component Evaluation", *Conference of the Centre for Advanced Studies on Collaborative Research*, Toronto, Ontario, Canada.
[46] Tracz, W. (1995), "Confessions of a Used Program Salesman: Lessons Learned", *Symposium on Software Reusability*, Seattle, WA.
[47] Anthes, G. (1993), "Software Reuse Bring Paybacks", *ComputerWorld*, Vol. 27, Issue 49.
[48] Wymore, A. and Bahill, A. (2000), "Can We Safely Reuse Systems, Upgrade Systems, or Use COTS Components?", *Systems Engineering*, Vol. 3, No. 2.
[49] Fortune, J., Valerdi, R. and Wang, G. (2008), "Systems Engineering Reuse: A Report on the State of the Practice", *23rd International Forum on COCOMO and Systems/Software Cost Modeling*, Los Angeles, CA.
[50] Rieff, J., Gaffney, J. and Roedler, G. (2007), "2007: The Breakout Year for COSYSMO", *PSM Users Group Conference*, Golden CO.
[51] Wang, G., Valerdi, R. and Fortune, J. (2009), "Reuse in Systems Engineering", Under Review, *IEEE Systems, Man, and Cybernetics Journal*.
[52] ISO/IEC (2002), "Systems Engineering - System Life Cycle Processes" *ISO/IEC 15288:2002(E)*.
[53] Gaffney, J. (2007), "COSYSMO-Risk/Reuse Model", *Lockheed Martin*.
[54] Wang, G. (2007), "COSYSMO Extension: Reuse", 2007 *COCOMO Forum*, *COSYSMO Working Group*, Los Angeles, CA.
[55] ANSI/EIA (1999), "ANSI/EIA-632-1988 Processes for Engineering a System", *ANSI/EIA*.
[56] Valerdi, R. and Wheaton, M. (2005), "ANSI/EIA 632 As a Standard WBS for COSYSMO", *AIAA 1st Infotech at Aerospace Conference*, Arlington, VA.
[57] Valerdi, R., Wang, G., Roedler, G., Rieff, J. and Fortune, J. (2007), "COSYSMO Reuse Extension", *22nd International Forum on COCOMO and Systems/Software Cost Modeling*, Los Angeles, CA.
[58] Wang, G., Valerdi, R., Ankrum, A., Millar, C. and Roedler, G. (2008), "COSYSMO Reuse Extension", *18th INCOSE Symposium*, Utrecht, the Netherlands.