# AC 2012-3389: SYSTEM ENGINEERING COMPETENCY: THE MISSING COURSE IN ENGINEERING EDUCATION

**Mr. Charles S. Wasson, Wasson Strategics, LLC**

Charles Wasson is an engineering textbook author, instructor, and consultant for Wasson Strategics, LLC, a professional training and consulting services firm specializing in systems engineering, technical project management, organizational development, and team development. In 2006, Wasson authored a new systems engineering text entitled System Analysis, Design, and Development: Concepts, Principles, and Practices as part of the John Wiley & Sons' System Engineering and Management series. The text received the Engineering Sciences Book of the Year Award from the International Academy of Astronautics (IAA) in Paris, France. As an internationally recognized author and instructor in system engineering and its organizational application, he is an invited guest speaker and panelist at professional meetings and symposia. Wasson champions the need to strengthen undergraduate engineering programs with a course in the fundamentals of system engineering. He holds B.S.E.E. and M.B.A. degrees from Mississippi State University and a certificate in systems engineering from Stevens Institute of Technology. His professional affiliations include the American Society for Engineering Education (ASEE), the International Council on System Engineering (INCOSE), and the Project Management Institute (PMI).

# Systems Engineering Competency
# The Missing Course in Engineering Education

**ABSTRACT** This paper addresses the need for and proposes solutions to bolster the competency of the engineering professionals at two levels: 1) strengthen undergraduate and graduate level engineering education to include a robust Systems Engineering(SE) problem solving / solution development course and 2) shift the Systems Engineering paradigm found in many organizations through education and training to employ scalable SE methodologies for projects ranging in size from small to large complex systems. The objective is to educate and train engineers to *efficiently* and *effectively* develop systems, products, and services that can be verified and validated as meeting user operational needs and expectations with acceptable risk and life cycle costs within project triple constraints – i.e., technical, cost, and schedule performance. To fill the void, the paper proposes a minimum set of topics required for a Fundamentals of Systems Engineering course and instructor / instructional team qualifications.

Universities, colleges, government, and industry often create or procure Systems Engineering courses that may provide *necessary* but *insufficient* content for educating and training engineers in concepts, principles, and practices required to achieve system, product, or service development success. The problem is exacerbated by the need for Systems Engineering instructors with industry experience, multi-disciplined systems thinking, and excellent instructional communications skills qualifications.

**Keywords:** Systems engineering competency, system engineering process, systems engineering paradigms, systems engineering fundamentals, systems engineering education and training.

## DEFINITIONS OF KEY TERMS

The following is a listing of definitions of key terms essential to topics discussed in this paper.

- **Competency** – Behaviors that encompass the knowledge, skills, and attributes required for successful performance. [1]
- **Paradigm** – A model representing a unique approach, perspective, or pattern of behavior for observing the world, making decisions, or communicating views.
- **Plug & Chug Paradigm** - Represents a traditional engineering teaching model in which students *Plug* a value into an equation and *Chug* out an answer for solving classical boundary condition problems.
- **Design-Build-Test-Fix Paradigm** – An *ad hoc*, *iterative* process traceable to scientific inquiry that lacks an insightful methodology in which engineers: 1) design an entity, 2) build it in the lab, 3) test it, and 4) fix, rework, or patch the design or its physical implementation in a seemingly endless loop until convergence at a final solution is achieved or schedule and cost resources are depleted.

- **Paradigm Shift**- A transformational change driven externally by: 1) the marketplace or technology, or 2) internally through visionary leadership to advance state of the practice or being from one paradigm to another over a planned period of time.
- **System Engineering** - "The multi-disciplined application of analytical, mathematical, and scientific principles to formulating, selecting, and developing a solution that has acceptable risk, satisfies user operational need(s), and minimizes development and life cycle costs while balancing stakeholder interests." [2]

## INTRODUCTION

One of the challenges of industrial enterprises operating in a highly competitive global economy is the capability to *efficiently* and *effectively* engineer systems that satisfy customer and user operational needs within budget, schedule, technology, and risk constraints. Unfortunately, the "engineering of systems" performed in many organizations is often characterized as chaotic, ineffective, and inefficient. Objective evidence of these characteristics is exemplified by non-compliance to requirements, cost overruns, and late schedule deliveries in program metrics for a project's contract or task triple performance constraints– i.e., technical, cost, and schedule.

Causal analysis of this performance reveals a number of contributory factors: a lack of technical leadership, a lack of understanding the user's problem / solution spaces, creation of single point design architectures and solutions without due analysis of alternatives (AoA), a lack of multi-disciplined decision making, poor documentation and configuration control, et al. Further analysis indicates these factors are symptomatic of a much larger competency issue traceable to engineering education - the lack of a Systems Engineering fundamentals course. Ideally, a course taught by seasoned instructors with in-depth industrial experience acquired from a diversity of small to large, complex systems.

To meet program accreditation requirements, industrial needs, and remain competitive, colleges and universities institute a Systems Engineering course or capstone project based on SE principles and practices. However, the outcomes of these projects tend to focus on domain-based design such as electrical, mechanical, software, et al with a minimal or no level of SE concepts, principles, and practices required to effectively and efficiently perform system development.

System development, especially for moderate to large, complex systems, adds a new dimension of complexity beyond *domain-centric* engineering concepts. It requires education and training in the "engineering of systems" – i.e., Systems Engineering. Where voids exist in this knowledge, the organization's ability to "engineer systems" is typically characterized as *ad hoc*, *chaotic*, and *dysfunctional* rather than *efficient* and *effective*.

In response to these operational needs, this paper explores the ad hoc, chaotic, and dysfunctional nature of Systems Engineering in many industry organizations. We trace its origins of the industrial Plug and Chug … Specify-Design-Build-Test-Fix Paradigm and its migration from the Plug and Chug … Design-Build-Test-Fix Paradigm acquired informally in engineering school. Whereas these paradigms may be effective for academic application, they are not suitable or scalable to larger, complex system, product, or service development efforts.

In contrast, Systems Engineering applies "up front" analysis and application of a proven methodology to translate and transform a user's abstract operational need or issue into the physical realization of a system that can be validated as meeting the need. For brevity, examples include:

- Identification of system / entity users and stakeholders
- Establishing system boundaries and interfaces with external systems and the operating environment
- Understanding the user's problem and solution spaces
- Understanding the user's use cases and scenarios
- Identification of requirements
- Development of deployment, operations and sustainment, and disposal concepts
- Derivation, allocation, and flow down of requirements to multiple levels and traceability to source or originating requirements
- Development of requirements, operations, behavioral, and physical architectures
- Development of system phases, modes, and states of operation and Mission event timelines (METs)
- Development of multi-level design solutions
- Analysis of alternatives (AoA)
- Modeling and simulation
- Integration and test engineering and specialty engineering – i.e., human factors, reliability, maintainability, et al - to avoid showstopper surprises that impact system acceptance, delivery, and user satisfaction.
- Verification and validation (V&V)
- Et al

Anecdotal evidence based on the author's experiences suggest that many engineers are estimated to spend on average from 50% to 75% of their total career hours collaborating with others concerning the engineering of systems – i.e., SE - for which they have no formal education. Aerospace and defense tends to be a higher percentage; commercial industry is generally less. Yet, engineers are perceived as having the requisite skills required for the "engineering of systems." This reality translates into technical cost, and schedule performance risks thereby impacting organizational performance and profitability, as applicable. Institution of a Fundamentals of Systems Engineering course in engineering curricula taught by instructors with both in-depth industrial and instructional experience qualifications would significantly reduce these risks, improve organizational engineering capabilities and performance, enterprise profitability, and customer and user satisfaction.

Analysis and assessment of organizational skills typically reveals competent domain discipline knowledge and understanding in the application of engineering, science, and math concepts in the development of "components" of systems. However, when these individuals and their skill sets are integrated into multi-discipline, problem-solving and solution-development, decision making teams dependent on human interpersonal skills, system development performance suffers.

To solve these problems, academia, industry, and government supported the development of SE capability assessment models, SE competency models, et al. The models serve as referential frameworks characterized by attributes for assessing levels of organizational and individual capabilities as *potential* indicators of future performance ... subject to a "willingness to perform."

This leads to the question: *Why do organizations that publicize higher capability maturity assessments have system development performance issues*? Answers to this question include a multitude of factors such as education and training, experience, leadership, organization, resources, technical planning, processes, tools, and methods. By virtue of the capability and competency assessments, one could theoretically say that this condition should not exist and there must be something fundamentally wrong with the referential models.

Filtering out factors such as leadership, resources to some degree, et al with a focus on engineering and training, you have to lead, work, and understand the daily lives of engineers performing system development and the state of the practice of SE to answer the question. More specifically, the answer to the question is multi-faceted requiring several answers addressing the disparity in organizational understandings of SE, the types of SE courses taught, and the need for SE instructors with both industry and instructional experience.

In general, most models employ attributes of expected outcomes as a basis for assessing levels of capabilities or competency. Since there are different ways of transforming system requirements into deliverable systems, products, or services, the models do not dictate *how* an organization or individual achieves those outcomes. For example, some organizations employ the Plug and Chug ... Specify-Build-Test-Fix Paradigm with the perception that it is SE because it is "highly iterative" and "recursive" similar to the application of the SE Process. Both the SE Process and the paradigm may result in the same documents by title. However, the Plug and Chug ... Specify-Build-Test-Fix Paradigm approach is typically *inefficient* and *ineffective*, especially for moderate to large, complex systems resulting in technical compliance issues, late deliveries, and overrun budgets. In this regard an organization could have a high capability assessment rating – i.e., exhibit and present the right attributes - but have poor performance on a program / project due to the inefficiency and ineffectiveness of the paradigm.

In this context, this paper provides personal observations common across many organizations based on the author's work in SE, project management, organizational development, and team development.

## STATEMENT OF THE PROBLEM

*Despite the formulation and development of Systems Engineering capability assessment and competency models, certifications, education and training courses, et al, system development projects continue to exhibit technical performance issues concerning the engineering of systems.*

Contributing to this overall problem are several contributory performance effecters:

1. Misperceptions that writing specifications, developing designs, performing integration and test, and then verifying and possibly validating a system is, by definition, "Systems Engineering."

2. Erroneous perception that the ad hoc, "Plug and Chug ... Specify-Design-Build-Test-Fix" Paradigm originating from scientific inquiry processes and engineering school is SE.
3. Failure to provide formal SE courses as a requirement as part of engineering degree programs allows graduates to enter academia, industry, and government to become part of the in-grained Plug and Chug ... Specify-Design-Build-Test-Fix Paradigms that continue to contribute to technical program performance issues.
4. Inability to differentiate the System Development Process from the SE Process.
5. Erroneous assumption that the SE Process is performed only at the system level.
6. Failure to recognize that specifications are more than a random sets of "shalls" organized in a standard outline structure.
7. Failure to recognize that SE work products capture the artifacts of the data-driven, decision-making processes rather than drive the central focus of SE activities.
8. Failure to understand that SE provides a framework of starting point templates for organizing the engineering of systems.
9. Failure to recognize that every entity at every level of abstraction within a system is characterized by four domain solutions: requirements, operations, behavioral, and physical – that when integrated comprise the overall system design solution.
10. Recognition that SE competency requires two levels of knowledge and experience: 1) understanding the SE concepts, principles, and practices concerning the engineering of systems (undergraduate level) and 2) understanding how to *efficiently* and *effectively* tailor SE practices to achieve project objectives within technical, technology, cost, schedule, and risk constraints (graduate level).
11. Misperception that documenting organizational processes and checklists unsupported by formal SE educational, training, and leadership – e.g., "paint-by-number" engineering [3] – will result in engineering and project success.

    Please note that documented processes, which capture organizational best practices, lessons learned, and checklists, should be in-grained as a mental reference for planning and performing tasks to minimize risk and support engineering decision making, not for substitution of informed engineering judgment.

Solutions to this overall problem and its subelements require consensus solutions by academia, industry, and government through a series of action-oriented steps that promote the awareness, recognition, and a willingness *to correct the problem. For additional information on many of these topics, please refer to Wasson* [2]. The scope of this paper focuses on three key aspects of the problem:

1. Misperceptions that the Plug and Chug ... Specify-Design-Build-Test-Fix Paradigm is SE.
2. Recognition of the differences between Systems Acquisition and Management SE courses versus System Development courses as the foundation for engineering education.
3. The need for seasoned SE course instructors with in-depth industrial experience and instructional ability to communicate that knowledge in a formal course setting.

In 2003 and 2006 by the National Defense Industrial Association (NDIA) Systems Engineering Division published a Task Group Report concerning SE issues within the U.S. Department of Defense (DoD) and the defense industry. The results of the 2006 survey are:

- **Issue #1** - Key systems engineering practices known to be effective are not consistently applied across all phases of the program life cycle.
- **Issue #2** - Insufficient systems engineering is applied early in the program life cycle, compromising the foundation for initial requirements and architecture development.
- **Issue #3** - Requirements are not always well-managed, including the effective translation from capabilities statements into executable requirements to achieve successful acquisition programs.
- **Issue #4** - The quantity and quality of systems engineering expertise is insufficient to meet the demands of the government and the defense industry.
- **Issue #5** - Collaborative environments, including SE tools, are inadequate to effectively execute SE at the joint capability, system of systems (SoS), and system levels.

Although these issues focus on the DoD, the author has observed comparable issues in other business domains.

## LITERATURE REVIEW

A survey of engineering literature revealed no papers and research explicitly targeting the: 1) Plug and Chug ... Specify-Design-Build-Test-Fix Paradigm and its impact on organizational performance and 2) the need for seasoned SE instructors with in-depth industry SE experience and instructional ability to communicate that knowledge in a formal course setting to engineering students. Some of the capability and competency models identified below differentiate system acquisition from system development.
A plethora of information, papers, and research abound concerning related topics that contribute to the overall problem of program / project performance, in general. These topics are summarized and elaborated below.

- Systems Engineering issues
- Emergence of Systems Engineering education
- Criteria for Accrediting Engineering Programs, 2012 - 2013
- Educating the Engineer of 2020
- Development of systems thinking skills in engineers.
- Integration of Systems Engineering courses into capstone courses
- Certification of Systems Engineers
- Establishment of Systems Engineering standards
- Assessment of organizational System Engineering capabilities
- Systems Engineering competency models
- Adequacy of project funding for Systems Engineering

**Systems Engineering Issues**
Compelling objective evidence of the lack of SE or the adequacy of SE competency and its roots are addressed in several papers. Examples include:

- NDIA [4] - Top Five Systems Engineering Issues within Department of Defense and Defense Industry.
- Castellano [5] - Program Support: Perspectives and Systemic Issues
- Bar Yam [6] - When Systems Engineering Fails --- Toward Complex Systems Engineering.
- Bahill and Henderson [7] - Requirements Development, Verification, and Validation Exhibited in Famous Failures.
- JPL [8] - Report on the Loss of the Mars Polar Lander and Deep Space 2 Missions.
- NASA [9] – Mars Climate Orbiter Mishap investigation Board Phase I Report.
- Honour [10] - Understanding the Value of Systems Engineering.
- Miller, et al [11] - The Strategic Management of Large Engineering Projects.

**Emergence of Systems Engineering Education**
Fabrycky [12] addresses how "Systems Engineering is now gaining international recognition as an effective technologically based interdisciplinary process for bringing human-made systems into being, and for improving systems already in being." The paper includes two tables: one for "Systems Centric SE programs" and the second for "Domain Centric SE programs" by institution and degree programs.

**Criteria for Accrediting Engineering Programs, 2012 - 2103**
The Accreditation Board of Engineering and Technology (ABET) [13] establishes criteria for Student Outcomes, et al requirements for accrediting engineering degree programs.

**Educating the Engineer of 2020**
The National Academies "Educating the Engineer of 2020" report [14] states "In the past, steady increases in knowledge have spawned new subspecialties within engineering (e.g., microelectronics, photonics, and biomechanics). However, contemporary challenges - from biomedical devices to complex manufacturing designs to large systems of networked devices—increasingly require a systems perspective. This drives a growing need to pursue collaborations with multidisciplinary teams of technical experts..." Hsu, Raghunathan, and Curran [15] observe that universities are challenged to meet the demands of industry by supplying graduates with a sound foundation in Systems Engineering.

Borrego and Bernhard [16] cite Patil and Codner's [17] analysis of engineering education accreditation in the U.S. Europe, and the Asia Pacific region concerning the "workplace performances of engineering graduates have been a constant subject of criticism. There is increasing evidence of a mismatch between graduate student's skills developed during their studies and those needed by gradate engineers in the workplace (p. 646)." Adams, et al [18] cite Duberstadt [19], Sheppard [20], et al concerning the need to align problem solving and knowledge acquisition with professional practice. Sheppard notes "Although engineering

education is strong on imparting some kinds of knowledge, it is not very effective in preparing students to integrate their knowledge, skills, and identity as developing professions … In the engineering science and technology courses, the tradition of putting theory before practice and the effort to cover technical knowledge comprehensively allow little opportunity for students to have the kind of deep learning experiences that mirror professional practice and problem solving."

**Development of Systems Thinking Skills**
Since "systems thinking" skills are an integral part of Systems Engineering attributes, Davidz and Nightingale [21] provide research data concerning the levels of significance of experiential learning.

**Integration of Systems Engineering Courses into Capstone Courses**
Several universities offer systems engineering instruction as part of the capstone course experience. Schmidt, et al [22], Nemes, et al [23], and Corns, et al [24] provide research and recommendations concerning the importance of Systems Engineering to capstone projects and its introduction prior to the capstone project courses.

**Certification of Systems Engineers**
The International Council on Systems Engineering (INCOSE) [25] provides certification of "Multi-Level Base Credentials" for systems engineers at the Entry Level, Foundation Level, and Expert Levels with extensions to "cover a specific domain or subset of systems engineering in more detail". Davidz and Nightingale [26] observe that the adequacy of certification programs remains controversial, primarily due to their newness for widespread certification.

**Establishment of Systems Engineering Standards**
Sheard [27] provides an evolutionary "frameworks quagmire" of Systems Engineering and Software Engineering standards established by organizations such as the US Department of Defense (DoD), the Institute of Electrical and Electronic Engineers (IEEE), Electronic Industries Alliance (EIA), the International Organization of Standards (ISO), et al. ISO examples include:

- ISO / IEC 12207:2008 – Software life cycle processes [28]
- ISO / IEC 15288:2008 – System life cycle processes [29]
- ISO / IEC 19760 – Guide for ISO /IEC 15288 System life cycle processes [30]
- Et al

**Assessment of Organizational System Engineering Capabilities**
The Software Engineering Institute (SEI) at Carnegie-Mellon University developed the Capability Maturity Model Integration for Development (CMMI-DEV) [31] for assessing organizational system development capabilities.

**Systems Engineering Competency Models and Certifications**
The following is a listing International Council on Systems Engineering (INCOSE), the Defense Acquisition University (DAU), NASA, and the Mitre Institute established a set of SE competency models as well as INCOSE certification of SEs.

- Defense Acquisition University (DAU) - Systems Planning Development Research and Engineering - Systems Engineering / Programs Systems Engineer (SPRDE-SE/PSE) Model Levels I – III [32]
- NASA – Academy for Program / Project & Engineering Leadership (APPEL) [33]
- The Mitre Institute – Systems Engineering Competency Model [34]
- INCOSE Systems Engineering Professional (SEP) Certifications [35]
- INCOSE United Kingdom (UK) Competency Model [36]

Squires, et al [37] present a competency taxonomy to guide the experience acceleration of program lead systems engineers.

**Adequacy of Project Funding for Systems Engineering**
Honour [38] provides recommendations based on research concerning the optimal amount of funding required to adequately ensure levels of success.

## TOPIC MOTIVATION

Since World War II, the need to engineer systems, especially moderate to large, complex systems motivated the US Department of Defense, NASA, et al to investigate, develop, and mature problem-solving / solution development methodologies to enable engineers to more efficiently and effectively develop systems derived from evolving best practices, lessons learned, program / project failures, etc. Over several decades, engineering education and training courses have been developed, SE standards established, certification of systems engineers, rating for organizational capability assessments established, et al. Yet, despite this infrastructure framework of prevention safeguards and improvements, technical programs often have technical problems that ultimately impact cost and schedule delivery performance.

Despite the plethora of knowledge frameworks, one question lingers. *Where is the shortfall in the current framework of educational courses, standards, certifications, organizational assessments, documented processes, et al that continues to result in technical program problems or failures*?

Based on the literature review, one could easily counter "what could be missing from the infrastructure framework that is already in place?" A number of factors related to engineering education and training, organizational leadership, project resources, technical planning, etc. contribute to this condition. Technically, the primary answer resides in the *ad hoc Plug and Chug … Specify-Design-Build-Test-Fi*x engineering paradigm that persists in many organizations. The paradigm is based on the erroneous perception that writing specifications; developing designs; performing integration, test, and verification is, by definition, Systems Engineering. As a result of this misperception, any engineer that has interfaced two components is "knighted" by organizations as a Systems Engineer, regardless of their education, training, knowledge, experience, and so forth. Many of the personnel who are members of an SE functional organization are typically System Analysts, not SEs.

The condition that exists in many organizations at the project, functional, and executive organizational levels is a range of misperceptions and beliefs that SE is being performed. To better understand the condition, let's employ a generalized case study to illustrate the two extremes of the condition.

## ORGANIZATIONAL SYSTEM DEVELOPMENT PERFORMANCE

Organizations vary significantly in terms of their Systems Engineering capabilities and degrees of SE influenced by their perceptions of what SE is. Based on the author's experiences, the extremes of the spectrum of organizations consist of those:

1. Considered "best of class."
2. That employ the Plug and Chug ... Specify-Design-Build Test-Fix Paradigm as SE.

Some will contend that SE concepts, principles, and practices are not applicable to engineering in their business domains. This premise is difficult to accept based on the fundamentals of SE. Every viable, mission-oriented, business entity – e.g., services organizations, non-profits, et al – serves a purpose, has interfaces with external systems in its operating environment – e.g., customers, suppliers, competitors, et al, and produces performance-based outcomes – e.g. systems, products, services, and behaviors – that are delivered to or sold in the marketplace for some form of return on investment (ROI) or to provide service benefits. This view is reflective of the mindset that SE applies only to physical systems and products such as cell phones, computers, etc. without recognition that organizations are also systems that produce products and services for both external and internal customers and users.

To illustrate the two organizational extremes, consider the example illustrated in Figure 1. Panel 1 represents what organizations communicate with good intentions via proposals and presentations to customers. They boldly proclaim that their planned strategy will be based on a progressive sequence of timely decisions that result in smooth, seamless, and deliver on-time performance within project management triple constraints – i.e., technical, cost, and schedule performance, each with its own but interdependent levels.

On contract award, the program / project organization embarks on the proposed system development process as illustrated in Panel 2 (Figure 1). However, the "effort has a delayed start due to the failure to perform a "progressive sequence of timely decisions" as indicated by the perturbations. For example, the development team may be staffed by leadership positions that may be undefined or unfilled, key technologies or personnel may not be mature or available when planned, new personnel may not agree with the proposed solution based on the proposal team's lack of understanding of the user's problem and solution spaces, and so forth.

As time progresses, the Project Manager and the Project Engineer become apprehensive about current budget and schedule performance because the development team has not started machining / bending metal, assembling hardware, or coding software when committed due to a lack of informed decision making. When this condition occurs, any objective evidence of a true Systems Engineering concepts, principles, and practices is summarily rejected as "philosophical theory and bureaucratic paperwork … we don't have time for this. The organization reverts to its fire-fighting mode of system development to meet schedules and budgets."

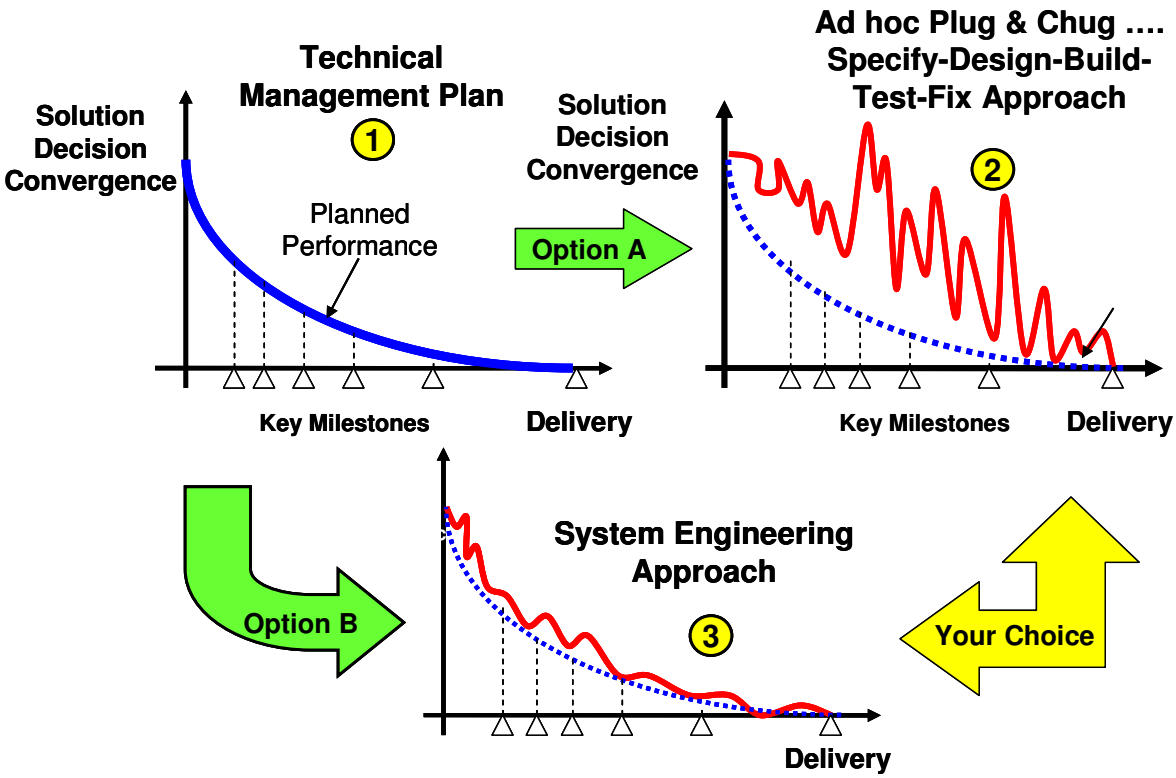


**Figure 1: Contrasting SE-Based System Development versus the Plug & Chug …Design-Build-Test-Fix Paradigms.**

The program reverts to their traditional business-as-usual Plug & Chug … Design-Build-Test-Fix Paradigm that leaps to a single point design solution decision, which lacks supporting objective evidence such as peer reviews, analyses of alternatives (AoA), trade studies, etc. Program / project decision-making occurs … one day they have selected a solution; the next day they are grasping for new solutions and so forth. As a result, we see major decision-making oscillations in Panel 2 (Figure 1). Panicked by schedule milestone commitments and customer perceptions of the lack of progress in bending metal, coding software, et al, the system or product is quickly rushed to manufacturing or vendors and rushed into integration and test with bold pronouncements to the customer of "being ahead of schedule."

Then … *a reality check occurs* …

The customer recognizes that despite overtures of the program / project performing Systems Engineering and being in the System Integration and Test Phase, the system developer is redesigning a major portion of the system or its interfaces – e.g., Plug and Chug ... Specify-Design-Build-Test-Fix - … due to a lack of understanding of the user's problem space and solution space(s), teams failed to approve interfaces or properly characterize the operating environment, et al. What began as the smooth, seamless technical plan of proposed "progressive sequence of timely decisions" in Panel 1 evolves into major decision oscillations illustrated in Panel 2.

The organization works extended hours nights, weekends, or holidays, attempting to rework, patch, or redesign the evolving design solution until the system developer or customer either totally deplete funding or negotiate a lesser capability delivery with the risk of discovering latent defects in fielded systems. The need for redesign and rework in many cases can often be attributed to:

- The lack of "up front" analysis and integration of specialty engineering – e.g., human factors, reliability, et al – integration & test, verification and validation V & V, operations and sustainment personnel into the "up-front", data-driven, decision-making.
- Failure to commit when planned to sound, data driven, technical decisions

In contrast, Panel 3 illustrates how SE can *minimize* these problems. Observe the decision-making oscillations in Panel 3; even SE is dependent on and subject to the human conditions of consensus decision-making. However, despite the frailties of the human condition to make and commit to timely decision-making, the amplitude of the oscillations in applying SE methods is relatively minor compared to Panel 2.The downward trend to delivery reflects the proposed technical plan to the customer of executing a "progressive sequence of timely decisions."

*So, HOW do organizations evolve into these types of performance*?

Panel 3 exemplifies an organization that employs a valid SE approach and has competent personnel that have been educated and trained as leaders, SEs, or domain engineers applying SE methods to their assigned role and tasks. Personnel and teams understand how to transform SE philosophy theory into practice. Using the approach and SE Process Model developed by Wasson [39] [40], the integrated team of stakeholders applies its SE problem solving / solution development methodology to develop a system design solution that will avoid many of the pitfalls of the Panel 2 organization.

In contrast, the Panel 2 organization begins with SE concepts, principles, and practices based process. However, over time, the program / project eventually defaults to the Plug & Chug … Design-Build-Test-Fix Paradigm. That is, well-intentioned engineers working on the program / project, each desiring to:

- Take a quantum leap to a single point solution – i.e., their preferred physical domain solution without due process. The phrase, "quantum leap to a single point solution," refers to reading requirements and immediately creating a physical solution without understanding the underlying problem the user is trying to solve or performing the up-front analysis to fully understand the integrated set of requirements, operational, behavioral, and physical capabilities required of the system [41]. They fail to satisfy the *necessary* and *sufficiency* criteria for understanding the capabilities, operations, behavioral, and physical interactions, outcomes, and performance their respective product is required to contribute within the overall system, product, or service.

- Employ engineering practices that are inefficient, ineffective, and unscalable to moderate or large projects of multi-discipline engineering, et al teams.

Competing priorities abound and chaos results due to a number of factors such as a lack of leadership, SE education and training, resources, processes, tools, and methods. Ultimately, the technical program becomes paralyzed. No one can orchestrate decision-making convergence and stabilize the progression toward a durable, long-term solution.

### CAUSAL ANALYSIS FOR THE PANEL 2 ORGANIZATION PERFORMANCE

If you analyze the Systems Engineering process implementations of both of these organizations, you will find similarities. Both prepare and approve specifications, develop designs responsive to the specifications, develop architectures, perform trade studies, etc. In general, they do all of the "right things" under the belief of performing Systems Engineering to impress their customers. However, further investigation reveals that the application of the SE Process differs significantly between the two organizations. The Panel 2 organization employs the ad hoc Plug and Chug … Specify-Design-Build-Test-Fix Paradigm; the Panel 3 organization consists of "system thinkers" that perform and scale the SE process *iteratively* and *recursively* at all levels of system decomposition. Let's explore each of these paradigms.

#### The Plug and Chug … Specify-Design-Build-Test-Fix Paradigm

Engineers naturally gravitate to component-centric design solutions. Caldwell [42] in addressing engineering curriculum reform describes the traditional engineering course presentation order as bottom-up - *Components - Interactions - Systems*. He observes that engineering courses focus on the analysis of engineering components, not integrated systems. The technical strategy that ensues occurs naturally: Specify, Design, Build, Test, Fix (and Rework) the system or product design in a seemingly *endless loop* until it complies with customer requirements – i.e., verification - but may not necessarily fulfill their operational need(s) – i.e., validation.

Consider, for example, the Panel 2 organization specifying graphical user interface (GUI) specification requirements. In the context that engineer can write specification requirements, the resulting requirements might state "... shall have a look and feel that is realistically representative of the real-world." Or suppose the organization specifies a requirement that a system interface "... shall be intuitively obvious." Requirements of this type cannot be quantified easily by the Plug and Chug … Specify-Design-Build-Test-Fix Paradigm. Problems such as these are often described as amorphous, dynamic, qualitative, and may have conflicting stakeholder views, etc. ... not classical boundary condition Plug and Chug problems.

The preceding discussion motivates a key question: *How does the Plug & Chug … Specify-Design-Build-Test-Fix* migrate into workplace organizations? The answer lies in the transition from engineering school to industry.

#### The Industrial Plug & Chug … Specify-Design-Build-Test-Fix Paradigm

The *Plug & Chug … Specify-Design-Build-Test-Fix Paradigm* is a colloquial expression that characterizes informal "catch phrases "engineers acquire in engineering school and later become workplace lingo expressions. The paradigm represents a convolution of two separate paradigms: 1) a *Plug & Chug Paradigm* from classroom exercises and 2) a *Design-Build-Test-Fix Paradigm* laboratory exercises. Let's examine each of these further.

### The Plug & Chug Paradigm

The *Plug & Chug Paradigm* represents an instructional teaching model for engineering students. Solutions to the classical boundary condition engineering problems require students to consider inputs, initial states and dynamic boundary conditions, constraints, and assumptions to arrive at solution / results.

### The Educational Design-Build-Test-Fix Paradigm

The educational *Design-Build-Test-Fix Paradigm* has origins in scientific inquiry methods and is often acquired informally and experientially through laboratory exercises. The paradigm evolves from students having a requirement to design a widget, verify, and validate the design solution. If the test fails, they enter an iterative fix / rework / patch cycle that involves rework or tweaking the configuration in a seemingly endless loop until the test article and test configuration produces operationally valid data. Test results are then documented in a lab report and submitted for grading.

The Design-Build-Test-Fix Paradigm instruction tends to be *component-centric*. Erwin [43] observes that projects in engineering schools tend to focus on the building aspects of systems. Then, when the projects are submitted for grading, most of the assessment is based on completion of the artifact, with design having lesser importance. He notes that this approach is often rationalized on the basis of allowing the students to be "creative." As a result, the student receives little or no guidance or direction concerning the design process.

Given this characterization of the Plug & Chug … Design-Build-Test-Fix Paradigm, let's shift our focus to an actual Systems Engineering paradigm represented by Panel 3 (Figure 1).

## STRATEGIC APPLICATION OF SYSTEM ENGINEERING

During our discussion of the Panel 3 organization, we noted that oscillations related to human interactions are still present in the planned technical performance. However, the magnitudes of the oscillations and downward trend indicate convergence in decision-making and the maturation of the system design solution toward delivery. The question is: *What is different in the Panel 3's organization application of SE*?

Panel 3's organization analyzes the user's operational need, bounds the problem space, and partitions the problem space into one or more candidate solution spaces at all levels of abstraction. At each level, the SE Process [39] is *iteratively* and *recursively* applied to select and develop operational, logical, and physical domain architectural solutions. Requirements are allocated and flowed down to each architectural element and subsequent levels. At the lowest levels, a point is reached whereby the engineering school *Plug & Chug … Design-Build-Test-Fix Paradigm* can be more appropriately applied. With seasoned, knowledgeable, and experienced SE leadership that understands how the SE is applied and scaled, the project can more reliably and predictably deliver systems, products, and services on schedule and within cost constraints.

## Contrasting SE Knowledge and Application in Both Organizations

In summary, we find that both the Figure 1 Panel 2 and Panel 3 organizations deliver all of the publicized work product decision artifacts – e.g., technical plans, specifications, designs, analyses and trade studies, test cases, test results, verification and validation (V & V), etc. in compliance with capability standards. Yet, both have significant differences in their depth of understanding, implementation of SE, and delivery performance. Table 1 summarizes a comparison of common attributes between the two organizational approaches.

**Table 1: Summarization of Differences in Key Attributes of Two Types of SE Approaches**

| **Attribute** | **Panel 2 Organization** (Perception of SE) | **Panel 3 Organization** (Actual SE) |
|---|---|---|
| Complexity Management Approach | Component-centric containment | Progressive multi-level system decomposition |
| SE Process Implementation | Plug and Chug … Specify-Design-Build-Test-Fix Paradigm | Iterative and recursive multi-level application of the SE Process [39] |
| Problem Solving / Solution Development Approach | Quantum leap to single point design solutions | Selection based on Analysis of Alternatives (AoA) of viable candidates |

One question that remains unanswered is: *Why is the Panel 2 (Figure 1) organization implementation of SE self-perpetuating*?

## THE SELF-PERPETUATING SPECIFY-DESIGN-BUILD-TEST-FIX PARADIGM

When engineering graduates enter the workforce, they bring their in-grained Plug & Chug … Design-Build-Test-Fix Paradigm along with their textbooks and continue to apply the paradigm … unless their organization retools them with a paradigm shift to System Engineering. For some organizations, the paradigm shift is unlikely to occur. *Why*?

First, functional management, which is accountable for the "care and feeding" of their personnel such as education and training, may not have had or understand the need for formal SE education and training. They will confidently tell you they have been "performing Systems Engineering for years" when, in fact, it is the ad hoc Plug & Chug … Specify-Design-Build-Test-Fix Paradigm. On recognition of the paradigm, engineers, managers, and executives will sometimes defend the approach as "you practice a different 'brand' of SE."

SE methods identify system users and stakeholders; establish system boundaries; identify and analyze user operational needs, use cases, and scenarios; translate needs into requirements; derive, allocate, and flow down specification requirements; develop multi-level operational, behavioral, and physical architectures; define modes and states of operations; etc. At its core foundation, is there really more than one 'brand' of SE? Functional and program / project leadership mindsets such as this perpetuate the paradigm with each set of new hires.

Secondly, pressures to meet highly aggressive contract or task milestone schedules or budgets from executive and program / project management preclude serious transformation from the ad hoc paradigm to a true Systems Engineering implementation. Lacking insightful management vision to stage a series of pilot projects to achieve the paradigm shift transformation, the response is "we'll do it next time."

## THE INSTRUCTIONAL SYSTEM DEVELOPMENT (ISD) MODEL FOR SE

Another factor that influences organizational competency in applying SE concepts, principles, and practices relate to Instructional System Development (ISD). Education and training often focus on *what* must be accomplished without adequately helping students understand *why*, *how to*, and *when to*. These learning objectives items are critical for providing insightful knowledge for scaling SE methodologies and application to meet project cost, schedule, and technical performance constraints.

Consider the Panel 2 (Figure 1) organization discussed earlier. SE *document-centric* organizations focus experiential learning on WHAT must be accomplished. Education in the organization evolves slowly through informal learning osmosis coupled with a managerial leadership "Go Do" culture– i.e., Go Do a spec, Go Do a plan, Go Do a trade study, etc.

Observe that the knowledge transfer of the experiential approach limits foundational knowledge for informed decision making – i.e., the *how tos*, *whys*, *where tos*, and *when tos*. Implicitly over time, personnel in the organization learn a few of the basics of SE through informal exposure to subject matter experts (SMEs) in meetings, personal study, or short courses. Yet, they often lack professional understanding of what constitutes a valid and acceptable specification, plan, design solution, trade study, et al – i.e., how to tailor SE practices. As a result, they gain experiential workplace knowledge – e.g., the *whats* - without understanding the *how tos*, *whys*, *where tos*, and *when* tos. Providing formal education and training SE courses more efficiently condenses what engineering students or professionals "need to know" into a structured course setting in a short period of time with a higher probability of having a more complete treatment of the subject matter.

## FILLING THE SE EDUCATIONAL VOID

Engineers complete curriculum requirements for engineering program degrees accredited by the Accreditation Board for Engineering and Technology (ABET). Over time, work experiences, education, and training enable them to maintain their engineering graduation level of competence in some engineering topics, less in others, or increase their understanding and proficiency in other topics.

On entry into the workforce, graduates soon discover they are now domain-specific contributors within a multi-level, multi-disciplined, system development framework that requires "systems thinking" knowledge beyond domain-specific engineering knowledge. Samson and Lowery [44] note that even though engineering graduates have an engineering specialty, to be successful they must understand the role of the specialty in contributing to "the broader arena of societal systems problems." These missing skills require understanding of fundamental system analysis, design, and development concepts, principles, and practices – e.g., Systems Engineering – addressed by Wasson [2] as well as communications and interpersonal skills.

## SHIFTING THE UNDERGRADUATE ENGINEERING EDUCATION PARADIGM

Based on the preceding discussions, it should be apparent that there is a need to shift the undergraduate engineering education paradigm to include a Fundamentals of Systems Engineering course. Instituting this instruction would fill an educational void that exists in most organizations. You may ask why this is required.

If we investigate typical engineering career paths, most engineers spend five to ten years in some industries of their 40+-year nominal careers applying "hands-on" knowledge obtained as part of their undergraduate engineering degree program. This knowledge will continue to serve as a foundation for decision-making for life.

From their fifth year onward, engineers spend significantly more of their work hours each day in meetings, collaborating with colleagues, participating in system development activities and decisions, and continue to build on their discipline knowledge. Examples include: system / product specification development, requirements analysis, interface analysis and definition, trade studies, technical reviews, integration & test, baseline management, risk management, et al. These topics exemplify the system analysis, design, and development concepts, principles, and practices.

To further illustrate this point, consider the structure of a typical undergraduate engineering curriculum. Most engineers receive engineering degrees from accredited institutions of higher learning based on: 1) a *domain-centric* curriculum - e.g., electrical engineering, mechanical engineering, et al - and 2) a set of general engineering courses required of all disciplines - e.g., engineering statics and dynamics, thermodynamics, strength of materials, engineering economy, engineering ethics, et al.

In general, engineers spend four years obtaining an engineering degree that may have an industrial "hands-on" life span of five to ten years, which is approximately 25% of an average career. At that point, they become team leaders, systems engineers, managers, et al, which place more emphasis on SE leadership, acquisition, and, management skills. They continue to build on their foundational domain engineering discipline knowledge and experience throughout their careers. However, anecdotal evidence and estimates suggest they spend from 50% to 75% of their total career-hours, on average, collaborating with others concerning the engineering of systems ... for which they may have no formal education or training. McCumber and Sloan [45] citing Friedman [46] observe that most engineers are educated and trained to be "domain engineers." They add that it requires about five years of industrial maturity to evolve engineers into System Engineering."
This raises the question: *What instruction should be required to fill the SE educational void?*

## SYSTEMS ENGINEERING COMPETENCY AREAS

Undergraduate education should provide a working knowledge of SE concepts, principles, and practices across all engineering degree curricula. This includes classroom instruction, real-world exercises, case studies, industry lessons learned, etc. More in-depth instruction can be provided at the engineering graduate school level. Understand the difference in Systems Engineering as a professional career discipline versus a domain engineers such as electrical, mechanical, software, et al that apply SE methods, processes, and tools to solve domain specific problems. Both

contextual roles are crucial to meeting the interdisciplinary team needs of organizations to develop complex systems.

The overarching objective is for the engineering student to be capable of developing a fully integrated mindset that instinctively links the subject matter of the topic examples listed below into a coherent solution with consistency throughout its documentation. Please be aware that these activities are more than instructional checklists; they information content in each must be logically linked, concise, consistent, and traceable within the overall system design solution framework.

1. Knowledge of Systems Engineering Terminology, Acronyms, and Abbreviations
2. System Development Strategies and Planning
3. System Stakeholder / User Identification
4. Stakeholder Needs Assessment
5. User Organizational and System Mission Statements
6. Organizational and System Roles and Responsibilities
7. Use Cases and Scenarios Identification & Definition
8. Requirements Elicitation & Development
9. Specification Development
10. Systems Engineering Process
11. Requirements Allocation & Flow Down
12. Requirements Traceability and Management
13. Concept of Operations (ConOps) / Operational Concept Description (OCD)
14. Mission Event Timelines (METs)
15. Understanding the System Element Architecture
16. System Architecture Formulation and Development
17. System Complexity Decomposition
18. System Stimulus-Behavioral Responses
19. System Interface Definition and Control
20. System Phases, Modes, and States
21. Analysis of Alternatives (AoA)
22. System Design & Development
23. System Integration & Test
24. System Test Cases & Scenarios
25. Model-Based Systems Engineering (MBSE)
26. System Performance Modeling
27. System Optimization versus Suboptimization
28. Reliability, Availability, and Maintainability (RAM)
29. Specialty Engineering Integration – e.g., human factors, safety, reliability, maintenance, logistics, et al
30. System Safety
31. System / Subsystem Design Descriptions
32. System Verification and Validation (V&V)
33. Version Descriptions of the Verified System
34. Systems Engineering and Development Metrics
    a. Measures of Effectiveness (MOEs)

b. Measures of Suitability (MOS)
c. Measures of Performance (MOPs)
d. Key Performance Parameters (KPPs)
e. Technical Performance Measures (TPMs)

35. Engineering Standards, Frames of Reference, & Coordinate Systems
36. Configuration and Data Management (CM / DM)
37. Best Value Concepts
38. Design-to-Cost (DTC)
39. System Life-Cycle Cost Estimating – i.e., Total Ownership Costs (TOCs)
40. Earned Value Management (EVM)
    a. Event Based Schedule Development
    b. Integrated Master Plans (IMPs)
    c. Integrated Master Schedules (IMSs)
41. Technical Reviews and Audits
42. Risk Mitigation and Management
43. Fundamentals of Project Management

**Differentiating Courses in SE**

The solution for many education institutions and training organizations is to provide Systems Engineering courses. However, there is a difference in the types of SE courses offered by many educational and training organizations. The predominate type of SE course is System Acquisition and Management; for near-term competency, engineering students need an SE course in system development.

- System Acquisition and Management courses focus on Systems Engineering activities and provide instruction in its philosophy and theory – e.g. awareness of *what* should be accomplished, not *how*. These courses often focus on documents to be produced – i.e., document-centric solutions. Students or trainees emerge from these courses with high level concepts and a vocabulary of semantics concerning Systems Engineering theory and key practices but lack the skills required to competently perform Systems Engineering activities – i.e., analyze, organize, structure, assimilate, and integrate systems in an efficient and effective manner.

- Systems Engineering & Development courses equip engineers with the requisite knowledge and skills required to transform the "SE philosophy and theory" into real-world application – e.g., what is to be accomplished and how to actually perform the tasks. This includes: 1) an in-depth understanding of SE terminology, acronyms, and abbreviations and their contexts of usage; and 2) integration of end-to-end Systems Engineering and Development life cycle concepts and strategies; system analysis and decomposition techniques; system architecture development; requirements analysis, allocation, flow down, and management methods; configuration management methods, system integration & test methods; verification and validation methods; etc.

  Please note that understanding System Engineering and Development concepts, principles, and practices represents only the first level of Systems Engineering knowledge. The second level requires making informed decisions concerning how to

tailor best practices and lessons learned to "engineer the system" within project technical requirements, technology, cost, and schedule constraints and acceptable risk levels.

## THE NEED FOR INSTRUCTORS WITH INDUSTRY AND INSTRUCTIONAL EXPERIENCE

One of the challenges in orchestrating an educational paradigm shift is qualifying instructors with in-depth industrial experience and instructional skills to teach the discipline. You may ask, *why does Systems Engineering instruction require instructors with in-depth industrial experience*? Surely academic instructors can study and teach top-down Systems Engineering concepts, principles, and practices.

The National Academy of Engineering's "Educating the Engineer of 2020" report [47] observed "... The great majority of engineering faculty, for example, have no industry experience. Industry representatives point to this disconnect as the reason that engineering students are not adequately prepared, in their view, to enter today's workforce." Fortunately, through accreditation, engineers enter the workforce competent in their degreed domains, which is a reflection of their instructor comfort levels with applying mathematical and scientific concepts and principles. However, the problem solving / solution development methods required to enable engineers to analyze, translate, and transform an abstract problem space into multiple levels of solution spaces with manageable risk and triple constrain compliance is not part of those mathematical and scientific comfort zones.

Systems Engineering embodies more than a top-down approach, which is only one aspect. As a problem solving / solution development and decision making methodology, it is a *highly iterative,* multi-faceted approach – i.e., top-down, bottom-up, left-right, right-left, etc. Knowledge and application of that methodology is tempered through robust, hands-on industrial experience over a number of years across a diversity of projects. Custer, Daughterty, and Meyer [48] cite Guskey [49] emphasis on effective professional development requiring teachers to better understand: 1) the content they teach and 2) how students learn that content.

There is a perception in some organizations that if you teach students and professionals to write "shall" specification requirements statements to an outline, create an ad hoc PowerPoint architecture graphic, perform a trade study, estimate recurring and non-recurring engineering costs, and design a system, you are by their definition, an SE. Although SEs do perform these activities, they do not embody the knowledge of SE concepts, principles, and practices required to be a competent SE.

The Fundamentals of Systems Engineering course proposed is intended to equip the engineering student with the basics of *what* must be accomplished. Domain engineers in industry who have no interest in learning Systems Engineering methods will lament that basics are bureaucratic; they want to create designs rather than ensuring designs are responsive and traceable to customer needs. The challenge in industry is how you scale the SE activities and formalities to meet reasonable project triple constraints while preserving the spirit and intent of SE concepts, principles, and practices. That subject matter should be relegated to graduate level Systems Engineering classes composed of students who already understand the fundamentals and have industry experience.

Caldwell [50] observes that students seldom see SE methods presented as an integrated concept. He proposes engineering course presentation should follow a systems - components - interactions approach. He notes that the goal of this sequence is to provide students with an overall sense of SE as a problem-solving method. This is accomplished by providing a general structure – e.g., "scaffolding" – that enables students to see how components and interactions fit within a general SE context. Davidz and Nightingale [51] note that systems educational and training programs should enhance engineers' understanding of the "componential, relational, contextual, and dynamic elements of systems."

Erwin [52] citing Cummings and Sayer [53] promotes the need to transition the traditional "sage on the stage" education instructor role to the "guide on the side." Why? The "sage" role engenders a dependence on authority. In contrast, Systems Engineering often requires stand-alone critical thinking "outside the box" for which there may be no authority or precedence. Industrial practitioners with academic qualifications are well positioned to serve in this instructional role.

Finally, seasoned instructors with industrial experience are needed to simplify the jargon. Felder and Brent [54] observe that jargon often becomes a barrier to new learning material. This includes terms that are unfamiliar and related to concepts that can be easily learned but sound challenging.

### MANAGING ENGINEERING STUDENT EXPECTATIONS

One of the outcomes in shifting educational and industrial paradigms is managing student expectations. A Systems Engineering Fundamentals course can better prepare engineering students for transition and their roles in industry or government.

Engineering students often indicate they pursue engineering degrees with the intent of fulfilling an internal desire to innovate, create, and design systems and products. Organizations do employ multiple engineering discipline professionals to innovate and create engineering solutions– e.g., Systems Engineering. However, these solutions may or may not require new design of specific components; modification of existing legacy or commercial-off-the-shelf (COTS) may be the solution

When new engineering graduates enter the workforce, surprises occur. They discover that "new design" is often a last resort strategy, not a full-time activity. They develop / review specifications, design-to requirements, etc. Chief, project, or lead systems engineers are then confronted with determining how to accomplish work tasks within constrained budgets and schedules with highly capable engineers lacking Systems Engineering problem solving / solution development skills. Introducing undergraduate engineers to Systems Engineering methods provides a reality check of what to expect when they graduate and enter the industrial workforce.

A final question is: *What are the academic challenges to introducing an SE Fundamentals course at the undergraduate level?*

## ACADEMIC CHALLENGES TO SE FUNDAMENTALS IN ENGINEERING CURRICULA

Systems Engineering or a subset is typically offered only at the engineering graduate school level. The defense of this approach states that a Systems Engineering Fundamentals course requires requisite knowledge and experience that are only gained through work experience in the public or private sectors. However, seasoned system engineers and managers who work with today's engineering graduates recognize this is a myth. *Why*? New college graduates entering the workforce today quickly learn and assimilate the fundamentals of Systems Engineering concepts and practices without a graduate level course.

Today's generation of engineering students have tremendous learning abilities and capacities. For organizations that have a strong engineering heritage, new engineering graduates entering the workforce rapidly assimilate SE concepts, principles, and practices rapidly. These results, which are based on observed performance in the workplace, dispel the notion that undergraduate students require years of industrial experience as a prerequisite for introduction to fundamental SE concepts, principles, and practices.

Finally, SE course(s) in the US must comply with the Accreditation Board of Engineering and Technology (ABET) *Criteria for Accrediting Engineering Programs* [13]. At this time, criteria for Systems Engineering programs have not been established beyond the General Criteria. [55] ABET Criterion 3 Student Outcomes [56] include increasing emphasis on "economic, environmental, social, political, ethical, health and safety, manufacturability, and sustainability," et al skills to be able to deal in multi-disciplinary team environments. These are critical skills required for 21st century system development. However, if engineering education is void of the SE skills, which consist of the problem-solving / solution-development methodologies that provide the "engine" for system development, the social and interpersonal skills will be diverted to unnecessarily resolving multi-disciplinary, solution strategy conflicts within or between teams rather than focus on the primary mission of *efficiently* and *effectively* solving the user's problem and solution spaces.

### Assessment Metrics

Measures of Effectiveness (MOE) and supporting Measures of Performance (MOPs) metrics for assessing the Fundamentals of Systems Engineering course should address three key areas:

- Instructor SE qualifications such as knowledge, experience, and communications skills.
- Student assessments via verification of activity-based outcomes including homework, examinations, reviews, project results, et al.
- Industry and government validation via feedback from interviews or surveys with project managers and project engineers, functional managers, et al who understand SE concepts, principles, and practices.

## FUTURE SE DIRECTIONS

A longer-term strategy is to begin introducing SE concepts in the K - 12 curriculums. The reality is foundations for Systems Engineering concepts have analogs in K-12 instruction. Examples include concepts such as:

- Understanding user needs
- Problem definition
- Problem solving
- Solution exploration
- Collaborative, data-driven, decision-making such as brainstorming

From a national perspective, we must promote engineering awareness and opportunities in K – 12. Once students enter engineering programs, we must educate and train engineers to enter the workforce prepared to be immediately productive in developing complex systems, products, and services. Vest [57] observes "... Indisputably, engineers of today and tomorrow must conceive and direct projects of enormous complexity that require a new, highly integrative view of engineering systems." These initiatives are imperative to compete in a highly competitive global environment.

## SUMMARY

In summary, this paper identifies two SE deficiencies in many organizations and addresses the need for a course in Systems Engineering Fundamentals current undergraduate engineering curricula. Integration of SE concepts, principles, and practices into engineering programs will significantly upgrade the knowledge and skills of new engineering graduates and fill the void in the public and private workforce. This knowledge will greatly improve the competencies, effectiveness, and efficiency of engineers as well as program organizations within the enterprise.

Our discussion highlighted the ad hoc, chaotic, and dysfunctional nature of ineffective and inefficient decision making in organizations that lead to poor contract performance. These results are often traceable to the educational *Plug & Chug … Design-Build-Test-Fix Paradigm* that is embedded in an industrial *Plug & Chug … Specify-Design-Build-Test-Fix* Paradigm common in many organizations. Whereas organizational capabilities and personnel excel in domain engineering competencies, their ability to efficiently and effectively "the engineering of systems" is significantly weakened by these paradigms that are unscalable due for moderate to large, complex programs / projects.

The lack of this undergraduate SE instruction becomes self-evident early in most engineers' careers. On average, engineers spend four years obtaining an engineering degree with domain knowledge that has a "hands-on" domain application of five to ten years and then diminishes over time. Although engineers build on this foundation throughout their careers, anecdotal evidence suggests engineers spend on average up to 50% to 75% of their total career hours performing SE activities collaborating with others concerning the engineering of systems, not necessarily component domain engineering. Later in their careers, they acknowledge that a large percentage of those hours were inefficient and ineffective due to a lack of understanding in how to "engineer systems" via Systems Engineering methods.

Based on this discussion, two recommendations emerge to solve many of the performance problems and issues that plague system development projects today:

- **Engineering Education** - Institute a Fundamentals of Systems Engineering course as a degree requirement for undergraduate engineering curricula that addresses concepts, principles, and practices. Specifically, a course focused on the SE concepts, principles, and practices of system development introduced prior to capstone project courses [22] – [24].
- **Organizations** - Shift the organizational Specify-Design-Build-Test-Fix Paradigm to a scalable SE methodology-based education and continuous improvement paradigm that goes beyond SE philosophy and theory.

As a consequence, organizational competency will be enhanced, and project performance will more predictably correlate with organizational capability maturity results. Organizations will be able to perform more effectively and efficiently in meeting the needs of their customers while operating a highly competitive global environment.

Today, there is tremendous competition among colleges and universities offering engineering degree programs to entice prospective students to enroll, graduate, and achieve success and recognition for the college or university. Based on industry needs for well-rounded engineers who understand complex problem solving and solution development, what better way to achieve to recognition by industry and government by offering a Fundamentals of Systems Engineering course that prepares engineers to enter the workplace, be immediately productive, and leverage those skills to become tomorrow's thought leaders and innovators of complex systems.

## REFERENCES


1. LaRocca, M., "Career and Competency Pathing: The Competency Modeling Approach," Proceedings of Linkage Conference on Emotional Intelligence, Chicago, IL, 1999.
2. Wasson, Charles S., *System Analysis, Design, and Development: Concepts, Principles, and Practices*, 1st ed., John Wiley & Sons, Inc. (New York), 2006.
3. Wasson, Charles S., "Systems Thinking: Bridging the Educational Red Zone Between System Engineering and Program Management Education", 2nd Annual INCOSE Great Lakes Conference, Mackinac Island, MI, Sept, 7 – 9, 2008.
4. NDIA, "Task Group Report: Top Five Systems Engineering Issues within Department of Defense and Defense Industry," National Defense Industrial Association (NDIA), Systems Engineering Division, July 2006.
5. Castellano, Dave, "Program Support: Perspectives and Systemic Issues," Systems & Software Engineering – Office of the Deputy Secretary of Defense for Acquisition & Technology, NDIA Systems Engineering Conference, 24 October 2006, p. 17 - 25.

6. Bar-Yam, Yaneer, "When Systems Engineering Fails --- Toward Complex Systems Engineering," New England Complex Systems Institute, IEEE Press, Piscataway, NJ, International Conference on Systems, Man & Cybernetics. Vol. 2, pp. 2021-2028.
7. Bahill, A. Terry and Henderson, Steven J. "Requirements Development, Verification, and Validation Exhibited in Famous Failures," *Journal of Systems Engineering*," Vol. 8, No. 1, 2005.
8. JPL D-18709, "Report on the Loss of the Mars Polar Lander and Deep Space 2 Missions," Jet Propulsion Laboratory (JPL), California Institute of Technology, 22 March 2000.
9. NASA, "Mars Climate Orbiter Mishap Investigation Board Phase I Report," November 10, 1999.
10. Honour, Eric C., "Understanding the Value of Systems Engineering," Honourcode, Inc., Cantonment, FL, 2004.
11. Miller, Roger, Floriscel, Serghei, and Lessard, Donald R., *The Strategic Management of Large Engineering Projects: Shaping Institutions, Risks, and Governance*, The MIT Press, January, 2001.
12. Fabrycky, Wolter J., "Systems Engineering: Its Emerging Academic and Professional Attributes," American Society of Engineering Education (ASEE) 2010 Annual Conference, AC 2010-1958, 2010, p. 1.
13. ABET, "Criteria for Accrediting Engineering Programs," 2012 - 2013, Accreditation Board of Engineering and Technology (ABET), Engineering Accreditation Commission, Baltimore, MD, October 29, 2011.
14. National Academy of Engineering (NAE) Report, "Educating Engineers of 2020 and Beyond: Adapting Engineering Education to the New Century," National Academies Press, Washington, DC, 2005, p. 10.
15. Hsu, John C., Raghunathan, S., and Curran, R. "Effective Learning in Systems Engineering," American Institute of Aeronautics and Astronautics (AIAA), 46th AIAA Aerospace Sciences Meeting and Exhibit, Reno, NV, January 7 – 10, 2008, p. 1.
16. Borrego, Maura, and Bernhard, Jonte, "The Emergence of Engineering Education Research as an Internationally Connected Field of Inquiry," *Journal of Engineering Education (JEE),* Vol. 100, January 2011, pp. 14 – 47.
17. Patil, A, and Codner, G., "Accreditation of Engineering Education: Review, Observations and Proposal for Global Accreditation," *European Journal of Engineering Education*, Vol. 100, No. 1, January 2011, pp. 18-19.
18. Adams, Robin, Evangelou, Demetra, English, Lyn, De Figueitedo, Antonio Dias, Mousoulides, Nicholas, Pawley, Alice L., Schifellite, Carmen, Stevens, Reed, Svinicki, Marilla, Trenor, Julie Martin, and Wilson, Denise M., "Multiple Perspectives on Engaging Future Engineers," *Journal of Engineering Education (JEE),* Vol. 100, No. 1, January 2011, p. 49.
19. Duberstadt, J. J., "Engineering for a Changing World: A Roadmap to the Future of Engineering Practice, Research, and Education," Ann Arbor, MI: Millenium Project, University of Michigan.

20. Sheppard, Sheri D., Macatangay, Kelly, Colby, Anne, & Sullivan, William M., "Educating Engineers: Designing for the Future of the Field." San Francisco: Jossey-Bass, 2008.
21. Davidz, Heidi L. and Nightingale, Deborah J. "Enabling Systems Thinking to Accelerate the Development of Senior Systems Engineers," International Council on Systems Engineering (INCOSE) *Journal of Systems Engineering*, Vol. II, No. 1, 2008, p. 11.
22. Schmidt, P., Zalewski, J., Murphy, G., Morris, T., Carmen, C., van Susante, P., "Case Studies in Application of System Engineering Practices to Capstone Projects," American Society for Engineering Education(ASEE) Conference and Exposition. June 26-29, 2011. Vancouver, Canada.
23. Nemes, J., Hochstedt, K., Brannon, M., Kisenwether, E., and Bilen, S., "SE Capstone – Introduction of Systems Engineering into an Undergraduate Multidisciplinary Capstone Course," American Society for Engineering Education (ASEE) *2011 Annual Conference Proceedings*, June 26 – 29, 2011, Vancouver, Canada.
24. Corns, Steven, and Dagli, Cihan H., "SE Capstone: Integrating Systems Engineering Fundamentals to Engineering Capstone Projects: Experiential and Active," American Society for Engineering Education (ASEE) *2011 Annual Conference Proceedings*, June 26 – 29, 2011, Vancouver, Canada.
25. International Council on Systems Engineering (INCOSE) "System Engineering Professional (SEP) Program Overview," February, 2012, pp. 28 – 32.
26. Davidz and Nightingale, p. 4.
27. Sheard, Sarah A., "The Frameworks Quagmire," Software Productivity Consortium, Computer, July 2011, pp. 96 – 98.
28. ISO / IEC 12207:2008, "Systems and software engineering -- Software life cycle processes," International Organization of Standardization (ISO) / the International Electrotechnical Commission (IEC), 2008.
29. ISO / IEC 15288:2008 – "Systems and Software Engineering -System Life Cycle Processes." International Organization of Standardization (ISO) / the International Electrotechnical Commission (IEC), 2008.
30. ISO / IEC 19760:2003 "Guide for ISO/IEC 15288 - System Life Cycle Processes," International Organization of Standardization (ISO) / the International Electrotechnical Commission (IEC), 2003.
31. Software Engineering Institute (SEI), "Capability Maturity Model Overview," Carnegie-Mellon University, http://www.sei.cmu.edu/cmmi/ accessed March 17, 2011.
32. DAU – "Systems Planning Development Research and Engineering - Systems Engineering / Programs Systems Engineer (SPRDE-SE/PSE) Model," Levels I – III, Defense Acquisition University (DAU), Ft. Belvoir, VA.
33. NASA, "Project Management and Systems Engineering Competencies," Academy of Program / Project Engineering Leadership (APPEL),
34. Mitre, "Mitre Systems Engineering Competency Model," The Mitre Institute, Mitre Corporation, Bedford, MA, Sept. 1, 2007.

35. INCOSE, “Systems Engineering Professional (SEP) – Multi-Level Base Credentials,” International Council on Systems Engineering (INCOSE).
36. INCOSE, “Systems Engineering Competency Framework,” International Council on Systems Engineering (INCOSE) United Kingdom (UK).
37. Squires, A., Wade, J., Dominick, P. and Gelosh, D., "Building a competency taxonomy to guide experience acceleration of lead program systems engineers″, proceedings of the 9th conference on systems engineering research (CSER 2011), Los Angeles, CA, 2011.
38. Honour, p. 15.
39. Wasson, Chapter 26, pp. 275 – 289.
40. Wasson,Charles, S., “Formulation and Development of the Wasson Systems Engineering Process Model,” American Society of Engineering Education (ASEE) Southeastern Section Conference. Mississippi State University, Starkville, MS, April 1 – 3, 2012.
41. Wasson, Charles S., *System Analysis, Design, and Development: Concepts, Principles, and Practices*, 1st ed., John Wiley & Sons, Inc. (New York), 2006, pp. 244 – 245, 430 – 479.
42. Caldwell, Barrett S. “Teaching Systems Engineering by Examining Educational Systems,” Proceedings of the Spring 2007 American Society for Engineering Education (ASEE) Illinois-Indiana Section Conference, p. 91-92.
43. Erwin, Ben. *K-12 Education and Systems Engineering: A New Perspective*, Proceedings of the American Society of Engineering Education (ASEE) National Conference, Session 1280, Seattle, WA, July 1998, p. 6.
44. Samson, Charles H, and Lowery, Lee L., “The Need for Systems Engineering Education,” Proceedings of the Third Annual International Symposium, National Council on Systems Engineering (NCOSE), Arlington, VA, July 26 – 28, 1993, p. 1.
45. McCumber, William H. and Sloan, Crystal, “Educating Systems Engineers: Encouraging Divergent Thinking,” Twelfth Annual Symposium of the International Council on Systems Engineering, August 2002, p. 1.
46. Friedman, George, J. “Managing Complexity: An Application of Constraint Theory,” Proceedings of the Fourth Annual International Symposium of the National Council on Systems Engineering, San Francisco, 1994.
47. National Academy of Engineering (NAE), “Educating the Engineer of 2020: Adapting Engineering Education to the New Century,” National Academies Press, Washington, DC, 2005, p. 21.
48. Custer, Rodney L., Daughterty, Jenny L., Meyer, Joseph P., “Formulating the Conceptual Base for Secondary Level Engineering Education: A Review and Synthesis,” *Research in Engineering and Technology Education*, National Center for Engineering and Technology Education, p. 3.
49. Guskey, T. “What Makes Professional Development Effective?,” *Phi Delta Kappan* 84, pp. 748-750.
50. Caldwell, p. 91-92.
51. Davidz and Nightingale, p. 11.

52. Erwin, p. 4.

53. Cummings, Jim and Sayer, Dennis. *Brave New Schools, Changing Cultural Illiteracy Through Global Learning Networks*, St. Martin's Press, April 1997.

54. Felder, Richard M. and Brent, Rebecca, "The ABC's of Engineering Education: ABET, Bloom's Taxonomy, Cooperative Learning, and So On," Proceedings of the 2004 American Society for Engineering Education (ASEE) Annual Conference & Exposition, p. 1.

55. ABET, p. 20.

56. ABET, p. 3.

57. Vest, Charles, "Educating Engineers of 2020 and Beyond: Adapting Engineering Education to the New Century," National Academies Press, Washington, DC, 2005, Appendix B, p. 165.