System Engineering Competency
The Missing Element in Engineering Education

Charles S. Wasson
Wasson Strategics, LLC
www.wassonstrategics.com
Email: wslse@cpws.net

Abstract. *The "engineering of systems" performed in many organizations is often characterized as chaotic, ineffective, and inefficient. Objective evidence of these characteristics is reflected in program performance metrics such as non-compliance to requirements, overrun budgets, and late schedule deliveries. Causal analysis reveals a number of factors contribute to this condition: a lack of technical leadership, a lack of understanding the user's problem / solution spaces, point design architectures and solutions, a lack of integrated decision making, et al. Further analysis indicates these factors are symptomatic of a much larger competency issue traceable to undergraduate engineering education - the lack of a Systems Engineering fundamentals course taught by seasoned instructors with robust, industrial experience acquired from a diversity of small to large, complex systems.*

This paper explores the ad hoc, chaotic, and dysfunctional nature of technical planning and execution. We trace its origins to the industrial Plug and Chug … Specify-Design-Build-Test-Fix Paradigm and its predecessor 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.

The solution is to bolster the competency of the engineering workforce at two stages: 1) upgrade undergraduate engineering education to include a System Engineering fundamentals course and 2) shift the industrial System Engineering paradigm through education and training to employ scalable SE problem solving / solution development methodologies for projects ranging in size from small to large complex systems.

Introduction

The capabilities of organizations to *effectively* and *efficiently* develop systems, products, or services often reach a level of diminishing returns based on a number of driving factors such as: leadership, personnel education and training, technology, processes, best practices, corporate bureaucracies, resources, et al. These factors share one common element … *humans, the "engine" for action, performance, and accomplishment*.

For technical organizations, engineering competency provides the "knowledgebase" for innovation, development, production, and so forth. The totality of collective organizational knowledge, understanding, and ability to "orchestrate and engineer systems" *efficiently* and *effectively* determines its overall core competence and capability to perform.

Analysis and assessment of an organization's skills typically reveals competent knowledge and understanding in engineering and math concepts. In contrast, analysis of system development performance reveals that programs often exhibit budget overruns, late schedule deliveries, and technical requirements compliance issues.

System development, especially for moderate to large, complex systems, adds a new dimension of complexity beyond *discipline-centric* engineering concepts. It requires education and training in the "engineering of systems" or 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*. Objective evidence of this condition and its roots is summarized in several documents. Examples include:

- Castellano [1], *Program Support: Perspectives and Systemic Issues.*
- Task Group Report [2], *Top Five System Engineering Issues within Department of Defense and Defense Industry*.
- Bar-Yam, Yaneer [3], *When Systems Engineering Fails --- Toward Complex Systems Engineering.*
- Bahill, A. Terry and Henderson, Steven J. [4], *Requirements Development, Verification, and Validation Exhibited in Famous Failures.*

This paper explores a condition occurring in many 21st Century organizations. *Why do organizations deemed compliant with capability maturity standards have system development performance issues*? The initial answer resides in the lack of System Engineering education as part of undergraduate engineering curricula. Hsu, Raghunathan, and Curran [5] observe that universities are challenged to meet the demands of industry by supplying graduates with a sound foundation in Systems Engineering.

"Checking the box" of having a System Engineering course does not guarantee a competency in System Engineering. *Why*? System Engineering courses generally occur in two forms:

- Acquisition System Engineering courses that teach philosophy and theory – e.g. awareness of WHAT should be accomplished, not HOW. Students emerge with a vocabulary of semantics but lack the skills to apply what they have learned.
- Development System Engineering courses that equip engineers with the requisite knowledge and skills 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.

Both types of instruction may cover the same topics. However, the differences reside in two areas:

- The seasoned knowledge, skills, and experience of the instructor in SE practices.
- The knowledge, efficiency, and effectiveness of the students to apply and scale what they have learned to real-world problems.

The degree of success of these two points, coupled with SE leadership abilities, may provide insights as to why some organizations achieve and publicize assessment standard ratings and SE training. Yet exhibit project performance that fails to correlate with the rating.

Definitions of Key Terms

- **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* in a value into an equation and *Chug* out an answer for solving classical boundary condition problems.
- **Design-Build-Test-Fix Paradigm** – An *iterative* process that lacks an insightful methodology in which engineers DESIGN an entity, BUILD it in the lab, TEST it, and FIX (rework or patch) the design or its physical implementation in a seemingly endless loop until convergence at a final solution is achieved.
- **Paradigm Shift** - A transformational change in perspective from one paradigm to another – e.g., Plug & Chug … Design-Build-Test-Fix Paradigm to a System Engineering Paradigm.
- **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." Wasson [6]

Organizational System Development Performance

Organizations vary significantly in terms of their System Engineering capabilities. The spectrum of organizations ranges from those who are considered "best of class" to others who naively believe or live in a state of denial that their "business as usual" approach is true System Engineering. In general, both cases are exemplified by their technical, cost, and schedule performance.

To illustrate the preceding point, 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 smooth, seamless, and deliver on-time performance within cost constraints and contract requirements.

On contract award, the program organization embarks on the proposed system development process as illustrated in Panel 2 (Figure 1). However, the effort has a delayed start due to indecision indicated by the oscillations. For example, the development team may be staffed by leadership positions that may be undefined or unfilled, key personnel may not be available when planned, new personnel may not agree with the proposed solution, 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 metal, assembling hardware, or coding software when planned. When this condition occurs, any objective evidence of an SE decision-making path is summarily rejected as "philosophical theory and bureaucratic paperwork … we don't have time for this."

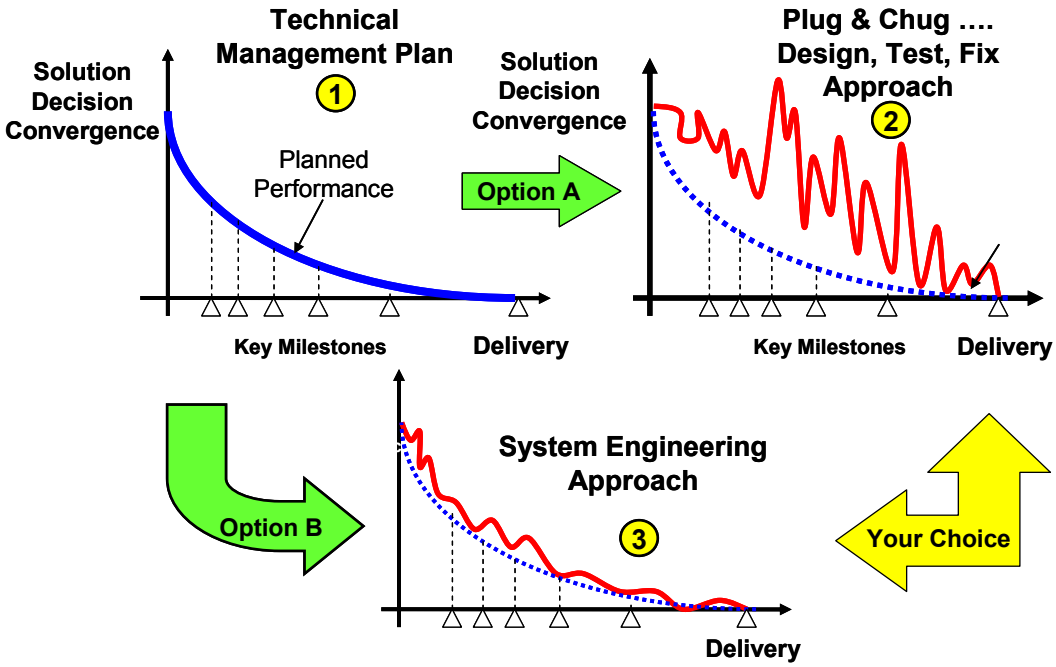


Figure 1: Contrasting SE-Based System Development versus the Plug & Chug …Design-Build-Test-Fix Paradigms.

The program reverts to their traditional "business as usual" Plug & Chug … Design-Build-Test-Fix Paradigm that leaps to a point design solution decision, which lacks supporting objective evidence such as peer reviews, analyses, trade studies, etc. Oscillatory decision making occurs … one day they have selected a solution; the next day they are grasping for new alternative solutions and so forth. As a result, we see major decision-making oscillations in Panel 2 (Figure 1). The system or product is quickly produced by 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 performing System Integration and Test Phase, the system developer is redesigning a major portion of the system or its interfaces … due to a lack of understanding of the user's problem space, teams failed to approve interfaces or properly characterize the operating environment, et al. What began as a smooth, seamless Technical Plan in Panel 1 evolves into major decision perturbations illustrated in Panel 2. The organization works nights, weekends, or holidays, attempting to rework or patch the evolving design solution until the organization or customer becomes exhausted and demands delivery.

In contrast, Panel 3 illustrates how SE can *minimize* these problems. Observe the decision-making oscillations in Panel 3; even SE is dependent on consensus decision making. Despite the degrees of uncertainty, the amplitude of the oscillations are relatively minor compared to Panel 2, and the downward trend to completion approximates the original plan.

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

Panel 3 exemplifies an organization that employs an SE approach and has qualified, trained SEs as leaders who understand how to transform SE philosophy theory into practice. Using the approach and SE Process Model introduced by Wasson [7], the integrated team of stakeholders applies its SE problem solving / solution development methodology.

In contrast, the organization in Panel 2 begins with an SE process but eventually defaults to the Plug & Chug … Design-Build-Test-Fix Paradigm, which focuses on individual accomplishments rather than team-based decision making. In general, the program is staffed by well-intentioned engineers, each desiring to:

- Take a quantum leap to their preferred physical domain solution – e.g. a point solution without due process. They fail to satisfy the necessary and sufficiency criteria for understanding the capabilities and behavioral interactions their respective product is required to contribute within the overall system architectural framework.
- Employ engineering practices that are not scalable to large group efforts.

Competing priorities abound and chaos results due to a lack of leadership, education, and training. Ultimately, the technical program becomes paralyzed. No one can orchestrate decision-making convergence and stabilize the progression toward a durable solution.

Causal Analysis for the Panel 2 Organization Performance

If you analyze the SE process implementation 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 guise of System 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 a Plug and Chug … Specify-Design-Build-Test-Fix Paradigm; the Panel 3 organization performs and scales 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 [8] in addressing engineering curriculum reform describes the traditional engineering course presentation order as bottom-up - COMPONENTS - INTERACTIONS - SYSTEMS. He notes that engineering courses focus on the analysis of engineering components, not integrated systems. The technical strategy that ensues occurs naturally: SPECIFY a system or product, DESIGN it, TEST it, and FIX it in an endless loop until it performs in accordance with customer requirements. We can construct a representative model of this process as shown in Figure 2.

The fallacy of the model resides in a failure to fully understand:

- The problem or issue space the user is trying to resolve.
- How the user desires to operate and integrate the system or product into their larger environment.
- Required outcomes and level of performance.

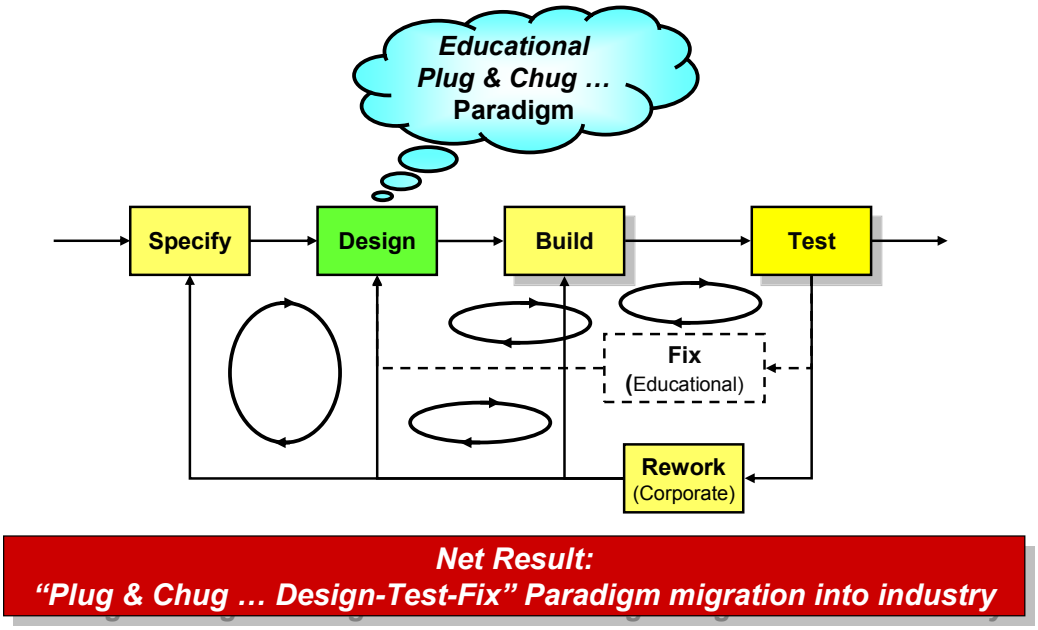


Figure 2: Typical Industrial Plug & Chug …Specify-Design-Build-Test-Fix Paradigm

Consider, for example, specifying or developing design solutions for a graphical user interface (GUI) with the "look and feel" that is "realistically representative of the real-world." Requirements of this type cannot be quantified easily by the traditional Plug and Chug … Specify-Design-Build-Test-Fix Paradigm. Problems such as these are often amorphous, dynamic, have conflicting stakeholder views, etc.; not classical boundary condition problems. These problems, which tend to be complex and qualitative rather than quantitative, must be translated into boundary conditions problems that have manageable risk and can be solved by the Plug and Chug … Specify-Design-Build-Test-Fix Paradigm.

This discussion initiates a key question: *How does the Plug & Chug … Specify-Design-Build-Test-Fix* infiltrate *these organizations*? The answer lies in a self-perpetuating migration from engineering school to industry.

The Origin of the Plug & Chug … Design-Build-Test-Fix Paradigm

The *Plug & Chug … Design-Build-Test-Fix Paradigm* is a colloquial expression that characterizes informal methods engineers acquire while in undergraduate engineering school. 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* from laboratory exercises. Let's characterize each of these.

The Plug & Chug Paradigm

The *Plug & Chug Paradigm* illustrated in Figure 3 represents a classroom teaching model for engineering students. Solutions to the classical boundary condition engineering problems require students to consider Inputs, Initial States & Dynamic Boundary Conditions, Constraints, and Assumptions to arrive at Solution / Results. The Solution should be Verified and Validated.

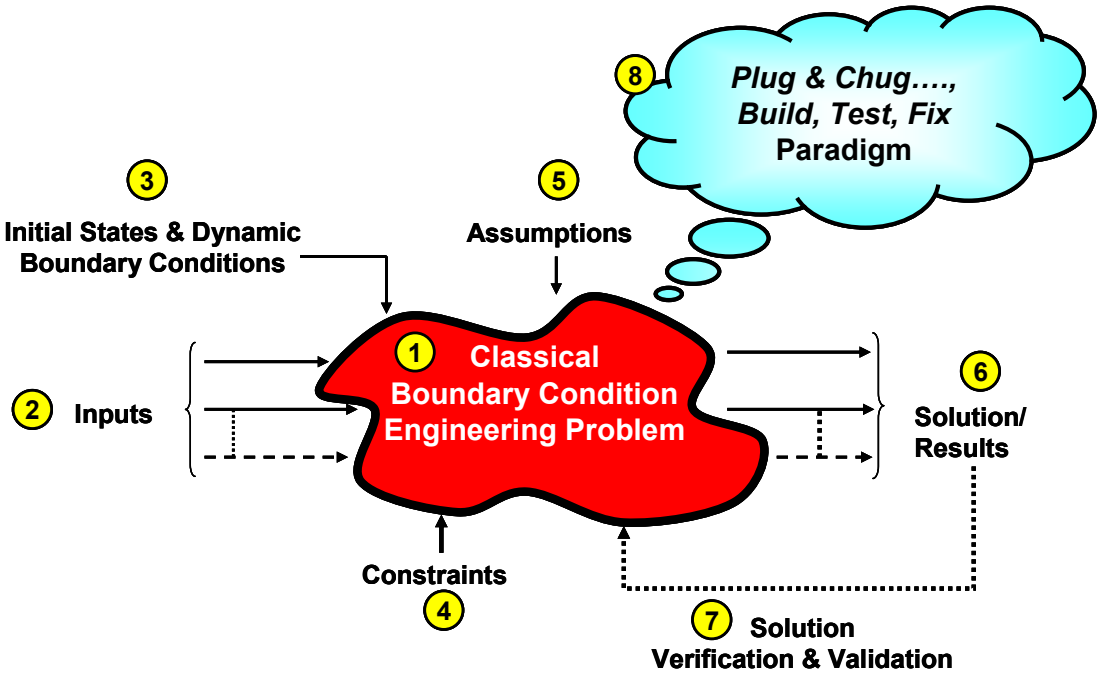


Figure 3: Educational Plug and Chug …Design-Build-Test-Fix Paradigm

The Educational Design-Build-Test-Fix Paradigm

The *Plug & Chug … Design-Build-Test-Fix Paradigm* is acquired informally and experientially through laboratory exercises. As illustrated in Figure 3, the paradigm evolves from students having a requirement to DESIGN a test article or experiment and perform TESTS. IF the test fails, they enter an iterative FIX 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 Plug & Chug … Design-Build-Test-Fix Paradigm tends to be *component-centric*. Erwin [9] 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 System 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*? Figure 4 provides insights.

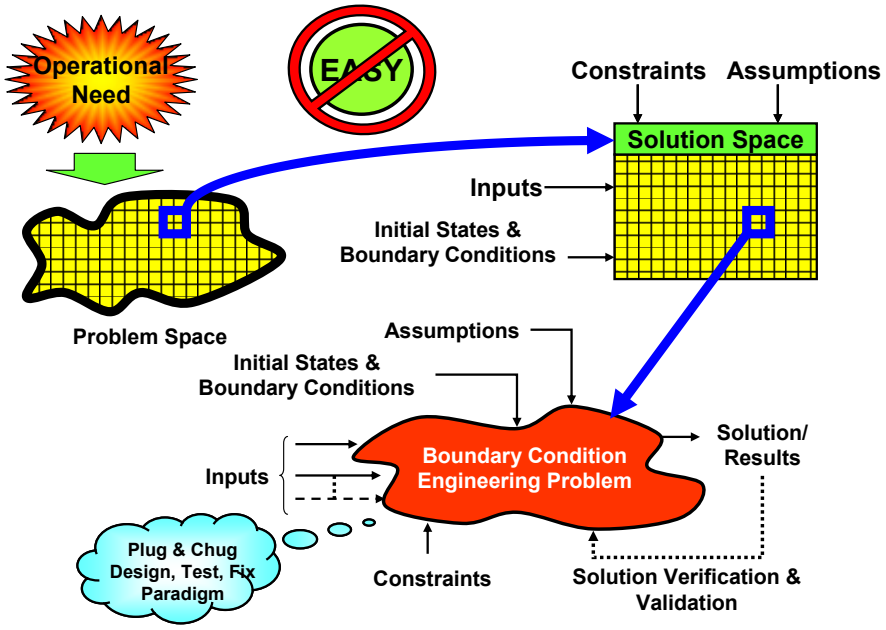


Figure 4: SE decomposition of complex operational needs into manageable boundary condition engineering problems

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 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 products – e.g., technical plans, specifications, designs, analyses and trade studies, test cases, test results, 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-Build-Test-Fix	Iterative and recursive multi-level application
Problem Solving / Solution Development Approach	Quantum leap to point design solutions	Selection based on Analysis of Alternatives (AoA) of viable candidates

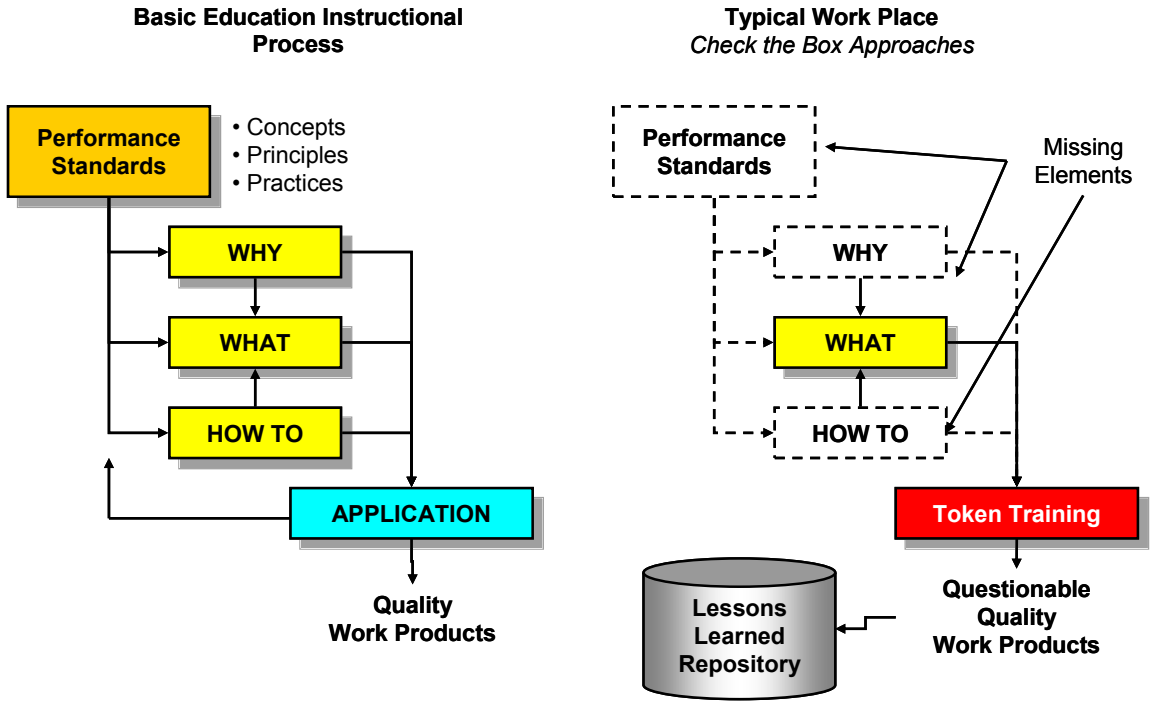


Figure 5: Contrasting ISD versus Experiential Workplace Learning Models

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

The Self-Perpetuating SPECIFY-BUILD-TEST-FIX Paradigm

When engineering graduates enter the industrial 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 engineering education and training, may not have had formal SE education and training and perpetuates the *Plug & Chug … Design-Build-Test-Fix* Paradigm based on their own engineering school and industry experiences or lack thereof.

Secondly, executive and program management, which are accountable for contract performance, apply immediate pressures to perform, precluding any serious transformation.

The Instructional System Development (ISD) Model for SE

Another factor that influences organizational competency in applying SE concepts, principles, and practices is an Instructional System Development (ISD) learning model illustrated in Figure 5. Education focuses too often 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 a form of 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 the missing requisite knowledge of the HOW TOs, WHYs, and WHEN TOs as indicated by the dashed boxes. 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 an acceptable specification, plan, trade study, et al; how to tailor approaches or procedures; etc. As a result, they gain experiential workplace knowledge – e.g., the WHATs - without understanding the HOW TOs, WHYs, and WHENs of the discipline required by instructional development models as illustrated by the left side of Figure 5.

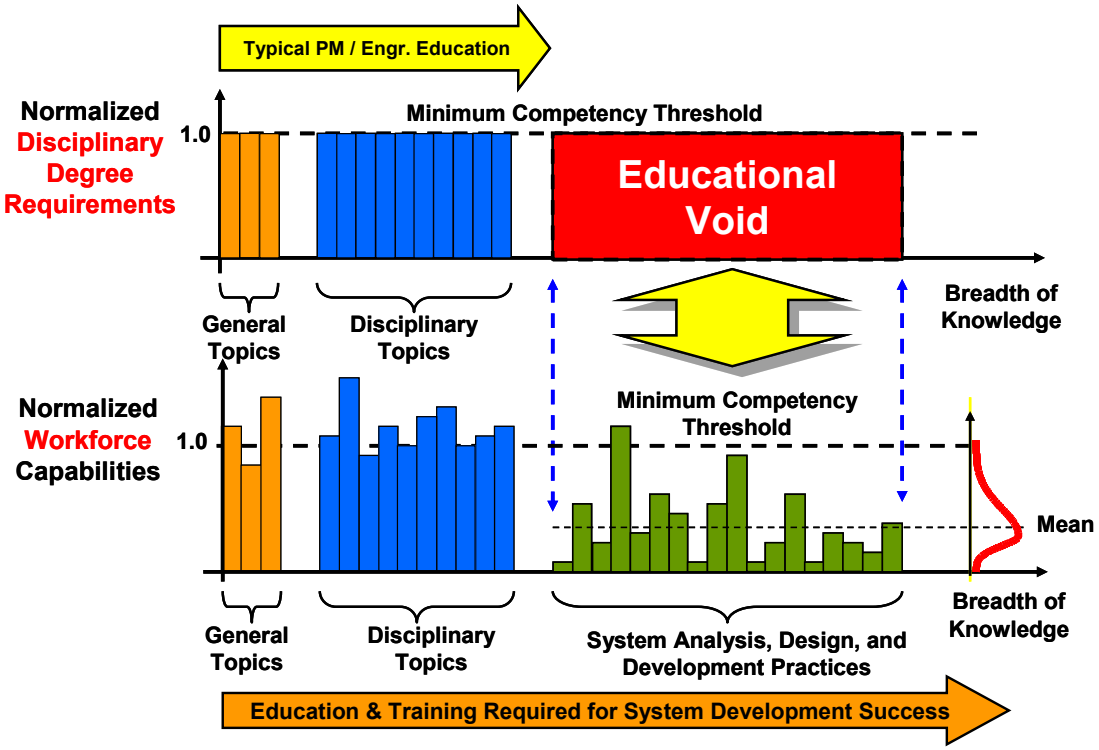


Figure 6: Understanding the System Engineering Educational Void in Engineering Curricula

Filling the SE Educational Void

Using Figure 6 as a guide, undergraduate engineers complete curriculum requirements for engineering degrees established by accreditation bodies such as 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 others as illustrated by the lower left portion of the figure.

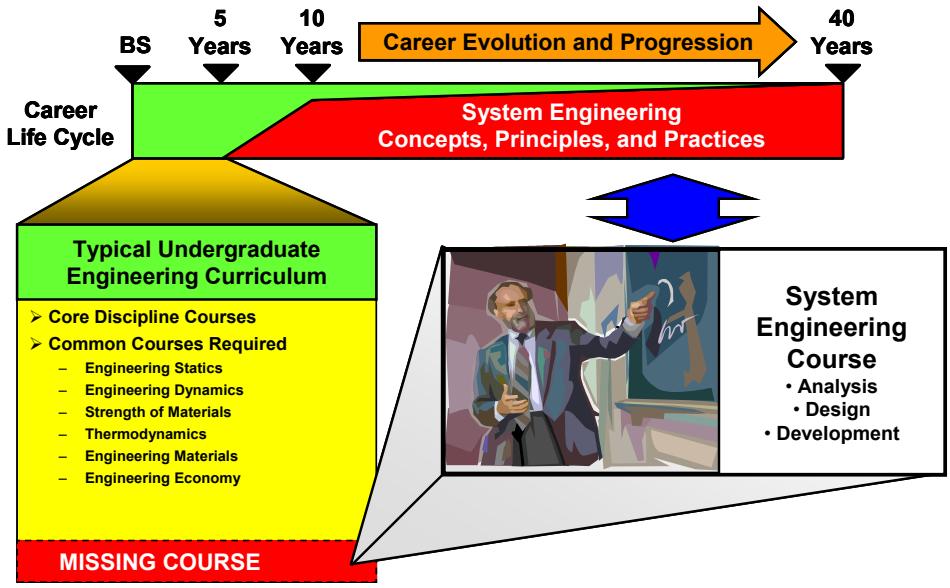


Figure 7: System Engineering – The Missing Element in Engineering Education

On entering the workforce, graduates soon discover they are now domain-specific contributors within a multi-level, multi-disciplined, system development framework that requires knowledge beyond domain-specific engineering knowledge. Samson and Lowery [10] note that even though engineering graduates have an engineering specialty, to be successful they must understand the role of the specialty in contributing to "societal systems problems." These missing skills, as illustrated in the lower right portion of the Figure 6, require understanding of fundamental System Analysis, Design, and Development concepts, principles, and practices – e.g., System Engineering – addressed by Wasson [7] 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 System Engineering Fundamentals course. Instituting this instruction would fill an educational void that exists in industry. You may ask why this is required. Consider the illustration shown in Figure 7.

If we analyze typical engineering career paths, most engineers spend only five to ten years of their 40+-year nominal careers directly applying the knowledge obtained as part of their undergraduate engineering degree program. McCumber and Sloan [11] citing (Friedman [12]) 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."

From their fifth year onward, engineers spend significantly more of their work hours each day in meetings, collaborating with colleagues, and participating in system development activities and decisions. 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 deficiency depicted in the lower right corner of Figure 6.

To further illustrate this point, consider the structure of a typical undergraduate engineering curriculum shown in the lower left portion of Figure 7. 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, et al.

Analysis of Figure 7 clearly indicates that engineers spend four years obtaining an engineering degree that has an application life span of five to ten years, which is approximately 25% of an average career. They continue to build on this aspect of this knowledge and experience throughout their careers. However, they spend the remaining 75% of their remaining career-hours, on average, performing activities – i.e., System Engineering - for which they had no formal education.

This raises the question: What instruction should be required to fill the SE educational void noted in Figure 6?

System Engineering Competency Areas

Undergraduate engineering schools should provide a working knowledge of SE concepts, principles, and practices across all degree programs. 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 System Engineering as a discipline versus being a domain engineer who applies System Engineering methods, processes, and tools to solve domain specific problems. Both roles are integral to this paper.

The following is a general list of topical examples for undergraduate SE instruction:

- Technical Strategies and Planning
- Stakeholder / User Identification
- Stakeholder Needs Assessment
- Use Case Identification & Definition
- System Engineering Process (Wasson)
- Specification Development
- Requirements Elicitation & Development
- Requirements Allocation & Flow Down
- Requirements Traceability and Mgt.
- System Architecture Development
- System Complexity Decomposition
- System Stimulus-Behavioral Responses
- System Interface Definition and Control
- System Phases, Modes, and States
- Analysis of Alternatives (AoA)
- System Design & Development
- System Integration & Test
- System Test Cases & Scenarios
- Model-Based System Engineering (MBSE)
- System Performance Modeling
- System Optimization
- Reliability, Availability, and Maintainability
- Specialty Engineering Integration
- System Safety
- System Verification and Validation (V&V)
- Technical Performance Measures (TPMs)
- System Engineering and Development Metrics
- Engineering Standards, Frames of Reference, & Coordinate Systems
- Configuration and Data Management (CM / DM)
- Best Value Concepts
- Design-to-Cost (DTC)
- System Life-Cycle Cost Estimating
- Total Ownership Costs (TOCs)
- Earned Value Management (EVM)
- Event Based Schedule Development
- Integrated Master Plans (IMPs)
- Integrated Master Schedules (IMSs)
- Technical Reviews and Audits
- Fundamentals of Project Management

The author suggests consideration of *System Analysis, Design, and Development* textbook [7] as a frame of reference for scoping the content of the technical topics listed above.

Shifting the Engineering Education Paradigm

One of the challenges in orchestrating an educational paradigm shift is establishing qualified instructors with robust industrial experience. You may ask, *why does System Engineering instruction require instructors with robust industrial experience*? Surely academic instructors can study and teach top-down System Engineering concepts, principles, and practices. Note the operative term "top-down."

SE 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 [13] cite Guskey [14] emphasis on effective professional development requiring teachers to better understand: 1) the content they teach and 2) how students learn that content.

Caldwell [15] 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 [16] note systems educational and training programs should enhance engineers' understanding of the "componential, relational, contextual, and dynamic elements of systems."

Erwin [17] citing (Cummings and Sayer, [18]) 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, System 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 [19] 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 Undergraduate Expectations

One of the outcomes in shifting educational and industrial paradigms is managing student expectations. A Systems Engineering Fundamentals course can provide insightful career knowledge.

Engineering students often indicate they pursue engineering degrees with the intent of satisfying an internal desire to innovate, create, and design systems and products. In sharp contrast, organizations employ multiple engineering disciplines to innovate engineering solutions at various levels of abstraction – e.g., Systems Engineering – which may or may not require custom design of specific components.

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 must 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 System Engineering problem solving / solution development skills. Introducing undergraduate engineers to System 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

System engineering is typically offered only at the engineering graduate school level. Academia defend this approach by stating that a System Engineering Fundamentals course requires requisite knowledge and experience that are only gained through work in the public or private sectors. Seasoned system engineers and managers recognize this is an academic myth. *Why*? New college graduates who enter the workforce of organizations with robust system engineering competencies quickly learn the fundamentals of System Engineering 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, approval of an SE course must meet curriculum requirements established by the ABET and other accreditation organizations.

Future SE Directions

A longer term strategy is to begin introducing SE concepts in the K – 12 curriculums. The reality is foundations for System Engineering concepts have analogs in K-12 instruction. Examples include concepts such as: identifying (teacher) operational needs, bounding user (teacher) requirements, teacher requirements traceability and compliance to teacher direction, innovating and creating designs in response to teacher requirements, thought processes for standard outlines, reviewing homework, testing knowledge, interpersonal skills, communicating, verification and validation, et al.

SUMMARY

In summary, this paper identifies a deficiency in current undergraduate engineering curricula and addresses how a course in System Engineering Fundamentals 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 traceable to the *Plug & Chug … Design-Build-Test-Fix Paradigm* that is embedded in a *Specify-Design-Build-Test-Fix* model common in many organizations. Whereas organizational capabilities excel in "engineering," their ability to efficiently and effectively "engineer systems" is significantly weakened by these paradigms that are not scalable due to a lack of a System Engineering Fundamentals course in undergraduate engineering curriculums.

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 shelf-life of five to ten years and then diminishes over time. For the remainder of their careers, anecdotal evidence suggests engineers spend up to 75% of their remaining career hours in meetings or collaborating with others on decision making concerning the engineering of systems, not necessarily components. Later in their careers, they acknowledge that a large percentage of those hours were inefficiently and ineffectively applied 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:

- Institute a System Engineering Fundamentals course as a curriculum degree requirement for undergraduate engineering instruction.
- 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.

REFERENCES

[1] 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.

[2] NDIA (2006) Task Group Report: *Top Five System Engineering Issues within Department of Defense and Defense Industry*, National Defense Industrial Association, System Engineering Division, July 2006.

[3] Bar-Yam, Yaneer. *When Systems Engineering Fails --- Toward Complex Systems Engineering*, New England Complex Systems Institute.

[4] Bahill, A. Terry and Henderson, Steven J. *Requirements Development, Verification, and Validation Exhibited in Famous Failures*, Systems Engineering, Vol. 8, No. 1, 2005.

[5] 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.

[6] Wasson, Charles S. *System Analysis, Design, and Development*, John Wiley & Sons, Inc. (New York), 2006, p. 24.

[7] Wasson, p. 275.

[8] 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.

[9] Erwin, Ben. *K-12 Education and Systems Engineering: A New Perspective*, Proceedings of the American Society of Engineering Education National Conference, Session 1280, Seattle, WA, July 1998, p. 6.

[10] 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, Arlington, VA, July 26 – 28, 1993, p. 1.

[11] 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.

[12] 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.

[13] Custer, Rodney L., Daughterty, Jenny L., Meyer, Joseph P. Formulating the Conceptual Base for Secondary Level Engineering Education: a Review and Synthesis, p. 3.

[14] Guskey, T. *What Makes Professional Development Effective*? Phi Delta Kappan 84, pp. 748-750.

[15] 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.

[16] Davidz, Heidi L. and Nightingale, Deborah J. *Enabling Systems Thinking to Accelerate the Development of Senior Systems Engineers*, Systems Engineering, Vol. II, No. 1, 2008, p. 11.

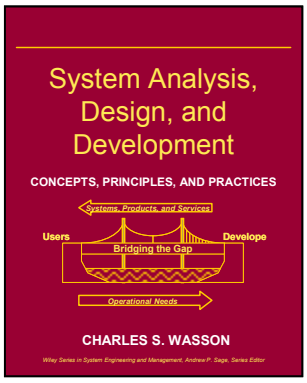
[17] Erwin, Ben. *K-12 Education and Systems Engineering: A New Perspective*, Proceedings of the American Society of Engineering Education National Conference, Session 1280, Seattle, WA, July 1998 p. 4.

[18] Cummings, Jim and Sayer, Dennis. Brave New Schools, Changing Cultural Illiteracy Through Global Learning Networks.

[19] 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 Annual Conference & Exposition, p. 1.

BIOGRAPHY

Mr. Wasson is a member of the International Council on System Engineering (INCOSE) and the American Society for Engineering Education (ASEE). His professional career experience spans over 38 years of proven leadership in program/project management; system, hardware, and software design, development, integration, and test; and organizational development with Lockheed Martin Corporation, Loral Corporation, Teledyne Brown Engineering, US Army Missile Research & Development Command, et al. As an internationally recognized author and instructor in system engineering and its organizational application, Charles is an invited guest speaker and panelist at professional meetings, conferences, and symposia.



As the author of numerous technical papers and papers, Mr. Wasson is the author of *System Analysis, Design, and Development*, a System Engineering textbook published by John Wiley & Sons, Inc. as part of its highly acclaimed System Engineering and Management Series. The textbook was selected by the International Academy of Astronautics (IAA) in Paris for its most prestigious of four book awards, the 2006 Engineering Sciences Book of the Year Award. The text was written for general application to any domain such as aerospace and defense, medical, financial, transportation, education, et al systems.