INCOSE

TWENTY-FIFTH 2015 ANNIVERSARY

SYSTEMS ENGINEERING HANDBOOK

A GUIDE FOR SYSTEM LIFE CYCLE PROCESSES AND ACTIVITIES

FOURTH EDITION

WILEY

SYSTEMS ENGINEERING HANDBOOK

SYSTEMS ENGINEERING HANDBOOK

A GUIDE FOR SYSTEM LIFE CYCLE PROCESSES AND ACTIVITIES

FOURTH EDITION

INCOSE-TP-2003-002-04
2015

Prepared by:

International Council on Systems Engineering (INCOSE)
7670 Opportunity Rd, Suite 220
San Diego, CA, USA 92111-2222

Compiled and Edited by:

DAVID D. WALDEN, ESEP
GARRY J. ROEDLER, ESEP
KEVIN J. FORSBERG, ESEP
R. DOUGLAS HAMELIN
THOMAS M. SHORTELL, CSEP

WILEY

Published by John Wiley & Sons, Inc., Hoboken, New Jersey
Published simultaneously in Canada

Limit of Liability/Disclaimer of Warranty: While the publisher and author have used their best efforts in preparing this book, they make no representations or warranties with respect to the accuracy or completeness of the contents of this book and specifically disclaim any implied warranties of merchantability or fitness for a particular purpose. No warranty may be created or extended by sales representatives or written sales materials. The advice and strategies contained herein may not be suitable for your situation. You should consult with a professional where appropriate. Neither the publisher nor author shall be liable for any loss of profit or any other commercial damages, including but not limited to special, incidental, consequential, or other damages.

For general information on our other products and services or for technical support, please contact our Customer Care Department within the United States at (800) 762-2974, outside the United States at (317) 572-3993 or fax (317) 572-4002.

Wiley also publishes its books in a variety of electronic formats. Some content that appears in print may not be available in electronic formats. For more information about Wiley products, visit our web site at www.wiley.com.

Library of Congress Cataloging-in-Publication Data:

Systems engineering handbook : a guide for system life cycle processes and activities / prepared by International Council on Systems Engineering (INCOSE) ; compiled and edited by, David D. Walden, ESEP, Garry J. Roedler, ESEP, Kevin J. Forsberg, ESEP, R. Douglas Hamelin, Thomas M. Shortell, CSEP. – 4th edition.
pages cm
Includes bibliographical references and index.
ISBN 978-1-118-99940-0 (cloth)
1. Systems engineering–Handbooks, manuals, etc. 2. Product life cycle–Handbooks, manuals, etc. I. Walden, David D., editor. II. Roedler, Garry J., editor. III. Forsberg, Kevin, editor. IV. Hamelin, R. Douglas, editor. V. Shortell, Thomas M., editor. VI. International Council on Systems Engineering.
TA168.S8724 2015
620.001′1–dc23

2014039630

ISBN: 9781118999400

Set in 10/12pt Times LT Std by SPi Publisher Services, pondicherry, India

Printed in the United States of America

10 9 8 7 6 5 4 3 2 1

1 2015

CONTENTS

INCOSE NOTICES

This International Council on Systems Engineering (INCOSE) Technical Product was prepared by the INCOSE Knowledge Management working group. It is approved by INCOSE Technical Operations Leadership for release as an INCOSE Technical Product.

General Citation Guidelines: References to this handbook should be formatted as follows, with appropriate adjustments for formally recognized styles:

INCOSE (2015). *Systems Engineering Handbook: A Guide for System Life Cycle Process and Activities* (4th ed.). D. D. Walden, G. J. Roedler, K. J. Forsberg, R. D. Hamelin, and, T. M. Shortell (Eds.). San Diego, CA: International Council on Systems Engineering. Published by John Wiley & Sons, Inc.

HISTORY OF CHANGES

Revision	Revision date	Change description and rationale
Original	Jun 1994	Draft *Systems Engineering Handbook* (SEH) created by INCOSE members from several defense/aerospace companies—including Lockheed, TRW, Northrop Grumman, Ford Aerospace, and the Center for Systems Management—for INCOSE review
1.0	Jan 1998	Initial SEH release approved to update and broaden coverage of SE process. Included broad participation of INCOSE members as authors. Based on Interim Standards EIA 632 and IEEE 1220
2.0	Jul 2000	Expanded coverage on several topics, such as functional analysis. This version was the basis for the development of the Certified Systems Engineering Professional (CSEP) exam
2.0A	Jun 2004	Reduced page count of SEH v2 by 25% and reduced the US DoD-centric material wherever possible. This version was the basis for the first publically offered CSEP exam
3.0	Jun 2006	Significant revision based on ISO/IEC 15288:2002. The intent was to create a country- and domain-neutral handbook. Significantly reduced the page count, with elaboration to be provided in appendices posted online in the INCOSE Product Asset Library (IPAL)
3.1	Aug 2007	Added detail that was not included in SEH v3, mainly in new appendices. This version was the basis for the updated CSEP exam
3.2	Jan 2010	Updated version based on ISO/IEC/IEEE 15288:2008. Significant restructuring of the handbook to consolidate related topics
3.2.1	Jan 2011	Clarified definition material, architectural frameworks, concept of operations references, risk references, and editorial corrections based on ISO/IEC review
3.2.2	Oct 2011	Correction of errata introduced by revision 3.2.1
4.0	Jan 2015	Significant revision based on ISO/IEC/IEEE 15288:2015, inputs from the relevant INCOSE working groups (WGs), and to be consistent with the Guide to the Systems Engineering Body of Knowledge (SEBoK)

PREFACE

The objective of the International Council on Systems Engineering (INCOSE) *Systems Engineering Handbook* (SEH) is to describe key process activities performed by systems engineers. The intended audience is the systems engineering (SE) professional. When the term *systems engineer* is used in this handbook, it includes the new systems engineer, a product engineer or an engineer in another discipline who needs to perform SE, or an experienced systems engineer who needs a convenient reference.

The descriptions in this handbook show what each SE process activity entails, in the context of designing for required performance and life cycle considerations. On some projects, a given activity may be performed very informally; on other projects, it may be performed very formally, with interim products under formal configuration control. This document is not intended to advocate any level of formality as necessary or appropriate in all situations. The appropriate degree of formality in the execution of any SE process activity is determined by the following:

1. The need for communication of what is being done (across members of a project team, across organizations, or over time to support future activities)
2. The level of uncertainty
3. The degree of complexity
4. The consequences to human welfare

On smaller projects, where the span of required communications is small (few people and short project life cycle) and the cost of rework is low, SE activities can be conducted very informally and thus at low cost. On larger projects, where the span of required communications is large (many teams that may span multiple geographic locations and organizations and long project life cycle) and the cost of failure or rework is high, increased formality can significantly help in achieving project opportunities and in mitigating project risk.

In a project environment, work necessary to accomplish project objectives is considered “in scope”; all other work is considered “out of scope.” On every project, “thinking” is always “in scope.” Thoughtful tailoring and intelligent application of the SE processes described in this handbook are essential to achieve the proper balance between the risk of missing project technical and business objectives on the one hand and process paralysis on the other hand. Chapter 8 provides tailoring guidelines to help achieve that balance.

APPROVED FOR SEH V4:

Kevin Forsberg, ESEP, Chair, INCOSE Knowledge Management Working Group

Garry Roedler, ESEP, Co-Chair, INCOSE Knowledge Management Working Group

William Miller, INCOSE Technical Director (2013–2014)

Paul Schreinemakers, INCOSE Technical Director (2015–2016)

Quoc Do, INCOSE Associate Director for Technical Review

Kenneth Zemrowski, ESEP, INCOSE Assistant Director for Technical Information

LIST OF FIGURES

LIST OF TABLES

1

SYSTEMS ENGINEERING HANDBOOK SCOPE

1.1 PURPOSE

This handbook defines the discipline and practice of systems engineering (SE) for students and practicing professionals alike and provides an authoritative reference to understand the SE discipline in terms of content and practice.

1.2 APPLICATION

This handbook is consistent with ISO/IEC/IEEE 15288:2015, *Systems and software engineering—System life cycle processes* (hereafter referred to as ISO/IEC/IEEE 15288), to ensure its usefulness across a wide range of application domains—man-made systems and products, as well as business and services.

ISO/IEC/IEEE 15288 is an international standard that provides generic top-level process descriptions and requirements, whereas this handbook further elaborates on the practices and activities necessary to execute the processes. Before applying this handbook in a given organization or project, it is recommended that the tailoring guidelines in Chapter 8 be used to remove conflicts with existing policies, procedures, and standards already in use within an organization. Processes and activities in this handbook do not supersede any international, national, or local laws or regulations.

This handbook is also consistent with the *Guide to the Systems Engineering Body of Knowledge* (SEBoK, 2014) (hereafter referred to as the SEBoK) to the extent practicable. In many places, this handbook points readers to the SEBoK for more detailed coverage of the related topics, including a current and vetted set of references.

For organizations that do not follow the principles of ISO/IEC/IEEE 15288 or the SEBoK to specify their life cycle processes (including much of commercial industry), this handbook can serve as a reference to practices and methods that have proven beneficial to the SE community at large and that can add significant value in new domains, if appropriately selected and applied. Section 8.2 provides top-level guidance on the application of SE in selected product sectors and domains.

1.3 CONTENTS

This chapter defines the purpose and scope of this handbook. Chapter 2 provides an overview of the goals and value of using SE throughout the system life cycle.

INCOSE Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities, Fourth Edition.
Edited by David D. Walden, Garry J. Roedler, Kevin J. Forsberg, R. Douglas Hamelin and Thomas M. Shortell.

Chapter 3 describes an informative life cycle model with six stages: concept, development, production, utilization, support, and retirement.

ISO/IEC/IEEE 15288 identifies four process groups to support SE. Each of these process groups is the subject of an individual chapter. A graphical overview of these processes is given in Figure 1.1:

- *Technical processes* (Chapter 4) include business or mission analysis, stakeholder needs and requirements definition, system requirements definition, architecture definition, design definition, system analysis, implementation, integration, verification, transition, validation, operation, maintenance, and disposal.
- *Technical management processes* (Chapter 5) include project planning, project assessment and control, decision management, risk management, configuration management, information management, measurement, and quality assurance.
- *Agreement processes* (Chapter 6) include acquisition and supply.
- *Organizational project-enabling processes* (Chapter 7) include life cycle model management, infrastructure management, portfolio management, human resource management, quality management, and knowledge management.

This handbook provides additional chapters beyond the process groups listed in Figure 1.1:

- *Tailoring processes and application of systems engineering* (Chapter 8) include information on how to adapt and scale the SE processes and how to apply those processes in various applications. Not every process will apply universally. Careful selection

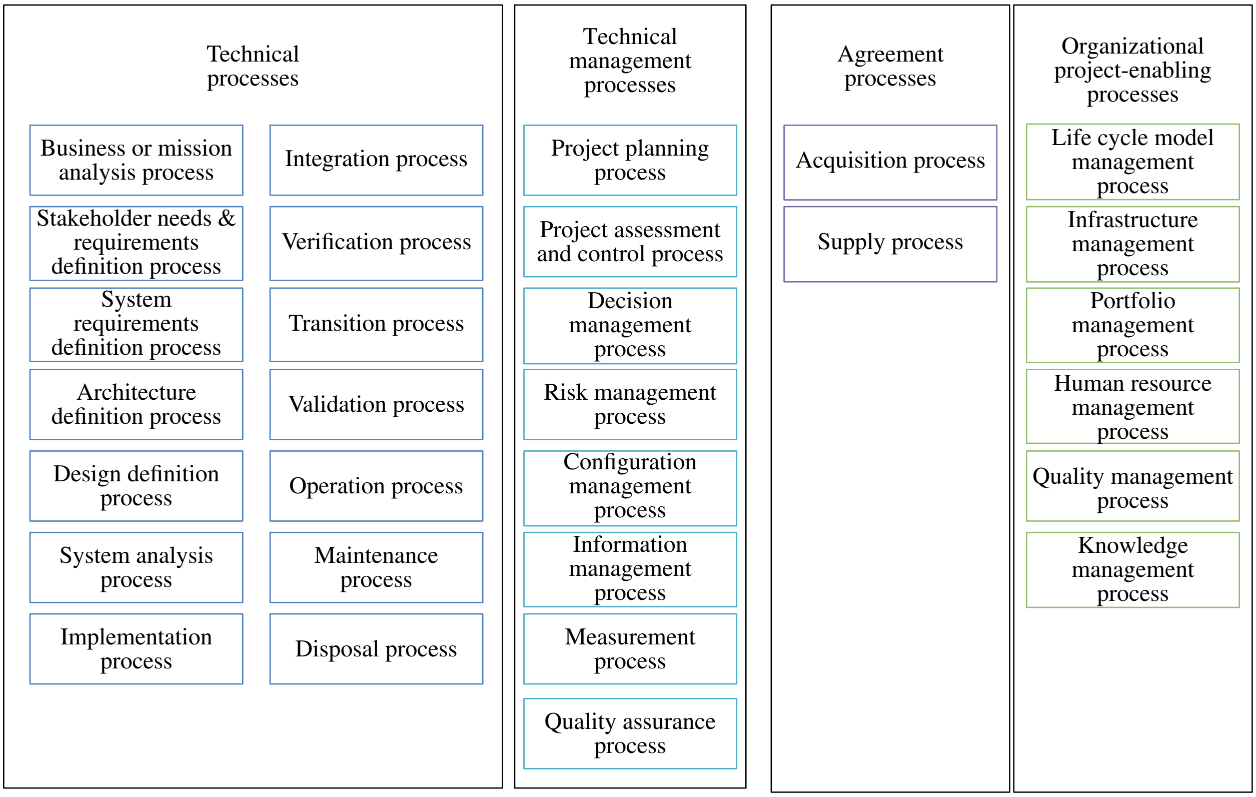
Technical processes
Business or mission analysis process
Integration process
Stakeholder needs & requirements definition process
Verification process
System requirements definition process
Transition process
Architecture definition process
Validation process
Design definition process
Operation process
System analysis process
Maintenance process
Implementation process
Disposal process
Technical management processes
Project planning process
Project assessment and control process
Decision management process
Risk management process
Configuration management process
Information management process
Measurement process
Quality assurance process
Agreement processes
Acquisition process
Supply process
Organizational project-enabling processes
Life cycle model management process
Infrastructure management process
Portfolio management process
Human resource management process
Quality management process
Knowledge management process

FIGURE 1.1 System life cycle processes per ISO/IEC/IEEE 15288. This figure is excerpted from ISO/IEC/IEEE 15288:2015, Figure 4 on page 17, with permission from the ANSI on behalf of the ISO. © ISO 2015. All rights reserved.

from the material is recommended. Reliance on process over progress will not deliver a system.

- *Crosscutting systems engineering methods* (Chapter 9) provide insights into methods that can apply across all processes, reflecting various aspects of the iterative and recursive nature of SE.
- *Specialty engineering activities* (Chapter 10) include practical information so systems engineers can understand and appreciate the importance of various specialty engineering topics.

Appendix A contains a list of references used in this handbook. Appendices B and C provide a list of acronyms and a glossary of SE terms and definitions, respectively. Appendix D provides an N^2 diagram of the SE processes showing where dependencies exist in the form of shared inputs or outputs. Appendix E provides a master list of all inputs/outputs identified for each SE process. Appendix F acknowledges the various contributors to this handbook. Errors, omissions, and other suggestions for this handbook can be submitted to the INCOSE using the comment form contained in Appendix G.

1.4 FORMAT

A common format has been applied in Chapters 4 through 7 to describe the system life cycle processes found in ISO/IEC/IEEE 15288. Each process is illustrated by an input–process–output (IPO) diagram showing key inputs, process activities, and resulting outputs. A sample is shown in Figure 1.2. Note that the IPO

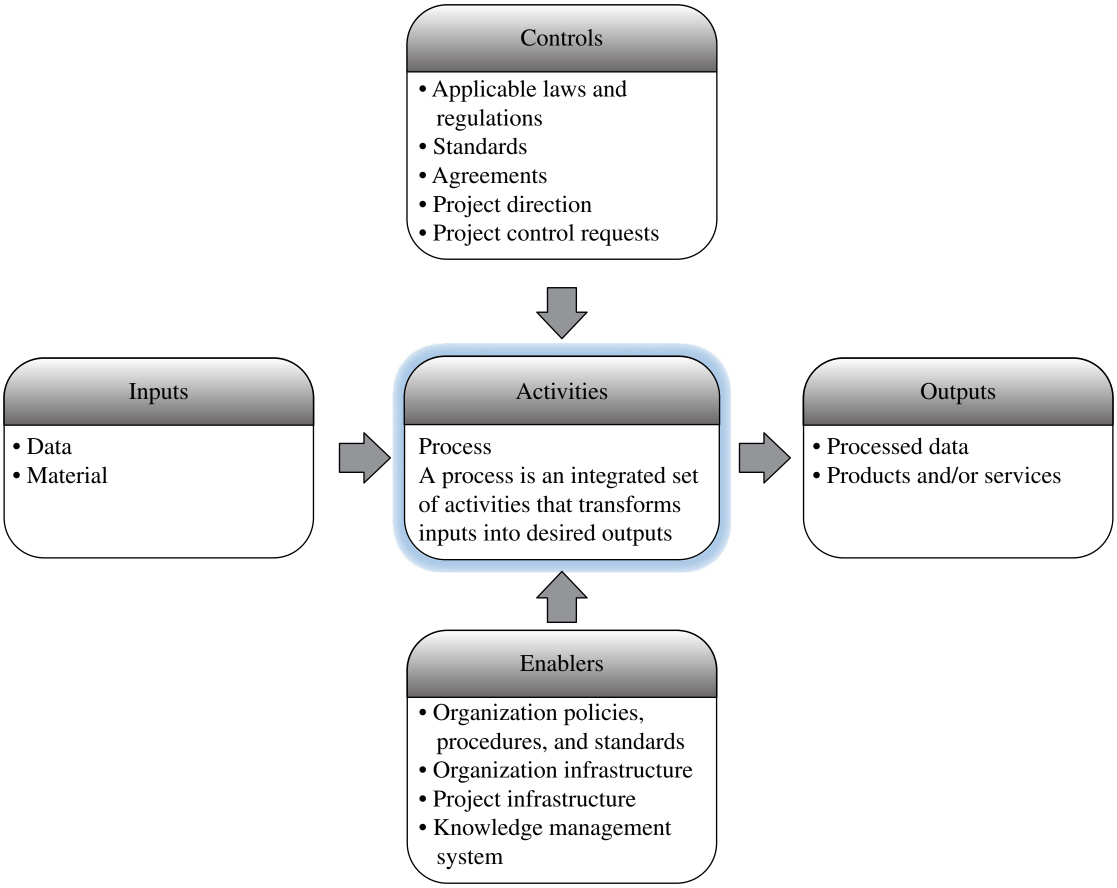
Controls
• Applicable laws and regulations
• Standards
• Agreements
• Project direction
• Project control requests
Inputs
• Data
• Material
Activities
Process
A process is an integrated set of activities that transforms inputs into desired outputs
Outputs
• Processed data
• Products and/or services
Enablers
• Organization policies, procedures, and standards
• Organization infrastructure
• Project infrastructure
• Knowledge management system

FIGURE 1.2 Sample of IPO diagram for SE processes. INCOSE SEH original figure created by Shortell and Walden. Usage per the INCOSE Notices page. All other rights reserved.

diagrams throughout this handbook represent “a” way that the SE processes can be performed, but not necessarily “the” way that they must be performed. The issue is that SE processes produce “results” that are often captured in “documents” rather than producing “documents” simply because they are identified as outputs. To understand a given process, readers are encouraged to study the complete information provided in the combination of diagrams and text and not rely solely on the diagrams.

The following heading structure provides consistency in the discussion of these processes:

- Process overview
- Purpose
- Description
- Inputs/outputs
- Process activities
- Process elaboration

To ensure consistency with ISO/IEC/IEEE 15288, the purpose statements from the standard are included verbatim for each process described herein. Inputs and outputs are listed by name within the respective IPO diagrams with which they are associated. A complete list of all inputs and outputs with their respective descriptions appears in Appendix E.

The titles of the process activities listed in each section are also consistent with ISO/IEC/IEEE 15288. In some cases, additional items have been included to provide summary-level information regarding industry best practices and evolutions in the application of SE processes.

The controls and enablers shown in Figure 1.2 govern all processes described herein and, as such, are not repeated in the IPO diagrams or in the list of inputs associated with each process description. Typically, IPO diagrams do not include controls and enablers, but since they are not repeated in the IPO diagrams throughout the rest of the handbook, we have chosen to label them IPO diagrams. Descriptions of each control and enabler are provided in Appendix E.

1.5 DEFINITIONS OF FREQUENTLY USED TERMS

One of the systems engineer’s first and most important responsibilities on a project is to establish nomenclature and terminology that support clear, unambiguous communication and definition of the system and its elements, functions, operations, and associated processes. Further, to promote the advancement of the field of SE throughout the world, it is essential that common definitions and understandings be established regarding general methods and terminology that in turn support common processes. As more systems engineers accept and use common terminology, SE will experience improvements in communications, understanding, and, ultimately, productivity.

The glossary of terms used throughout this book (see Appendix C) is based on the definitions found in ISO/IEC/IEEE 15288; ISO/IEC/IEEE 24765, *Systems and Software Engineering—Vocabulary* (2010); and SE VOCAB (2013).

2

SYSTEMS ENGINEERING OVERVIEW

2.1 INTRODUCTION

This chapter offers a brief overview of the systems engineering (SE) discipline, beginning with a few key definitions, an abbreviated survey of the origins of the discipline, and discussions on the value of applying SE. Other concepts, such as systems science, systems thinking, SE leadership, SE ethics, and professional development, are also introduced.

2.2 DEFINITIONS AND CONCEPTS OF A SYSTEM

While the concepts of a *system* can generally be traced back to early Western philosophy and later to science, the concept most familiar to systems engineers is often traced to Ludwig von Bertalanffy (1950, 1968) in which a system is regarded as a "whole" consisting of interacting "parts." The ISO/IEC/IEEE definitions provided in this handbook draw from this concept.

2.2.1 General System Concepts

The systems considered in ISO/IEC/IEEE 15288 and in this handbook

> [5.2.1] … are man-made, created and utilized to provide products or services in defined environments for the benefit of users and other stakeholders.

The definitions cited here and in Appendix C refer to systems in the real world. A system concept should be regarded as a shared "mental representation" of the actual system. The systems engineer must continually distinguish between systems in the real world and system representations. The INCOSE and ISO/IEC/IEEE definitions draw from this view of a system:

> … an integrated set of elements, subsystems, or assemblies that accomplish a defined objective. These elements include products (hardware, software, firmware), processes, people, information, techniques, facilities, services, and other support elements. (INCOSE)
>
> [4.1.46] … combination of interacting elements organized to achieve one or more stated purposes. (ISO/IEC/IEEE 15288)

Thus, the usage of terminology throughout this handbook is clearly an elaboration of the fundamental idea that a system is a purposeful whole that consists of interacting parts.

INCOSE Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities, Fourth Edition.
Edited by David D. Walden, Garry J. Roedler, Kevin J. Forsberg, R. Douglas Hamelin and Thomas M. Shortell.

An external view of a system must introduce elements that specifically do not belong to the system but do interact with the system. This collection of elements is called the *operating environment or context* and can include the users (or operators) of the system.

The internal and external views of a system give rise to the concept of a *system boundary*. In practice, the system boundary is a "line of demarcation" between the system itself and its greater context (to include the operating environment). It defines what belongs to the system and what does not. The system boundary is not to be confused with the subset of elements that interact with the environment.

The *functionality* of a system is typically expressed in terms of the interactions of the system with its operating environment, especially the users. When a system is considered as an integrated combination of interacting elements, the functionality of the system derives not just from the interactions of individual elements with the environmental elements but also from how these interactions are influenced by the organization (interrelations) of the system elements. This leads to the concept of *system architecture*, which ISO/IEC/IEEE 42010 (2011) defines as

> the fundamental concepts or properties of a system in its environment embodied in its elements, relationships, and in the principles of its design and evolution.

This definition speaks to both the internal and external views of the system and shares the concepts from the definitions of a system.

2.2.2 Scientific Terminology Related to System Concepts

In general, *engineering* can be regarded as the practice of creating and sustaining services, systems, devices, machines, structures, processes, and products to improve the quality of life—getting things done effectively and efficiently. The repeatability of experiments demanded by science is critical for delivering practical engineering solutions that have commercial value. Engineering in general and SE in particular draw heavily from the terminology and concepts of science.

An *attribute* of a system (or system element) is an observable characteristic or property of the system (or system element). For example, among the various attributes of an aircraft is its air speed. Attributes are represented symbolically by variables. Specifically, a *variable* is a symbol or name that identifies an attribute. Every variable has a domain, which could be but is not necessarily measurable. A *measurement* is the outcome of a process in which the system of interest (SOI) interacts with an observation system under specified conditions. The outcome of a measurement is the assignment of a *value* to a variable. A system is in a *state* when the values assigned to its attributes remain constant or steady for a meaningful period of time (Kaposi and Myers, 2001). In SE and software engineering, the *system elements* (e.g., software objects) have *processes* (e.g., operations) in addition to attributes. These have the binary logical values of being either *idle* or *executing*. A complete description of a system state therefore requires values to be assigned to both attributes and processes. *Dynamic behavior* of a system is the time evolution of the system state. *Emergent behavior* is a behavior of the system that cannot be understood exclusively in terms of the behavior of the individual system elements.

The key concept used for problem solving is the *black box/white box* system representation. The black box representation is based on an external view of the system (attributes). The white box representation is based on an internal view of the system (attributes and structure of the elements). There must also be an understanding of the relationship between the two. A system, then, is represented by the (external) attributes of the system, its internal attributes and structure, and the interrelationships between these that are governed by the laws of science.

2.2.3 General Systems Methodologies

Early pioneers of SE and software engineering, such as Yourdon (1989) and Wymore (1993), long sought to bring discipline and precision to the understanding and management of the dynamic behavior of a system by seeking relations between the external and internal representations of the system. Simply stated, they believed that if the flow of dynamic behavior (the system state evolution) could be mapped coherently into the flow of states of the constituent elements of the system, then emergent behaviors could be better understood and managed.

Klir (1991) complemented the concepts of a system in engineering and science with a general systems methodology. He regarded problem solving in general to rest upon a principle of alternatively using abstraction and interpretation to solve a problem. He considered that his methodology could be used both for system inquiry (i.e., the representation of an aspect of reality) and for system definition (i.e., the representation of purposeful man-made objects).

2.3 THE HIERARCHY *WITHIN* A SYSTEM

In the ISO/IEC/IEEE usage of terminology, the *system elements* can be *atomic* (i.e., not further decomposed), or they can be *systems on their own merit* (i.e., decomposed into further subordinate system elements). The *integration* of the system elements must establish the relationship between the effects that *organizing* the elements has on their *interactions* and how these effects enable the system to achieve its *purpose*.

One of the challenges of system definition is to understand what level of detail is necessary to define each system element and the interrelations between elements. Because the SOIs are in the real world, this means that the response to this challenge will be domain specific. A system element that needs only a black box representation (external view) to capture its requirements and confidently specify its real-world solution definition can be regarded as atomic. Decisions to make, buy, or reuse the element can be made with confidence without further specification of the element. This leads to the concept of hierarchy within a system.

One approach to defining the elements of a system and their interrelations is to identify a complete set of distinct system elements with regard only to their relation to the whole (system) by suppressing details of their interactions and interrelations. This is referred to as a partitioning of the system. Each element can be either atomic or it can be a much higher level that could be viewed as a system itself. At any given level, the elements are grouped into distinct subsets of elements subordinated to a higher level system, as illustrated in Figure 2.1. Thus, hierarchy within a system is an organizational representation of system structure using a partitioning relation.

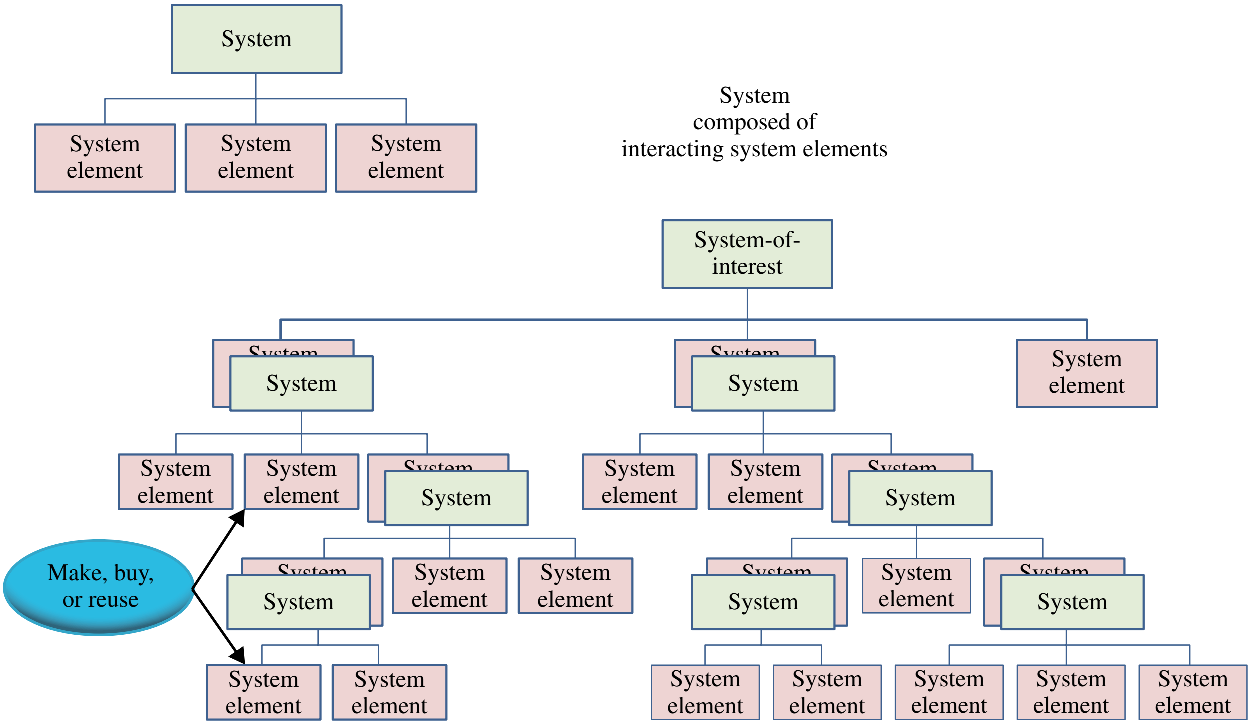

FIGURE 2.1 Hierarchy within a system. This figure is adapted from ISO/IEC/IEEE 15288:2015, Figure 1 on page 11 and Figure 2 on page 12, with permission from the ANSI on behalf of the ISO. © ISO 2015. All rights reserved.

The concept of a system hierarchy described in ISO/IEC/IEEE 15288 is as follows:

> [5.2.2] The system life cycle processes … are described in relation to a system that is composed of a set of interacting system elements, each of which can be implemented to fulfill its respective specified requirements.

The art of defining a hierarchy within a system relies on the ability of the systems engineer to strike a balance between clearly and simply defining span of control and resolving the structure of the SOI into a complete set of system elements that can be implemented with confidence. Urwick (1956) suggests that a possible heuristic is for each level in the hierarchy to have no more than 7 ± 2 elements subordinate to it. Others have also found this heuristic to be useful (Miller, 1956). A level of design with too few subordinate elements is unlikely to have a distinct design activity. In this case, both design and verification activities may contain redundancy. In practice, the nomenclature and depth of the hierarchy can and should be adjusted to fit the complexity of the system and the community of interest.

2.4 DEFINITION OF SYSTEMS OF SYSTEMS

A "system of systems" (SoS) is an SOI whose elements are managerially and/or operationally independent systems. These interoperating and/or integrated collections of constituent systems usually produce results unachievable by the individual systems alone. Because an SoS is itself a system, the systems engineer may choose whether to address it as either a system or as an SoS, depending on which perspective is better suited to a particular problem.

The following characteristics can be useful when deciding if a particular SOI can better be understood as an SoS (Maier, 1998):

- Operational independence of constituent systems
- Managerial independence of constituent systems
- Geographical distribution
- Emergent behavior
- Evolutionary development processes

Figure 2.2 illustrates the concept of an SoS. The air transport system is an SoS comprising multiple aircraft,

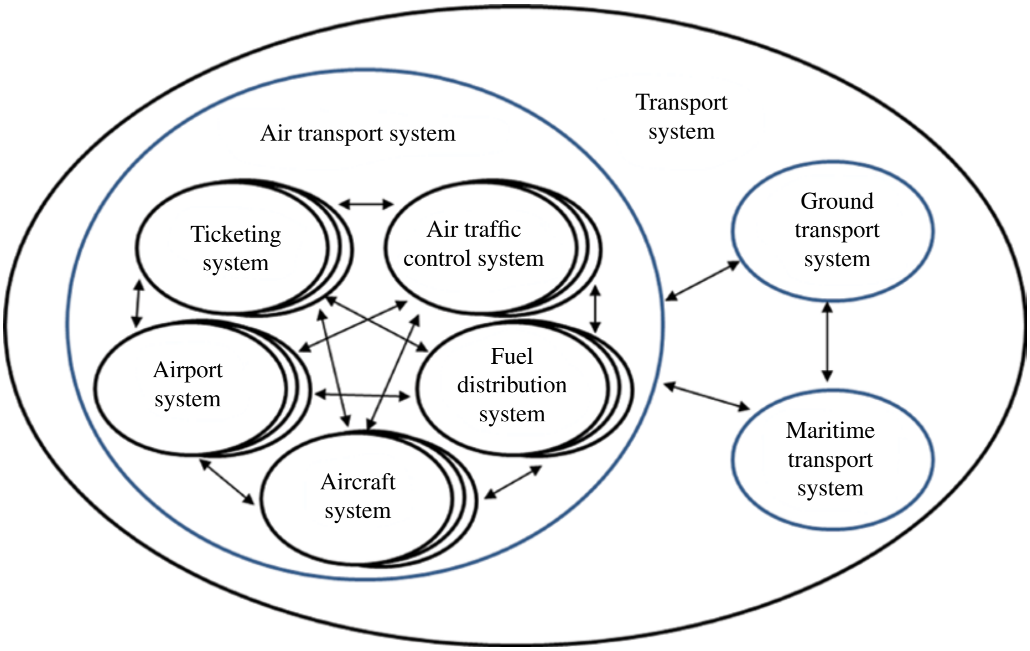

FIGURE 2.2 Example of the systems and systems of systems within a transport system of systems. Reprinted with permission from Judith Dahmann. All other rights reserved.

airports, air traffic control systems, and ticketing systems, which along with other systems such as security and financial systems facilitate passenger transportation. There are equivalent ground and maritime transportation SoS that are all in turn part of the overall transport system (an SoS in the terms of this description).

The SoS usually exhibits complex behaviors, often created by the existence of the aforementioned Maier's characteristics. "Complexity" is essentially different from "complicated." In complicated systems, such as an automobile, the interactions between the many parts are governed by fixed relationships. This allows reasonably reliable prediction of technical, time, and cost issues. In complex systems, such as the air transport system, interactions between the parts exhibit self-organization, where local interactions give rise to novel, nonlocal, emergent patterns. Complicated systems can often become complex when the behaviors change, but even systems of very few parts can sometimes exhibit surprising complexity.

The best way to understand a complicated system is to break it down into parts recursively until the parts are so simple that we understand them and then to reassemble the parts to understand the whole. However, this approach does not help us to understand a complex system, because the emergent properties that we really care about disappear when we examine the parts in isolation. A fundamentally different approach is required to understand the whole in context through iterative exploration and adaptation. As a result, SE requires a balance of linear, procedural methods for sorting through complicatedness ("systematic activity") and holistic, nonlinear, iterative methods for harnessing complexity ("systemic" or systems thinking and analysis—always required when dealing with SoS). The tension between breaking things apart and keeping them in context must be dynamically managed throughout the SE process.

The following challenges all influence the engineering of an SoS (Dahmann, 2014):

1. *SoS authorities*—In an SoS, each constituent system has its own local "owner" with its stakeholders, users, business processes, and development approach. As a result, the type of organizational structure assumed for most traditional SE under a single authority responsible for the entire system is absent from most SoS. In an SoS, SE relies on crosscutting analysis and on composition and integration of constituent systems, which in turn depend on an agreed common purpose and motivation for these systems to work together toward collective objectives that may or may not coincide with those of the individual constituent systems.
2. *Leadership*—Recognizing that the lack of common authorities and funding poses challenges for SoS, a related issue is the challenge of leadership in the multiple organizational environment of an SoS. This question of leadership is experienced where a lack of structured control normally present in SE requires alternatives to provide coherence and direction, such as influence and incentives.
3. *Constituent systems' perspectives*—SoS are typically composed, at least in part, of in-service systems, which were often developed for other purposes and are now being leveraged to meet a new or different application with new objectives. This is the basis for a major issue facing SoS SE, that is, how to technically address issues that arise from the fact that the systems identified for the SoS may be limited in the degree to which they can support the SoS. These limitations may affect initial efforts at incorporating a system into an SoS, and systems' commitments to other users may mean that they may not be compatible with the SoS over time. Further, because the systems were developed and operate in different situations, there is a risk that there could be a mismatch in understanding the services or data provided by one system to the SoS if the particular system's context differs from that of the SoS.
4. *Capabilities and requirements*—Traditionally (and ideally), the SE process begins with a clear, complete set of user requirements and provides a disciplined approach to develop a system to meet these requirements. Typically, SoS are comprised of multiple independent systems with their own requirements, working toward broader capability objectives. In the best case, the SoS capability needs are met by the constituent systems as they meet their own local requirements. However, in many cases, the SoS needs may not be consistent with the requirements for the constituent systems. In these cases, SoS SE needs to identify alternative approaches to meeting those needs either through

changes to the constituent systems or through the addition of other systems to the SoS. In effect, this is asking the systems to take on new requirements with the SoS acting as the "user."

5. *Autonomy, interdependencies, and emergence*—The independence of constituent systems in an SoS is the source of a number of technical issues facing SE of SoS. The fact that a constituent system may continue to change independently of the SoS, along with interdependencies between that constituent system and other constituent systems, adds to the complexity of the SoS and further challenges SE at the SoS level. In particular, these dynamics can lead to unanticipated effects at the SoS level leading to unexpected or unpredictable behavior in an SoS even if the behavior of the constituent systems is well understood.
6. *Testing, validation, and learning*—The fact that SoS are typically composed of constituent systems that are independent of the SoS poses challenges in conducting end-to-end SoS testing, as is typically done with systems. First, unless there is a clear understanding of the SoS-level expectations and measures of those expectations, it can be very difficult to assess the level of performance as the basis for determining areas that need attention or to ensure users of the capabilities and limitations of the SoS. Even when there is a clear understanding of SoS objectives and metrics, testing in a traditional sense can be difficult. Depending on the SoS context, there may not be funding or authority for SoS testing. Often, the development cycles of the constituent systems are tied to the needs of their owners and original ongoing user base. With multiple constituent systems subject to asynchronous development cycles, finding ways to conduct traditional end-to-end testing across the SoS can be difficult if not impossible. In addition, many SoS are large and diverse, making traditional full end-to-end testing with every change in a constituent system prohibitively costly. Often, the only way to get a good measure of SoS performance is from data collected from actual operations or through estimates based on modeling, simulation, and analysis. Nonetheless, the SoS SE team needs to enable continuity of operation and performance of the SoS despite these challenges.
7. *SoS principles*—SoS is a relatively new area, with the result that there has been limited attention given to ways to extend systems thinking to the issues particular to SoS. Work is needed to identify and articulate the crosscutting principles that apply to SoS in general and to develop working examples of the application of these principles. There is a major learning curve for the average systems engineer moving to an SoS environment and a problem with SoS knowledge transfer within or across organizations.

Beyond these general SE challenges, in today's environment, SoS pose particular issues from a security perspective. This is because constituent system interface relationships are rearranged and augmented asynchronously and often involve commercial off-the-shelf (COTS) elements from a wide variety of sources. Security vulnerabilities may arise as emergent phenomena from the overall SoS configuration even when individual constituent systems are sufficiently secure in isolation.

The SoS challenges cited in this section require SE approaches that combine both the systematic and procedural aspects described in this handbook with holistic, nonlinear, iterative methods.

2.5 ENABLING SYSTEMS

Enabling systems are systems that facilitate the life cycle activities of the SOI. The enabling systems provide services that are needed by the SOI during one or more life cycle stages, although the enabling systems are not a direct element of the operational environment. Examples of enabling systems include collaboration development systems, production systems, logistics support systems, etc. They enable progress of the SOI in one or more of the life cycle stages. The relationship between the enabling system and the SOI may be one where there is interaction between both systems or one where the SOI simply receives the services it needs when it is needed. Figure 2.3 illustrates the relationship of the SOI, enabling systems, and the other systems in the operational environment.

During the life cycle stages for an SOI, it is necessary to concurrently consider the relevant enabling systems and the SOI. All too often, it is assumed that the enabling

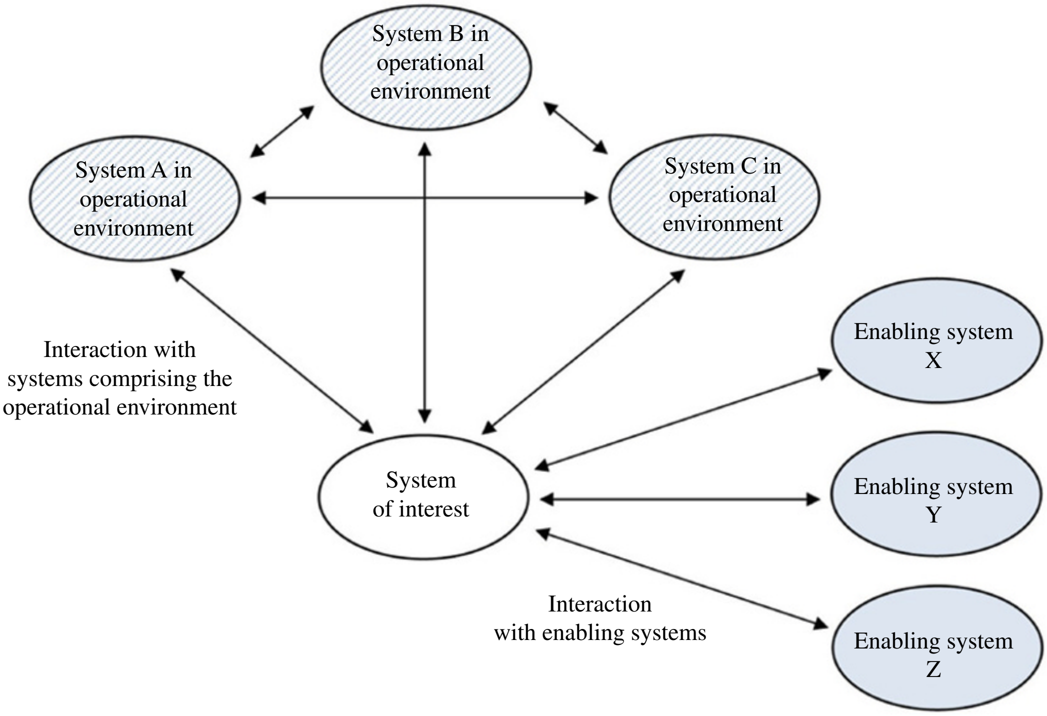
System B in operational environment
System A in operational environment
System C in operational environment
Enabling system X
Interaction with systems comprising the operational environment
System of interest
Enabling system Y
Interaction with enabling systems
Enabling system Z

FIGURE 2.3 System of interest, its operational environment, and its enabling systems. This figure is excerpted from ISO/IEC/IEEE 15288:2015, Figure 3 on page 13, with permission from the ANSI on behalf of the ISO. © ISO 2015. All rights reserved.

systems will be available when needed and are not considered in the SOI development. This can lead to significant issues for the progress of the SOI through its life cycle.

2.6 DEFINITION OF SYSTEMS ENGINEERING

SE is a perspective, a process, and a profession, as illustrated by these three representative definitions:

> Systems engineering is an interdisciplinary approach and means to enable the realization of successful systems. It focuses on defining customer needs and required functionality early in the development cycle, documenting requirements, and then proceeding with design synthesis and system validation while considering the complete problem: operations, cost and schedule, performance, training and support, test, manufacturing, and disposal. Systems engineering integrates all the disciplines and specialty groups into a team effort forming a structured development process that proceeds from concept to production to operation. Systems engineering considers both the business and the technical needs of all customers with the goal of providing a quality product that meets the user needs. (INCOSE, 2004)

> Systems engineering is an iterative process of top-down synthesis, development, and operation of a real-world system that satisfies, in a near optimal manner, the full range of requirements for the system. (Eisner, 2008)

> Systems engineering is a discipline that concentrates on the design and application of the whole (system) as distinct from the parts. It involves looking at a problem in its entirety, taking into account all the facets and all the variables and relating the social to the technical aspect. (FAA, 2006)

Certain keywords emerge from this sampling—interdisciplinary, iterative, sociotechnical, and wholeness.

The SE perspective is based on systems thinking. Systems thinking (see Section 2.9.2) is a unique perspective on reality—a perspective that sharpens our awareness of wholes and how the parts within those wholes interrelate. When a system is considered as a combination of system elements, systems thinking acknowledges the primacy of the whole (system) and the primacy of the relation of the interrelationships of the system elements to the whole. Systems thinking occurs through discovery, learning, diagnosis, and dialogue that lead to sensing, modeling, and talking about the real world to better understand, define, and work with systems. A systems thinker knows how systems fit into the larger context of day-to-day life, how they behave, and how to manage them.

The SE process has an iterative methodology that supports discovery, learning, and continuous improvement. As the process unfolds, systems engineers gain insights into the relationships between the specified requirements for the system and the emergent properties of the system. Insights into the emergent properties of a system can therefore be gained from understanding the interrelationships of the system elements and the relation of these to the whole (system). Due to circular causation, where one system variable can be both the cause and effect of another, even the simplest of systems can have unexpected and unpredictable emergent properties. Complexity, as discussed in Section 2.4, can further exacerbate this problem; hence, one of the objectives of the SE process is to minimize undesirable consequences. This can be accomplished through the inclusion of and contributions from experts across relevant disciplines coordinated by the systems engineer.

SE includes both technical and management processes, and both processes depend upon good decision making. Decisions made early in the life cycle of a system, whose consequences are not clearly understood, can have enormous implications later in the life of a system. It is the task of the systems engineer to explore these issues and make the critical decisions in a timely manner. The roles of the systems engineer are varied, and Sheard's "Twelve Systems Engineering Roles" (1996) provides one description of these variations.

2.7 ORIGINS AND EVOLUTION OF SYSTEMS ENGINEERING

The modern origins of SE can be traced to the 1930s followed quickly by other programs and supporters (Hughes, 1998). Tables 2.1 and 2.2 (Martin, 1996) offer a thumbnail of some important highlights in the origins and standards of SE. A list of current significant SE standards and guides is provided in Table 2.3.

TABLE 2.1 Important dates in the origins of SE as a discipline

1937	British multidisciplinary team to analyze the air defense system
1939–1945	Bell Labs supported NIKE missile project development
1951–1980	SAGE air defense system defined and managed by Massachusetts Institute of Technology (MIT)
1954	Recommendation by the RAND Corporation to adopt the term "systems engineering"
1956	Invention of systems analysis by RAND Corporation
1962	Publication of *A Methodology for Systems Engineering* by Hall
1969	Modeling of urban systems at MIT by Jay Forrester
1990	National Council on Systems Engineering (NCOSE) established
1995	INCOSE emerged from NCOSE to incorporate international view
2008	ISO, IEC, IEEE, INCOSE, PSM, and others fully harmonize SE concepts on ISO/IEC/IEEE 15288:2008

TABLE 2.2 Important dates in the origin of SE standards

1969	Mil-Std 499
1979	Army Field Manual 770-78
1994	Perry Memorandum urges military contractors to adopt commercial practices. EIA 632 IS (interim standard) and IEEE 1220 (trial version) issued instead of Mil-Std 499B
1998	EIA 632 released
1999	IEEE 1220 released
2002	ISO/IEC 15288 released, adopted by IEEE in 2003
2012	*Guide to the Systems Engineering Body of Knowledge (SEBoK)* released

TABLE 2.3 Current significant SE standards and guides

ISO/IEC/IEEE 15288	Systems and software engineering—System life cycle processes
ANSI/EIA-632	Processes for engineering a system
ISO/IEC/IEEE 26702	Systems engineering—Application and management of the systems engineering process (replaces IEEE 1220™)
SEBoK	Guide to the systems engineering body of knowledge
ISO/IEC TR 24748	Systems and software engineering—Life cycle management—Part 1, guide for life cycle management; Part 2, guide to the application of ISO/IEC 15288 (system life cycle processes)
ISO/IEC/IEEE 24765	Systems and software engineering—Vocabulary
ISO/IEC/IEEE 29148	Software and systems engineering—Life cycle processes—Requirements engineering
ISO/IEC/IEEE 42010	Systems and software engineering—Architecture description (replaces IEEE 1471)
ISO 10303-233	Industrial automation systems and integration—Product data representation and exchange—Part 233: Application protocol: Systems engineering
OMG SysML™	Object management group (OMG) systems modeling language (SysML™)
CMMI-DEV v1.3	Capability maturity model integration (CMMI®) for development
ISO/IEC 15504-6	Information technology—Process assessment—Part 6: An exemplar system life cycle process assessment model
ISO/IEC/IEEE 15289	Systems and software engineering—Content of systems and software life cycle process information products (documentation)
ISO/IEC/IEEE 15939	Systems and software engineering—Measurement process
ISO/IEC/IEEE 16085	Systems and software engineering—Life cycle processes—Risk management
ISO/IEC/IEEE 16326	Systems and software engineering—Life cycle processes—Project management
ISO/IEC/IEEE 24748-4	Life cycle management—Part 4: Systems engineering planning
ISO 31000	Risk management—Principles and guidelines
TechAmerica/ANSI EIA-649-B	National consensus standard for configuration management
ANSI/AIAA G-043A-2012e	ANSI/AIAA guide to the preparation of operational concept documents
ISO/IEC/IEEE 15026	Systems and software assurance—Part 1, concepts and vocabulary; Part 2, assurance case; Part 4, assurance in the life cycle

With the introduction of the international standard ISO/IEC 15288 in 2002, the discipline of SE was formally recognized as a preferred mechanism to establish agreement for the creation of products and services to be traded between two or more organizations—the supplier(s) and the acquirer(s). But even this simple designation is often confused in a web of contractors and subcontractors since the context of most systems today is as a part of an “SoS” (see Section 2.4).

2.8 USE AND VALUE OF SYSTEMS ENGINEERING

Even on its earliest projects, SE emerged as an effective way to manage complexity and change. As both complexity and change continue to escalate in products, services, and society, reducing the risk associated with new systems or modifications to complex systems continues to be a primary goal of the systems engineer. This is illustrated in Figure 2.4. The percentages along the timeline represent the actual life cycle cost (LCC) accrued over time based on a statistical analysis performed on projects in the US Department of Defense (DOD), as reported by the Defense Acquisition University (DAU, 1993). For example, the concept stage of a new system averages 8% of the total LCC. The curve for committed costs represents the amount of LCC committed by project decisions and indicates that when 20% of the actual cost has been accrued, 80% of the total LCC has already been committed. The diagonal arrow under the curve reminds us that errors are less expensive to remove early in the life cycle.

Figure 2.4 also demonstrates the consequences of making early decisions without the benefit of good information and analysis. SE extends the effort performed in concept exploration to reduce the risk of hasty commitments without adequate study. Though shown as

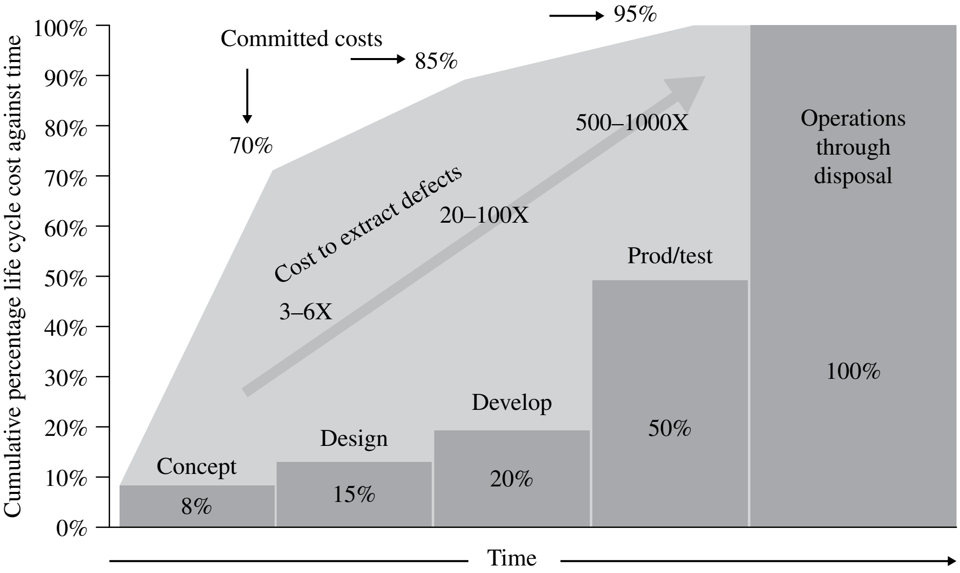

FIGURE 2.4 Committed life cycle cost against time. Reprinted with permission from DAU. All other rights reserved.

linear, the execution of the various life cycle stages associated with modern product development is, in actual application, recursive. Nonetheless, the consequences of ill-formed decisions throughout the life cycle are the same.

Another factor driving the need for SE is that complexity has an ever increasing impact on innovation. Few new products represent the big-bang introduction of new invention; rather, most products and services in today's market are the result of incremental improvement. This means that the life cycle of today's products and services is longer and subject to increasing uncertainty. A well-defined SE process becomes critical to establishing and maintaining a competitive edge in the twenty-first century.

As illustrated in Figure 2.5, the development and market penetration of technology has accelerated by more than a factor of four over the past 140 years. In this sample of products, the time it took to achieve 25% market penetration was reduced from about 50 years to below 12 years. On average, development from prototype to 25% market penetration went from 44 to 17 years.

Two studies have demonstrated the value of SE from an effectiveness and return on investment (ROI) perspective. These studies are summarized in the following.

2.8.1 SE Effectiveness

A 2012 study by the National Defense Industrial Association, the Institute of Electrical and Electronic Engineers (IEEE), and the Software Engineering Institute of Carnegie Mellon surveyed 148 development projects and found clear and significant relationships between the application of SE activities and the performance of those projects, as seen in Figure 2.6 and explained in the following (Elm and Goldenson, 2012).

The left column represents those projects deploying lower levels of SE expertise and capability, as measured by the quantity and quality of specific SE work products. Among these projects, only 15% delivered higher levels of project performance, as measured by satisfaction of budget, schedule, and technical requirements, and 52% delivered lower levels of project performance. The second column represents those projects deploying moderate levels of SE expertise and capability. Among these projects, 24% delivered higher levels of project performance, and only 29% delivered lower levels of performance. The third column represents those projects deploying higher levels of SE expertise and capability. For these projects, the number delivering higher levels of project performance increased substantially to 57%, while those delivering lower levels decreased to 20%.

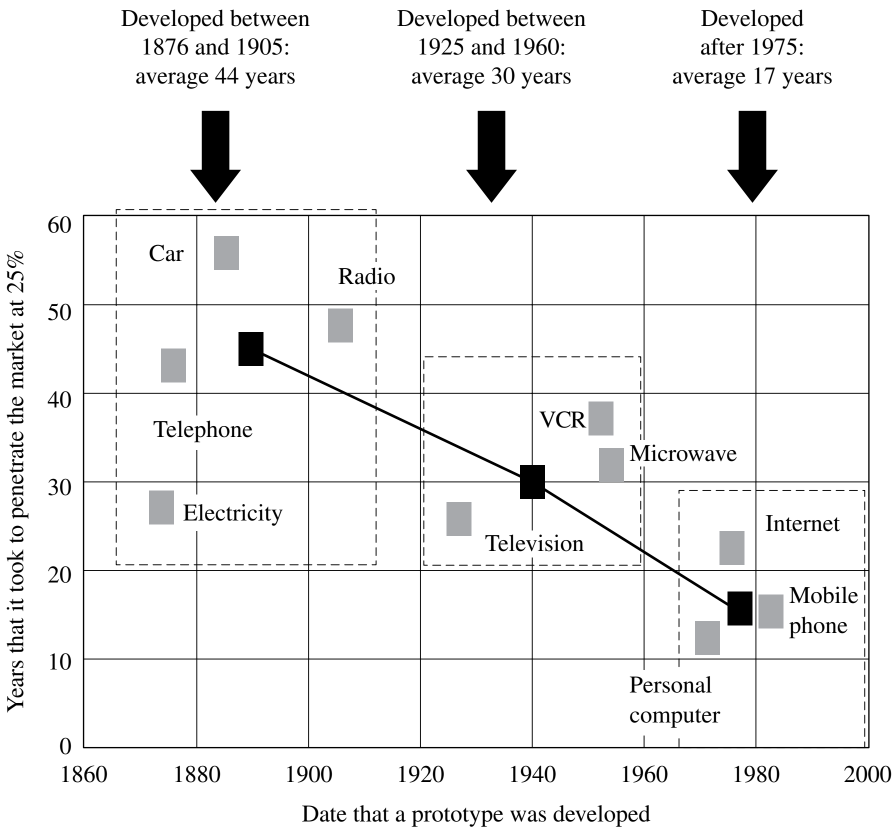

FIGURE 2.5 Technology acceleration over the past 140 years. INCOSE SEH original figure created by Michael Krueger. Usage per the INCOSE Notices page. All other rights reserved.

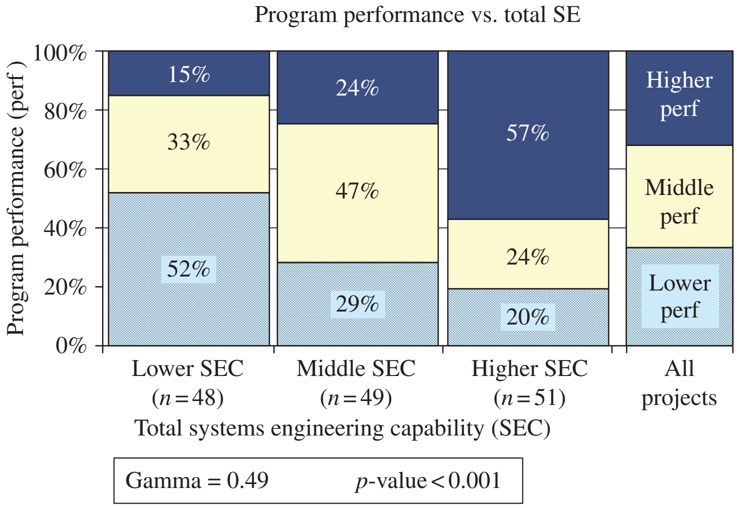

FIGURE 2.6 Project performance versus SE capability. From Elm and Goldenson (2012). Reprinted with permission from Joseph Elm. All other rights reserved.

2.8.2 SE ROI

A quantitative research project was completed by Eric Honour and the University of South Australia to quantify the ROI of SE activities (Honour, 2013). This project gathered data on 43 survey points and from 48 detailed interviews, each point representing the total results of a single system development project. Projects had a wide variety of domains, sizes, and success levels. Figure 2.7 compares the total SE effort with cost compliance (top figure) and schedule performance (bottom figure). Both graphs show that SE effort has a significant, quantifiable effect on program success, with correlation factors as high as 80%. In both graphs, increasing the percentage of SE within the project results in better success up to an optimum level, above which additional SE effort results in poorer performance.

The research results showed that the optimum level of SE effort for a normalized program is 14% of the total program cost. In contrast, the median SE effort of the actual interviewed programs was only 7%, showing that programs typically operate at about half the optimum level of SE effort. The optimum level for any program can also be predicted by adjusting for the program characteristics, with levels ranging from 8 to 19% of the total program cost.

The ROI of adding additional SE activities to a project is shown in Table 2.4, and it varies depending on the level of SE activities already in place. If the project is using no SE activities, then adding SE carries a 7:1 ROI; for each cost unit of additional SE, the project total cost will reduce by 7 cost units. At the median level of the programs interviewed, additional SE effort carries a 3.5:1 ROI.

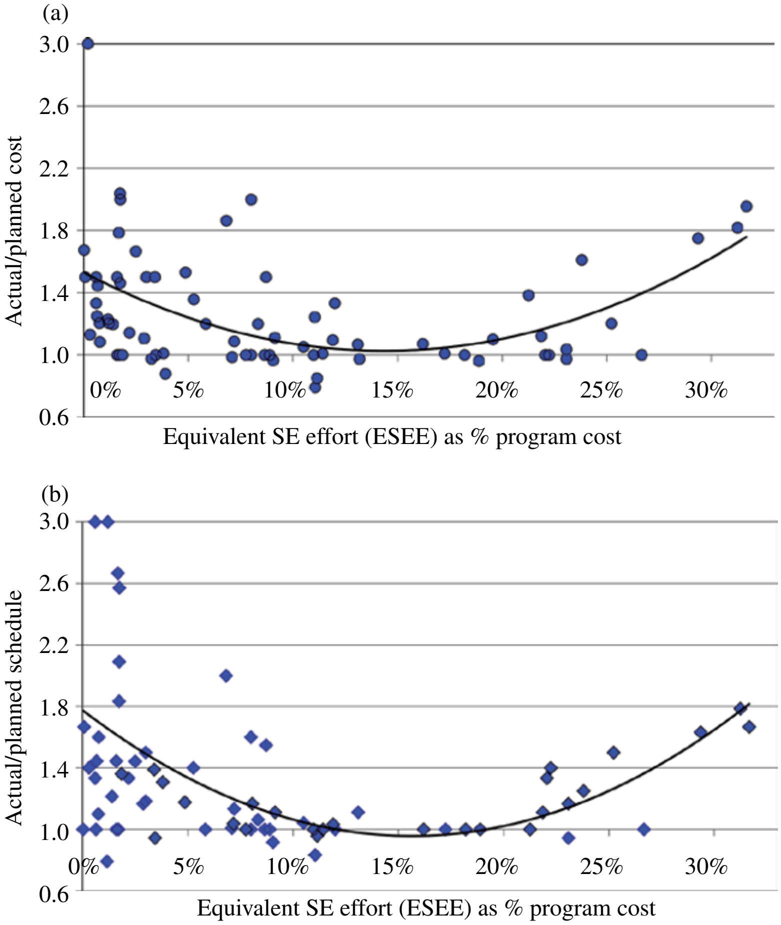

FIGURE 2.7 Cost (a) and schedule (b) overruns correlated with SE effort. From Honour (2013). Reprinted with permission from Eric Honour. All other rights reserved.