8th Conference on Systems Engineering Research
March 17-19, 2010, Hoboken, NJ

Paper #nnn

Better Requirements Decomposition Guidelines Can Improve Cost Estimation of Systems Engineering and Human Systems Integration

Kevin Liu
MIT
77 Mass. Ave.
Cambridge MA | E38-546
k_liu@mit.edu

Ricardo Valerdi
MIT
77 Mass. Ave.
Cambridge MA | 41-205
rvalerdi@mit.edu

Phillip A. Laplante
Penn State
30 E. Swedesford Road
Malvern, PA 19355
plaplante@psu.edu

Abstract

If program managers are to fund efforts to improve the quality of systems engineering and requirements decomposition in programs, they need a way to express the costs and benefits of their decisions. This paper details proposed updates to requirements decomposition guidelines that will help generate the "number of system requirements" input to the systems engineering cost model COSYSMO. The updated guidelines give better consideration of human systems integration and other nonfunctional requirements. They also incorporate the quantitative and qualitative feedback of sixteen experts who participated in a validation exercise.

Introduction

As systems become more complex and software-intensive, the practice of rigorous systems engineering becomes more critical to program performance. The modern practice of systems engineering spans both technical and social disciplines. Likewise, requirements engineering is both an engineering and a humanistic endeavor, since understanding individual human behavior and social dynamics is critical to delivering systems that meet users' needs and expectations.

Effort spent on systems engineering has been shown to directly correlate to program schedule and performance (Elm, Goldenson et al. 2008). It follows, therefore, that the quality of requirements engineering should be of similar importance, as related work has supported (Kamata and Tamai 2007) (Hofmann and Lehner 2001).

While previous studies have emphasized the importance of quality systems engineering and requirements engineering practice, there continues to be a disconnect between academic research and industry practice (Muller 2008). One study of software requirements showed that inspection of requirements was mostly informal and ad hoc about one-quarter of the time (Neill and Laplante 2003).

One contributor to this problem is that academic research is difficult to validate without industry support. Empirical research in particular is lacking and any that does exist tends to be within the context of a classroom environment (Hofer and Tichy 2007; España, Condori-Fernandez et al. 2009).

While this paper addresses the question of requirements quality, its goal is not to lay out the characteristics that requirements *should* possess. We begin by making three assumptions: (1) that systems engineering practice will vary between and even within companies (Stem, Boito et al. 2006); (2) that program managers are incentivized to assess requirements quality as quickly and cheaply as possible; (3) that requirements elicited from stakeholders in government and industry will arrive to the systems engineer at varying levels of decomposition and quality.

The framework we propose addresses each of the above assumptions in turn: (1) The framework is intentionally high-level, so that it can be adapted by many systems engineering organizations; (2) The decomposition steps we propose are meant to be used with a cost model, so that any insights gleaned from the process will be associated with cost and benefit; (3) The framework is iterative, incentivizing systems engineers to engage the customer and avoid requirements creep.

The COSYSMO Cost Model

Across industry and government, program managers are the individuals tasked with ensuring a product or program is delivered to the customer within a set schedule and budget. As in any other engineering discipline, the needs of program managers to better understand systems engineering drives research in the field. However, since PM's seek to reduce risk and cost, they are unlikely to volunteer their programs to try out untested models and procedures.

Although systems engineering is critical to program success, more systems engineering effort does not necessarily guarantee ideal program performance (Elm, Goldenson et al. 2008). The Constructive Systems Engineering Cost Model (COSYSMO) gives program stakeholders metrics with which to assess systems engineering planning and budgeting decisions.

Previous work has shown that most early estimates of systems engineering are made by analogy, expert opinion, or heuristics, and that those parametric models that are used apply weighting factors not directly related to systems engineering effort (Stem, Boito et al. 2006). COSYSMO is a parametric model that uses four size drivers and fourteen effort multipliers most important to systems engineering, as identified and validated by industry, academia, and government stakeholders (Valerdi 2008).

The COSYSMO size driver upon which each of the other size drivers are weighted is "number of system requirements". Systems engineering as a discipline developed to translate stakeholder requirements into a desired system; despite expanding definitions of what systems engineering encompasses, requirements engineering remains one of its core functions.

COSYSMO's cost estimating relationship (CER) establishes a link between the number of requirements used to deliver a system and the systems engineering effort needed to define the system. Of course, not all requirements will need the same amount of systems engineering to realize. Therefore, counting rules must be applied. The decomposition guidelines explored in this paper were previously developed by Valerdi and Eiche (2005) and Valerdi (2008).

Requirements Decomposition

Requirements decomposition is not a new concept; the guidelines for decomposition in the context of COSYSMO are meant to supplement and not replace existing documentation. The five decomposition steps defined for COSYSMO are summarized as:

1. Determine the system of interest.
2. Decompose system objectives, capabilities, or measures of effectiveness into requirements that can be tested, verified, or designed.
3. Provide a graphical or narrative representation of the system of interest and how it relates to the rest of the system.
4. Count the number of requirements in the system/marketing specification or the verification test matrix for the level of de-

sign in which systems engineering is taking place in the desired system of interest.
5. Determine the volatility, complexity, and reuse of requirements. (Valerdi 2008)

The goal of these decomposition guidelines is to decompose requirements to a level where useful data exists to assist in cost estimation. Like any cost model, COSYSMO must be calibrated using historical data, ideally from systems similar to the system being designed. Proper decomposition ensures that available data is properly understood and results in an accurate estimate.

Human Systems Integration Requirements

As previous work has shown, early systems engineering decisions make significant impacts on system life cycle costs (Bahill and Henderson 2005). Oftentimes, costs are driven by "nonfunctional" requirements, which are generally defined as requirements that must be met but are not central to defining a system's core function or capability (Neill and Laplante 2003). The understood definition of a nonfunctional requirement varies across organizations, but it is clear that all requirements, functional or not, are the responsibility of the systems engineer to realize (Glinz 2007).

The research presented in this paper is part of a larger study into cost estimation of Human Systems Integration (HSI). Previous work on this project has shown that HSI effort, like other systems engineering effort, is driven by requirements early in a system's development (Liu, Valerdi et al. 2009). One reason HSI is commonly overlooked during early system development is that HSI requirements are often defined as nonfunctional requirements.

The U.S. Air Force Human Systems Integration Office defines HSI requirements as those that "include, but are not limited to, any requirement pertaining to one or more domains of HSI, or the integration of those domains. Broadly, the term encompasses any requirement that contributes to the integration of human considerations into the system being developed," (Air Force Human Systems Integration Office 2009). The domains of HSI are listed in Table 1.

Table 1. Domains of HSI

HSI Domains
Manpower
Personnel
Training
Environment
Safety
Occupational Health
Habitability
Survivability
Human Factors Engineering

Applied Exercise Using Decomposition Guidelines

Part of the effort to expand upon COSYSMO's existing decomposition guidelines is to explicitly identify the impact of nonfunctional requirements, particularly HSI requirements, on systems engineering effort.

Empirical research in systems engineering is difficult for many reasons, such as the availability of data and non-standardized systems engineering methods (Valerdi and Davidz 2009). The Annual International Forum on COCOMO and Systems/Software Cost Modelling brings together representatives from industry, academia, and government interested in furthering the development of cost models such as COSYSMO. In order to capitalize on the expertise of the Forum's participants, we set up an exercise on requirements decomposition as part of a workshop on COSYSMO presented during the Forum.

A total of 16 people participated in the exercise and 14 provided demographic information. Figure 1 summarizes the experience of each participant in the domains of systems engineering, requirements managements/engineering, requirements decomposition, and COSYSMO.

The data show that most of the respondents thought highly of their knowledge of systems engineering and requirements. Familiarity with COSYSMO, however, was more varied. Six respondents considered themselves experts in COSYSMO or had worked on its development, but seven respondents reported little familiarity with the model.

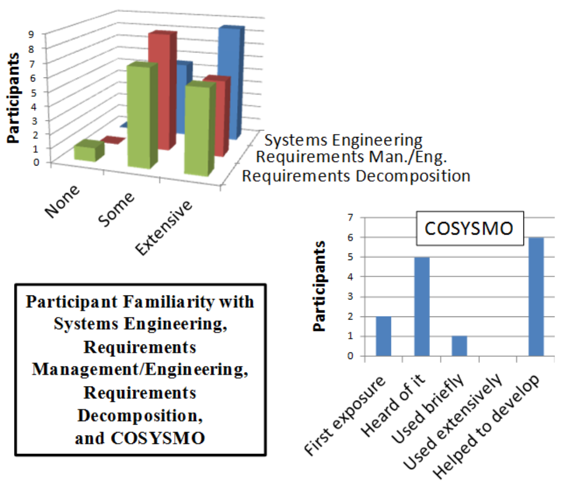

Figure 1. Workshop participant experience with SE, requirements, and COSYSMO.

Exercise Design

None of the participants in the exercise worked on the same programs or even for the same organizations. Many other factors, such as gender, age, and experience, could not be controlled for. However, these conditions are similar to those that would be found in industry. In addition, participants were asked to split into eight pairs to simulate the discussion that would naturally have occurred in a requirements integrated product team (IPT). These considerations helped to improve

The groups were given a twenty-minute overview of the requirements decomposition process outlined in COSYSMO. They were then shown three sequences of eight requirements and asked to perform one of the decomposition steps on each requirement.

The requirements used during the exercise related to a fictional "glass console" likened to the glass displays that have replaced traditional displays in airliner cockpits (see Figure 2).

Figure 2. Participants were asked to design a glass replacement for an SUV console.

The format and content of the displays was modeled after requirements taken from government-furnished documents. An example is shown in Figure 3.

Cautions and Warnings. Method for displaying system warnings, cautions, and alarms must be appropriate given the importance of the situation (**Threshold**).

Figure 3. Sample requirement used during exercise.

Some quantitative results collected from the participants' responses are presented in the next section.

Phase 1: System of Interest

The first phase asked participants to judge "is this requirement at the level of the glass console? If not, is it too high or too low?" The quantitative data taken from participants' responses are presented here.

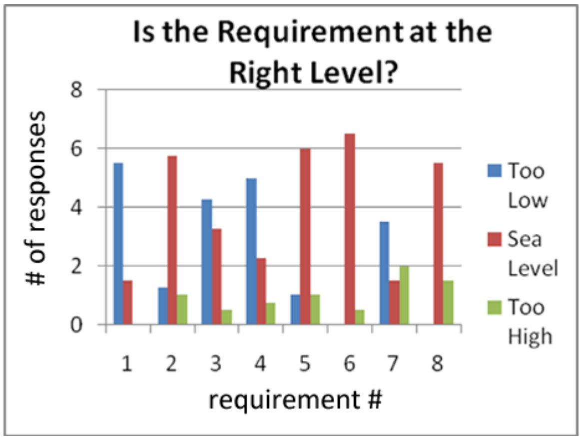

Figure 4. Phase 1 exercise results.

Phase 2: Can the Requirement be Tested, Designed, or Verified?

Participants were asked to provide a "yes" or "no" response separately as to whether each of the requirements in question could be tested, verified, or designed. This section generated the most feedback from participants, as different stakeholders understood these terms to have different meanings.

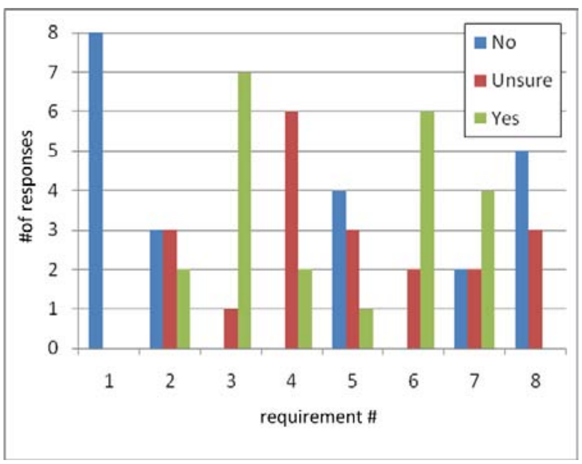

Figure 5. Phase 2 exercise results.

Phase 3: How do Higher HSI requirements affect the System of Interest?

Participants were asked whether the requirement in question, once decomposed, would correspond to *zero*, *one*, or *many* requirements at the level of the system of interest. Respondents were not asked to estimate an exact figure for "many requirements" as such an estimate would have required more time and analysis than was available.

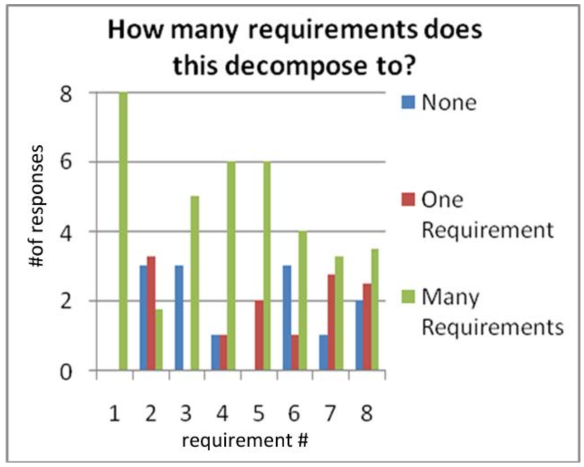

Figure 6. Phase 3 exercise results.

Discussion

In the course of our exercise, it became apparent that even systems engineers with high amounts of experience with requirements management and engineering could come to very different conclusions about the same high-level requirement. This was in large part due to the fact that we used requirements in our exercise that often times were intentionally vague, redundant, or wordy; essentially, the exercise requirements were not ideal requirements.

However, we chose these requirements because they were similar to ones we had reviewed in government-furnished documents and were validated by experts in industry practices. Stakeholder requirements, particularly those that appear in draft form early in a system's development, cannot be expected to be ideal from a requirements engineering perspective. Our decomposition process is intended to allow the early cost estimator or systems engineer to better grasp the expected need for systems engineering in a system using only whatever imperfect requirements are available.

Many insights relevant to our decomposition process emerged from our exercise and they are discussed by decomposition step below.

Step 1: Determine the System-of-Interest

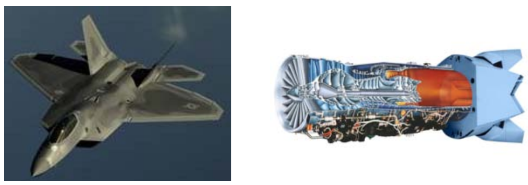

Figure 7. Which is the appropriate system of interest?

Choosing a system-of-interest in COSYSMO is a pragmatic necessity. Consider a case in which you want to estimate the systems engineering effort needed to build a jet engine, a subsystem of a military aircraft system. If you have no experience building jet engines and no historical data from similar projects, you will not be able to calibrate COSYSMO. On the other hand, if you want to estimate systems engineering effort for an aircraft but only have requirements documents available for an engine, then you will not be able to generate a useful requirements count. The ISO/IEC 15288 standard has additional useful information on system-of-interest identification (ISO/IEC 2008).

The decomposition of requirements must be performed by the organization using COSYSMO. The level of decomposition of interest for COSYSMO is the level in which the system will be designed and tested; which is equivalent to the TYPE A, System/Segment Specification (Department of Defense 1985).

The qualitative feedback from our exercise was that the first decomposition step also needed to ask "is the requirement in question at the right decomposition level to be analyzed using COSYSMO?" This assessment should apply the use case framework developed by Cockburn for software use cases and expanded upon by Valerdi (Cockburn 2001) (Valerdi 2008).

Step 2: Assess Ability to Test, Verify, or Design

The Defense Acquisition Guidebook (2009) defines verification as the process that "confirms that the system element meets the design to or build-to specifications as defined in the functional, allocated, and product baselines,". Verification can be accomplished through 4 different means: (1) Demonstration, (2) Inspection, (3) Analysis, and (4) Test. Verification principally happens during the test and evaluation phase of systems development, but systems engineers are responsible for constant verification of requirements throughout the life cycle of a system.

In certain cases, verification of a requirement within the context of a system of interest may not be possible, but the requirement may still impact systems engineering effort needed. Consider, for example, if the system of interest was a car steering wheel and the requirement being considered dealt with crash safety. The designers of the steering wheel should make design considerations for safety, but verification of safety requirements would probably be best done as part of vehicle-level crash tests.

Both the quantitative and qualitative feedback gathered during our exercise confirmed that there was disagreement between the participants over the definitions of "test," "verify", and "design.

Step 3. Assess System of Interest Relationship with Rest of the System

This step focuses on the hierarchical relationship between the system elements. This information can help describe the size of the system and its levels of design. It serves as a sanity check for the previous two steps.

Once a system of interest has been identified and the appropriate requirements have been decomposed to the target level, the user of COSYSMO might assume that she can proceed by simply counting the resulting requirements and inputting the result into the parametric model. However, requirements decomposition is an iterative process. As the initial set of requirements for a system of interest are decomposed from a higher system's requirements, the decomposition process itself will likely reveal missing requirements, requirements requiring clarification, or unexpected connections to other systems, the environment, or external pressures (Laplante 2009).

The term nonfunctional incorrectly implies that requirements that are not functional must have a smaller impact on a system than functional requirements. Reliability, user interface, and safety are just some of the types of nonfunctional requirements that clearly defy such a simplification.

While the definition of nonfunctional requirements contributes to the difficulties of properly accounting for their impact on systems engineering effort, the difficulties are exacerbated by the requirements elicitation structure of the Department of Defense (DoD). In the United States, the DoD is the largest government customer of large systems acquisitions and is a leading agency in the evolution of systems engineering best practices and standards. The DoD defines system requirements in a document called a Capabilities Development Document (CDD). CDD's assign a hierarchy of importance to requirements explicitly using a name scheme and implicitly, via their position within the CDD.

In the case where a nonfunctional requirement results in one requirement at a system's "sea level", then the resulting requirement should be counted as a derived requirement. In general, a derived requirement is a requirement that results from external constraints. Consider, for example, a coffee pot system. One of its original requirements is that it must be able to produce boiling water, but an external constraint requires that water temperature not exceed 205 degrees Fahrenheit for safety reasons. The original requirement would be modified and the resulting requirement would be a derived requirement. This modification may impact the complexity rating of the requirement, but would not affect the overall count.

However, as our exercise showed, participants disagreed at times as to whether a nonfunctional requirement corresponded to one requirement at the sea level of a system or many. In instances where this distinction is not clear, counting a nonfunctional requirement as a single requirement would produce incorrect results. Instead, if a nonfunctional requirement results in a derived requirement at a higher level in the system being developed, that requirement must be decomposed to be counted properly at the sea level.

Step 4. Count Requirements and Step 5. Assess Weighting Factors.

Our experiment did not address the final two steps of the decomposition process. Ideally, if the first three steps of decomposition are completed correctly, counting the resulting requirements should be a straightforward task.

Once the quantity of requirements has been determined, giving them a weighting of "easy," "nominal," or "difficult" serves to capture the knowledge and experience of experts familiar with the system being developed. The weights for these factors were determined using expert opinion through the use of a Delphi survey (Valerdi 2008). The volatility and reuse factors are optional and depend on the version of COSYSMO implementation being used.

Conclusions

The observations taken from our exercise have prompted us to make some changes to COSYSMO's decomposition guidelines.

In step 1, we will make a more clear distinction between the terms "system-of-interest" and "sea level". Determining the system-of-interest establishes whether a given requirement is relevant. Determining sea level establishes whether a relevant requirement is at the correct level of decomposition to be counted in COSYSMO.

In step 2, we will change the guideline to simply read "Assess Ability to Verify or Design," and to summarize the guidance provided on these terms by the Defense Acquisition Guidebook and other sources.

We intend to expand upon step 3 of the decomposition process. Systems engineers using this process need to be able to determine whether an HSI or other nonfunctional requirement is relevant to the system-of-interest. Once this is done, a derived requirement at the highest level must be written. This derived requirement must then be decomposed to a set of sea level requirements to be counted in COSYSMO. Step 3 is therefore actually a multistep process.

Qualitatively, participants were immediately able to judge the "quality" of a requirement. In several cases, participants objected to answering a question on the grounds that the requirement presented would need to be rewritten in order for the question to be answered. These results highlight the need for the guidelines presented herein. The requirements presented in these exercises were all modeled off of actual requirements found in government documents. As such, there would likely have not been opportunity to modify the requirements. The requirements decomposition guidelines help to mitigate issues in requirements quality by quantifying the systems engineering impact each requirement poses. Understanding these impacts helps to identify those requirements that are least understood or have least quality.

Since decomposition of requirements for COSYSMO ultimately leads to an input to a cost model, the output of the model can in turn help systems engineers justify the cost and benefit of quality requirements to decision-makers. Better requirements ultimately result in a higher-performing system.

Next Steps

This exercise has confirmed the need for better requirements decomposition guidelines. Our participants expressed frustration and confusion over their tasks, either because the sample requirements were not written well or because they did not understand the decomposition guidelines. Such frustrations evoke the experiences of actual requirements engineers working on real systems.

At the same time, our exercise shows that even participants with minimal systems engineering or requirements engineering training can benefit from basic guidelines. This is important, as systems' requirements must capture the needs of varied stakeholders, most of whom may not be systems engineering professionals.

A follow-on exercise using the modified decomposition guidelines is planned for during the USC Annual Research Review, in March of 2010 to gain further insights.

References

Air Force Human Systems Integration Office, A. (2009). Human Systems Integration Requirements Pocket Guide. Washington, D.C.

Bahill, A. and S. Henderson (2005). "Requirements development, verification, and validation exhibited in famous failures." Systems Engineering 8(1): 1-14.

Cockburn, A. (2001). Writing effective use cases. Boston, Addison-Wesley.

Department of Defense (1985). MIL-STD 490A: Specification Practices. Washington, DC, Government Printing Office.

Elm, J. P., D. R. Goldenson, et al. (2008). A Survey of Systems Engineering Effectivenesss - Initial Results. N. S. E. Committee, Carnegie Mellon University & National Defense Industrial Association.

Glinz, M. (2007). "On non-functional requirements." Proc. RE 7: 21-26.

Hofmann, H. F. and F. Lehner (2001). "Requirements engineering as a success factor in software projects." Software, IEEE 18(4): 58-66.

ISO/IEC (2008). ISO/IEC 15288: System Life Cycle Processes. Geneva, Switzerland, International Organization for Standards.

Kamata, M. I. and T. Tamai (2007). How Does Requirements Quality Relate to Project Success or Failure? Requirements Engineering Conference, 2007. RE '07. 15th IEEE International.

Laplante, P. A. (2009). Requirements engineering for software and systems. Boca Raton, CRC Press.

Liu, K. K., R. Valerdi, et al. (2009). Economics of Human Systems Integration: A Systems Engineering Perspective. Conference on Systems Engineering Research (CSER). Loughborough, UK.

Muller, G. (2008). Industry and Academia: Why Practitioners and Researchers are Disconnected.

Neill, C. J. and P. A. Laplante (2003). "Requirements Engineering: The State of the Practice." Software, IEEE 20(6): 40-46.

Stem, D. E., M. Boito, et al. (2006). Systems engineering and program management : trends and costs for aircraft and guided weapons programs. Santa Monica, CA, RAND.

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

Valerdi, R. and H. L. Davidz (2009). "Empirical research in systems engineering: challenges and opportunities of a new frontier." Systems Engineering 12(2): 169-181.

Valerdi, R. and B. Eiche (2005). On Counting Requirements. 3rd Conference on Systems Engineering Research. Hoboken, NJ.

Biography

Kevin Liu is a graduate research assistant at the Systems Engineering Advancement Research Initiative at the Massachusetts Institute of Technology. He is currently pursuing a M.S. in Technology in Policy from MIT and received a B.S. in Electrical Engineering from the U.S. Naval Academy, from where he was commissioned a 2ndLt. in the U.S. Marine Corps. His research interests include cost estimation and human-systems integration.

Dr. Ricardo Valerdi is a Research Associate at the Lean Advancement Initiative at the Massachusetts Institute of Technology. He holds a Ph.D. in Industrial & Systems Engineering from the University of Southern California. His research interests are in systems engineering, cost estimation, decision making, enterprise transformation, and enterprise performance measurement. Dr. Valerdi was on the Board of Directors of INCOSE and is the founding co-Editor-in-Chief of the Journal of Enterprise Transformation.

Dr. Phil Laplante is Professor of Software Engineering at Penn State. His research interests are in requirements engineering, software testing, and project management. He holds a PhD in Computer Science from Stevens Institute of Technology, is a Fellow of the IEEE and. He is a licensed professional engineer in Pennsylvania and a Certified Software Development Professional.