THE ENGINEERING DESIGN OF SYSTEMS
THE ENGINEERING DESIGN OF SYSTEMS
MODELS AND METHODS
Second Edition

DENNIS M. BUDE
In memory of my Mother and Father
# Contents

**Preface**  
ix  

**Part 1  Introduction, Overview, and Basic Knowledge**  
Chapter  1  Introduction to Systems Engineering  
Chapter  2  Overview of the Systems Engineering Design Process  
Chapter  3  Modeling and SysML Modeling  
Chapter  4  Discrete Mathematics: Sets, Relations, and Functions  
Chapter  5  Graphs and Directed Graphs (Digraphs)  

**Part 2  Design and Integration**  
Chapter  6  Requirements and Defining the Design Problem  
Chapter  7  Functional Architecture Development  
Chapter  8  Physical Architecture Development  
Chapter  9  Allocated Architecture Development  
Chapter  10  Interface Design  
Chapter  11  Integration and Qualification  

**Part 3  Supplemental Topics**  
Chapter  12  Graphical Modeling Techniques  
Chapter  13  Decision Analysis for Design Trades
Preface

This book is meant to be a basic text for courses in the engineering design of systems at both the upper division undergraduate and beginning graduate levels. The book is the product of many years of consulting on numerous portions of the system development process, research into the use of systems engineering in industry, and six years developing a course on the engineering design of systems. During the development of this book, I found that many engineers did not understand systems engineering. Even those that do may not have a good perspective on a complete and unified process for engineering a system. The desire to suppress the number of decisions being made during design is quite strong in most engineers. While engineers have learned modeling throughout their academic life, and most have developed models during the practice of engineering, very few engineers working on systems are knowledgeable of the modeling techniques required in systems engineering. In addition, most engineers are not aware of methods for using models during the systems engineering process. As a result, I adopted the following themes in formulating this book:

1. Defining the design problem in systems engineering is one of several keys to success and can be approached systematically using engineering techniques.
2. The design problem in systems engineering is defined in terms of requirements. These requirements evolve from a high-level set of mission and stakeholders’ requirements to detailed sets of derived requirements.
3. The design process will fail if the requirements are defined too narrowly, leaving little if any room for design decisions and raising the possibility
that no feasible solution exists. The design problem should be well
defined and decision rich.

4. For the design problem to be well defined, the evolving sets of
requirements must be complete (none missing), consistent (no contra-
dictions), correct (valid for an acceptable solution), and attainable (an
acceptable solution exists). While it is not possible at this time to state
requirements mathematically and prove these properties, it is possible to
develop mathematical and heuristic representations of the design
problem to assist in evaluating the presence of these properties.

5. The characteristics of the requirements will not be achieved if scenarios
defining how the system will be used are not elaborated in detail, the
interactions among the system and other systems are not defined, and
the stakeholders’ objectives are not understood. Each of these requires a
different kind of modeling to be successful.

6. The design problem is not likely to be well defined if the requirements do
not address every relevant phase of the system’s life cycle.

7. The design problem is not likely to be well defined if the requirements do
not contain stakeholder preferences for comparing feasible designs
against each other.

8. The keys to understanding many of the modeling techniques for
developing requirements, defining architectures, and deriving require-
ments are found in discrete mathematics: set theory, relations and
functions, and graph theory.

9. Integration requires a well-defined design, including a design of the
qualification process for verification, validation, and acceptance. A
systematic process of design provides all of the necessary inputs for
defining the qualification process.

10. Early validation of the evolution of the definition of the design problem
needs to be pursued vigorously to ensure that the definition of the design
problem does not change as the problem is defined in greater detail.

11. Qualification of the system is the key issue in integration. Qualification
includes verification and validation of both the requirements and the
system design, followed by the stakeholders’ acceptance. There are many
methods for qualifying the system; these methods must be chosen
judiciously.

12. Successful qualification also requires that decisions about what should be
tested be made in a systematic way that balances the two conflicting
objectives of not wasting resources and obtaining stakeholder acceptance.

The major changes for the second edition are descriptions of The Object
Management Group’s Systems Modeling Language (OMG SysML™) and the
introduction of new terminology. SysML is introduced in Chapter 1, defined in
some detail in Chapter 3, and referenced in other chapters. The major changes in terminology were motivated by suggestions from readers to be less focused on specific application domains. Originating requirements has become stakeholders’ requirements. Originating Requirements Document has become Stakeholders’ Requirements Document. The operational architecture has become the allocated architecture. New material has been added in Chapter 1 to enhance the introduction of the engineering of systems. Additional material in Chapter 1 describes different types of systems and outlines the various attributes of the value provided by systems engineering. Minor changes have been made to several other chapters as well. Finally, I have added a large selection of historical references for systems engineering.

The book is divided into three major parts: (1) Introduction, Overview, and Basic Knowledge; (2) Design and Integration Topics; and (3) Supplemental Topics. The first part provides an introduction to the issues associated with the engineering of a system. Next, an overview of the engineering process is provided so that readers will have a context for the more detailed material. Finally, basic knowledge needed for the core material is presented. Homework problems are provided at the end of each chapter.

Chapter 1 defines a system, systems engineering, the life cycle of a system, and then introduces systems engineering processes. This material sets the stage for the details that follow.

Chapter 2 provides an overview of the details that are to come by presenting a number of basic concepts; these concepts include an operational concept, objectives, requirements, functions, items, components, interfaces verification, validation, and acceptance. The relations among these concepts are also addressed.

Chapter 3 provides an overview of modeling and the types of modeling needed in engineering systems. Modeling methods associated with SysML are then introduced and described. While IDEF0 is not part of SysML, this topic has been kept in Chapter 3 as an important part of the modeling concepts described in this book.

Chapter 4 presents basic discrete mathematics. The purpose of the discrete mathematics is to demonstrate the mathematical rigor for which systems engineering must strive and to provide a language with which we can discuss key issues. Examples of such important concepts are the distinction between a relation and a function and why this is critical for engineering a system; a partition of the elements of a set that can be applied to many systems engineering concepts (e.g., requirements); and partial orders of functional execution.

Chapter 5 extends the discussion of discrete mathematics to graph theory so that the graphical communication structures commonly used in the engineering of systems can be seen to have substantial problems as rigorous mathematical representations. On the other hand, the difficult concepts in Chapter 4 can be effectively represented with graphs for analysis and communication.
Part 2 covers the critical material required to understand the major elements needed in the engineering design of any system: requirements, architectures (functional, physical, and allocated), interfaces, and qualification.

Requirements development is approached as a systematic process in Chapter 6. This systematic process involves the definition of an operational concept of the system (including usage scenarios), a description of the involvement of the system with other systems, and an objectives hierarchy of the stakeholders across all phases of the system’s life cycle. A partition of requirements is employed to discuss the systematic approach for defining requirements.

Definitions of the functional, physical, and allocated architectures are provided as well as the detailed methods for developing these architectures in Chapters 7 through 9. Chapter 7 begins with several definitions that are needed to enable a meaningful discussion of the topic. The notion of a functional architecture is defined. An emphasis is placed on process modeling in Chapter 7. However, additional material is presented in Chapters 3 and 12 on data and behavioral modeling methods, as well as other approaches for process modeling. (This material can be used while discussing Chapters 7 through 9.)

Modeling approaches for partitioning a function into segments are discussed. Key topics are feedback and control within the functional decomposition and evaluating the architecture for shortfalls and overlaps. Chapter 7 also addresses the functionality needed for error detection and recovery as well as tracing the input/output requirements to functions and items.

Chapter 8 introduces the distinction between the generic and instantiated physical architectures. The morphological box is used to demonstrate the generation of multiple instantiated physical architectures. The graphical representation of the physical architecture is discussed along with notions of centralized, decentralized, and distributed architectures. Finally, fault-tolerant architectures are described.

Chapter 9 defines the allocated architecture and discusses the allocation of functions to components, the tracing and derivation of requirements, the analysis of activation and control structures, and the conduct of various analyses (risk, performance, and trade-off).

Chapter 10 characterizes interfaces; discusses the functions associated with interfaces in several contexts (communications systems and software design); describes interface architectures; and discusses interface design as it impacts system performance as part of the design process.

Finally, qualification of the system (Chapter 11) during integration requires the understanding of the stakeholders’ needs and the qualification methods that are typically used. Deciding what to test and how to test it is critical in this phase of the development process. All of the topics in Chapters 6 to 11 are addressed in a rigorous and systematic manner, consistent with the general, practical application of systems engineering in industry.

Homework exercises are provided on each of these topics from Part 2 for several real but simple systems that are familiar to all students: an automatic teller machine (ATM), an air bag, and the OnStar system of Cadillac. A case
study is available over the web to give the students a sample of the solutions to the homework. Readers are encouraged to access a limited version of a commercial system engineering software product (CORE) to enhance the conduct of these homework exercises and the educational mission of this book.

Finally, two additional key topics are introduced in the third part: methods for data, process, and behavior modeling and decision analysis. Chapter 12 addresses the topics of data modeling, process modeling, and behavior modeling. Many alternate approaches for each of these modeling areas are described in detail so that teachers using this text can substitute the material most relevant to their program for the IDEF0 process modeling in Chapter 3. (A few minutes of IDEF0 instruction will be required in any course because of the extensive use that I have made of an IDEF0 model of the systems engineering process in Appendix B.)

Chapter 13 presents the key topics of decision analysis as an integrative way of supporting the many decisions that are part of the design and integration of a system. These decision analytic topics include the development and quantification of values (objectives, value functions, and trade-offs), and the modeling of uncertainty regarding facts.

The homework problems and the case study of the elevator are defined with the express purpose of having the student demonstrate the level of understanding necessary to perform the engineering activities described in the book. In developing these homework exercises I have taken the position that demonstrating an ability to discuss how to do systems engineering is a necessary but not a sufficient level of understanding. The CORE software (that is appropriate for use with this book is available via the web: http://www.vitechcorp.com) takes the tedium out of performing these systems engineering activities as well as reinforcing the basic concepts behind the activities. The case material related to an elevator system can be downloaded from the following web site: http://www.theengineeringdesignofsystems.com.

Many of the ideas for this book have originated with Alexander Levis. I have benefited greatly from my conversations with him. Jim Long introduced me to much of the systems engineering process and has since provided many thought-provoking concepts and ideas since we first met in 1991. Ron Howard guided me through the Ph.D. process and provided me with a wonderful foundation in decision analysis. This foundation in decision analysis is at the heart of the methods proposed in this book. I have worked on several consulting over the last 20 years with Terry Bresnick; Terry’s comments and questions have helped shape much of my thinking on the application of decision analysis to the engineering design of a system. Andrew Sage has seen several drafts of the book and provided many very useful comments, including suggestions for its title. Many faculty members who taught from the first edition have provided useful comments that led to improvements.

Sanford Friedenthal and Abe Melich were kind enough to review portions of the original manuscript when it was near completion. Both Sandy and Abe provided very valuable comments for improving the quality of the material
and its presentation. Sandy has given me a great deal of information and encouragement to include the SysML material in this second edition.

Several colleagues at George Mason University and Stevens Institute of Technology have provided many useful comments and suggestions. I wish to thank Kathryn Laskey, William Miller, and Mike Pennotti.

Several students and teaching assistants have contributed to sections of these notes. Cathy Brown provided a substantial extension of the requirements for the elevator case study. John Van Ormer extended the physical architecture of the elevator. Jahan Araghi extended my initial case study on the ATM as part of his Master's project. Tong Zhang and Parham Pasha provided some examples on sets, relations, and graphs. Christine Salter provided extensive support in addressing topics that needed revision, developed solutions for homework problems, and provided solution material for the OnStar and ATM problems. Several student groups provided material on which the air bag case is based. Meg Giordana and Barry Liner provided extensive comments on the qualification material. Tim Parker developed two case studies for use in Chapters 8 and 9: the FBI Fingerprint Identification System and the Wide-Area Augmentation System of the Federal Aviation Administration. Steve Charbonneau provided interesting insights about state charts as part of his M.S. Thesis. The SYST 520 class at George Mason University during the spring of 1998 provided many extensive and useful comments on an early draft of the first edition.

I wish to thank all of these individuals, as well as many others with whom I have conversed on these topics, for stimulating me to complete this effort.

One of the most difficult aspects of writing this book has been to decide which material to include and which to leave out. There is still a great deal more to be said on the topics covered in this book and on some additional topics that were not included. More importantly, there is still a great deal more to discover, at least on my part.

DENNIS M. BUEDE

Reston, Virginia
November 2008
Part 1

Introduction, Overview, and Basic Knowledge
Chapter 1

Introduction to Systems Engineering

1.1 INTRODUCTION

A system is commonly defined to be “a collection of hardware, software, people, facilities, and procedures organized to accomplish some common objectives.” The stakeholders for the system hold these objectives. Never forget that the system being addressed by one group of engineers is the subsystem of another group and the supersystem of yet a third group. The objective of the engineers for a system is to provide a system that accomplishes the primary objectives set by the stakeholders, including those objectives associated with the creation, production, and disposal of the system. To accomplish this engineering task, the engineers must identify the system’s stakeholders throughout the system’s life cycle and define the objectives of all of these stakeholders. These objectives typically address the triad of cost, schedule, and performance — cheaper, faster, and better.

A major characteristic of the engineering of systems is the attention devoted to the entire life cycle of the system. This life cycle has been characterized as “birth to death,” and “lust to dust.” That is, the life cycle begins with the gleam in the eyes of the users or stakeholders, is followed by the definition of the stakeholders’ needs by the systems engineers, includes developmental design and integration, goes through production and operational use, usually involves refinement, and finishes with the retirement and disposal of the system. Ignoring any part of this life cycle while engineering the system can lead to sufficiently negative consequences, including failure at the extreme. In
particular, developing a system that has not adequately addressed the stakeholders’ needs leads to failures such as the “highway to nowhere” near San Francisco, which was stopped by political pressure brought to bear by homeowners on the surrounding hills overlooking the bay. The view of the bay that these homeowners enjoyed and thought was an associated right of the property they owned would have been blocked by the highway. Similar commercial failures that did not consider the needs of the stakeholders in sufficient detail include the personal computers IBM PC Jr. and the Apple LISA. This is not to say that the adherence to methods and models put forth in this book or any other will guarantee success or even the absence of failure. Rather the methods and models proposed here do attend to the entire life cycle of the system and provide a process that makes sense, can be tailored to various levels of detail as dictated by the complexity of the system being addressed, and attend to all of the details that many engineers during years of practice in systems engineering have determined to be useful.

The concepts of design and integration are critical to the methods addressed in this chapter and the book. The word *design* is used by many professions (artists, architects, all disciplines of engineering) and is claimed by each.

The *American Heritage Dictionary* [Berube, 1991] defines design as:

**design (di-zin’)** v. **-signed, -signing, -signs.** –tr. 1. To conceive in the mind; invent: designed his dream vacation. 2. To form a plan for: designed a marketing strategy for the new product. 3. To have a goal or purpose; intend. 4. To plan by making a preliminary sketch, outline, or drawing. 5. To create or execute in an artistic or highly skilled manner. –intr. 1. To make or execute plans. 2. To create designs. –n. 1. A drawing or sketch. 2. The invention and disposition of the forms, parts, or details of something according to a plan. 3. A decorative or artistic work. 4. A visual composition; pattern. 5. The art of creating designs. 6. A plan; project. 7. A reasoned purpose; intention. 8. Often designs. A sinister or hostile scheme: He has designs on my job. …

All but the third and eighth definitions for the noun usage will apply at various times during the course of this book. *Design* during the engineering of a system as discussed in this book is the preliminary activity that has the purpose of satisfying the needs of the stakeholders, begins in the mind of the lead engineer but has to be transformed into models employing visual formats in a highly skilled manner for success to be achieved. While this book addresses the engineering methods and models used during the design process, there is always an element of artistry that is required for the design process and the system to be successful.

Integration brings all of the detailed elements of the overall design together through a process of testing (or qualification) to achieve a valid system for meeting the needs of the stakeholders. Engineers of appropriate disciplines perform integration according to the specifications defined by the design of the system’s engineers. The integration process involves testing or qualification of both the elements of the system and the system itself to ensure that the system meets the ultimate needs of the stakeholders.
This chapter first provides an overview of the issues and process associated with the engineering of a system. This overview addresses the phases of the system’s life cycle, describes the importance of performing the engineering of a system well, provides a definition for the engineering of a system, introduces the key process model for the engineering of a system called the Vee model, describes the richness of decisions that are inherent in the engineering process, and discusses the diversity of expertise required for this engineering process. Section 1.3 describes process models that have been adopted by the software engineering community. Architectures play a key role in the engineering of systems and are introduced next. Requirements, Section 1.7, play a major role in the engineering of a system because they serve the role of defining the engineering design problem and capturing the key information needed to describe design decisions. The life cycle of the system is next examined in more detail. Finally, the Vee model for engineering a system is described in more detail.

The key method addressed in this chapter is the process used to perform the engineering of systems. Supplementing this discussion of the engineering method are discussions of the key concepts needed to understand the method at an introductory level. This method is presented as a process model; models and modeling are discussed in detail in Chapter 3 so the reader is asked to accept the notion of the process discussion as a discussion of a model until more detail on models can be provided in Chapter 3.

1.2 OVERVIEW OF THE ENGINEERING OF SYSTEMS

The development process in systems engineering is commonly viewed [Forsberg and Mooz, 1992; Lake, 1992] as a decomposition (or design) process followed by a recomposition (or integration) process (see Sidebar 1.1). During the decomposition process, the stakeholders’ requirements are analyzed and defined in engineering terms and then partitioned into a set of specifications (or specs) for several segments, elements, or components. It is critical that this design process be *broad in perspective* so that nothing is left out and every contingency is considered. Systems engineers must be “big picture” people. Depth is only achieved by much iteration through the design process, as many as are needed until the system’s specifications are sufficiently detailed for individual configuration items (CIs) to be built or purchased. This design process defines what the system must do, how well the system must do it, and how the system should be tested to verify and validate the system’s performance. To do this the systems engineers must maintain a very clear focus on the objectives that the system’s stakeholders (users, owners, manufacturers, maintainers, trainers, etc.) have defined for the system.

One of many possible representations of the life cycle of a system is shown in Figure 1.1, beginning with the identification of the need for the system and progressing through the retirement of the system. Some of the phases of the life cycle are described in more detail.
cycle are accomplished in parallel, as the diagram tries to depict; exactly which phases occur in parallel depends upon the type of system, the organization, and the context. For additional detail see Driscoll [2007].

As shown in Figure 1.1, design includes the preliminary system design as well as parts of the identification of need and concept definition. Parts of the identification of need and concept definition include the development of a basic idea and the first embodiment of the idea; these two initial activities are often called invention and are usually not part of the engineering of a system. Invention has a heavy technological and scientific focus. The last portions of the identification of need and concept design phases, plus preliminary system design, address the initial or follow-on commercialization of the idea based upon a specific statement of stakeholders’ needs.

SIDEBAR 1.1

The term systems engineering dates back to Bell Telephone Laboratories in the 1940s [Schlager, 1956; Hall, 1962; Fagen, 1978]. Fagen [1978] traces the concepts of systems engineering within Bell Labs back to early 1900s and describes major applications of systems engineering during World War II. RCA used the “systems approach” during the research and development of the electronically scanned, black and white television [Engstrom, 1957]. In 1943 the National Defense Research Committee established a Systems Committee with Bell Laboratories support to guide a project called C-79, the first task of which was to improve the communication system of the Air Warning Service. An unpublished chapter on systems engineering in the Bell system suggested that the first use of the phrase “systems engineering”
within the Bell system was in a memo in the summer of 1948. Systems engineering was identified as a unique function in the organizational structure of Bell Laboratories in 1951.

Involvement in the earliest intercontinental ballistic missile (ICBM) program, starting with Atlas, is the most well-known of early systems engineering activities.

Hall [1962] asserts that the first attempt to teach systems engineering as we know it today came in 1950 at MIT by Mr. Gilman, Director of Systems Engineering at Bell. The first book on Systems Engineering was written by Goode and Machol in 1957, titled *System Engineering – An Introduction to the Design of Large-Scale Systems*.

Hall [1962] defined systems engineering as a function with five phases: (1) system studies or program planning; (2) exploratory planning, which includes problem definition, selecting objectives, systems synthesis, systems analysis, selecting the best system, and communicating the results; (3) development planning, which repeats phase 2 in more detail; (4) studies during development, which includes the development of parts of the system and the integration and testing of these parts; and (5) current engineering, which is what takes place while the system is operational and being refined.

The RAND Corporation was founded in 1946 by the United States Air Force and created *systems analysis*, which is certainly an important part of systems engineering.

The Department of Defense entered the world of systems engineering in the late 1940s with the initial development of missiles and missile-defense systems [Goode and Machol, 1957].

Paul Fitts addressed the allocation of the system’s functions to the physical elements of the system in the late 1940s and early 1950s [Fitts, 1951].

There is special bibliography at the back of the book devoted to historical references.

The products of the design process serve as the inputs to the hardware and software design of detailed configuration item (CI) design. The CIs then reenter the systems engineering process during system integration for integration testing, verification, and validation. Further adjustments to the design occur during the refinement phase. The life-cycle phases associated with the engineering of the system are shaded in Figure 1.1. The term *concurrent engineering* simply means that the systems engineering process should be done with all of the phases (and their associated requirements) of the system life cycle in mind [Prasad, 1996]. This notion of concurrent engineering is a key concept addressed in this book.

The importance of systems engineering is highlighted by examining a generally accepted relationship between the phases of the system life cycle
and the commitment versus the incursion of costs. The time associated with the system’s life cycle is plotted on the x-axis; note the time increments are notional and should not be interpreted as equal to the relative length of the four stages being addressed. See Prang [1992] for an illustration based on computer boards. (Prang is also referenced in Scheiber [1995].) Figure 1.2 shows the major phases of the system life cycle on the horizontal axis. The curves represent the cost committed, based upon engineering design decisions, and the cost incurred, based upon actual expenditures. As can be seen, about 80% of the cost of the system is committed by the end of design and integration, while only about 20% of the actual cost for the system has been spent. Obviously, mistakes made in the front end of the system life cycle can have substantially negative impacts on the total cost of the system and its success with the users and bill payers.

There have been many definitions of systems engineering put forward since the 1950s when systems engineering became a profession. Table 1.1 provides several of these definitions. There are two important trends to note over the 20-year span of these definitions. First, the role of management in the systems engineering process is made explicit in the definitions from the 1990s. Second, the three pillars of engineering success (cost, schedule, and technical

![FIGURE 1.2 Cost commitment and incursion in the system life cycle.](image-url)
<table>
<thead>
<tr>
<th>Source</th>
<th>Definitions of Systems Engineering</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mil-Std 499A, 1974</td>
<td>The application of scientific and engineering efforts to: (1) transform an operational need into a description of system performance parameters and a system configuration through the use of an iterative process of definition, synthesis, analysis, design, test, and evaluation; (2) integrate related technical parameters and ensure compatibility of all related, functional, and program interfaces in a manner that optimizes the total system definition and design; (3) integrate reliability, maintainability, safety, survivability, human, and other such factors into the total technical engineering effort to meet cost, schedule, and technical performance objectives.</td>
</tr>
<tr>
<td>Sailor, 1990</td>
<td>Both a technical and management process; the technical process is the analytical effort necessary to transform an operational need into a system design of the proper size and configuration and to document requirements in specifications; the management process involves assessing the risk and cost, integrating the engineering specialties and design groups, maintaining configuration control, and continuously auditing the effort to ensure that cost, schedule, and technical performance objectives are satisfied to meet the original operational need.</td>
</tr>
<tr>
<td>Sage, 1992</td>
<td>The design, production, and maintenance of trustworthy systems within cost and time constraints.</td>
</tr>
<tr>
<td>Forsberg &amp; Mooz, 1992</td>
<td>The application of the <em>system analysis and design process</em> and the <em>integration and verification process</em> to the logical sequence of the <em>technical aspect of the project life cycle</em>.</td>
</tr>
<tr>
<td>Wymore, 1993</td>
<td>The intellectual, academic, and professional discipline the primary concern of which is the responsibility to ensure that all requirements for a bioware/hardware/software system are satisfied throughout the life cycle of the system.</td>
</tr>
<tr>
<td>Mil-Std 499B draft, 1993</td>
<td>An interdisciplinary approach encompassing the entire technical effort to evolve and verify an integrated and life-cycle balanced set of system people, product, and process solutions that satisfy customer needs. Systems engineering encompasses: (a) the technical efforts related to the development, manufacturing, verification, deployment, operations, support, disposal of, and user training for system products and processes; (b) the definition and management of the system configuration; (c) the translation of the system definition into work breakdown structures; and (d) development of information for management decision making.</td>
</tr>
<tr>
<td>INCOSE*, 1996</td>
<td>An interdisciplinary approach and means to enable the realization of successful systems.</td>
</tr>
</tbody>
</table>

*INCOSE is the International Council on Systems Engineering, a professional society of systems engineers. INCOSE’s definition of a system is an interacting combination of elements, viewed in relation to function.
performance) from the 1970s evolve to concerns over the life cycle, namely concurrent engineering.

The *American Heritage Dictionary* [Berube, 1991] defines engineering as:

> The application of scientific and mathematical principles to practical ends such as the design, construction, and operation of efficient and economical structures, equipment, and systems.

The following definitions of engineering and the engineering of systems are adopted here:

**Engineering:** discipline for transforming scientific concepts into cost-effective products through the use of analysis and judgment.

**Engineering of a System:** engineering discipline that develops, matches, and trades off requirements, functions, and alternate system resources to achieve a cost-effective, life-cycle-balanced product based upon the needs of the stakeholders.

Figure 1.3 shows the design and integration process as a “Vee” with the emphasis of this model of the engineering process for a system being on the activities that the engineers perform. The left or decomposition side of the Vee

![Figure 1.3](image_url)

**FIGURE 1.3** Systems engineering “Vee” (after Forsberg and Mooz [1992]).
coincides with the three phases at the beginning of the life cycle from Figure 1.1. Time proceeds from left to right in Figure 1.3, just as it did in Figure 1.1. The process is initiated at the top left of the Vee with the definition of the operational need of the stakeholders. The focus of the decomposition and definition process (or design) is the movement from an operational need to system-level requirements to specifications for each component to the specifications (or specs) for each CI. Since time is moving from left to right in Figure 1.3, parallel work on high- and low-level design activities is not only permitted but encouraged. The iterative nature of this design process, from high-level issues such as stakeholders’ requirements to low-level issues such as component and CI design, is accomplished by moving vertically in the Vee over short increments of time. This vertical movement during the design process is critical to success and has been observed in studies of expert designers [Guindon, 1990]. Note, this Vee model does not emphasize the interaction with the stakeholders even though that interaction is assumed to occur in order to enable the engineering processes depicted in the Vee model.

The horizontal line, drawn just under the middle intersection of the Vee in Figure 1.3, depicts the hand off of the final products of the design process, the CI specs, to the discipline (or design) engineers, those engineers whose orientation is electrical, mechanical, chemical, civil, aerospace, computer science, and the like and whose job it is to produce a physical entity. This dividing line can be drawn higher or lower to signify decreasing or increasing overlap between design and integration activities. As the dividing line is drawn in Figure 1.3, the sloping lines of the middle portion of the Vee can be extended until they meet the dividing line, with the resulting very modest overlap between design and integration. If the dividing line is raised above the intersection of the sloping lines of the Vee, there would be no intersection of design and integration. This complete separation of design and integration is often sought in practice to enhance contractual relationships between procurer and supplier of the system; however, this separation negatively impacts the schedule and cost associated with the development of the system. There is significant integration and qualification activity that should take place during design, as is discussed in Chapter 11. In many systems engineering activities the horizontal dividing line between systems engineering and the discipline engineers is drawn significantly lower than shown in Figure 1.3.

The right-hand side of the Vee depicts the integration and qualification activities of the engineering of a system. Integration involves the assembly of the CIs into components, the assembly of lower level components into higher level components, and the assembly of high-level components into the system. All of this assembly involves testing (or qualification) of the newly assembled system elements to determine whether the assembled element meets the set of requirements (or spec) that the design phase had established for that element; this qualification is called verification. Finally, after the system is verified against the system requirements, the system must be validated. After validation, the stakeholders determine whether the system is acceptable. Naturally,
there are problems throughout this process that require modifications to be made either to the design of the elements of the system or to the requirements that were developed during design. Recall that time is running from left to right in Figure 1.3; the Vee process allows for the low level of verification of CIs to be happening in parallel with some high-level validation and even acceptance activities.

A sample of the movement from operational need to CI specs is given for a race car in Table 1.2. The first column states the operational need or mission requirement: Win the Indianapolis 500. Associated with this need are stakeholders’ requirements concerning the pretrial average speed and the average speed during the race with the expected number of yellow flags and pit stops (note the numbers in Table 1.2 are notional and are not accurate reflections of race conditions). System-level requirements can then be derived that are more meaningful during engineering. As an example, the key system-level requirement involves the \( g-g \) space of a vehicle [Milliken and Milliken, 1995]. Race cars, when driven by experienced drivers, are always changing velocity in speed or direction. (Recall that speed is the velocity you are traveling in your direction of travel. But when traveling around a curve, you also have a component of velocity perpendicular to your direction of travel.) Therefore the acceleration ability of the car in both longitudinal and lateral directions (see Fig. 1.4) is critical in the design process. Figure 1.4 portrays the \( g-g \) curve for a single car driven by three racers (charts a-c); the bottom right space (chart d) is the inferred \( g-g \) space of the vehicle. Finally, each of these system-level requirements is “flowed down” to component-level requirements, such as the engine’s horsepower and the drag coefficient of the body of the race car. (Note the true values of these parameters are closely guarded secrets of racing teams.) This process continues until the requirements for CIs are defined, establishing a hierarchy of requirements, from mission or need down to the CIs.

The system integration process starts during the decomposition and definition (or design) process. As part of design, the integration and qualification
plans are developed. The purpose of qualification is the verification and validation of the system’s design. Verification addresses the following question: Does the component, element, segment, or system meet its requirements, or have we built the component, ..., system right? On the other hand, validation, which is often combined with acceptance testing, demonstrates that the system satisfies the users’ needs, or have we built the right system? Note, as verification moves farther from the CIs and closer to the system, it is not possible to conduct enough testing to prove anything statistically. Demonstration is often

FIGURE 1.4 “g-g” design region for a racecar (from Milliken and Milliken [1995]).
the best that can be done. It is expected, though not desired, that there will be issues and problems that arise as part of this qualification process. Decisions must be made concerning relaxation of requirements versus design changes to specific CIs and components. During the design phase, integration activities should be planned to maximize the effectiveness of qualification within the resources and time available. These planned activities are then carried out during integration, with adaptations as needed. There should have been some thought given during design about what the most likely adaptations would be so that the integration phase has sufficient, built-in flexibility.

To be successful the engineering design of systems must embrace the notion that many decisions are made during the development process. This is not a controversial position to take. However, adopting the notion that these decisions should be made via a rational, explicit process is not consistent with much of the current practice in the engineering of systems. Table 1.3 lists a sample of the many categories of development decisions. Chapter 13 provides a method for addressing these decisions. An important philosophical point in decision making is that decisions have to be made with the best information available at the time, realizing that the outcomes associated with the decision remain uncertain when the decision is made. Therefore, distinguishing between a good decision and a good outcome is important. The material in this book will also distinguish between the level of detail needed to make decisions in the engineering of a system and the level of detail needed to ensure proper implementation of the system’s components and CIs.

In order to accomplish this difficult job of engineering a system, people with many different specialties must be involved on the systems engineering team. The stakeholders are central to the success of this effort and need to be represented on the systems engineering team. Discipline engineers with knowledge of the technologies associated with the system’s concept are needed to provide the expertise needed for design and integration decisions throughout development. Discipline engineers not only come from traditional engineering fields such as electrical, mechanical, and civil but also from the social sciences to address psychological, informational, physical, and cultural issues. In addition, systems engineers who model and estimate system-level parameters such as cost and reliability fall in the category of discipline engineers. Analysts skilled in modeling and simulation, more and more of which is done on the computer rather than with scaled-down mock-ups of the system, are also important members of this team. Engineers skilled in the processes (or methods) of systems engineering form the nucleus of this collection of skills. These processes and associated models are the nucleus of this book. Finally, managers that are in charge of meeting cost and schedule milestones need to be present. These five disciplines are depicted in the Venn diagram in Figure 1.5. Sidebar 1.2 describes Joe Shea, who was hired by the National Aeronautics and Space Administration (NASA) in 1961 to take charge of systems engineering for the Office of Manned Space Flight.