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

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

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

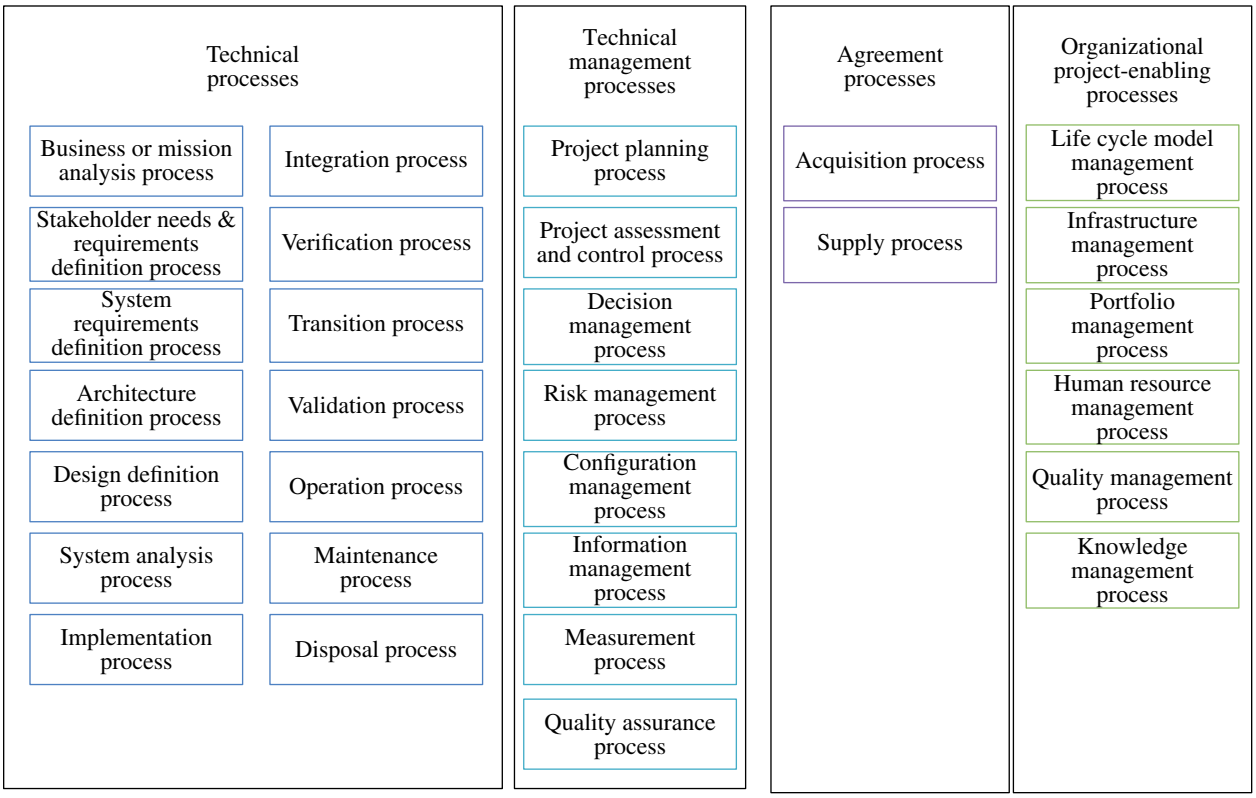


FIGURE 1.1 System life cycle processes per ISO/IEC/IEEE 15288. This figure is excerpted from ISO/IEC/IEEE 15288:2015, Figure 4 on page 17, with permission from the ANSI on behalf of the ISO. © ISO 2015. All rights reserved.

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

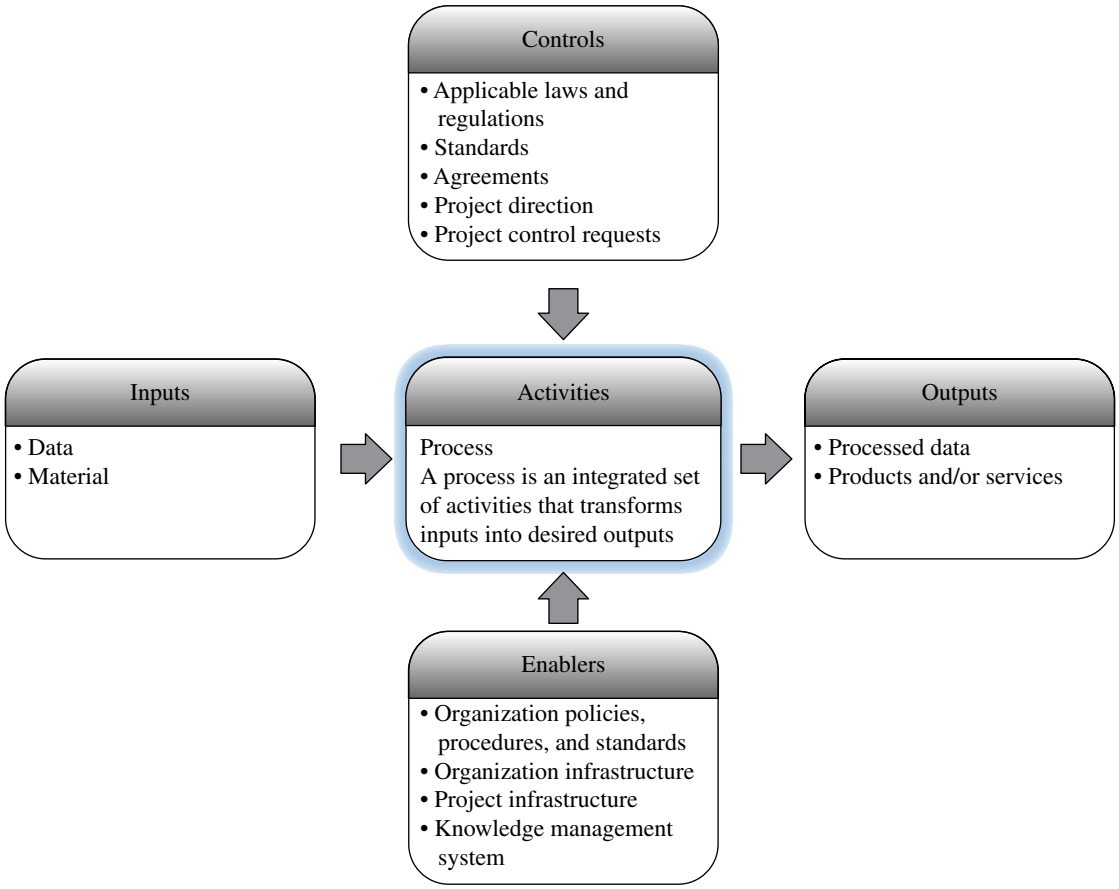


FIGURE 1.2 Sample of IPO diagram for SE processes. INCOSE SEH original figure created by Shortell and Walden. Usage per the INCOSE Notices page. All other rights reserved.

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.

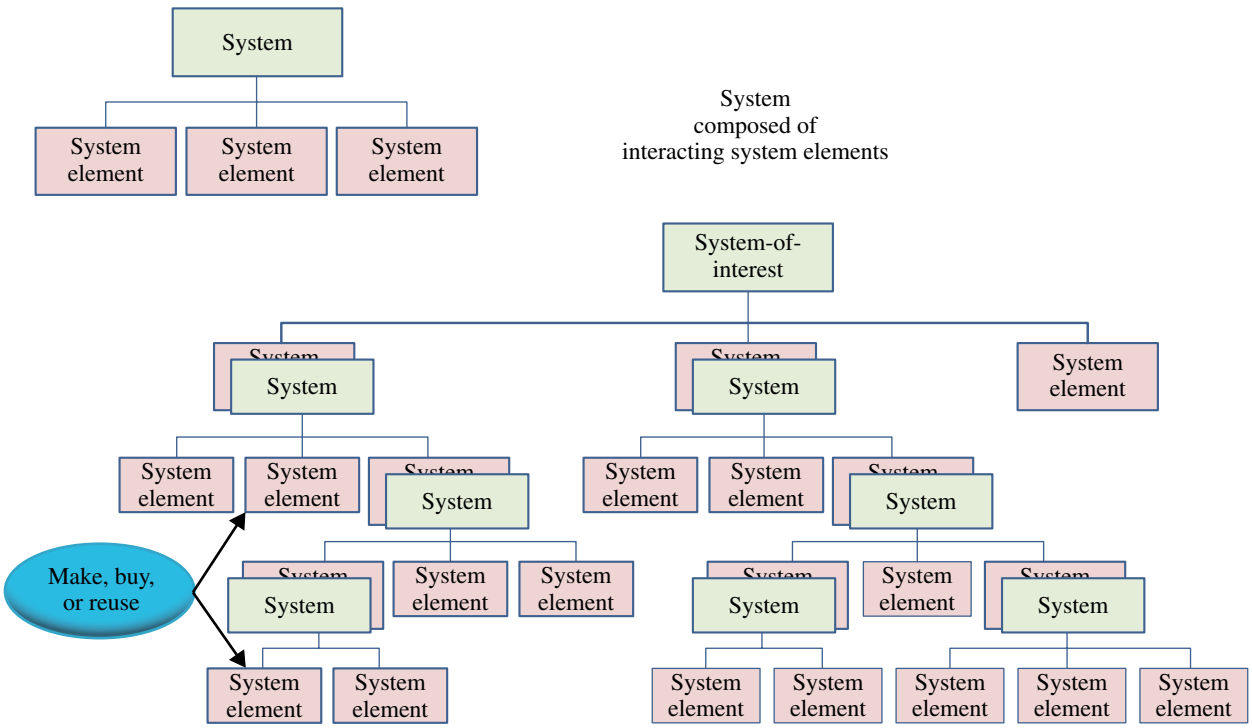


FIGURE 2.1 Hierarchy within a system. This figure is adapted from ISO/IEC/IEEE 15288:2015, Figure 1 on page 11 and Figure 2 on page 12, with permission from the ANSI on behalf of the ISO. © ISO 2015. All rights reserved.

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,

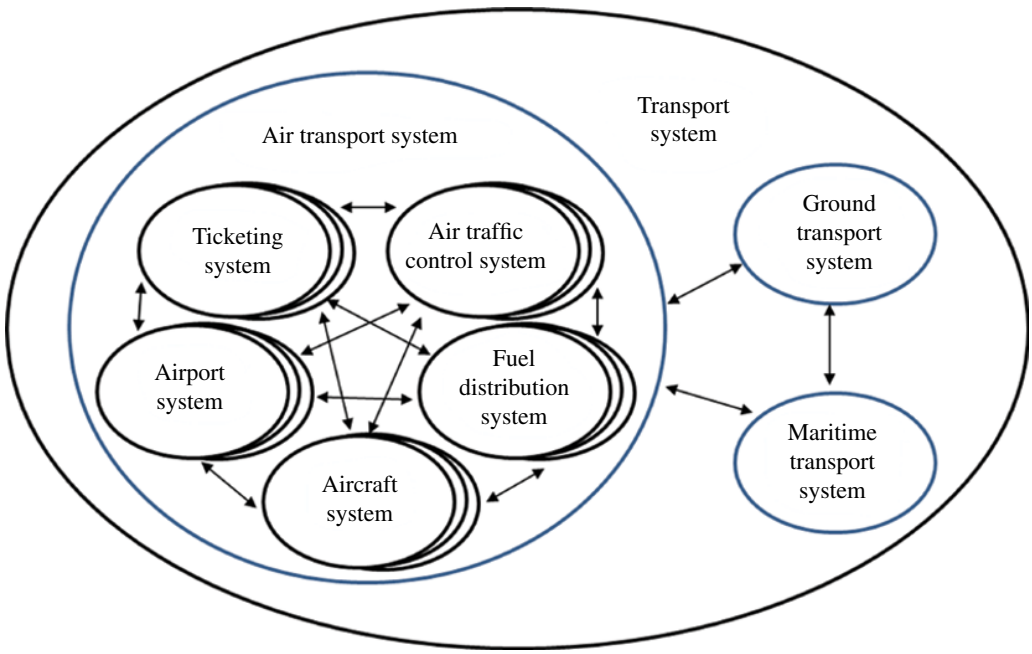


FIGURE 2.2 Example of the systems and systems of systems within a transport system of systems. Reprinted with permission from Judith Dahmann. All other rights reserved.

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

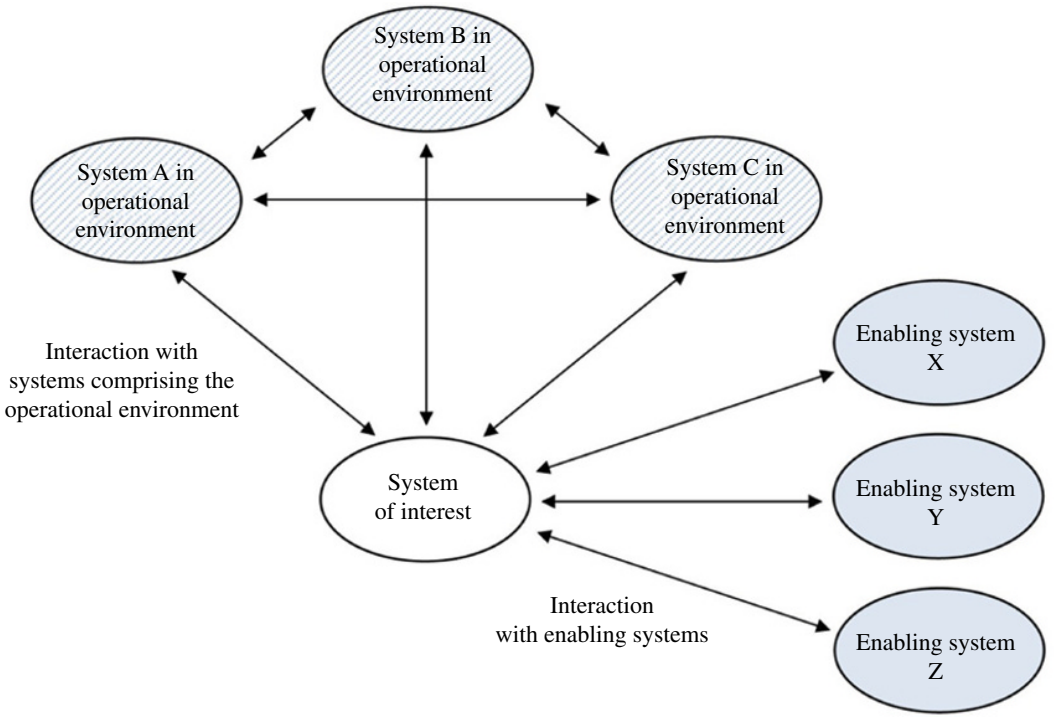


FIGURE 2.3 System of interest, its operational environment, and its enabling systems. This figure is excerpted from ISO/IEC/IEEE 15288:2015, Figure 3 on page 13, with permission from the ANSI on behalf of the ISO. © ISO 2015. All rights reserved.

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

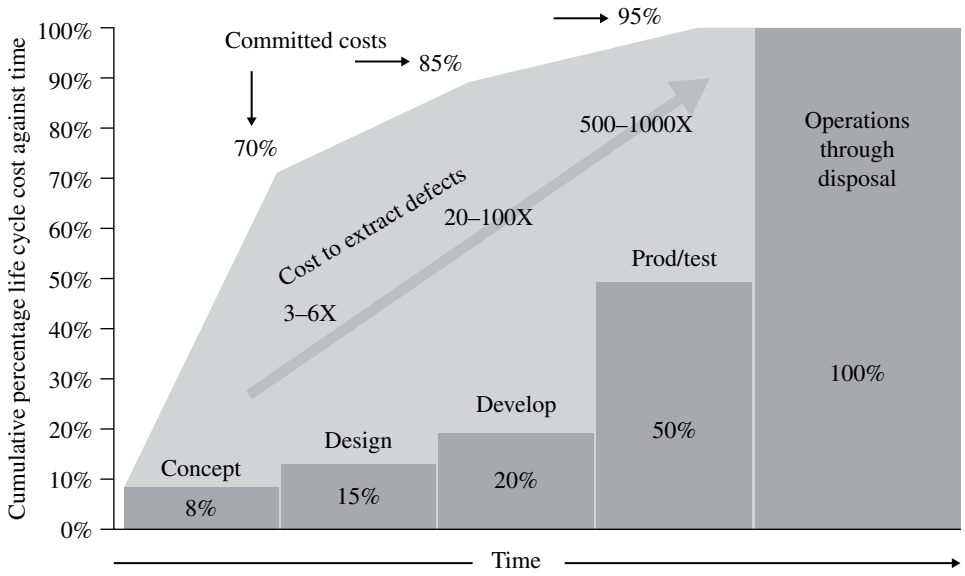


FIGURE 2.4 Committed life cycle cost against time. Reprinted with permission from DAU. All other rights reserved.

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%.

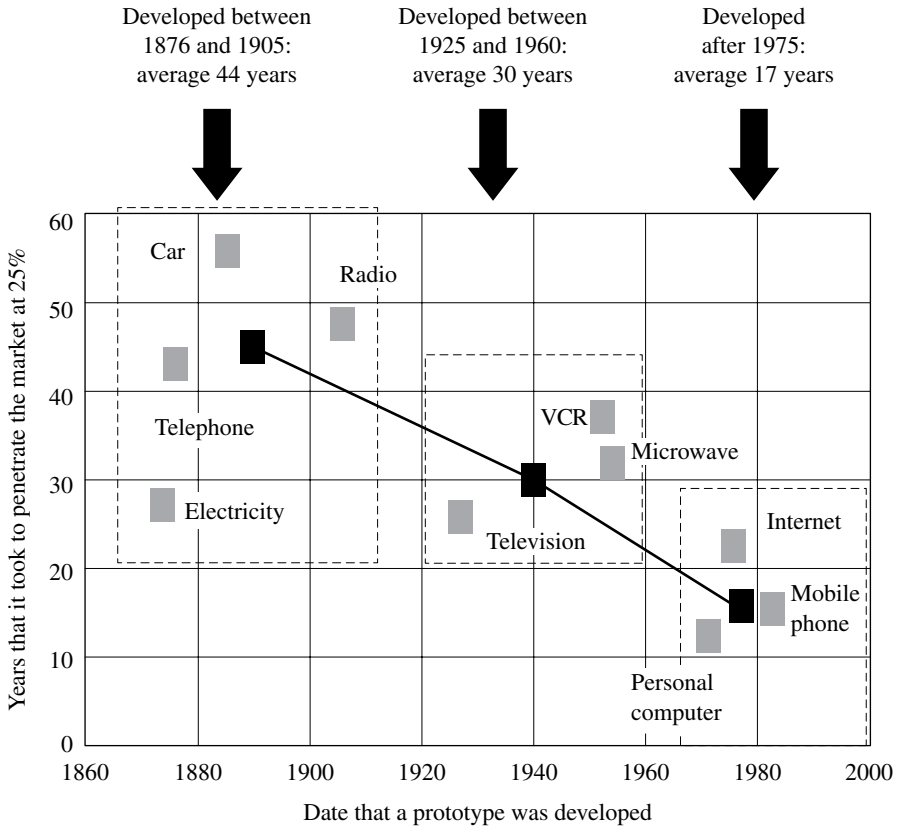


FIGURE 2.5 Technology acceleration over the past 140 years. INCOSE SEH original figure created by Michael Krueger. Usage per the INCOSE Notices page. All other rights reserved.

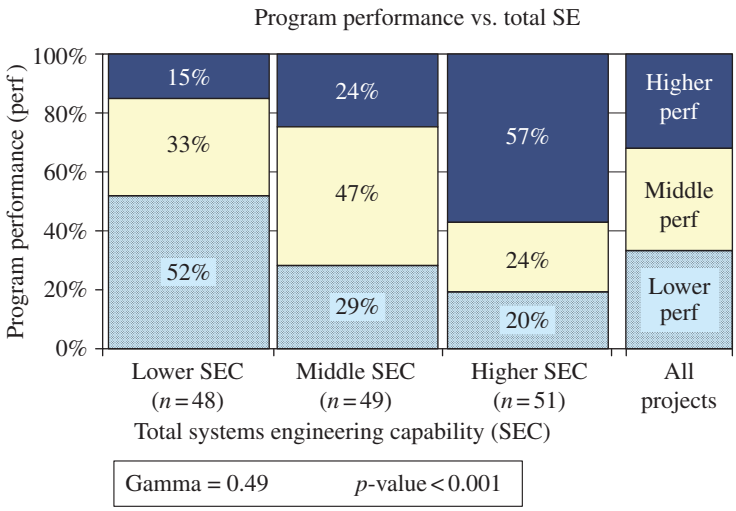


FIGURE 2.6 Project performance versus SE capability. From Elm and Goldenson (2012). Reprinted with permission from Joseph Elm. All other rights reserved.

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.

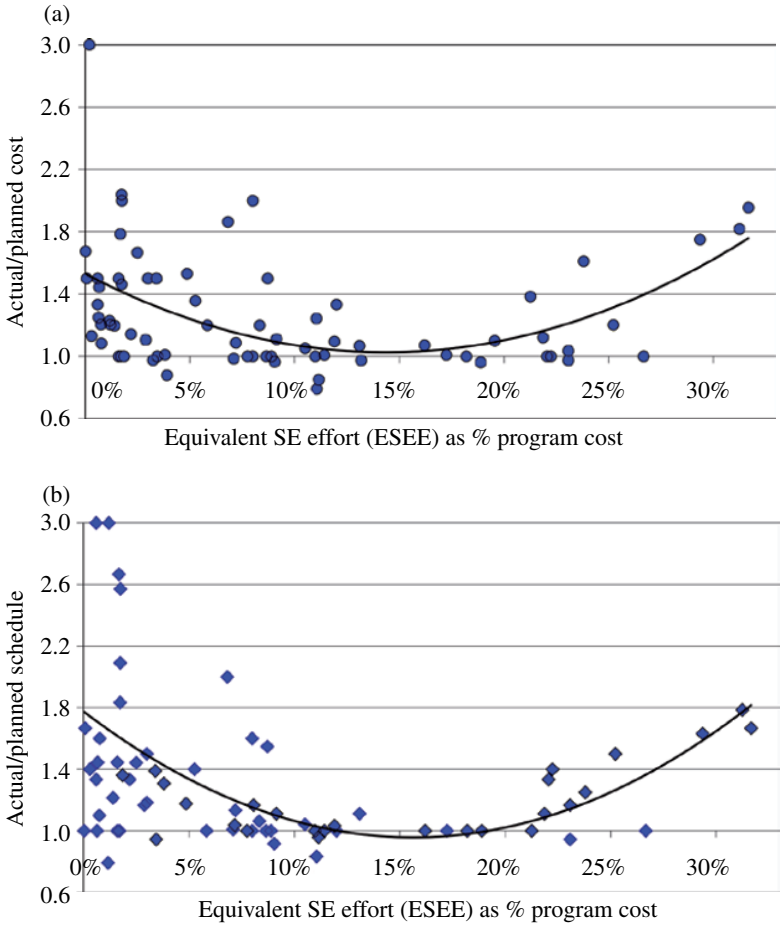


FIGURE 2.7 Cost (a) and schedule (b) overruns correlated with SE effort. From Honour (2013). Reprinted with permission from Eric Honour. All other rights reserved.

TABLE 2.4 SE return on investment

Current SE effort (% of program cost)	Average cost overrun (%)	ROI for additional SE effort (cost reduction $ per $ SE added)
0	53	7.0
5	24	4.6
7.2 (median of all programs)	15	3.5
10	7	2.1
15	3	–0.3
20	10	–2.8

From Honour (2013). Reprinted with permission from Eric Honour. All other rights reserved.

2.9 SYSTEMS SCIENCE AND SYSTEMS THINKING

This section summarizes the nature of systems science and systems thinking. It also describes how they relate to SE.

2.9.1 Systems Science

Systems science brings together research into all aspects of systems with the goal of identifying, exploring, and understanding patterns of complexity that cross disciplinary fields and areas of application. It seeks to develop interdisciplinary foundations that can form the basis of theories applicable to all types of systems (e.g., in nature, society, and engineering) independent of element type or application.

Additionally, systems science can help to provide a common language and intellectual foundation for SE and make practical system concepts, principles, patterns, and tools accessible to practitioners of the "systems approach." An integrated systems approach for solving complex problems needs to combine elements of systems science, systems thinking, and SE. As such, systems science can serve as the foundation for a metadiscipline that unifies the traditional scientific specializations.

The information in this section is extracted from the systems science article in Part 2 of the SEBoK (SEBoK, 2014). Figure 2.8 illustrates the relationships between systems science, systems thinking, and general systems approach as applied to engineered systems.

Systems science is an integrative discipline that brings together ideas from a wide range of sources sharing in a common systems theme. Some fundamental concepts now used in systems science have been present in other disciplines for many centuries, while equally fundamental concepts have independently emerged as recently as 40 years ago (Flood and Carson, 1993).

Systems science is both the "science of systems" and the "systems approach to science," covering theories and methods that contrast with those of other sciences, which are generally reductionist in nature. Where it is appropriate, the reductionist approach has been very successful in using the methods of separating and isolating in search of simplicity. However, where those methods are not appropriate, systems science relies on connecting and contextualizing to identify patterns of organized complexity.

Questions about the nature of systems, organization, and complexity are not specific to the modern age. As John Warfield (2006) put it,

> Virtually every important concept that backs up the key ideas emergent in systems literature is found in ancient literature and in the centuries that follow.

It was not until the middle of the twentieth century, however, that there was a growing sense of a need for, and possibility of, a scientific approach to the problems of organization and complexity in a "science of systems" per se.

Biologist Ludwig von Bertalanffy was one of the first to argue for and develop a broadly applicable scientific research approach based on *open system theory* (Bertalanffy, 1950). He explained the scientific need for systems research in terms of the limitations of analytical procedures in science. These limitations are based on the idea that an entity can be resolved into and reconstituted from its parts, either materially or conceptually:

> This is the basic principle of "classical" science, which can be circumscribed in different ways: resolution into isolable causal trains or seeking for "atomic" units in the various fields of science, etc.

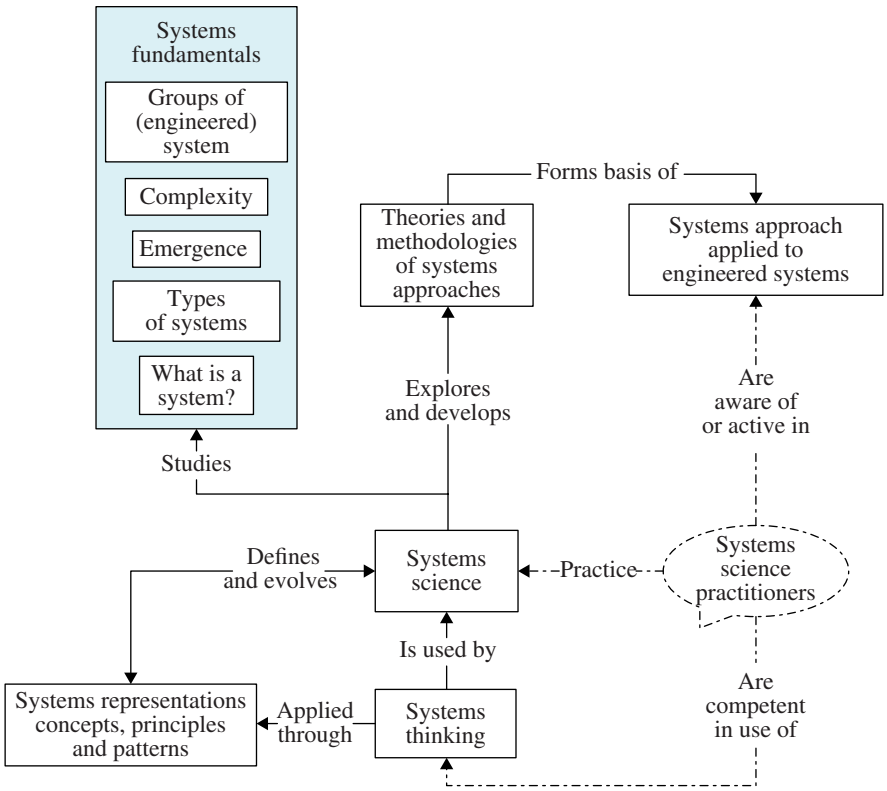


FIGURE 2.8 Systems science in context. From SEBoK (2014). Reprinted with permission from the BKCASE Editorial Board. All other rights reserved.

Research in systems science attempts to compensate for the inherent limitations of classical science, most notably the lack of ways to deal with emergence. Systems science has developed—and continues to develop—hand in hand with practice, each maturing and learning from the other. Various efforts have taken on complementary or overlapping issues of the new "systems approach" as progress has been made over time:

- Cybernetics (Ashby, 1956; Wiener, 1948)
- Open system and general system theory (Bertalanffy, 1950, 1968; Flood, 1999)
- Operations research (Churchman et al., 1950)
- Hard and soft systems thinking (Checkland, 1998; Lewin, 1958)
- Organizational cybernetics (Beer, 1959; Flood, 1999)
- Critical systems thinking (Jackson, 1989)
- System dynamics (Forrester, 1961; Senge, 1990)
- SE (Hall, 1962)
- System analysis (Ryan, 2008)
- Service science and service SE (Katzan, 2008)

A broader discussion of contrasts between analytical procedures and integrative system concepts is provided in *Model-Oriented Systems Engineering Science* (Hybertson, 2009) from the perspective of a traditional versus the complex SE views of systems.

2.9.2 Systems Thinking

As a systems engineer, it is vital to develop knowledge and skills that can be utilized in performing a deep analysis of problem or opportunity situations for which system responses are required. As noted earlier, systems science has contributed to the development of such knowledge. However, during the twentieth century, a number of

approaches to performing deep analysis have arisen under the title of "systems thinking." While it is difficult to put a precise boundary around systems thinking and differentiate it from systems science, many systems thinking methods and tools have become popular and have been successfully utilized in multidisciplinary contexts.

2.9.2.1 System Dynamics Jay Forrester of MIT developed the DYNAMO simulation language and observed that a common means of analyzing complex systems could be used in multiple disciplines (1961). Several students of Forrester refined these ideas into methodologies and tools that have provided a useful basis for analysis. Peter Senge, with his popular book *The Fifth Discipline* (1990), established systems thinking as a discipline. He also developed the link, loop, and delay language as a means of graphically representing system dynamics. Based upon two primary loops (growth and limit), a number of so-called archetypes have been developed to describe a variety of situations. Another student, Barry Richmond, further developed archetypes by adding flow mechanisms in the simulation languages STELLA and iTHINK, which are commercially available.

2.9.2.2 Soft Systems and Action Research Peter Checkland (1975) observed that the use of classical engineering approaches to complex problems falls down since there are many soft factors (attitudes, practices, procedures, etc.) that affect systems. He also observed that the path to improvement must come through the development and analysis of alternative models. Based upon analysis, discussion, and dialogue, a course of action is planned and executed, and the results observed as feedback for further analysis. John Boardman has been inspired by Checkland's work and has developed a version of the soft systems approach supported by a tool called Systemigrams (Boardman and Sauser, 2008).

2.9.2.3 Discovering Patterns Central to systems thinking is the discovery of patterns. A pattern is a representation of similarities in a set of problems, solutions, or systems. Systems thinking captures and exploits what is common in a set of problems and corresponding solutions in the form of patterns of various types, as noted in the archetypes described previously.

Systems engineers use the general information provided by patterns to understand a specific system problem and to develop a specific system solution. Examples include, but are not limited to, the following:

- Software design patterns, such as adapter
- System architecture patterns, such as layered architecture and single sign-on
- Community or urban design patterns, such as ring road, pedestrian street, and food court
- Security and safety patterns, such as fault-tolerant design and role-based access control
- Interaction patterns, such as publish–subscribe
- Domain-specific patterns, such as the suspension bridge pattern

Pattern categories include domain taxonomies, standards, templates, architecture styles, reference architectures, product lines, abstract data types, and classes in class hierarchies.

Another important category of patterns is associated with complex, counterintuitive systems. An example is "shifting the burden." When a problem appears in a complex system, the intuitive response is to apply a quick short-term fix, rather than take the time to develop a long-term solution. The problem with that approach is that there is often a trade-off between the short-term fix and the long-term solution, such that the initial success of the quick fix reduces the chances the real solution will be developed. This pattern appears in the typical short-term/long-term priority struggle between the systems engineer, project manager, discipline engineering communities, and other stakeholders within a project. Shifting the burden is one of a set of "system archetypes," a class of patterns (sometimes called patterns of failure or antipatterns) that illustrate the underlying systems pathology behind many of the barriers to effective SE. Awareness and understanding of these patterns are important for people trying to embed the systems approach in their organization and take positive action to counter these challenges. Additional examples and references are given in the article "Patterns of Systems Thinking" in (SEBoK, 2014).

2.9.2.4 Habits of a Systems Thinker While there are several additional aspects of systems thinking introduced by multiple contributors, the following list

summarizes essential properties of a systems thinker (Waters Foundation, 2013):

- Seeks to understand the big picture
- Observes how elements within the system change over time, generating patterns and trends
- Recognizes that a systems' structure (elements and their interactions) generates behavior
- Identifies the circular nature of complex cause-and-effect relationships
- Surfaces and tests assumptions
- Changes perspective to increase understanding
- Considers an issue fully and resists the urge to come to a quick conclusion
- Considers how mental models affect current reality and the future
- Uses understanding of system structure to identify possible leverage actions
- Considers both short- and long-term consequences of actions
- Finds where unintended consequences emerge
- Recognizes the impact of time delays when exploring cause-and-effect relationships
- Checks results and changes actions if needed: "successive approximation"

People who think and act in a "systems way" are essential to the success of both research and practice (Lawson, 2010). Successful systems research will not only apply systems thinking to the topic being researched but should also consider a systems thinking approach to the way the research is planned and conducted. It would also be of benefit to have people involved in research who have, at a minimum, an awareness of systems practice and ideally are involved in practical applications of the theories they develop.

For a more thorough description of the "discipline" of systems thinking, refer to Part 2 of SEBoK (2014) as well as the popular website *Systems Thinking World* (Bellinger, 2013).

2.9.3 Considerations for Systems Engineers

Sillitto (2012) provides a useful digest of concepts from systems science and systems thinking, organized so as to be immediately useful to the SE practitioner. Properties that are generally true of the sort of systems that systems engineers find themselves involved with are as follows:

1. A system exists within a wider "context" or environment.
   - The context includes an "operational environment," a "threat environment," and a "resource environment" (Hitchens, 2003).
   - The context may also contain collaborating and competing systems.
2. A system is made up of parts that interact with each other and the wider context.
   - The parts may be any or all of hardware, software, information, services, people, organizations, processes, services, etc.
   - Interactions may include exchange of information, energy, and resources.
3. A system has system-level properties ("emergent properties") that are properties of the whole system not attributable to individual parts.
   - Emergent properties depend on the structure (parts and relationships between them) of the whole system and on its interactions with the environment.
   - This structure determines the interactions between functions, behavior, and performance of the parts and interaction of the system with the environment—in ways both intended and unintended.
4. A system has the following:
   - A life cycle
   - Function, which can be characterized following Hitchens as "operate – maintain viability – manage resources" or as "observe – orient – decide – act" (Hitchens, 2003)
   - Structure, including the following:
     - A boundary, which may be static or dynamic and physical or conceptual
     - A set of parts
     - The set of relationships and potential interactions between the parts of the system and across the boundary (interfaces)
   - Behavior, including state change and exchange of information, energy, and resources

- Performance characteristics associated with function and behavior in given environmental conditions and system states

5. A system both changes and adapts to its environment when it is deployed (inserted into its environment).
6. Systems contain multiple feedback loops with variable time constants, so that cause-and-effect relationships may not be immediately obvious or easy to determine.

Additional properties that are "sometimes true" that systems engineers are likely to encounter are as follows:

1. A system may exist independent of human intentionality.
2. A system may be part of one or several wider "containing systems."
3. A system may be self-sustaining, self-organizing, and dynamically evolving (such systems include "complex adaptive systems").
4. A system may offer "affordances"—features that provide the potential for interaction by "affording the ability to do something" (Norman, 1990):
   - Affordances will lead to interactions whether planned or not. For example, the affordance of a runway to let planes land and take off also leads to a possibly unintended affordance to drive vehicles across it, which may get in the way of planes, leading to undesirable emergent whole-system behavior.
5. A system may be:
   - Clearly bounded and distinct from its context (the solar system, Earth, planes, trains, automobiles, ships, people)
   - Closely coupled with or embedded in its context (a bridge, a town, a runway, the human cardiovascular system, the Internet)
   - Of fluid and dynamic makeup (a club, team, social group, ecosystem, flock of geese, and again the Internet)
6. A system may be technical (requiring one or multiple disciplines to design), social, ecological, environmental, or a compound of any or all of these.

These lists help explain the need for systems engineers to think about the SOI in its wider context, so as to ensure they understand both the properties important to the system's purpose and those that might give rise to undesirable unintended consequences.

2.10 SYSTEMS ENGINEERING LEADERSHIP

Many of the processes in this handbook rightly discuss management (e.g., decision management, risk management, portfolio management, knowledge management), and these are all important aspects of the SE process. However, leadership is an equally important topic to systems engineers. In a paper entitled "What Leaders Really Do," J. P. Kotter (2001) states that "leadership is different from management, but not for the reason most people think." Kotter defines the key differences between leaders and managers as:

- Coping with change versus coping with complexity
- Setting a direction versus planning and budgeting
- Aligning people versus organizing and staffing
- Motivating people versus controlling and problem solving

A quote often attributed to Peter Drucker is: "Managers do things right. Leaders do the right things." Compare this to the informal definitions of the SE verification and validation processes: "Verification ensures you built the system right. Validation ensures you built the right system." Both verification and validation are important for the development of systems. Likewise, both management and leadership are important for systems engineers and their teams. Different phases of a project demand emphasis on different aspects of leadership.

Aspects of leadership that are particularly relevant for systems engineers include:

- Thinking strategically and looking at the long-term implications of decisions and actions to set vision and course
- Seeing the "big picture"
- Casting or capturing the vision for the organization and communicating it (the systems engineer may be working in support of the identified leader, or sometimes, it isn't the leader's prerogative to "cast" the vision)

- Defining the journey from the "as is" of today to the "to be" of tomorrow
- Turning ambiguous problem statements into clear, precise solution challenges for the team
- Working with the stakeholders (including customers), representing their points of view to the team and the team's point of view to them
- Maximizing customer value by ensuring a direct tie of all engineering effort to the customer business or mission needs
- Establishing an environment for harmonious teams while working to leverage the potential benefits of diversity (including bridging cultural and communication differences in multidisciplinary teams)
- Challenging conventional wisdom at all levels
- Managing conflicts and facilitating healthy conflict around ideas and alternatives
- Facilitating decision making
- Demanding and enabling excellence

Leadership is both an opportunity and a critical responsibility of the systems engineer. The SE leader must have a systems view that takes into account the context, boundaries, interrelationships, and scope. They drive better solutions through the holistic understanding of the problem and its context and environment. SE leaders highlight the risks of unintended consequences in a proactive manner. Many times, SE leaders need to move the conversation from "price and cost" to "value and ROI." SE leaders need to serve as a model for the adaptability, agility, and resilience that is sought in both the systems and the teams that develop them. After all, SE leaders have the "best seat in the house" for seeing the broader systems view (Long, 2013).

2.11 SYSTEMS ENGINEERING PROFESSIONAL DEVELOPMENT

To efficiently and cost-effectively deliver differentiated products to the market, an organization needs to know what gaps exist in their overall capability. An individual needs to know what skills would enable them to be more effective, to develop those skills, and to have a standard to demonstrate and communicate their skill levels. The overall system for optimizing SE delivery is shown in Figure 2.9.

Most development, but especially for SE, is achieved through experience and on-the-job training. Typically, 70% of development is achieved through experience, 20% through mentoring, and only 10% through training (Lombardo and Eichinger, 1996). Training creates an

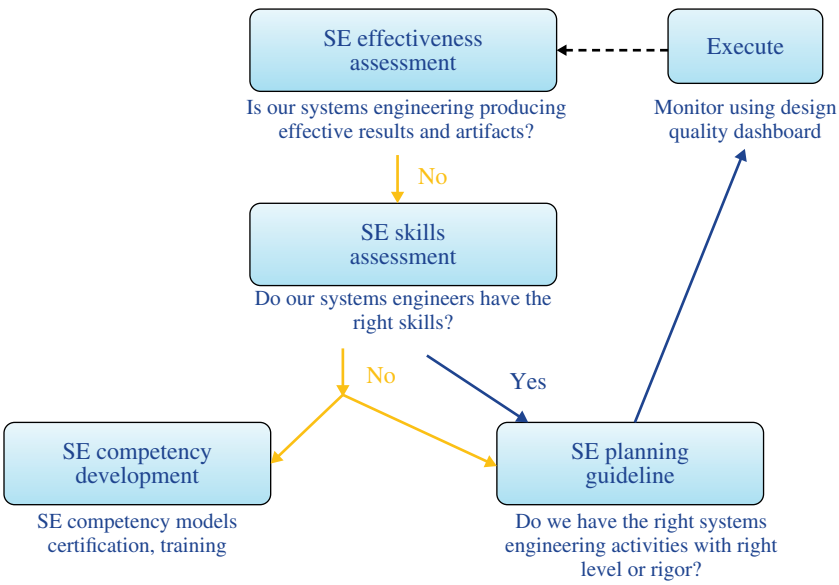


FIGURE 2.9 SE optimization system. Reprinted with permission from Chris Unger. All other rights reserved.

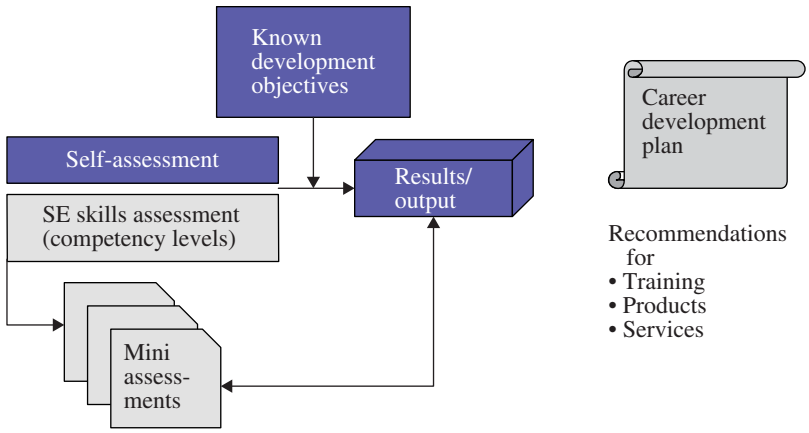


FIGURE 2.10 Professional development system. Reprinted with permission from Chris Unger. All other rights reserved.

understanding of basic concepts, while the mentor helps developing systems engineers absorb the appropriate lessons from practical experience. The model in Figure 2.10 shows how SE development can work for an individual (either pursuing certification or in a development discussion with their manager).

INCOSE has developed an SE Competency Framework based on the work of the INCOSE United Kingdom Chapter. Use of the framework can enable employees to analyze their skills and evaluate the need for training, coaching, or new job assignments to fill any gaps found in the assessment (INCOSE UK, 2010). The framework defines three classes of competencies and aligns relevant competencies to each class. The classes are systems thinking, holistic life cycle view, and SE management. The framework has defined four skill levels: awareness, supervised practitioner, practitioner, and expert. The framework is tailorable to the needs of the organization.

2.11.1 SE Professional Ethics

There will always be pressure to cut corners to deliver programs faster or at lower costs, especially for a profession such as SE. As stated in the INCOSE Code of Ethics,

> The practice of Systems Engineering can result in significant social and environmental benefits, but only if unintended and undesired effects are considered and mitigated.

Part of the role of the systems engineer as a leader and professional is knowing when unacceptable risks or trade-offs are being made, knowing how to influence key stakeholders, and having the courage to stand up for the customers, community, and profession when necessary. The INCOSE Code of Ethics contains sections on "Fundamental Principles," "Fundamental Duties to Society and Public Infrastructure," and "Rules of Practice" to help the SE professional in practical applications of ethics to their work and daily lives (INCOSE, 2006).

2.11.2 Professional Certification

INCOSE offers a multilevel SE professional certification program to provide a formal method for recognizing the knowledge and experience of systems engineers throughout the world. Three certification levels are available through INCOSE:

- *Associate Systems Engineering Professional (ASEP)*—Applicants are required to successfully complete a knowledge examination.
- *Certified Systems Engineering Professional (CSEP)*—Requires a minimum of 5 years of practical SE experience, a technical degree (additional years of SE experience can be used in lieu of a technical degree), three professional references covering the candidate's cumulative years of experience, and successful completion of a knowledge examination.

- *Expert Systems Engineering Professional (ESEP)*—Requires a minimum of 25 years of practical SE experience, a minimum of 5 years of professional leadership credits, a technical degree (additional years of experience can be used in lieu of a technical degree), and three professional references covering at least the most recent 10 years of experience. The ESEP award is based on panel review and approval.

Additional details on the requirements for SE professional certification are available on the INCOSE website at http://www.incose.org/.

3

GENERIC LIFE CYCLE STAGES

3.1 INTRODUCTION

Every man-made system has a life cycle, even if it is not formally defined. A life cycle can be defined as the series of stages through which something (a system or manufactured product) passes. In keeping with increased awareness of environmental issues, the life cycle for any system of interest (SOI) must encompass not only the development, production, utilization, and support stages but also provide early focus on the retirement stage when decommissioning and disposal of the system will occur. The needs for each of the subsequent stages must be considered during the earlier stages, especially during the concept and development stages, in order to make the appropriate trades and decisions to accommodate the needs of later stages in an affordable and effective manner.

The role of the systems engineer encompasses the entire life cycle for the SOI. Systems engineers orchestrate the development of a solution from requirements definition through design, build, integration, verification, operations, and ultimately system retirement by assuring that domain experts are properly involved, that all advantageous opportunities are pursued, and that all significant risks are identified and mitigated. The systems engineer works closely with the project manager in tailoring the generic life cycle, including key decision gates, to meet the needs of their specific project. Per ISO/IEC/IEEE 15288,

> 5.4.2—Life cycles vary according to the nature, purpose, use and prevailing circumstances of the system …

The purpose in defining the system life cycle is to establish a framework for meeting the stakeholders' needs in an orderly and efficient manner for the whole life cycle. This is usually done by defining life cycle stages and using decision gates to determine readiness to move from one stage to the next. Skipping stages and eliminating "time-consuming" decision gates can greatly increase the risks (cost, schedule, and performance) and may adversely affect the technical development as well by reducing the level of the SE effort, as discussed in Section 2.8.

Systems engineering (SE) tasks are usually concentrated at the beginning of the life cycle, but both industry and government organizations recognize the need for SE throughout the systems' life span, often to modify or change a system product or service after it enters production or is placed in operation. Consequently, SE is an important part of all life cycle stages. During the utilization and support stages, for example, SE executes

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.

performance analysis, interface monitoring, failure analysis, logistics analysis, tracking, management, etc. that is essential to ongoing operation and support of the system.

3.2 LIFE CYCLE CHARACTERISTICS

3.2.1 Three Aspects of the Life Cycle

Every system life cycle consists of multiple aspects, including the business aspect (business case), the budget aspect (funding), and the technical aspect (product). The systems engineer creates technical solutions that are consistent with the business case and the funding constraints. System integrity requires that these three aspects are in balance and given equal emphasis at all decision gate reviews. For example, when Motorola's Iridium project started in the late 1980s, the concept of satellite-based mobile phones was a breakthrough and would clearly capture a significant market share. Over the next dozen years, the technical reviews ensured a highly successful technical solution. In fact, in the first decade of the twenty-first century, the Iridium project is proving to be a good business venture for all except for the original team who had to sell all the assets—at about 2% of their investment—through the bankruptcy court. The original team lost sight of the competition and changing consumer patterns that substantially altered the original business case. Figure 3.1 highlights two critical parameters that engineers sometimes lose sight of: time to breakeven (indicated by the circle) and return on investment (indicated by the lower curve).

3.2.2 Decision Gates

Decision gates, also known as control gates, are often called "milestones" or "reviews." A decision gate is an approval event in the project cycle, sufficiently important to be defined and included in the schedule by the project manager, executive management, or the customer. Entry and exit criteria are established for each gate at the time they are included into the project management baseline. Decision gates ensure that new activities are not pursued until the previously scheduled activities, on which new activities depend, are satisfactorily completed and placed under configuration control. Proceeding beyond the decision gate before the project is ready entails risk. The project manager may decide to accept that risk, as is done, for instance, with long-lead item procurement.

All decision gates are both reviews and milestones; however, not all reviews and milestones are decision gates. Decision gates address the following questions:

- Does the project deliverable still satisfy the business case?
- Is it affordable?
- Can it be delivered when needed?

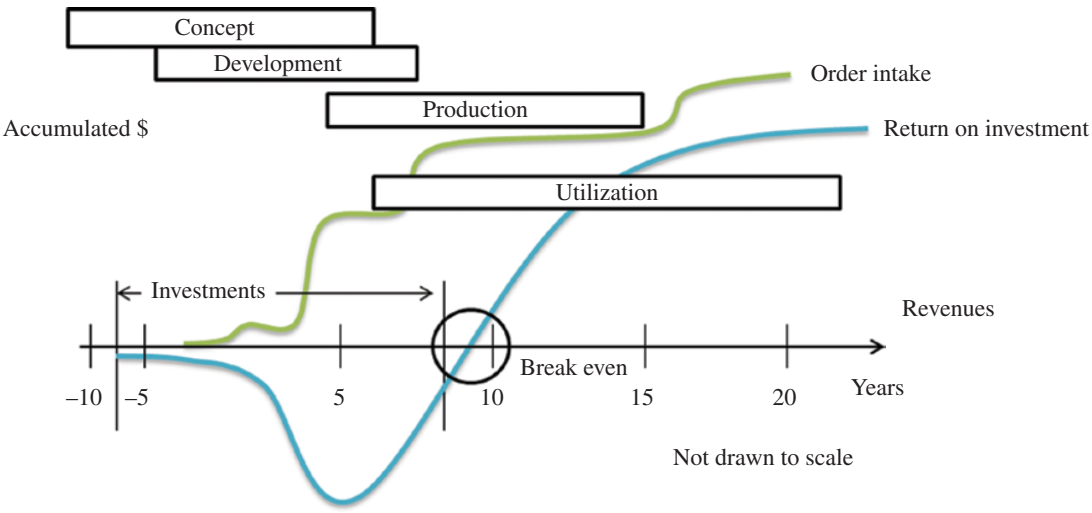


FIGURE 3.1 Generic business life cycle. From Stoewer (2005). Reprinted with permission from Heinz Stoewer. All other rights reserved.

Decision gates represent major decision points in the system life cycle. The primary objectives of decision gates are to:

- Ensure that the elaboration of the business and technical baselines are acceptable and will lead to satisfactory verification and validation (V&V)
- Ensure that the next step is achievable and the risk of proceeding is acceptable
- Continue to foster buyer and seller teamwork
- Synchronize project activities

There are at least two decision gates in any project: authority to proceed and final acceptance of the project deliverable. The project team needs to decide which life cycle stages are appropriate for their project and which decision gates beyond the basic two are needed. Each decision gate must have a beneficial purpose; "pro forma" reviews waste everyone's time.

Even in agile development (see Section 9.9), frequent interaction with stakeholders may minimize, but not eliminate, the need for decision gates. The consequences of conducting a superficial review, omitting a critical discipline, or skipping a decision gate are usually long term and costly.

The project business case issues of market demand, affordability, and realistic schedules are important decision criteria influencing concept selection, and they should be updated and evaluated at every decision gate. Inadequate checks along the way can set up subsequent failures—usually a major factor in cost overruns and delays. At each gate, the decision options are typically similar to the following:

- *Acceptable*: Proceed with the next stage of the project.
- *Acceptable with reservations*: Proceed and respond to action items.
- *Unacceptable: Do not proceed*—continue this stage and repeat the review when ready.
- *Unacceptable: Return to a preceding stage.*
- *Unacceptable: Put a hold on project activity.*
- *Unsalvageable: Terminate the project.*

Decision gate descriptions should identify the:

- Purpose and scope of the decision gate
- Entry and exit criteria
- Host and chairperson
- Attendees
- Location
- Agenda and how the decision gate is to be conducted
- Evidence to be evaluated
- Actions resulting from the decision gate
- Method of closing the review, including timing for resolution of open action items

Decision gate approval follows review by qualified experts and involved stakeholders and is based on hard evidence of compliance to the criteria of the review. Balancing the formality and frequency of decision gates is seen as a critical success factor for all SE process areas. On large or lengthy projects, decisions and their rationale are maintained using an information management process.

Upon successful completion of a decision gate, some artifacts (e.g., documents, models, or other products of a project life cycle stage) have been approved as the basis upon which future work must build. These artifacts are placed under configuration management.

3.3 LIFE CYCLE STAGES

ISO/IEC/IEEE 15288 states:

> 5.4.1—A system progresses through its life cycle as the result of actions, performed and managed by people in organizations, using processes for execution of these actions.

A system "progresses" through a common set of life cycle stages where it is conceived, developed, produced, utilized, supported, and retired. The life cycle model is the framework that helps ensure that the system meets its required functionality throughout its life. For example, to define system requirements and develop system solutions during the concept and development stages, experts from other stages are needed to perform trade-off analyses, help make decisions, and arrive at a balanced solution. This ensures that a system has the necessary attributes as early as possible. It is also essential to have the enabling systems available to perform required stage functions.

Table 3.1 lists six generic life cycle stages (ISO/IEC TR 24748-1, 2010). The purpose of each is briefly

TABLE 3.1 Generic life cycle stages, their purposes, and decision gate options

Life cycle stages	Purpose	Decision gates
Concept	Define problem space 1. Exploratory research 2. Concept selection Characterize solution space Identify stakeholders' needs Explore ideas and technologies Refine stakeholders' needs Explore feasible concepts Propose viable solutions	Decision options • Proceed with next stage • Proceed and respond to action items • Continue this stage • Return to preceding stage • Put a hold on project activity • Terminate project
Development	Define/refine system requirements Create solution description—architecture and design Implement initial system Integrate, verify, and validate system	
Production	Produce systems Inspect and verify	
Utilization	Operate system to satisfy users' needs	
Support	Provide sustained system capability	
Retirement	Store, archive, or dispose of the system	

This table is excerpted from ISO/IEC TR 24748-1 (2010), Table 1 on page 14, with permission from the ANSI on behalf of the ISO. © ISO 2010. All rights reserved.

identified, and the options from decision gate events are indicated. Note that stages can overlap and the utilization and support stages run in parallel. Note also that the outcome possibilities for decision gates are the same for all decision gates. Although the stages in Table 3.1 are listed as independent, nonoverlapping, and serial, the activities constituting these stages can be in practice interdependent, overlapping, and concurrent.

Consequently, a discussion of system life cycle stages does not imply that the project should follow a predetermined set of activities or processes unless they add value toward achieving the final goal. Serial time progression is not inherently part of a life cycle model (stages do not necessarily occur serially one after another in time sequence). One possible example of the "progression" of a system through its life cycle is shown in Figure 3.2. When, in this handbook, reference is made to an earlier, prior, next, subsequent, or later stage, this type of model must be kept in mind to avoid confusion by inferring serial time sequencing. Subsequent chapters of this handbook will define processes and activities to meet the objectives of these life cycle stages. Because of the iterative nature of SE, specific processes are not aligned to individual life cycle stages. Rather, the entire set of SE processes is considered and applied at each stage of life cycle development as appropriate to the scope and complexity of the project.

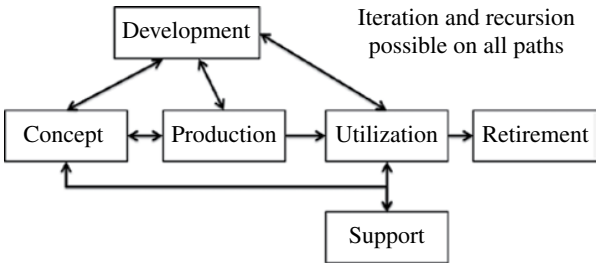
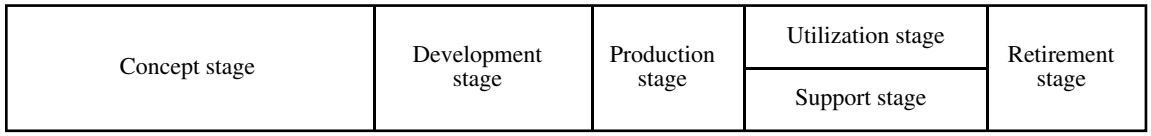


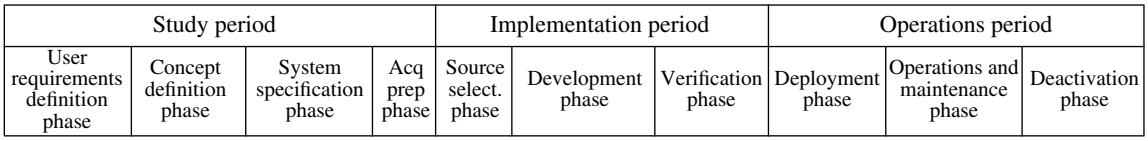
FIGURE 3.2 Life cycle model with some of the possible progressions. This figure is excerpted from ISO/IEC TR 24748-1 (2010), Figure 7 on page 13, with permission from the ANSI on behalf of the ISO. © ISO 2010. All rights reserved.

Figure 3.3 compares the generic life cycle stages to other life cycle viewpoints. For example, the concept stage is aligned with the study period for commercial projects and with the presystem acquisition and the project planning period in the US Departments of Defense and Energy, respectively. Typical decision gates are presented in the bottom line.

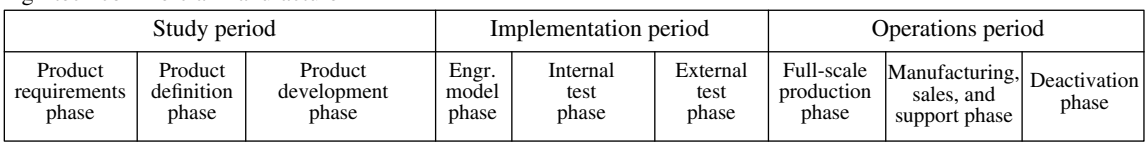
Generic life cycle (ISO/IEC/IEEE 15288:2015)



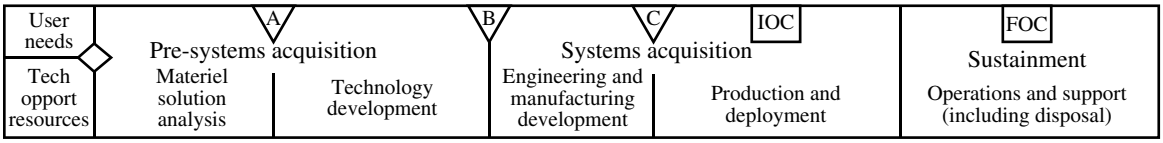
Typical high-tech commercial systems integrator



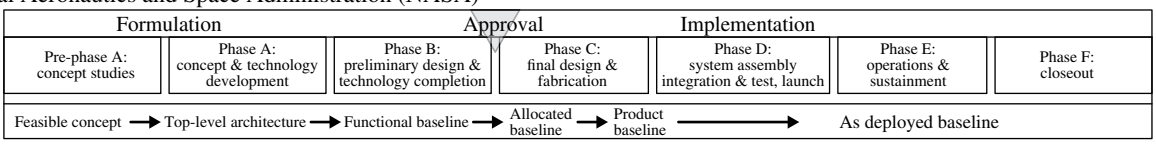
Typical high-tech commercial manufacturer



US Department of Defense (DoD)



National Aeronautics and Space Administration (NASA)



US Department of Energy (DoE)

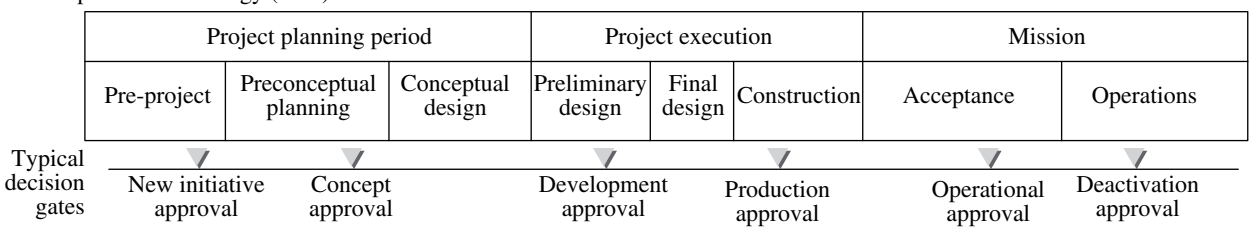


FIGURE 3.3 Comparisons of life cycle models. Derived from Forsberg et al. (2005), Figure 7.2. Reprinted with permission from Kevin Forsberg. All other rights reserved.

3.3.1 Concept Stage

The concept stage begins with some recognition of a need for new or modified SOI (ISO/IEC TR 24748-1, 2010). Many industries employ an exploratory research activity in the concept stage to study new ideas or enabling technologies and capabilities, which then mature into the initiation of a new project (for the SOI). A great deal of creative SE is done in this stage, and the systems engineer leading these studies is likely to follow a new idea into the concept selection, perhaps as project champion. Often, the exploratory research activity identifies the enabling technologies. If the work is done properly in early stages of the life cycle, it is possible to avoid recalls and rework in later stages.

Many life cycle models show the process beginning with "requirements" or "user requirements." In fact, the

process begins earlier with interactions and studies to understand potential new organizational capabilities, opportunities, or stakeholder needs. It is critical that in these early studies, a high-level, preliminary concept be created and explored to whatever depth is necessary to identify technological risks and to assess the technology readiness level (TRL) of the project. The focus is on studying potential technologies and determining the state of what is possible and what is not. In some instances, the project may be an outgrowth of research activities where the research engineer or scientist has no connection to a user-supported need (Forsberg, 1995). The preliminary concept and enabling technologies need to be identified early, and issues arising from the studies need to be addressed during the development stage, according to the National Research Council of the (US) National Academies (NRC, 2008). One of the challenges in developing alternate concepts is that we often build on what has worked well for us in the past, without considering true alternatives, and thereby miss opportunities to make dramatic improvements. This problem has been widely recognized (Adams, 1990; Christensen, 2000).

The preliminary concept will also be used to generate early cost and schedule projections for the project if it moves ahead. Key activities during exploratory research are to clearly define the problem space, characterize the solution space, identify business or mission requirements and stakeholder needs, and, while avoiding any design work, provide an estimate of the cost and schedule for the full-scale development. Incomplete SE in this stage can lead to poor cost and schedule projections, as well as poor understanding of technical alternatives, resulting in poor trades among the alternatives. For example, the Mars Science Laboratory Rover, scheduled for launch in 2009, had to be "delayed because of technical glitches." This resulted in missing the launch window, causing a 2-year delay and a 35% cost growth over the approved development costs. Program critics, however, claimed a 400% cost growth based on the early concept studies, and they threatened the project with cancellation as a result (Achenbach, 2009).

The preliminary concept is a starting point, not an end point, as the project moves into the concept selection activity of the concept stage. The preliminary concept is not put under configuration control, and the key output from exploratory research is a clearer understanding of the business or mission requirements and the stakeholder needs, an assessment of the technology's readiness to move to the next stage, and a rough estimate of the project cost and schedule requirements and technical feasibility to first article delivery.

Concept selection is the second activity of the concept stage. The concept selection activity is a refinement and broadening of the studies, experiments, and engineering models pursued during the exploratory research activity. The first step is to identify, clarify, and document the stakeholders' conceptual operation of the system across the different stages of use and the environments it is to be used in. The operational concept (OpsCon) effort should be undertaken to include any changes caused by changes in the manufacture processes or materials, changes in interface standards, or new feature enhancements being added that can drive various aspects of concept selection of the system.

During the concept stage, the team begins in-depth studies that evaluate multiple candidate concepts and eventually provide a substantiated justification for the system concept that is selected. As part of this evaluation, mock-ups may be built (for hardware) or coded (for software), engineering models and simulations may be executed, and prototypes of critical elements may be built and tested. Engineering models and prototypes of critical elements are essential to verify the feasibility of concepts, to aid the understanding of stakeholder needs, to explore architectural trade-offs, and to explore risks and opportunities. These studies expand the risk and opportunity evaluation to include affordability assessment, environmental impact, failure modes, hazard analysis, technical obsolescence, and system disposal. Issues related to integration and verification must also be explored for each alternate system concept, since these can be discriminators in system selection. The systems engineer facilitates these analyses by coordinating the activities of engineers from many disciplines. Key objectives are to provide confidence that the business case is sound and the proposed solutions are achievable.

The concept stage may include system and key system element-level concept and architecture definition and integration, verification, and validation (IV&V) planning. Early validation efforts align requirements with stakeholder expectations. The system capabilities specified by the stakeholders will be met by the combination of system elements. Problems identified for individual system element-level concepts should be addressed early to minimize the risk that they fall short of the required functionality or performance when the elements are finally designed and verified.

FIGURE 3.4 Importance of the concept stage. DILBERT © 1997 Scott Adams. Used with permission from UNIVERSAL UCLICK. All rights reserved.

Many projects are driven by eager project champions who want "to get on with it." They succumb to the temptation to cut short the concept stage, and they use exaggerated projections to support starting development without adequate understanding of the challenges involved, as comically illustrated in Figure 3.4. Many commissions reviewing failed systems after the fact have identified insufficient or superficial study in the concept stage as a root cause of failure.

3.3.2 Development Stage

The development stage defines and realizes a SOI that meets its stakeholder requirements and can be produced, utilized, supported, and retired. The development stage begins with the outputs of the concept stage. The primary output of this stage is the SOI. Other outputs can include a SOI prototype, enabling system requirements (or the enabling systems themselves), system documentation, and cost estimates for future stages (ISO/IEC TR 24748-1, 2010).

Business and mission needs, along with stakeholder requirements, are refined into system requirements. These requirements are used to create a system architecture and design. The concept from the previous stage is refined to ensure all system and stakeholder requirements are satisfied. Requirements for production, training, and support facilities are defined. Enabling systems' requirements and constraints are considered and incorporated into the design. System analyses are performed to achieve system balance and to optimize the design for key parameters.

One of the key activities of the development stage is to specify, analyze, architect, and design the system so that the system elements and their interfaces are understood and specified. Hardware and software elements are fabricated and coded.

Operator interfaces are specified, tested, and evaluated during the development stage. Operator and maintainer procedures and training are developed and delivered to ensure humans can interface with the SOI.

Feedback is obtained from both external and internal stakeholders through a series of technical reviews and decision gates. Projects that are not showing acceptable progress may be redirected or even terminated.

The development stage includes detailed planning and execution of IV&V activities. The planning for these activities needs to take place early to ensure that adequate facilities and other resources are available when needed. A source of additional information about IV&V and the significance for project cost and risk when these activities are optimized was the subject of the European Union SysTest program (Engel, 2010).

3.3.3 Production Stage

The production stage is where the system is produced or manufactured. Product modifications may be required to resolve production problems, to reduce production costs, or to enhance product or system capabilities. Any of these may influence system requirements and may require system reverification or revalidation. All such changes require SE assessment before changes are approved.

3.3.4 Utilization Stage

The utilization stage is where the system is operated in its intended environment to deliver its intended services. Product modifications are often planned for introduction throughout the operation of the system. Such upgrades enhance the capabilities of the system. These changes should be assessed by systems engineers to ensure smooth integration with the operational system.

For large complex systems, midlife upgrades can be substantial endeavors requiring SE effort equivalent to a major program.

3.3.5 Support Stage

The support stage is where the system is provided services that enable continued operation. Modifications may be proposed to resolve supportability problems, to reduce operational costs, or to extend the life of a system. These changes require SE assessment to avoid loss of system capabilities while under operation.

3.3.6 Retirement Stage

The retirement stage is where the system and its related services are removed from operation. SE activities in this stage are primarily focused on ensuring that disposal requirements are satisfied. Planning for retirement is part of the system definition during the concept stage. Experience has repeatedly demonstrated the consequences when system retirement is not considered from the outset. Early in the twenty-first century, many countries have changed their laws to hold the developer of a SOI accountable for proper end-of-life disposal of the system.

3.4 LIFE CYCLE APPROACHES

Various life cycle models, such as the Waterfall (Royce, 1970), Spiral (Boehm, 1986), and Vee (Forsberg and Mooz, 1991), are useful in defining the start, stop, and process activities appropriate to the life cycle stages.

Graphical representations of life cycle stages tend to be linear, but this hides the true incremental, iterative, and recursive nature of the underlying processes. The approaches that follow imply full freedom to choose a development model and are not restricted to sequential methods.

3.4.1 Iteration and Recursion

Too often, the system definition is viewed as a linear, sequential, single pass through the processes. However, valuable information and insight need to be exchanged between the processes, in order to ensure a good system definition that effectively and efficiently meets the mission or business needs. The application of iteration and recursion to the life cycle processes with the appropriate feedback loops helps to ensure communication that accounts for ongoing learning and decisions. This facilitates the incorporation of learning from further analysis and process application as the technical solution evolves.

Figure 3.5 shows an illustration of iteration and recursion of the processes. Iteration is the repeated application of and interaction between two or more processes at a given level in the system structure or hierarchy. Iteration is needed to accommodate stakeholder decisions and evolving understanding, account for architectural decisions/constraints, and resolve trades for affordability, adaptability, feasibility, resilience, etc. Although the figure only shows a subset of the life cycle technical processes, there can be iteration between any of the processes. For example, there is often iteration between system requirements definition and architecture definition. In this case, there is a concurrent application of the processes with iteration between them, where the evolving system requirements help to shape the architecture through identified constraints and functional and quality requirements. The architecture trades, in turn, may identify requirements that are not feasible, driving further requirements analysis with trades that change some requirements. Likewise, the design definition could identify the need to reconsider decisions and trades in the requirements definition or architecture definition processes. Any of these can invoke additional application of the system analysis and decision management processes.

Recursion is the repeated application of and interaction of processes at successive levels in the system structure. The technical processes are expected to be recursively applied for each successive level of the system structure until the level is reached where the decision is made to make, buy, or reuse a system element. During the recursive application of the processes, the outputs at one level become inputs for the next successive level (below for system definition, above for system realization).

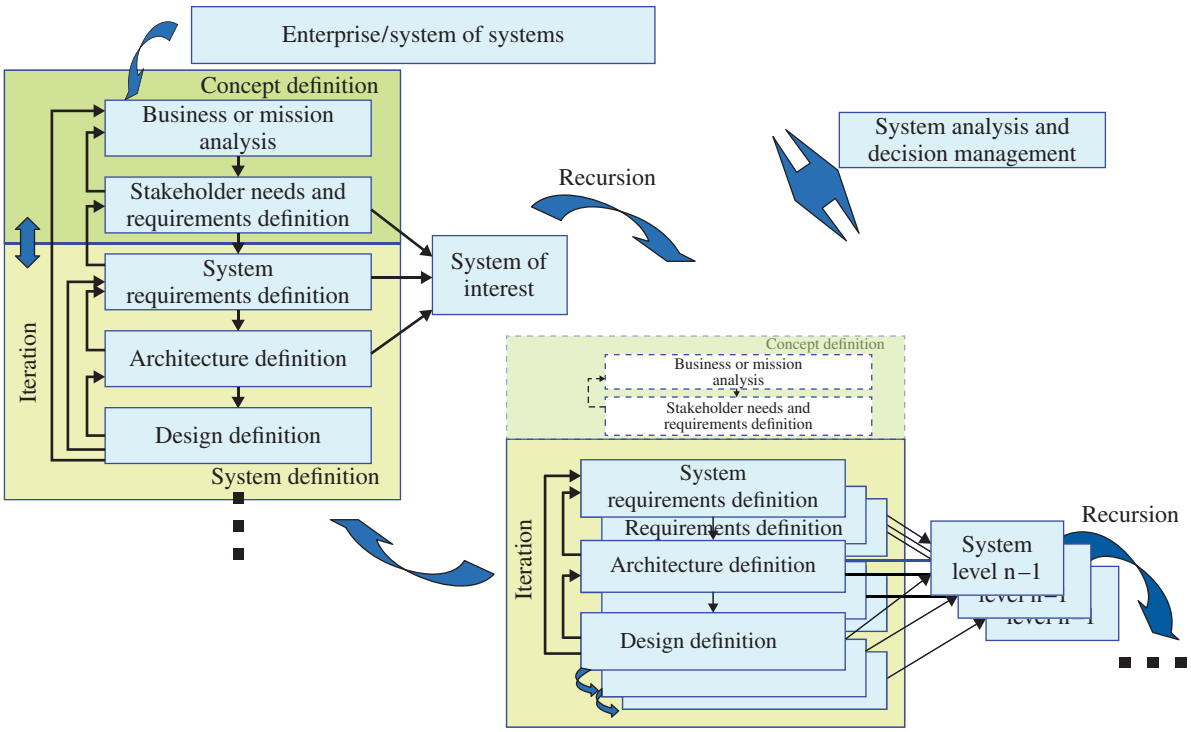


FIGURE 3.5 Iteration and recursion. Reprinted with permission from Garry Roedler. All other rights reserved.

3.4.2 Sequential Methods

On projects where it is necessary to coordinate large teams of people working in multiple companies, sequential approaches provide an underlying framework to provide discipline to the life cycle processes. Sequential methods are characterized by a systematic approach that adheres to specified processes as the system moves through a series of representations from requirements through design to finished product. Specific attention is given to the completeness of documentation, traceability from requirements, and verification of each representation after the fact.

The strengths of sequential methods are predictability, stability, repeatability, and high assurance. Process improvement focuses on increasing process capability through standardization, measurement, and control. These methods rely on the "master plans" to anchor their processes and provide project-wide communication. Historical data is usually carefully collected and maintained as inputs to future planning to make projections more accurate (Boehm and Turner, 2004).

Safety-critical products, such as the Therac-25 medical equipment described in Section 3.6.1, can only meet modern certification standards by following a thorough, documented set of plans and specifications. Such standards mandate strict adherence to process and specified documentation to achieve safety or security. However, unprecedented projects or projects with a high rate of unforeseeable change, poor predictability, and lack of stability often degrade, and a project may incur significant cost trying to keep documentation and plans up to date.

The Vee model, introduced in Forsberg and Mooz (1991), described in Forsberg et al. (2005), and shown in Figure 3.6, is a sequential method used to visualize various key areas for SE focus, particularly during the concept and development stages. The Vee highlights the need for continuous validation with the stakeholders, the need to define verification plans during requirements development, and the importance of continuous risk and opportunity assessment.

The Vee model provides a useful illustration of the SE activities during the life cycle stages. In this version of

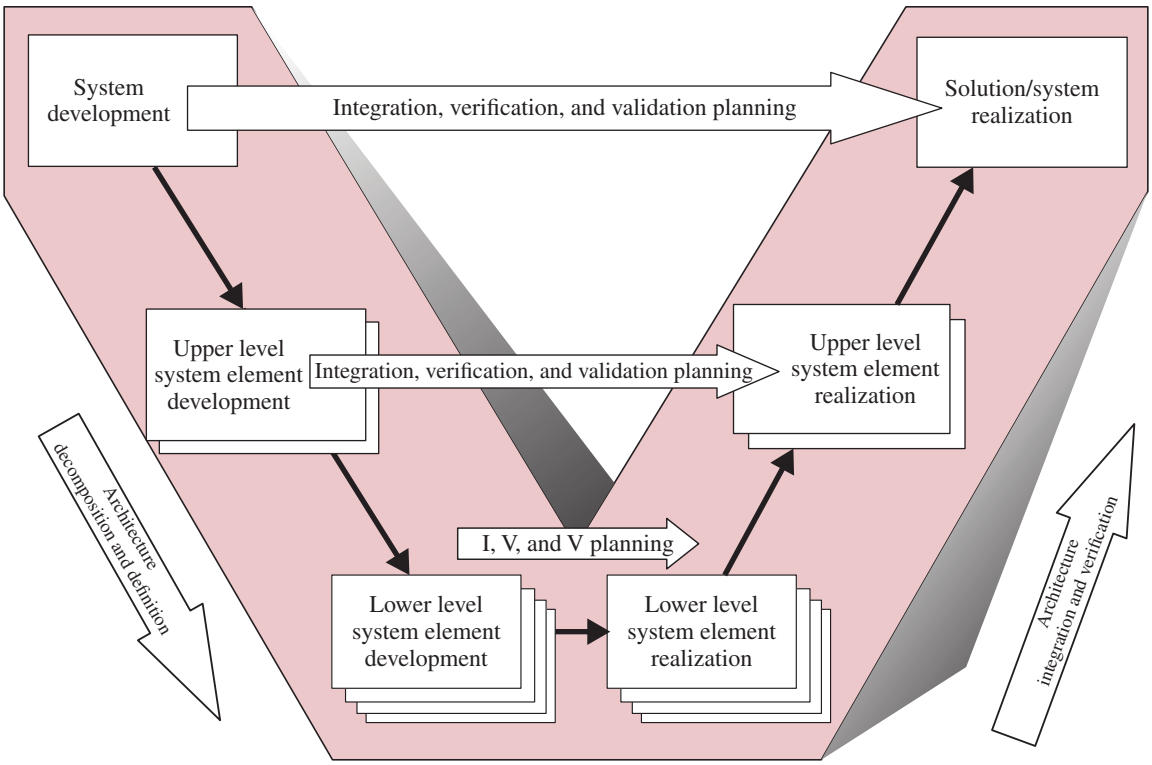


FIGURE 3.6 Vee model. Derived from Forsberg et al. (2005), Figure 7.10. Reprinted with permission from Kevin Forsberg. All other rights reserved.

the Vee model, time and system maturity proceed from left to right. The core of the Vee (i.e., those products that have been placed under configuration control) depicts the evolving baseline from stakeholder requirements agreement to identification of a system concept to definition of elements that will comprise the final system. With time moving to the right, the evolving baseline defines the left side of the core of the Vee, as shown in the shaded portion of Figure 3.7.

A key attribute of the Vee model is that time and maturity move from the left to the right across the diagram, as shown in Figure 3.7. At any instant of time, the development team then can move their perspective only along the vertical arrow, from the highest level of the system requirements down to the lowest level of detail. The off-core opportunity and risk management investigations going downward are addressing development options to provide assurance that the baseline performance being considered can indeed be achieved and to initiate alternate concept studies at the lower levels of detail to determine the best approach. These downward off-core investigations and development efforts are entirely under control of the development team.

On the other hand, the essential upward off-core stakeholder discussions (in-process validation) ensure that the proposed baselines are acceptable to management, customer, user, and other stakeholders. Changes to enhance system performance or to reduce risk or cost are welcome for consideration, but these must go through formal change control, since others outside the development team may be building on previously defined and released design decisions. The power of understanding the significance of the off-core studies is illustrated in a National Aeronautics and Space Administration (NASA) Jet Propulsion Laboratory (JPL) report (Briedenthal and Forsberg, 2007).

As entities are implemented, verified, and integrated, the right side of the core of the Vee is executed. Figure 3.8 illustrates the evolving baseline as system elements are integrated and verified. Since one can never go backward

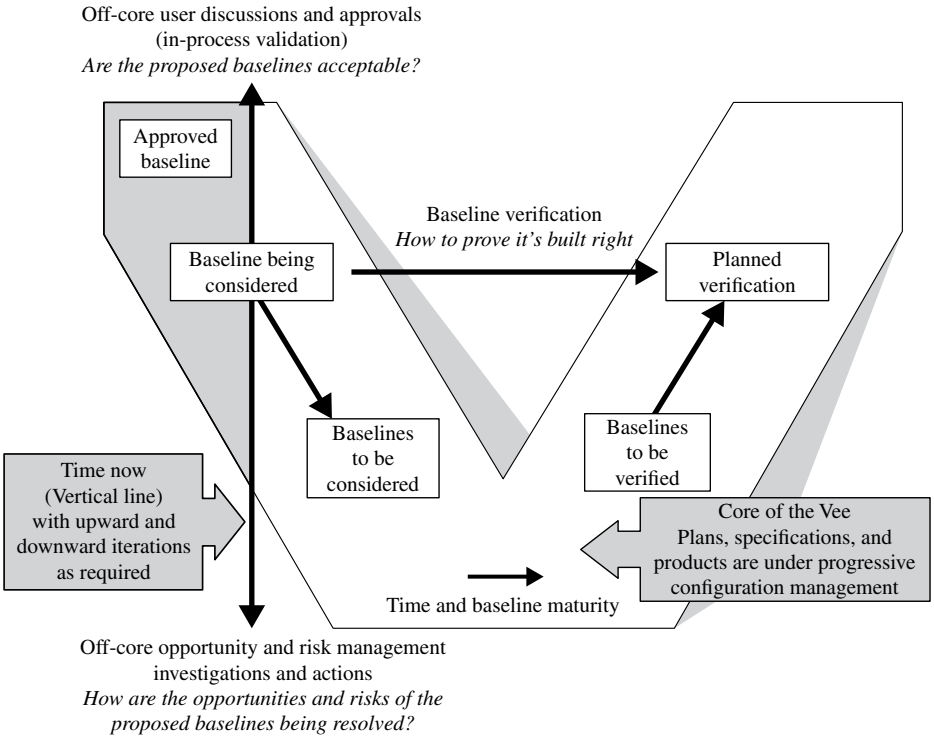


FIGURE 3.7 Left side of the Vee model. Derived from Forsberg et al. (2005), Figure 7.11. Reprinted with permission from Kevin Forsberg. All other rights reserved.

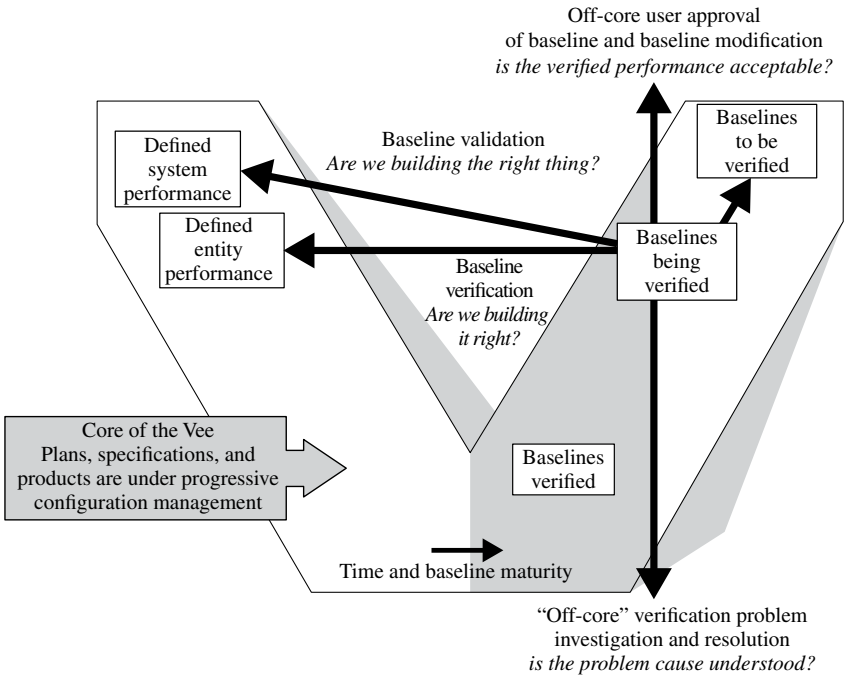


FIGURE 3.8 Right side of the Vee model. Derived from Forsberg et al. (2005), Figure 7.12. Reprinted with permission from Kevin Forsberg. All other rights reserved.

in time, all iterations in the Vee are performed on the vertical "time now" line. Upward iterations involve the stakeholders and are the in-process validation activities that ensure that the proposed baselines are acceptable. The downward vertical iterations are the essential off-core opportunity and risk management investigations and actions. In each stage of the system life cycle, the SE processes iterate to ensure that a concept or design is feasible and that the stakeholders remain supportive of the solution as it evolves.

3.4.3 Incremental and Iterative Methods

Incremental and iterative development (IID) methods have been in use since the 1960s (Larman and Basili, 2003). They represent a practical and useful approach that allows a project to provide an initial capability followed by successive deliveries to reach the desired SOI. The goal is to provide rapid value and responsiveness.

The IID approach is used when the requirements are unclear from the beginning or the stakeholder wishes to hold the SOI open to the possibilities of inserting new technology. Based on an initial set of assumptions, a candidate SOI is developed and then assessed to determine if it meets the stakeholder needs or requirements. If not, another evolutionary round is initiated, and the process is repeated until a system is delivered to satisfied stakeholders or until the organization decides to terminate the effort.

Most literature agrees that IID methods are best applied to smaller, less complex systems or to system elements. The focus is on flexibility and on allowing selected events to be taken out of sequence when the risk is acceptable. Tailoring in this way highlights the core activities of product development.

The features that distinguish IID from the sequential approaches are velocity and adaptability. While market strategies often emphasize that "time to market" or "speed" is critical, a more appropriate criterion is "velocity," which considers direction in addition to speed. By incorporating the stakeholders into the working-level teams, the project receives continuous feedback that they are going in a direction that satisfies the stakeholders' highest needs first. One downside is that reactive project management with a stakeholder that often changes direction can result in an unstable, chaotic project. On one hand, this approach avoids the loss of large investments in faulty assumptions; on the other hand, emphasis on a tactical viewpoint may generate short-term or localized solution optimizations.

IIDs may also be "plan driven" in nature when the requirements are known early in the life cycle, but the development of the functionality is performed incrementally to allow for the latest technology insertion or potential changes in needs or requirements. A specific IID methodology called evolutionary development (Gilb, 2005) is common in research and development (R&D) environments. Figure 3.9 illustrates how this approach was used in the evolution of the tiles for the NASA space shuttle.

An example of an incremental and iterative method is the Incremental Commitment Spiral Model (ICSM) (Boehm et al., 2014). The ICSM builds on the strengths of current process models, such as early V&V concepts in the Vee model, concurrency concepts in the concurrent engineering model, lighter-weight concepts in the agile and lean models, risk-driven concepts in the spiral model Boehm, 1996, the phases and anchor points in the rational unified process (RUP) (Kruchten, 1999), and recent extensions of the spiral model to address SoS capability acquisition (Boehm and Lane, 2007).

A view of the ICSM is shown in Figure 3.10. In the ICSM, each increment addresses requirements and solutions concurrently, rather than sequentially. ICSM also considers products and processes; hardware, software, and human factor aspects; and business case analyses of alternative product configurations or product line investments. The stakeholders consider the risks and risk mitigation plans and decide on a course of action. If the risks are acceptable and covered by risk mitigation plans, the project proceeds into the next spiral.

Figure 3.11 presents another view of the ICSM. The top row of activities indicates that a number of system aspects are being concurrently engineered at an increasing level of understanding, definition, and development.

3.5 WHAT IS BEST FOR YOUR ORGANIZATION, PROJECT, OR TEAM?

Conway's law suggests that "organizations which design systems … are constrained to produce designs which are copies of the communication structures of those organizations" (Conway, 1968). Systems thinking and SE help organizations avoid the pitfall of Conway's law by ensuring that system designs are appropriate to the problem being addressed.

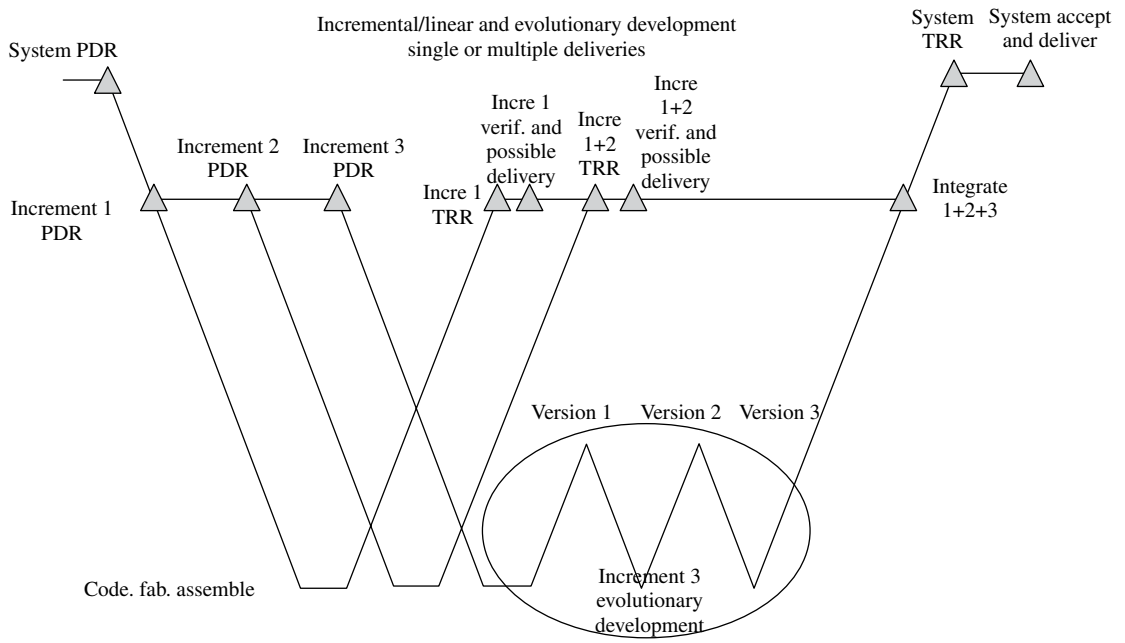


FIGURE 3.9 IID and evolutionary development. Derived from Forsberg et al. (2005), Figure 19.18. Reprinted with permission from Kevin Forsberg. All other rights reserved.

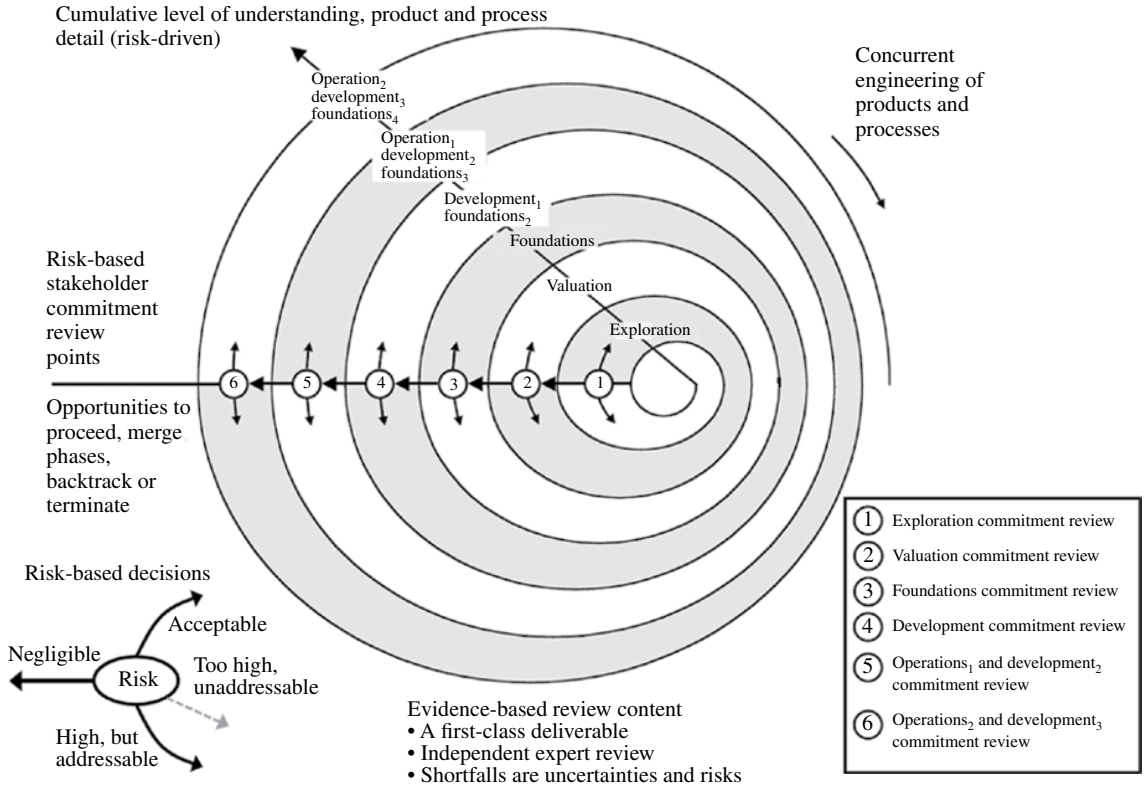


FIGURE 3.10 The incremental commitment spiral model (ICSM). From Boehm et al. (2014). Reprinted with permission from Barry Boehm. All other rights reserved.

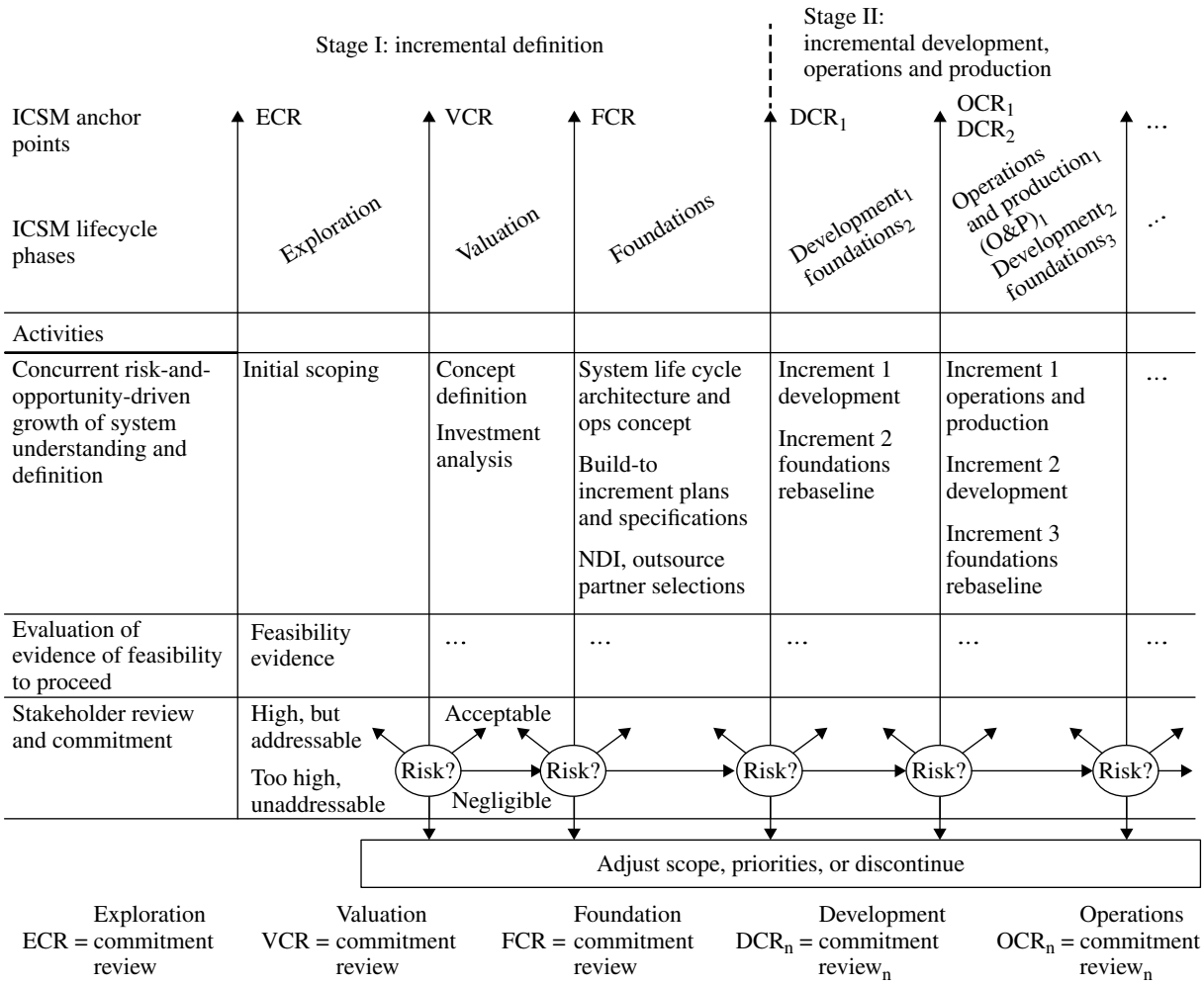


FIGURE 3.11 Phased view of the generic incremental commitment spiral model process. From Boehm et al. (2014). Reprinted with permission from Barry Boehm. All other rights reserved.

One of the earliest books on SE management (Chase, 1974) identified three simple criteria for such organizations: facilitate communications, streamline controls, and simplify paperwork. The way to effective SE management is not "in the direction of formal, formidable, massive documentation. It does, however, reside in the direction of creating a total environment which is conducive to the emergence and effective utilization of creative and inventive talents oriented toward achieving a system approach with a minimum of management encumbrances" (Chase, 1974).

Whenever someone (be it an individual or a company) wants to reach a desired end, they must perform a series of actions or operations. Further, they must consider the order of those actions, their dependencies, who will perform them, what they require and what they will generate, how long it will take to complete them, and what tools they will employ. Thus, individuals and organizations follow processes, be they predefined or ad hoc. Because process components (activities, products, agents, tools) and their interactions (information flow, artifacts flow, control, communication, timing, dependencies, and concurrency) can vary, processes will differ—even if the performing organizations have the same level, scope, and goal.

So why should an organization care about processes? In short, this is to better understand, evaluate, control,

learn, communicate, improve, predict, and certify the work performed (McConnell, 1998). For a given organizational level, the processes vary with the project's goals and available resources. At a high level, the company's business strategy determines the business approach, with the main goals of profitability, time to market, minimum cost, higher quality, and customer satisfaction setting the priorities. Similarly, the company's size; the number, knowledge, and experience of people (both engineers and support personnel); and hardware resources determine how to achieve those goals (Cockburn, 2000). The application domain and the corresponding system requirements, together with other constraints, form another important factor in defining and applying processes.

So what really is best for my organization? The answer is that it depends on the situation. Depending on the perspective, different processes are defined for entire organizations, teams, or individuals. A "one-size-fits-all" approach does not work when defining processes; thus, organizations must continuously document, define, measure, analyze, assess, compare, and change processes to best meet project goals. One would hardly expect to find the same processes used in a startup e-commerce company as in NASA. The intended goal shapes a process in terms of scope (namely, the stages and activities covered) and organizational level. In any case, the selected processes should help guide people on what to do—how to divide and coordinate the work—and ensure effective communication. Coordination and communication, for example, form the main problems in large projects involving many people, especially in distributed projects where people cannot communicate face to face (Lindvall and Rus, 2000).

3.6 INTRODUCTION TO CASE STUDIES

Real-world examples that draw from diverse industries and types of systems are provided throughout this handbook. Five case studies have been selected to illustrate the diversity of systems to which SE principles and practices can be applied: medical therapy equipment, a bridge, a superhigh-speed train, a breach of a cybersecurity system, and a redesign of a high-tech medical system for low-tech maintenance. They represent examples of failed, successful, and prototype systems that all define(d) the state of the art. These studies may be categorized as medical, infrastructure, and transportation applications; in the manufacturing and construction industry domains; with and without software elements; complex; and subject to scrutiny in the concept, development, utilization, and support stages as all have a need to be safe for humans and are constrained by government regulations.

3.6.1 Case 1: Radiation Therapy—The Therac-25

3.6.1.1 Background Therac-25, a dual-mode medical linear accelerator (LINAC), was developed by the medical division of the Atomic Energy Commission Limited (AECL) of Canada, starting in 1976. A completely computerized system became commercially available in 1982. This new machine could be built at lower production cost, resulting in lower prices for the customers. However, a series of tragic accidents led to the recommended recall and discontinuation of the system.

The Therac-25 was a medical LINAC, or particle accelerator, capable of increasing the energy of electrically charged atomic particles. LINACs accelerate charged particles by introducing an electric field to produce particle beams (i.e., radiation), which are then focused by magnets. Medical LINACs are used to treat cancer patients by exposing malignant cells to radiation. Since malignant tissues are more sensitive than normal tissues to radiation exposure, a treatment plan can be developed that permits the absorption of an amount of radiation that is fatal to tumors but causes relatively minor damage to surrounding tissue.

Six accidents involving enormous radiation overdoses to patients took place between 1985 and 1987. Tragically, three of these accidents resulted in the death of the patients. This case is ranked in the top 10 worst software-related incidents on many lists. Details of the accidents and analysis of the case are available from many sources (Jacky, 1989; Leveson and Turner, 1993; Porrello, n.d.).

3.6.1.2 Approach Therac-25 was a revolutionary design compared to its predecessors, Therac-6 and Therac-20, both with exceptional safety records. It was based on a double-pass concept that allowed a more powerful accelerator to be built into a compact and versatile machine. AECL designed Therac-25 to fully utilize the potential of software control. While Therac-6 and Therac-20 were built as stand-alone machines and could be operated without a computer, Therac-25 depended on a tight integration of software and hardware. In the new, tightly coupled system, AECL used software to monitor

the state of the machine and to ensure its proper operations and safety. Previous versions had included independent circuits to monitor the status of the beam as well as hardware interlocks that prevented the machine from delivering radiation doses that were too high or from performing any unsafe operation that could potentially harm the patient. In Therac-25, AECL decided not to duplicate these hardware interlocks since the software already performed status checks and handled all the malfunctions. This meant that the Therac-25 software had far more responsibility for safety than the software in the previous models. If, in the course of treatment, the software detected a minor malfunction, it would pause the treatment. In this case, the procedure could be restarted by pressing a single "proceed" key. Only if a serious malfunction was detected was it required to completely reset the treatment parameters to restart the machine.

The software for Therac-25 was developed from the Therac-20's software, which was developed from the Therac-6's software. One programmer, over several years, evolved the Therac-6 software into the Therac-25 software. A stand-alone, real-time operating system was added along with application software written in assembly language and tested as a part of the Therac-25 system operation. In addition, significant adjustments had been made to simplify the operator interface and minimize data entry, since initial operators complained that it took too long to enter a treatment plan.

At the time of its introduction to market in 1982, Therac-25 was classified as a Class II medical device. Since the Therac-25 software was based on software used in the earlier Therac-20 and Therac-6 models, Therac-25 was approved by the Federal Drug Administration under Premarket Equivalency.

3.6.1.3 Conclusions The errors were introduced in the concept and early development stages, when the decisions were made to create the software for Therac-25 using the modification of existing software from the two prior machines. The consequences of these actions were difficult to assess at the time, because the starting point (software from Therac-6) was a poorly documented product and no one except the original software developer could follow the logic (Leveson and Turner, 1993). This case illustrates the importance of the off-the-Vee-core studies early in development (see Fig. 3.7).

The issues from the Therac case are, unfortunately, still relevant, as evidenced by similar deaths for similar reasons in 2007 upon the introduction of new LINAC-based radiation therapy machines (Bogdanich, 2010).

3.6.2 Case 2: Joining Two Countries—The Øresund Bridge

3.6.2.1 Background The Øresund Region is composed of eastern Denmark and southern Sweden and since 2000 has been linked by the Øresund Bridge. The area includes two major cities, Copenhagen and Malmö, has a population of 3 million, and counts as Europe's eighth largest economic center. One fifth of the total Danish and Swedish Gross National Product (GNP) is produced in the region. The official name of the bridge is translated "the Øresund Connection" to underscore the full integration of the region. For the first time ever, Sweden is joined permanently to the mainland of Europe by a 10 min drive or train ride. The cost for the entire Øresund Connection construction project was calculated at 30.1 billion DKK (3 billion USD), and the investment is expected to be paid back by 2035.

The Øresund Bridge is the world's largest composite structure, has the longest cable-stayed bridge span in the world carrying motorway and railway traffic, and boasts the highest freestanding pylons. The 7.9 km (5 miles) long bridge crosses the international navigation route between the Baltic Sea and the North Sea. A cable-stayed high bridge rises 57 m (160 ft) above the surface of the sea, with a main span of 490 m (0.3 miles). Both the main span and the approach bridges are constructed as a two-level composite steel-concrete structure. The upper deck carries a four-lane motorway, and the lower deck carries a two-track railway for both passenger trains and freight trains. The rest of the distance is spanned by the artificial island Peberholm ("Pepper" islet, named to complement the Saltholm islet to the north) and a tunnel on the Danish side that is the longest immersed concrete tunnel in the world. Since completion, Peberholm has become a natural habitat for colonies of rare birds, one of the largest of its kind in Denmark and Sweden.

Nations other than Denmark and Sweden also contributed to this project. Canada provided a floating crane, aptly named Svanen (the swan), to carry prefabricated bridge sections out to the site and place them into position. Forty-nine steel girders for the approach bridges were fabricated in Cádiz, Spain. A specially designed catamaran was built to handle transportation of the foundations for the pylons, which weighed 19,000 tons each.

3.6.2.2 Approach As noted in the many histories of bridge, the development stage of the project began with well-defined time, budget, and quality constraints. The design evolved over more than 7 years, from start to delivery of final documentation and maintenance manuals. More than 4000 drawings were produced. The consortium dealt with changes, as necessary, using a combination of technical competence and stakeholder cooperation. Notably, there were no disputes and no significant claims against the owners at the conclusion, and this has been attributed to the spirit of partnership.

What is not often reported is that the success of the development stage is clearly based on the productive, focused, creative effort in the concept stage that began when the royal families of Denmark and Sweden finally agreed in 1990 to move ahead with a bridge project connecting their two countries. That SE effort shaped the approach to the project with well-defined time, budget, and quality constraints at the transition to the development stage. During the concept stage, the SE team also recognized that the concerns of environmental groups would—and should—impact the approach to the construction of the bridge. The owners took a creative approach by inviting the head of a key environmental group to be part of the board of directors.

From the beginning of the development stage, the owners defined comprehensive requirements and provided definition drawings as part of the contract documents to ensure a project result that not only fulfilled the quality requirements on materials and workmanship but also had the envisioned appearance. The contractor was responsible for the detailed design and for delivering a quality-assured product in accordance with the owners' requirements. The following are representative of the requirements levied at the start of the project:

- Schedule: Design life, 100 years; construction time, 1996–2000
- Railway: Rail load, International Union of Railways (UIC) 71; train speed, 200 km/h
- Motorway: Road axle load, 260 kN; vehicle speed, 120 km/h
- Ambient environment: Wind speed (10 min), 61 m/s; wave height, 2.5 m; ice thickness, 0.6 m; temperature, +/− 27°C
- Ship impact: To pylons, 560 MN; to girder, 35 MN

In addition to established requirements, this project crossed national boundaries and was thereby subject to the legislations of each country. Technical requirements were based on the Eurocodes, with project-specific amendments made to suit the national standards of both countries. Special safety regulations were set up for the working conditions, meeting the individual safety standards of Denmark and Sweden.

The railway link introduced yet another challenge. In Denmark, the rail traffic is right handed, as on roadways, whereas the trains in Sweden pass on the left-hand side. The connection needed to ensure a logical transition between the two systems, including safety aspects. In addition, the railway power supply differs between the two countries; thus, it was necessary to develop a system that could accommodate power supply for both railway systems and switch between them on the fly.

The design of a major cable-stayed bridge with approach spans for both road and railway traffic involves several disciplines, including, but not limited to, geotechnical engineering, aerodynamics, foundation engineering, wind tunnel tests, design of piers and pylons, design of composite girders, design of cables and anchorages, design of structural monitoring system, ship impact analysis, earthquake analysis, analysis of shrinkage and creep of concrete, ice load analysis, fatigue analysis, pavement design, mechanical systems, electrical systems, comfort analysis for railway passengers, traffic forecast, operation and maintenance aspects, analysis of construction stages, risk analysis for construction and operation, quality management, and environmental studies and monitoring.

Comprehensive risk analyses were carried out in connection with the initial planning studies, including specification of requirements to secure all safety aspects. Important examples of the results of these studies for the Øresund Bridge were as follows:

- Navigation span was increased from 330 to 490 m.
- The navigation channel was realigned and deepened to reduce ship groundings.
- Pier protection islands were introduced to mitigate bridge/ship accidents.

Risks were considered in a systematic way, using contemporary risk analysis methods such as functional safety analysis using fault tree and "what-if" techniques.

Three main issues were considered under the design–build contract:

- General identification and assessment of construction risks
- Ship collision in connection with realignment of navigation channel
- Risks in connection with 5-year bridge operation by contractor

A fully quantified risk assessment of the human safety and traffic delay risks was carried out for a comprehensive list of hazards, including fire, explosion, train collisions and derailments, road accidents, ship collisions and groundings, aircraft collisions, environmental loads beyond design basis, and toxic spillages. An example of a consequence of this analysis was the provision of passive fire protection on the tunnel walls and ceilings.

Both Denmark and Sweden are proud of being among the cleanest industrial countries in the world. Their citizens, and therefore the politicians, would not allow for any adverse environmental impact from the construction or operation of a bridge. The Great Belt and Øresund Strait both constitute corridors between the salty Kattegat and the sweeter water of the Baltic Sea. Any reduction in water exchange would reduce the salt content and, therefore, the oxygen content of the Baltic Sea and would alter its ecological balance. The Danish and Swedish authorities decided that the bridge should be designed in such a way that the flow through of water, salt, and oxygen into the Baltic was not affected. This requirement was designated the zero solution. To limit impacts on the local flora and fauna in Øresund during the construction, the Danish and Swedish authorities imposed a restriction that the spillage of seabed material from dredging operations should not exceed 5% of the dredged amounts. The zero solution was obtained by modeling with two different and independent hydrographical models.

In total, 18 million cubic meters of seabed materials were dredged. All dredged materials were reused for reclamation of the artificial peninsula at Kastrup and the artificial island, Peberholm. A comprehensive and intensive monitoring of the environment was performed to ensure and document the fulfillment of all environmental requirements. In their final status report from 2001, the Danish and Swedish authorities concluded that the zero solution as well as all environmental requirements related to the construction of the link had been fulfilled. Continual monitoring of eel grass and common mussels showed that, after a general but minor decline, populations had recovered by the time the bridge was opened. Overall, the environment paid a low price at both Øresund and the Great Belt because it was given consideration throughout the planning and construction stages of the bridges.

3.6.2.3 Conclusions This award-winning bridge is the subject of numerous articles and a PhD thesis, where details of the construction history and collaboration among all the stakeholders are provided (Jensen, 2014; Nissen, 2006; Skanska, 2013). This project provides a clear example of the benefit of a solid concept stage where the management team was able to resist the customer-driven temptation to jump prematurely into the development stage.

3.6.3 Case 3: Prototype System—The Superhigh-Speed Train in China

3.6.3.1 Background Shanghai Transrapid is the first commercial high-speed commuting system using the state-of-the-art electromagnetic levitation (or maglev) technology. The train runs from Shanghai's financial district to Pudong International Airport, and the total track length is about 30 km (20 miles). The train takes 7 min and 20 s to complete the journey, can reach almost 320 km/h (200 mph) in 2 min, and reaches its maximum speed of 430 km/h (267 mph) within 4 min. The Shanghai Transrapid project cost 10 billion Yuan (1.2 billion USD) and took 2.5 years to complete. Construction began in March 2001, and public service commenced on January 1, 2003. Critics argue that the speed over such a short distance is unnecessary and that the line may never recoup this cost. However, the speculation is that this is a prototype to gather operational data assessing the feasibility of a Shanghai to Beijing maglev train route. From this perspective, this project makes perfect sense.

Prior to this installation, many countries had argued over the feasibility of maglev trains. They do not have wheels or use a traditional rail. Rather, powerful magnets lift the entire train about 10 mm above the special track, called a guideway, which mainly directs the passage of the train. Electromagnetic force is used to make the train

hover and to provide vertical and horizontal stabilization. The frequency, intensity, and direction of the electrical current in the track control the train's movement, while the power for the levitation system is supplied by the train's onboard batteries, which recharge whenever the train is moving. Maglev trains also do not have an onboard motor. The guideway contains a built-in electric motor that generates an electromagnetic field that pulls the train down the track. Putting the propulsion system in the guideway rather than onboard the trains makes the cars lighter, which enables the train to accelerate quickly. The superhigh speeds are attained largely due to the reduction of friction.

Despite the high speed, the maglev system runs more quietly than a typical commuter train, consumes less energy, and is nearly impossible to derail because of the way the train's underside partially wraps around the guideway, like a giant set of arms hugging the train to the elevated platform. Passengers experience a comfortable and quiet ride due to the maglev technology and the specially designed window; noise level is less than 60 decibels at a speed of 300 km/h.

3.6.3.2 Approach The Chinese authorities considered the economical operation, low energy consumption, less environmental impact, and high speed when choosing a solution suitable for ground transport between hubs that range from hundreds to over 1000 km apart. But the same solution also needed to be suitable for modern mass rapid passenger transportation between a center city and adjacent cities. Despite the many advantages, in 1999, the technology was considered to be in an experimental stage—its technological superiority, safety, and economic performance not yet proven by commercialized operation. The current line is the result of a compromise; it was built as a demonstration to verify the maturity, availability, economics, and safety of a high-speed maglev transportation system.

The basic technology to create a maglev system has been around since 1979, but until this project, it had never been realized, mostly due to the expense of developing a new train system. Many experts believe that superfast steel-wheel rail systems—such as those in France and Japan—have reached the limits of this technology and cannot go any faster. Maglev proponents describe the system as "the first fundamental innovation in the field of railway technology since the invention of the railway" and are watching proposals for maglev installations in Germany and the United States (BBC, 2002; McGrath, 2003; SMTDC, 2005; Transrapid International, 2003).

3.6.3.3 Conclusions From an SE perspective, this case illustrates the fact that a project that goes through all the stages from concept to operations may in fact simply be part of the concept stage of a larger effort.

3.6.4 Case 4: Cybersecurity Considerations in Systems Engineering—The Stuxnet Attack on a Cyber-Physical System

3.6.4.1 Background As our world becomes increasingly digital, the issue of cybersecurity is a factor that systems engineers need to take into account. Both hardware and software systems are increasingly at risk for disruption or damage caused by threats taking advantage of digital technologies. Stuxnet, a cyber attack on Iran's nuclear capabilities, illustrates the need for systems engineers to be comprehensive in their assessment of vulnerabilities and rigorous in their mitigation of attack potential (Failliere, 2011; Langner, 2012).

This case study discusses a new degree of attack sophistication previously unseen—a new level of malware complexity at military-grade performance, nearly no side effects, and pinpoint accuracy. However, though the creation and deployment of Stuxnet were expensive undertakings, the strategy, tactical methods, and code mechanisms are now openly available for others to reuse and build upon at much less expense. Cyber-physical system attacks are becoming increasingly prevalent, and SE must consider the implications of cybersecurity to reduce the vulnerabilities.

Iran's Natanz nuclear fuel enrichment plant (FEP) is a military-hardened facility, with a security fence surrounding a complex of buildings, which are in turn each protected by a series of concrete walls. The complex contains several "cascade halls" for the production of enriched uranium in gas centrifuges. This facility was further hardened with a roof of several meters of reinforced concrete and covered with a thick layer of earth.

Each of the cascade halls is a cyber-physical system, with an industrial control system (ICS) of programmable logic controllers (PLCs), computers, an internal network with no connections to the outside world, and capacity for thousands of centrifuges. Though the internal network is isolated from the outside world by an "air gap,"

possible vulnerabilities still include malicious insider collusion, nonmalicious insider insertion of memory devices brought in from the outside, visiting service technicians, and supply chain intervention. It has been suggested that all of these breech vectors may have played a role in the massive centrifuge damage that began occurring in 2009 and continued at least through 2010.

Malware, now known as Stuxnet, was introduced into the ICS of at least one of the cascade halls and managed to take surreptitious control of the centrifuges, causing them to spin periodically and repeatedly at rates damaging to sustained physical operation. The net effect of the attack is still unclear, but at a minimum, it ranged from disruption of the production process up to potential permanent damage to the affected centrifuges.

3.6.4.2 Approach Many characteristics of Stuxnet are unprecedented and stand as the inflection point that ushers in a new era of system attack methodology and cyber-physical system targeting. Illuminating forensic analysis of the Stuxnet code was conducted by several well-known cybersecurity firms, with detailed postmortems covered in two documents from the Institute for Science and International Security, "Did Stuxnet Take Out 1000 Centrifuges at the Natanz Enrichment Plant?" (Albright et al., 2010) and "Stuxnet Malware and Natanz: Update of ISIS December 22, 2010 Report" (Albright et al., 2011). This analysis is beneficial in expanding the risk landscape that systems engineers should consider during design. Below are some concepts that are concerned in the context of Stuxnet:

- *Knowing what to do (intelligence)*—To be successful, a threat has to be able to take advantage of the targeted system(s). It is uncertain how the perpetrators knew what specific devices were employed in what configuration at Natanz; but after the Stuxnet code was analyzed, Natanz was clearly identified as the target. Stuxnet infected many sites other than Natanz, but it would only activate if that site was configured to certain specific system specifications. The perpetrators needed specific system configuration information to know how to cause damage and also to know how to single out the target among many similar but not identical facilities elsewhere. Systems engineers need to consider that adversaries will attempt to gain intelligence on a system and must consider methods to prevent this.
- *Crafting the code*—A zero-day attack is one that exploits a previously unknown vulnerability in a computer application, one that developers have had no time to address and patch. Stuxnet attacked Windows systems outside the FEP using a variety of zero-day exploits and stolen certificates to get proper insertion into the operating system and then initiated a multistage propagation mechanism that started with Universal Serial Bus (USB) removable media infected outside the FEP and ended with code insertion into the ICS inside the FEP. Systems engineers need to be prepared for many different attack vectors (including internal threats) and must consider them during system design.
- *Jumping the air gap*—It is widely believed that Stuxnet crossed the air gap on a USB removable media device, which had been originally infected on a computer outside of the FEP and carried inside. But it is also suggested that the supply chain for PLCs may have been at least one additional infection vector. Whatever the methods, the air gap was crossed multiple times. USB removable media could have also affected a bidirectional transfer of information, sending out detailed intelligence about device types connected to the FEP network subsequently relayed to remote servers outside of the control of the facility. Systems engineers always need to remember that threats to the system are both inside and outside the system boundary.
- *Dynamic updating*—Analysis shows that the attack code, once inserted, could be updated and changed over time, perhaps to take advantage of new knowledge or to implement new objectives. Stuxnet appears to be continuously updated, with new operational parameters reintroduced as new air gap crossings occur. Systems engineers need to prepare for situations after a successful attack has occurred.

3.6.4.3 Conclusions As the complexity and technology of systems change, the systems engineer's perspective needs to adjust accordingly. The increasing use of digital-based technologies in system design offers enormous benefits to everyone. However, the introduction of digital technologies also brings different risks than previously dealt with by SE. The case study earlier illustrates a point in time behind us, and the adversarial community continues to evolve new methods. The lesson

of this case study is that systems engineers need to understand the threats toward their system(s), be cognizant that attacks can and will occur, and be proactive in protecting their system(s). Robust and dynamic system security needs full engagement of SE. A database that systems engineers should be aware of is maintained by the National Institute of Standards and Technology (NIST, 2012).

3.6.5 Case 5: Design for Maintainability—Incubators

Note: This case study is excerpted from "Where Good Ideas Come From: The Natural History of Innovation" (Johnson, 2010).

3.6.5.1 Background In the late 1870s, a Parisian obstetrician named Stephane Tarnier was visiting the Paris Zoo where they had farm animals. While there, he conceived the idea of adapting a chicken incubator to use for human newborns, and he hired "the zoo's poultry raiser to construct a device that would perform a similar function for human newborns." At the time, infant mortality was staggeringly high "even in a city as sophisticated as Paris. One in five babies died before learning to crawl, and the odds were far worse for premature babies born with low birth weights." Tarnier installed his incubator for newborns at Maternité de Paris and embarked on a quick study of 500 babies. "The results shocked the Parisian medical establishment: while 66 percent of low-weight babies died within weeks of birth, only 38 percent died if they were housed in Tarnier's incubating box. … Tarnier's statistical analysis gave newborn incubation the push that it needed: within a few years the Paris municipal board required that incubators be installed in all the city's maternity hospitals."

"Modern incubators, supplemented with high-oxygen therapy and other advances, became standard equipment in all American hospitals after the end of World War II, triggering a spectacular 75 percent decline in infant mortality rates between 1950 and 1998."… "In the developing world, however, the infant mortality story remains bleak. Whereas infant deaths are below ten per thousand births throughout Europe and the United States, over a hundred infants die per thousand (births) in countries like Liberia and Ethiopia, many of them premature babies that would have survived with access to incubators. But modern incubators are complex, expensive things. A standard incubator in an American hospital might cost more than $40,000 (about €30,000). But the expense is arguably the smaller hurdle to overcome. Complex equipment breaks and when it breaks you need the technical expertise to fix it, and you need replacement parts. In the year that followed the 2004 Indian Ocean tsunami, the Indonesian city of Meulaboh received eight incubators from a range of international relief organizations. By late 2008, when an MIT professor named Timothy Prestero visited the hospital, all eight were out of order, the victims of power surges and tropical humidity, along with the hospital staff's inability to read the English repair manual. The Meulaboh incubators were a representative sample: some studies suggest that as much as 95% of medical technology donated to developing countries breaks within the first 5 years of use."

3.6.5.2 Approach "Prestero had a vested interest in those broken incubators, because the organization he founded, Design that Matters, had been working for several years on a scheme for a more reliable, and less expensive, incubator, one that recognized complex medical technology was likely to have a very different tenure in a developing world context than it would in an American or European hospital. Designing an incubator for a developing country wasn't just a matter of creating something that worked; it was also a matter of designing something that would break in a non-catastrophic way. You couldn't guarantee a steady supply of spare parts, or trained repair technicians. So instead, Prestero and his team decided to build an incubator out of parts that were already abundant in the developing world. The idea had originated with a Boston doctor named Jonathan Rosen, who had observed that even the smaller towns of the developing world seemed to be able to keep automobiles in working order. The towns might lack air conditioning and laptops and cable television, but they managed to keep their Toyota 4Runners on the road. So Rosen approached Prestero with an idea: What if you made an incubator out of automobile parts?"

"Three years after Rosen suggested the idea, the Design that Matters team introduced a prototype device called NeoNurture. From the outside, it looked like a streamlined modern incubator, but its guts were automotive. Sealed-beam headlights supplied the crucial warmth; dashboard fans provided filtered air circulation; door chimes sounded alarms. You could power the device via an adapted cigarette lighter, or a standard-issue

motorcycle battery. Building the NeoNurture out of car parts was doubly efficient, because it tapped both the local supply of parts themselves and the local knowledge of automobile repair. These were both abundant resources in the developing world context, as Rosen liked to say. You didn't have to be a trained medical technician to fix the NeoNurture; you didn't even have to read the manual. You just needed to know how to replace a broken headlight."

3.6.5.3 Conclusions Systems engineers need to consider issues like maintainability, producibility, and supportability at the project outset in the concept stage. It is too late to add these in during the production stage.

4

TECHNICAL PROCESSES

The ISO/IEC/IEEE 15288 technical processes and supporting process activities are invoked throughout the life cycle stages of a system. Technical processes are defined in ISO/IEC/IEEE 15288 as follows:

> [6.4] The Technical Processes are used to define the requirements for a system, to transform the requirements into an effective product, to permit consistent reproduction of the product where necessary, to use the product to provide the required services, to sustain the provision of those services and to dispose of the product when it is retired from service.

Technical processes enable systems engineers to coordinate the interactions between engineering specialists, other engineering disciplines, system stakeholders and operators, and manufacturing. They also address conformance with the expectations and legislated requirements of society. These processes lead to the creation of a sufficient set of requirements and resulting system solutions that address the desired capabilities within the bounds of performance, environment, external interfaces, and design constraints. Without the technical processes, the risk of project failure would be unacceptably high. As illustrated in Figure 4.1, the technical processes begin with the development of needs and requirements (Ryan, 2013):

- *Needs*—Per the Oxford English Dictionary, a need is a thing that is wanted or required. For a system, needs are often capabilities or things that are lacking but wanted or desired by one or more stakeholders. These can be viewed in at least three contexts in which SE is performed: (i) projects with customers internal to the enterprise that is doing the engineering, (ii) development under an agreement with an external entity, and (iii) entrepreneurial product development in anticipation of future sales.
- *Requirements*—Requirements are formal structured statements that can be verified and validated. There may be more than one requirement defined for each need.

> *Note: One underlying principle illustrated by Figure* 4.1 *is that when a decision is made to satisfy a need, that need gives rise to a corresponding requirement or set of requirements.*

ISO/IEC/IEEE 15288 includes 14 technical processes, the roles of the first four of which are illustrated in Figure 4.1:

- *Business or mission analysis process*—Requirements definition begins with the business vision of the

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.

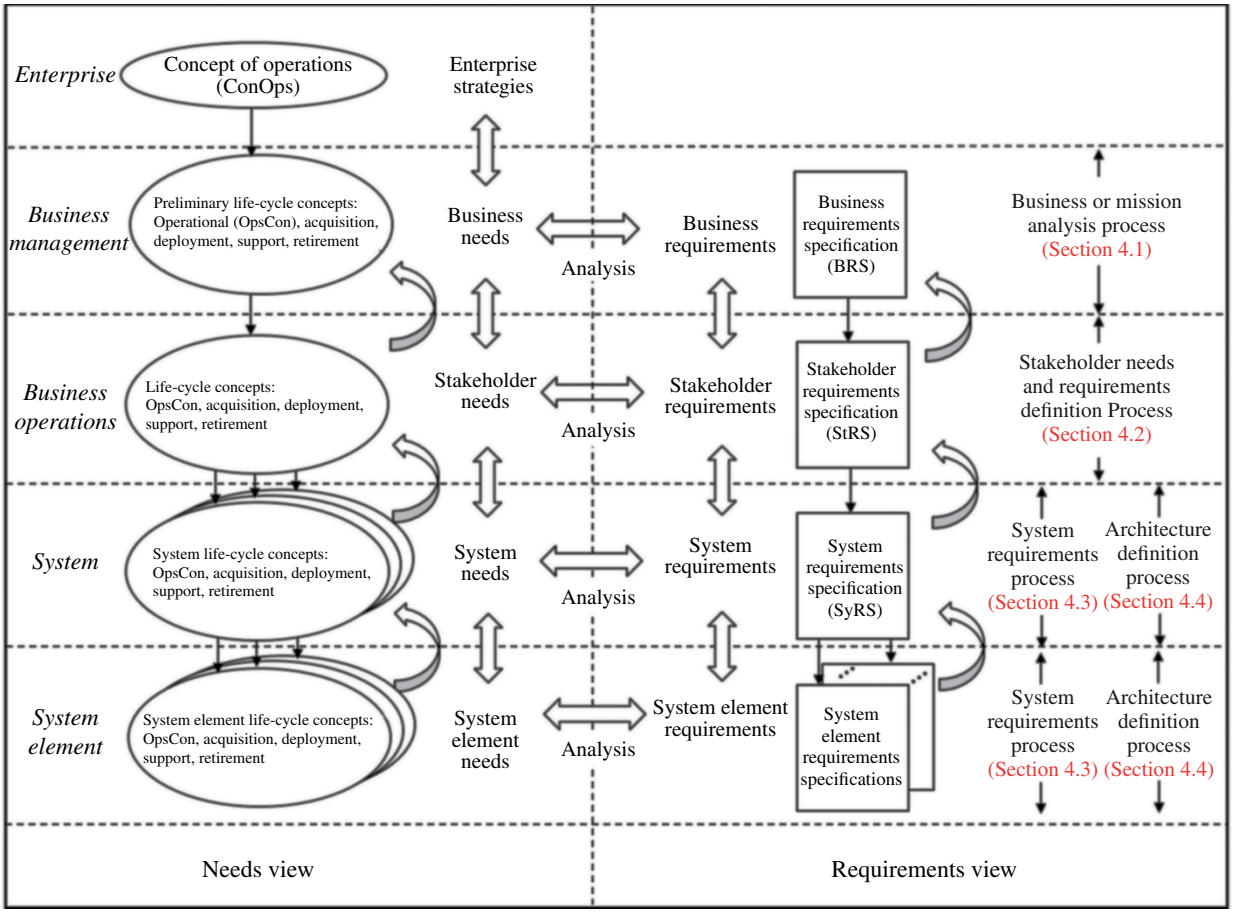


FIGURE 4.1 Transformation of needs into requirements. Reprinted with permission from Mike Ryan. All other rights reserved.

organization or enterprise, the concept of operations (ConOps), and other organization strategic goals and objectives from which business management define business needs (aka mission needs). These needs are supported by preliminary life cycle concepts—acquisition concept, deployment concept, operational concept (OpsCon), support concept, and retirement concept—see Section 4.1.2.2 for a detailed description of the roles of the ConOps and the OpsCon. Business needs are then elaborated and formalized into business requirements, which are often captured in a Business Requirements Specification (BRS).

- *Stakeholder needs and requirements definition process*—Using the enterprise-level ConOps from the acquiring enterprise and the system-level preliminary OpsCon from the development enterprise as guidance, requirements engineers lead stakeholders from business operations through a structured process to elicit stakeholder needs (in the form of a refined system-level OpsCon and other life cycle concepts). Stakeholder needs are then transformed by requirements engineers into a formal set of stakeholder requirements, which are often captured in a Stakeholder Requirements Specification (StRS).
- *System requirements definition process*—The stakeholder requirements in the StRS are then transformed by requirements engineers into system requirements, which are often contained in a System Requirements Specification (SyRS).
- *Architecture definition process*—Alternative system architectures are defined and one is selected.

- *Design definition process*—System elements are defined in sufficient detail to enable implementation consistent with the selected system architecture.
- *System analysis process*—Mathematical analysis, modeling, and simulation are used to support the other technical processes.
- *Implementation process*—System elements are realized to satisfy system requirements, architecture, and design.
- *Integration process*—System elements are combined into a realized system.
- *Verification process*—Evidence is provided that the system, the system elements, and the work products in the life cycle meet the specified requirements.
- *Transition process*—The system moves into operations in a planned, orderly manner.
- *Validation process*—Evidence is provided that the system, the system elements, and the work products in the life cycle will achieve their intended use in the intended operational environment.
- *Operation process*—The system is used.
- *Maintenance process*—The system is sustained during operations.
- *Disposal process*—The system or system elements are deactivated, disassembled, and removed from operations.

4.1 BUSINESS OR MISSION ANALYSIS PROCESS

4.1.1 Overview

4.1.1.1 Purpose As stated in ISO/IEC/IEEE 15288,

> [6.4.1.1] The purpose of the Business or Mission Analysis process is to define the business or mission problem or opportunity, characterize the solution space, and determine potential solution class(es) that could address a problem or take advantage of an opportunity.

4.1.1.2 Description Business or mission analysis initiates the life cycle of the system of interest (SOI) by defining the problem domain; identifying major stakeholders; identifying environmental conditions and constraints that bound the solution domain; developing preliminary life cycle concepts for acquisition, operations, deployment, support, and retirement; and developing the business requirements and validation criteria. Figure 4.2 is the IPO diagram for the business or mission analysis process.

4.1.1.3 Inputs/Outputs Inputs and outputs for the business or mission analysis process are listed in Figure 4.2. Descriptions of each input and output are provided in Appendix E.

4.1.1.4 Process Activities The business or mission analysis process includes the following activities and tasks:

- *Prepare for business or mission analysis.*
  - Establish a strategy for business or mission analysis, including the need for and requirements of any enabling systems, products, or services.
- *Define the problem or opportunity space.*
  - Review identified gaps in the organization strategy with respect to desired organization goals or objectives.
  - Analyze the gaps across the trade space.
  - Describe the problems or opportunities underlying the gaps.
  - Obtain agreement on the problem or opportunity descriptions.
- *Characterize the solution space.*
  - Nominate major stakeholders (individuals or groups). Business owners nominate the major stakeholders who are to be involved in the acquisition, operation, support, and retirement of the solution.
  - Define preliminary OpsCon. An OpsCon describes how the system works from the operator's perspective. The preliminary OpsCon summarizes the needs, goals, and characteristics of the system's user and operator community. The OpsCon also identifies the system context and system interfaces (i.e., the operational environment; see Elaboration for more detail).
  - Define other preliminary life cycle concepts. The business owners identify preliminary life cycle concepts in so far as they may wish to scope any aspect of the acquisition, deployment, support, and retirement of the solution.
  - Establish a comprehensive set of alternative solution classes.

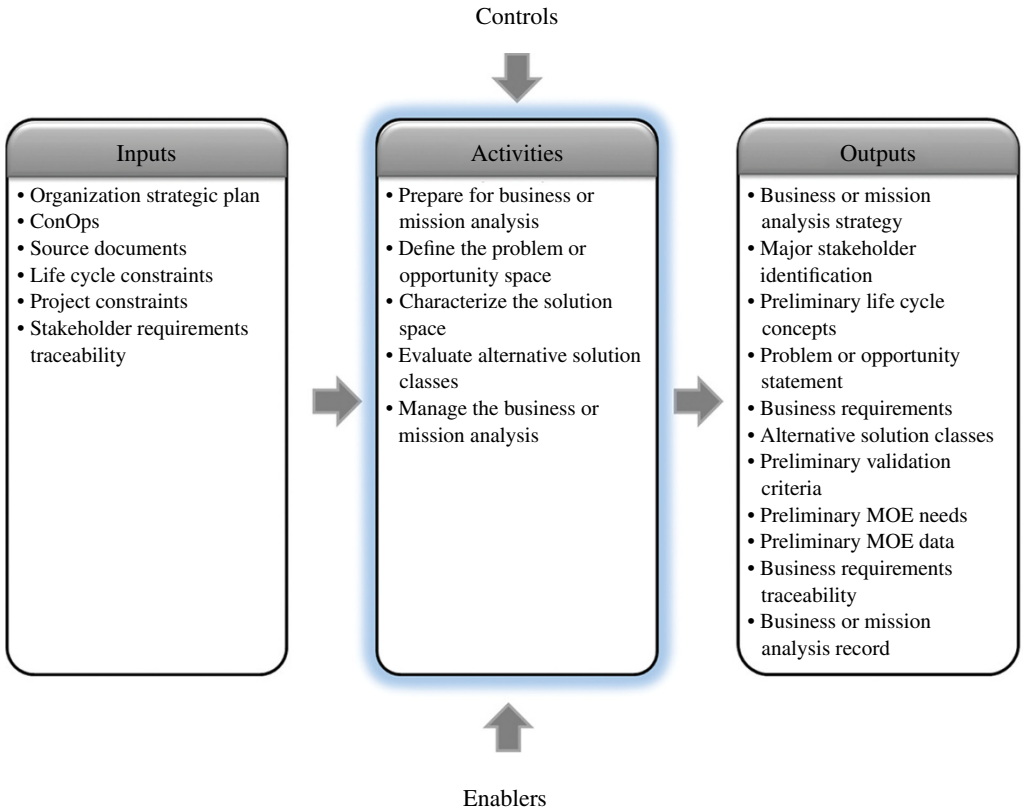


FIGURE 4.2 IPO diagram for business or mission analysis process. INCOSE SEH original figure created by Shortell and Walden. Usage per the INCOSE Notices page. All other rights reserved.

- *Evaluate alternative solution classes.*
  - Evaluate the set of alternative solution classes and select the preferred class(es). Appropriate modeling, simulation, and analytical techniques help determine the feasibility and value of the alternative candidate solutions.
  - Ensure that the preferred alternative solution class(es) has been validated in the context of the proposed business or mission strategy. Feedback on feasibility, market factors, and alternatives is also provided for use in completing the organization strategy and further actions.
- *Manage the business or mission analysis.*
  - Establish and maintain traceability of analysis results, such as requirements and preliminary life cycle concepts.
  - Provide baseline information for configuration management.

4.1.2 Elaboration

4.1.2.1 Nominate Major Stakeholders Although the detailed identification of stakeholders is undertaken in the stakeholder needs and requirements definition process, during business and mission analysis, the business managers are responsible for nominating major stakeholders and often for establishing a stakeholder board. It is fundamentally a business management function to ensure stakeholders are available and willing to contribute to the system development—most stakeholders are heavily occupied in business operations and must be given permission to expend effort and resources on other than their operational tasks.

4.1.2.2 ConOps and OpsCon ANSI/AIAA G-043A-2012 states that the terms "concept of operations" and "operational concept" are often used interchangeably but notes that an important distinction exists in that each has

a separate purpose and is used to meet different ends. This handbook uses these terms so that they are consistent with ANSI/AIAA G-043A-2012 and ISO/IEC/IEEE 29148:2011, the same way in which they are used in the US Department of Defense (DoD) and many other defense forces.

ISO/IEC/IEEE 29148 describes the ConOps as

> The ConOps, at the organization level, addresses the leadership's intended way of operating the organization. It may refer to the use of one or more systems, as black boxes, to forward the organization's goals and objectives. The ConOps document describes the organization's assumptions or intent in regard to an overall operation or series of operations of the business with using the system to be developed, existing systems, and possible future systems. This document is frequently embodied in long-range strategic plans and annual operational plans. The ConOps document serves as a basis for the organization to direct the overall characteristics of the future business and systems, for the project to understand its background, and for the users of [ISO/IEC/IEEE 29148] to implement the stakeholder requirements elicitation.

ISO/IEC/IEEE 29148 describes the OpsCon as

> A System Operational Concept (OpsCon) document describes what the system will do (not how it will do it) and why (rationale). An OpsCon is a user-oriented document that describes system characteristics of the to-be-delivered system from the user's viewpoint. The OpsCon document is used to communicate overall quantitative and qualitative system characteristics to the acquirer, user, supplier and other organizational elements.

Both the ConOps and the OpsCon are prepared by the organization that has the business need for the SOI. The ConOps is developed by/for the leadership at the enterprise level of the organization using the SOI. In some acquisitions, the ConOps may not be formalized, but rather implied by other business concepts and/or strategies. The OpsCon is prepared at the business level. Business management begins with the preparation of the preliminary OpsCon, which summarizes business needs from an operational perspective for the solution classes that address the problem or opportunity. The preliminary OpsCon is then elaborated and refined by business operations into the OpsCon by engagement with the nominated stakeholders during the stakeholder needs and requirements definition process—the final OpsCon therefore contains both the business needs and the stakeholder needs. The OpsCon may be iteratively refined as a result of feedback obtained through the conduct of the system requirements definition and architecture definition processes.

4.1.2.3 Other Life Cycle Concepts The OpsCon is just one of the life cycle concepts required to address the stakeholder needs across the system life cycle. Preliminary concepts are established in the business or mission analysis process to the extent needed to define the problem or opportunity space and characterize the solution space. These concepts are further refined in the stakeholder needs and requirements definition process. In addition to the operational aspects, other related life cycle concepts are required to address:

- *Acquisition concept*—Describes the way the system will be acquired including aspects such as stakeholder engagement, requirements definition, design, production, and verification. The supplier enterprise(s) may need to develop more detailed concepts for production, assembly, verification, transport of system, and/or system elements.
- *Deployment concept*—Describes the way the system will be validated, delivered, and introduced into operations, including deployment considerations when the system will be integrated with other systems that are in operation and/or replace any systems in operation.
- *Support concept*—Describes the desired support infrastructure and manpower considerations for supporting the system after it is deployed. A support concept would address operating support, engineering support, maintenance support, supply support, and training support.
- *Retirement concept*—Describes the way the system will be removed from operation and retired, including the disposal of any hazardous materials used in or resulting from the process and any legal obligations—for example, regarding IP rights protection, any external financial/ownership interests, and national security concerns.

4.1.2.4 Business Requirements and Validation *Specify business requirements*—It is often helpful to specify business requirements as part of the business and

mission analysis process. Business requirements are often contained in a BRS, which the *Guide to the Business Analysis Body of Knowledge* (IIBA, 2009) calls the business requirement document. The term "specification" has some variation in use in various industries, but it is used here to be synonymous with "document"—that is, business requirements are captured in the BRS, stakeholder requirements in the StRS, and system requirements in the SyRS.

Define business validation criteria The business must define how it will know that the solution provided will meet the OpsCon. Validation criteria establish critical and desired system performance—thresholds and objectives for system performance parameters that are critical for system success and those that are desired but may be subject to compromise to meet the critical parameters.

4.2 STAKEHOLDER NEEDS AND REQUIREMENTS DEFINITION PROCESS

4.2.1 Overview

4.2.1.1 Purpose As stated in ISO/IEC/IEEE 15288,

> [6.4.2.1] The purpose of the Stakeholder Needs and Requirements Definition process is to define the stakeholder requirements for a system that can provide the capabilities needed by users and other stakeholders in a defined environment.

4.2.1.2 Description Successful projects depend on meeting the needs and requirements of the stakeholders throughout the life cycle. A stakeholder is any entity (individual or organization) with a legitimate interest in the system. When nominating stakeholders, business management will take into account all those who may be affected by or able to influence the system—typically, they would consider users, operators, organization decision makers, parties to the agreement, regulatory bodies, developing agencies, support organizations, and society at large (within the context of the business and proposed solution). When direct contact is not possible, systems engineers find agents, such as marketing or nongovernmental organizations, to represent the concerns of a class of stakeholders, such as consumers or future generations.

After identifying the stakeholders, this process elicits the stakeholder needs that correspond to a new or changed capability or new opportunity. These needs are analyzed and transformed into a set of stakeholder requirements for the operation and effects of the solution and its interaction with the operational and enabling environments. The stakeholder requirements are the primary reference against which the operational capability is validated.

To achieve good results, systems engineers involve themselves in nearly every aspect of a project, pay close attention to interfaces where two or more systems or system elements work together, and establish an interaction network with stakeholders and other organizational units of the organization. Figure 4.3 shows the critical interactions for systems engineers.

The stakeholder requirements govern the system's development and are an essential factor in further defining or clarifying the scope of the development project. If an organization is acquiring the system, this process provides the basis for the technical description of the deliverables in an agreement—typically in the form of a system-level specification and defined interfaces at the system boundaries. Figure 4.4 is the IPO diagram for the stakeholder needs and requirements definition process.

4.2.1.3 Inputs/Outputs Inputs and outputs for the stakeholder needs and requirements definition process are listed in Figure 4.4. Descriptions of each input and output are provided in Appendix E.

4.2.1.4 Process Activities The stakeholder needs and requirements definition process includes the following activities:

- *Prepare for stakeholder needs and requirements definition.*
  - Determine the stakeholders or classes of stakeholders who will participate with systems engineering to develop and define the stakeholder needs and translate these into system requirements, phased throughout the entire life cycle. Capture these results in the ConOps.
  - Determine the need for and requirements of any enabling systems, products, or services.
- *Define stakeholder needs.*
  - Elicit stakeholder needs from the identified stakeholders.

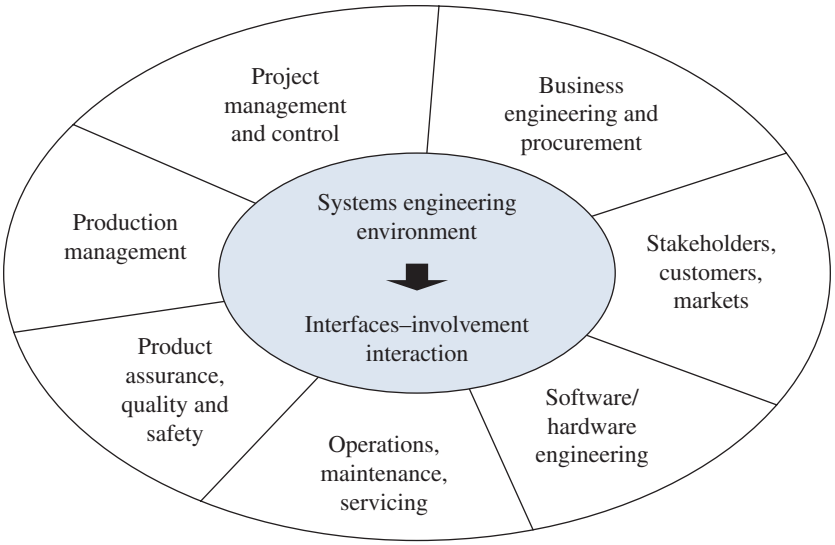


FIGURE 4.3 Key SE interactions. From Stoewer (2005). Figure reprinted with permission from Heinz Stoewer. All other rights reserved.

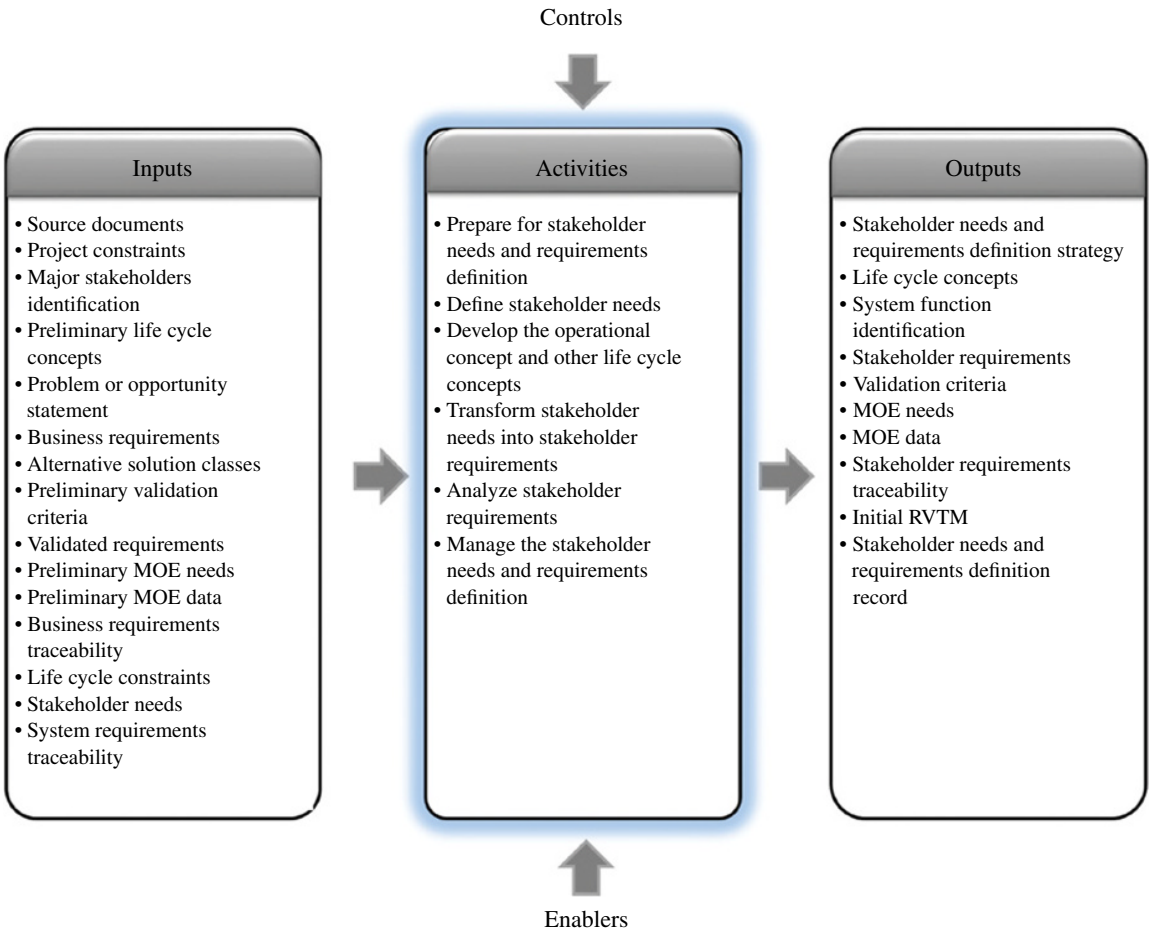


FIGURE 4.4 IPO diagram for stakeholder needs and requirements definition process. INCOSE SEH original figure created by Shortell and Walden. Usage per the INCOSE Notices page. All other rights reserved.

  - Prioritize the stakeholder needs to identify which to focus on.
  - Specify the stakeholder needs.
- *Develop the operational concept and other life cycle concepts.*
  - Identify the expected set of operational scenarios and associated capabilities, behaviors, and responses of the system or solution and environments across the life cycle (in acquisition, deployment, operations, support, and retirement). The scenarios are built to define the life cycle concept documents; the range of anticipated uses of system products; the intended operational environment and the systems' impact on the environment; and interfacing systems, platforms, or products. Scenarios help identify requirements that might otherwise be overlooked. Social and organizational influences also emerge from using scenarios.
  - Define the interactions of the system or solution with the users and the operating, support, and enabling environments.
- *Transform stakeholder needs into stakeholder requirements.*
  - Identify constraints on the solution (imposed by agreements or interfaces with legacy or interoperating systems). The constraints need to be monitored for any interface changes (external or internal) that could alter the nature of the constraint.
  - Specify health, safety, security, environment, assurance, and other stakeholder requirements and functions that relate to critical qualities.
  - Specify stakeholder requirements, consistent with scenarios, interactions, constraints, and critical qualities.
- *Analyze stakeholder requirements.*
  - Define validation criteria for stakeholder requirements. Stakeholder validation criteria include measures of effectiveness (MOEs) and measures of suitability (MOSs), which are the "operational" measures of success that are closely related to the achievement of the mission or operational objective being evaluated, in the intended operational environment under a specified set of conditions (i.e., how well the solution achieves the intended purpose). These measures reflect overall customer/user satisfaction (e.g., performance, safety, reliability, availability, maintainability, and workload requirements).
  - Analyze the set of requirements for clarity, completeness, and consistency. Include review of the analyzed requirements to the applicable stakeholders to ensure the requirements reflect their needs and expectations.
  - Negotiate modifications to resolve unrealizable or impractical requirements.
- *Manage the stakeholder needs and requirements definition.*
  - Establish with stakeholders that their requirements are expressed correctly.
  - Record stakeholder requirements in a form suitable for maintenance throughout the system life cycle (and beyond for historical or archival purposes).
  - Establish and maintain through the life cycle a traceability of stakeholder needs and requirements (e.g., to the stakeholders, other sources, organizational strategy, and business or mission analysis results).
  - Provide baseline information for configuration management.

4.2.2 Elaboration

Within the context of ISO/IEC/IEEE 15288, requirements (business, stakeholder, and system) are drivers for the majority of the system life cycle processes. Depending on the system development model, stakeholder requirements capture should be conducted nominally once near the beginning of the development cycle or as a continuous activity. Regardless, the reason for eliciting and analyzing requirements is the same—understand the needs of the stakeholders well enough to support the architecture definition and design definition processes.

4.2.2.1 Identify Stakeholders Systems engineers engage with legitimate stakeholders of the system. The major stakeholders at the business management level will have been nominated during the business or mission analysis process—here, systems engineers are interested in identifying stakeholders from the business operations level.

One of the biggest challenges in system development is the identification of the set of stakeholders from whom

requirements should be elicited. Customers and eventual end users are relatively easy to identify, but regulatory agencies and other interested parties that may reap the consequences of the deployed system should also be sought out and heard. Stakeholders can include the stakeholders of interoperating systems and enabling systems as these will usually impose constraints that need to be identified and considered. In sustainable development, this includes finding representation for future generations.

4.2.2.2 Elicit Stakeholder Needs Determining stakeholder needs requires the integration of a number of disparate views, which may not necessarily be harmonious. As the SE process is applied, a common paradigm for examining and prioritizing available information and determining the value of added information should be created. Each of the stakeholder's views of the needed systems can be translated to a common top-level system description that is understood by all participants, and all decision-making activities recorded for future examination. Under some circumstances, it may not be practical to elicit needs from the stakeholder but rather from the marketing organization or other surrogates. There may be stakeholders who oppose the system. These stakeholders or detractors of the system are first considered in establishing consensus needs. Beyond this, they are addressed through the risk management process, the threat analysis of the system, or the system requirements for security, adaptability, or resilience.

Systems engineering should support program and project management in defining what must be done and gathering the information, personnel, and analysis tools to elaborate the business requirements. This includes gathering stakeholder needs, system/project constraints (e.g., costs, technology limitations, and applicable specifications/legal requirements), and system/project "drivers," such as capabilities of the competition, military threats, and critical environments.

The outputs of the stakeholder needs and requirements definition process should be sufficient definition of the business and stakeholder needs and requirements to gain authorization and funding for program initiation through the portfolio management process. The output should also provide necessary technical definition to the acquisition process to generate a request for proposal (RFP) if the system is to be acquired through a contract acquisition process or to gain authorization to develop and market the system if market driven. These outputs can be captured in life cycle concept documents (particularly the OpsCon) and the StRS, which often are used to support the generation of a statement of work (SOW), and/or an RFP, both of which are artifacts of the acquisition process. Contributing users rely on well-defined completion criteria to indicate the successful definition of user and stakeholder needs:

- User organizations have gained authorization for new system acquisition.
- Program development organizations have prepared a SOW, StRS, and gained approval for new system acquisition. If they are going to use support from outside the company, they have issued an RFP and selected a contractor.
- Potential contractors have influenced the acquisition needs, submitted a proposal, and have been selected to develop and deliver the system.
- If the system is market driven, the marketing group has learned what consumers want to buy. For expensive items (e.g., aircraft), they have obtained orders for the new systems.
- If the system is market and technology driven, the development team has obtained approval to develop the new system from the corporation.

Since requirements come from multiple sources, eliciting and capturing requirements constitutes a significant effort on the part of the systems engineer. The OpsCon describes the intended operation of the system to be developed and helps the systems engineer understand the context within which requirements need to be captured and defined. Techniques for requirements elicitation include interviews, focus groups, the Delphi method, and soft systems methodology, to name a few. Trade-off analysis and simulation tools can also be used to evaluate mission operational alternatives and select the desired mission alternative. Tools for capturing and managing requirements are many and varied.

The source requirements captured by carrying out this activity are only a portion of the total stakeholder requirements. As such, source requirements will be expanded by a number of activities designed to break down the broad requirement statements and reveal the need for additional clarification, which will lead to either revision of the written source material or additional source documents, such as meeting minutes.

4.2.2.3 Initialize the Requirements Database It is essential to establish a database of baseline requirements traceable to the source needs (and subsequently to system requirements) to serve as a foundation for later refinement and/or revision by subsequent activities in the SE process. The requirements database must first be populated with the source documents that provide the basis for the total set of system requirements that will govern its design.

4.2.2.4 Develop the Life Cycle Concepts The word "scenario" is often used to describe a single thread of behavior; in other cases, it describes a superset of many single threads operating concurrently. Scenarios and what-if thinking are essential tools for planners who must cope with the uncertainty of the future. Scenario thinking can be traced back to the writings of early philosophers, such as Plato and Seneca (Heijden et al., 2002). As a strategic planning tool, scenario techniques have been employed by military strategists throughout history. Building scenarios serves as a methodology for planning and decision making in complex and uncertain environments. The exercise makes people think in a creative way, observations emerge that reduce the chances of overlooking important factors, and the act of creating the scenarios enhances communications within and between organizations. Scenario building is an essentially human activity that may involve interviews with operators of current/similar systems, potential end users, and meetings of an Interface Working Group (IFWG). The results of this exercise can be captured in many graphical forms using modeling tools and simulations.

Creation or upgrade of a system shares the same uncertainty regarding future use and emergent properties of the system. The stakeholder needs and requirements definition process captures the understanding of stakeholder needs in a series of life cycle concept documents, each focused on a specific life cycle stage: acquisition concept, deployment concept, OpsCon, support concept, and retirement concept. Each of these categories of life cycle concepts is discussed in Section 4.1.2.3. A primary goal of a concept document is to capture, early in the system life cycle, an implementation-free understanding of stakeholder needs by defining what is needed, without addressing how to satisfy the need. It captures behavioral characteristics required of the system in the context of other systems with which it interfaces, and captures the manner in which people will interact with the system for which the system must provide capabilities. Understanding these operational needs typically produces:

- A source of specific and derived requirements that meet the needs and objectives of the customer and user
- Invaluable insight for SE and designers as they define design, develop, verify, and validate the system
- Diminished risk of latent system defects in the delivered operational systems

If the system is for a military customer, there may be several required views of the system driven by architectural frameworks. These are defined, for example, in the US Department of Defense Architecture Framework (DoDAF, 2010) and in the UK Ministry of Defense Architecture Framework (MoDAF, n.d.) (OMG, 2013a).

The primary objective is to communicate with the end user of the system during the early specification stages to ensure that stakeholder needs (particularly the operational needs) are clearly understood and the rationale for performance requirements is incorporated into the decision mechanism for later inclusion in the system requirements and lower-level specifications. Interviews with operators of current/similar systems, potential users, interface meetings, IPO diagrams, functional flow block diagrams (FFBD), timeline charts, and N^2 charts provide valuable stakeholder input toward establishing a concept consistent with stakeholder needs. Other objectives are as follows:

1. To provide traceability between operational needs and the captured source requirements
2. To establish a basis for requirements to support the system over its life, such as personnel requirements, support requirements, etc.
3. To establish a basis for verification planning, system-level verification requirements, and any requirements for environmental simulators
4. To generate operational analysis models to test the validity of external interfaces between the system and its environment, including interactions with external systems
5. To provide the basis for computation of system capacity, behavior under-/overload, and mission-effectiveness calculations

6. To validate requirements at all levels and to discover implicit requirements overlooked from other sources

Since the preliminary life cycle concepts have provided a broad description of system behavior, a starting point for further developing the concept is to begin by identifying outputs generated by external systems (modified as appropriate by passing through the natural system environment), which act as stimuli to the SOI and cause it to take specified actions and produce outputs, which in turn are absorbed by external systems. These single threads of behavior eventually cover every aspect of operational performance, including logistical modes of operation, operation under designated conditions, and behavior required when experiencing mutual interference with multiobject systems.

Aggregation of these single threads of behavior represents a dynamic statement of what the system is required to do and how it is to be acquired, deployed, operated, supported, and retired. No attempt is made at this stage to define a complete OpsCon or to allocate functions to hardware or software elements (this comes later during architectural design). The life cycle concepts are essentially definitions of the functional concepts and rationale *from the stakeholder perspective*. The life cycle concepts are further developed as follows:

1. Start with the source operational requirements; deduce a set of statements describing the higher-level, mission-oriented needs.
2. Review the system needs with stakeholders and record the conflicts.
3. Define and model the operational boundaries.
4. For each model, generate a context diagram to represent the model boundary.
5. Identify all of the possible types of observable input and output events that can occur between the system and its interacting external systems.
6. If the inputs/outputs are expected to be significantly affected by the environment between the system and the external systems, add concurrent functions to the IPO diagram to represent these transformations and add input and output events to the database to account for the differences in event timing between when an output is emitted and when an input is received.
7. Record the existence of a system interface between the system and the environment or external system.
8. For each class of interaction between a part of the system and an external system, create a functional flow diagram to model the sequence of interactions as triggered by the stimuli events generated by the external systems.
9. Add information to trace the function timing from performance requirements and simulate the timing of the functional flow diagrams to confirm operational correctness or to expose dynamic inconsistencies. Review results with users and operational personnel.
10. Develop timelines, approved by end users, to supplement the source requirements.

4.2.2.5 Generate the StRS A draft StRS should be generated to formally represent the stakeholder requirements. The StRS should be traceable to the stakeholder needs and to the BRS.

4.3 SYSTEM REQUIREMENTS DEFINITION PROCESS

4.3.1 Overview

4.3.1.1 Purpose As stated in ISO/IEC/IEEE 15288,

> [6.4.3.1] The purpose of the System Requirements Definition process is to transform the stakeholder, user-oriented view of desired capabilities into a technical view of a solution that meets the operational needs of the user.

4.3.1.2 Description System requirements are the foundation of the system definition and form the basis for the architecture, design, integration, and verification. Each requirement carries a cost. It is therefore essential that a complete but minimum set of requirements be established from defined stakeholder requirements early in the project life cycle. Changes in requirements later in the development cycle can have a significant cost impact on the project, possibly resulting in cancellation.

The system requirements definition process generates a set of system requirements from the supplier's perspective using the stakeholder requirements that reflect the user's perspective as the basis. The system

requirements specify the system characteristics, attributes, functions, and performance that will meet the stakeholder requirements.

Requirements definition is both iterative and recursive (see Section 3.4.1). According to ISO/IEC/IEEE 29148, *Requirements engineering* (2011),

> When the application of the same process or set of processes is repeated on the same level of the system, the application is referred to as iterative. Iteration is not only appropriate but also expected. New information is created by the application of a process or set of processes. Typically this information takes the form of questions with respect to requirements, analyzed risks or opportunities. Such questions should be resolved before completing the activities of a process or set of processes.
>
> When the same set of processes or the same set of process activities are applied to successive levels of system elements within the system structure, the application form is referred to as recursive. The outcomes from one application are used as inputs to the next lower (or higher) system in the system structure to arrive at a more detailed or mature set of outcomes. Such an approach adds value to successive systems in the system structure.

Thus, iteration between this process and others is expected as more information is available and analysis is performed. This process continues to be recursively applied to define the requirements for each system element.

The output of the process must be compared for traceability to and consistency with the stakeholder requirements, without introducing implementation biases, before being used to drive the architecture definition process. The system requirements definition process adds the verification criteria to the defined system requirements. Figure 4.5 is the IPO diagram for the system requirements definition process.

4.3.1.3 Inputs/Outputs Inputs and outputs for the system requirements definition process are listed in Figure 4.5. Descriptions of each input and output are provided in Appendix E.

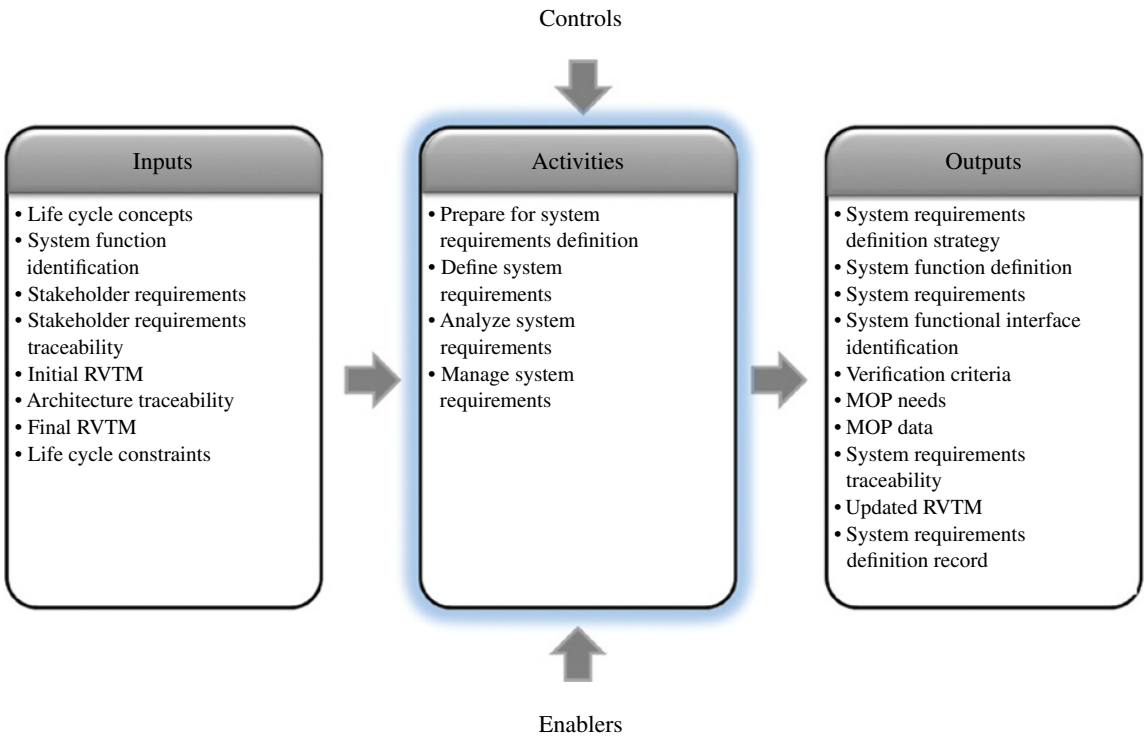


FIGURE 4.5 IPO diagram for the system requirements definition process. INCOSE SEH original figure created by Shortell and Walden. Usage per the INCOSE Notices page. All other rights reserved.

4.3.1.4 Process Activities The system requirements definition process includes the following activities:

- *Prepare for system requirements definition.*
  - Establish the approach for defining the system requirements. This includes system requirements methods, tools, and the need for and requirements of any enabling systems, products, or services.
  - In conjunction with the architecture definition process, determine the system boundary, including the interfaces, that reflects the operational scenarios and expected system behaviors. This task includes identification of expected interactions of the system with systems external to the system (control) boundary as defined in negotiated interface control documents (ICDs).
- *Define system requirements.*
  - Identify and define the required system functions. These functions should be kept implementation independent, not imposing additional design constraints. Define conditions or design factors that facilitate and foster efficient and cost-effective life cycle functions (e.g., acquisition, deployment, operation, support, and retirement). Also, include the system behavior characteristics.
  - Identify the stakeholder requirements or organizational limitations that impose unavoidable constraints on the system and capture those constraints.
  - Identify the critical quality characteristics that are relevant to the system, such as safety, security, reliability, and supportability.
  - Identify the technical risks that need to be accounted for in the system requirements.
  - Specify system requirements, consistent with stakeholder requirements, functional boundaries, functions, constraints, critical performance measures, critical quality characteristics, and risks. System requirements may be captured in the SyRS. Additionally, a documentation tree may be developed to define the hierarchy of system definition products being developed. The documentation tree evolves with the interaction of the system requirements definition, architecture definition, and design definition processes. Capture the associated rationale, as the requirements are specified.
- *Analyze system requirements.*
  - Analyze the integrity of the system requirements to ensure that each requirement or set of requirements possess overall integrity. (See Section 4.3.2.2 for characteristics of a good requirement or set of requirements.)
  - Provide analysis results to applicable stakeholders to ensure that the specified system requirements adequately reflect the stakeholder requirements.
  - Negotiate modifications to resolve issues identified in the requirements.
  - Define verification criteria—critical performance measures that enable the assessment of technical achievement. System verification criteria include measures of performance (MOPs) and technical performance measures (TPMs), which are the "implementation" measures of success that should be traceable to the MOEs and MOSs (operational perspective) with the relationships defined.
- *Manage system requirements.*
  - Ensure agreement among key stakeholders that the requirements adequately reflect the stakeholder intentions.
  - Establish and maintain traceability between the system requirements and the relevant elements of the system definition (e.g., stakeholder requirements, architecture elements, interface definitions, analysis results, verification methods or techniques, and allocated, decomposed, and derived requirements.)
  - Maintain throughout the system life cycle the set of system requirements together with the associated rationale, decisions, and assumptions.
  - Provide baseline information for configuration management.

4.3.2 Elaboration

This section elaborates and provides "how-to" information on the requirements analysis and management. Other key information on requirements can be found in ISO/IEC TR 19760, *Systems engineering—A guide to the application of ISO/IEC 15288* (2003), and ISO/IEC/IEEE 29148, *Requirements engineering* (2011), and in EIA 632, *Standard—Processes for engineering a system*

(ANSI/EIA, 2003), Requirements 14, 15, and 16, and Annex C3.1 a, b, and c.

4.3.2.1 Requirements Definition and Analysis Concepts Requirements definition and analysis, like the set of SE processes, is an iterative activity in which new requirements are identified and constantly refined as the concept develops and additional details become known. The requirements are analyzed, and deficiencies and cost drivers are identified and reviewed with the customer to establish a requirements baseline for the project.

An objective of requirements analysis is to provide an understanding of the interactions between the various functions and to obtain a balanced set of requirements based on user objectives. Requirements are not developed in a vacuum. An essential part of the requirements development process is the OpsCon, the implicit design concept that accompanies it, and associated demands of relevant technology. Requirements come from a variety of sources, including the customer/users, regulations/codes, and corporate entities.

This complex process employs performance analysis, trade studies, constraint evaluation, and cost–benefit analysis. System requirements cannot be established without determining their impact (achievability) on lower-level elements. Therefore, requirements definition and analysis is an iteration and balancing process that works both "top-down" (called allocation and flowdown) and "bottom-up." Once the top-level set of system requirements has been established, it is necessary to allocate and flow them down to successively lower levels. As the allocation and flowdown process is repeated, it is essential that traceability be maintained to ensure that all system-level requirements are satisfied in the resulting design. The resulting requirements database usually contains many attributes for each requirement and is also used in verification. Although it is an objective to avoid requirements that constrain or define aspects of the system implementation, it is not always possible. There are often necessary constraints that need to be reflected. This includes the following:

- *Standards*—Identify standards required to meet quality or design considerations imposed as defined stakeholder requirements or derived to meet organization, industry, or domain requirements.
- *Utilization environments*—Identify the utilization environments and all environmental factors (natural or induced) that may affect system performance, impact human comfort or safety, or cause human error for each of the operational scenarios envisioned for system use.
- *Essential design considerations*—Identify design considerations including human systems integration (e.g., manpower, personnel, training, environment, safety, occupational health, survivability, habitability), system security requirements (e.g., information assurance, antitamper provisions), and potential environmental impact.
- *Design constraints*—Identify design constraints including physical limitations (e.g., weight, form/fit factors), manpower, personnel, and other resource constraints on operation of the system and defined interfaces with host platforms and interacting systems external to the system boundary, including supply, maintenance, and training infrastructures.

4.3.2.2 Characteristics and Attributes of Good Requirements In defining requirements, care should be exercised to ensure the requirement is appropriately crafted. The following characteristics should be considered for every requirement based on ISO/IEC/IEEE 29148, *Systems and software engineering—Life cycle processes—Requirements engineering* (2011), and the *INCOSE Guide for Writing Requirements* (INCOSE RWG, 2012):

- *Necessary*—Every requirement generates extra effort in the form of processing, maintenance, and verification. Only necessary requirements should therefore be included in specifications. Unnecessary requirements are of two varieties: (i) unnecessary specification of design, which should be left to the discretion of the designer, and (ii) a redundant requirement covered in some other combination of requirements.
- *Implementation independent*—Customer requirements may be imposed at any level they desire; however, when customer requirements specify design, it should be questioned. A proper requirement should deal with the entity being specified as a "black box" by describing what transformation is to be performed by the "box." The requirement should specify "what"

is to be done at that level, not "how" it is to be done at that level.

- *Unambiguous*—Requirements must convey what is to be done to the next level of development. Its key purpose is to communicate. Is the requirement clear and concise? Is it possible to interpret the requirement in multiple ways? Are the terms defined? Does the requirement conflict with or contradict another requirement? Each requirement statement should be written to address one and only one concept. Requirements with "and," "or," "commas," or other forms of redundancy can be difficult to verify and should be avoided as it can be difficult to ensure all personnel have a common understanding. Requirements must therefore be written with extreme care. The language used must be clear, exact, and in sufficient detail to meet all reasonable interpretations. A glossary should be used to precisely define often-used terms or terms, such as "process," that could have multiple interpretations.
- *Complete*—The stated requirement should be complete and measurable and not need further amplification. The stated requirement should provide sufficient capability or characteristics.
- *Singular*—The requirement statement should be of only one requirement and should not be a combination of requirements or more than one function or constraint.
- *Achievable*—The requirement must be technically achievable within constraints and requires advances in technology within acceptable risk. It is best if the implementing developer participate in requirements definition. The developer should have the expertise to assess the achievability of the requirements. In the case of items to be subcontracted, the expertise of potential subcontractors is very valuable in the generation of the requirements. Additionally, participation by manufacturing and customers/users can help ensure achievable requirements. When it is not possible to have the right mix of developer, subcontractor, and/or manufacturing expertise during the requirements generation, it is important to have their review at the first point possible to ensure achievability.
- *Verifiable*—Each requirement must be verified at some level by one of the four standard methods (inspection, analysis, demonstration, or test). A customer may specify, "The range shall be as long as possible." This is a valid but unverifiable requirement. This type of requirement is a signal that a trade study is needed to establish a verifiable maximum range requirement. Each verification requirement should be verifiable by a single method. A requirement requiring multiple methods to verify should be broken into multiple requirements. There is no problem with one method verifying multiple requirements; however, it indicates a potential for consolidating requirements. When the system hierarchy is properly designed, each level of specification has a corresponding level of verification during the verification stage. If element specifications are required to appropriately specify the system, element verification should be performed.
- *Conforming*—In many instances, there are applicable government, industry, and product standards, specifications, and interfaces with which compliance is required. An example might be additional requirements placed on new software developments for possible reusability. Another might be standard test interface connectors for certain product classes. In addition, the individual requirements should conform to the organization's standard template and style for writing requirements—when all requirements within the same organization have the same look and feel, each requirement is easier to write, understand, and review.

> *Note: ISO/IEC/IEEE 29148 uses the term "consistent" instead of "conforming," stating that the requirement must be free of conflicts with other requirements. While that is correct, a requirement cannot, in and of itself, be consistent—consistency is more correctly a characteristic of the set of requirements (as described in the following paragraph).*

In addition to the characteristics of individual requirements, the characteristics of a set of requirements as a whole should be addressed to ensure that the set of requirements collectively provides for a feasible solution that meets the stakeholder intentions and constraints. These include:

- *Complete*—The set of requirements contains everything pertinent to the definition of system or system element being specified.

- *Consistent*—The set of requirements is consistent in that the requirements are not contradictory nor duplicated. Terms and abbreviations are used consistently in all requirements, in accordance with the glossary.
- *Feasible/affordable*—The set of requirements can be satisfied by a solution that is obtainable within LCC, schedule, and technical constraints.

> *Note: ISO/IEC/IEEE 29148 uses the term "affordable" instead of "feasible," stating that the set of requirements are feasible within the life cycle constraints of cost, schedule, technical, and regulatory. The INCOSE Guide for Writing Requirements (INCOSE RWG, 2012) includes the word "feasible," stating that "Feasible/Affordable is a more appropriate title for this characteristic of the requirement set."*

- *Bounded*—The set of requirements define the required scope for the solution to meet the stakeholder needs. Consequently, all necessary requirements must be included; irrelevant requirements must be excluded.

In addition to the characteristics listed above, individual requirement statements may have a number of attributes attached to them (either as fields in a database or through relationships with other artifacts):

- *Trace to parent*—A child requirement is one that has been derived or decomposed from the parent—the achievement of all the children requirements will lead to the achievement of the parent requirement. Each of the children requirements must be able to be traced to its parent requirement (and thence to any antecedent requirement and ultimately to the system need/mission).
- *Trace to source*—Each requirement must be able to be traced to its source—this is different from tracing to a parent because it identifies where the requirement came from and/or how it was arrived at (rather than which other requirement is its parent).
- *Trace to interface definition*—The interactions between two systems are described in interface definitions that are often contained in a document that has a title such as an ICD. The interface requirements contained in each of the interacting systems will include reference to where the interaction is defined. This attribute provides a link trace between any of the interface requirements to where the interaction is defined.
- *Trace to peer requirements*—This attribute links requirements that are related to each other (other than parent/child) at the same level. Peer requirements may be related for a number of reasons, such as the following: they may be in conflict, or codependent, or bundled, or a complimentary interface requirement of another system to which the system has an interface.
- *Trace to verification method*—This could be a simple statement of the way in which the requirement is to be verified (inspection, demonstration, test, analysis, simulation), or it could be a more elaborate statement that effectively provides the outline of an appropriate test plan.
- *Trace to verification requirement(s)*—The verification method for each requirement simply states the planned method of verification (inspection, demonstration, test, analysis, simulation). In addition to stating the verification method, some organizations write a set of verification requirements in addition to the system requirements. This is beneficial as it forces the systems engineer to consider how each requirement is to be verified and in doing so helps identify and remove requirements that are not verifiable.
- *Trace to verification results*—The results of each verification will most often be contained in a separate document. This attribute traces each requirement to the associated verification results.
- *Requirements verification status*—It is useful to include an attribute that indicates whether the requirement has been verified or not.
- *Requirements validation status*—Requirements validation as the process of ensuring requirements and sets of requirements meet the rules and characteristics in the guide. Some organizations include an attribute field "requirement validated" to indicate whether the individual requirement has been validated.
- *Priority*—This is how important the requirement is to the stakeholder. It may not be a critical requirement (i.e., one the system must possess or it won't work at all), but simply something that the stakeholder(s)

holds very dear. Priority may be characterized in terms of a level (1, 2, 3 or high, medium, low). Priority may be inherited from a parent requirement.

- *Criticality*—A critical requirement is one that the system must achieve or the system cannot function at all—perhaps can be viewed as one of the set of minimum essential requirements. Criticality may be characterized in terms of a level (1, 2, 3 or major, medium, minor). Criticality may be inherited from a parent requirement.
- *Risk*—This is the risk that the requirement cannot be achieved within technology, schedule, and budget. For example, the requirement may be possible technically (i.e., the requirement may be feasible) but have risk of achievement within available budget and schedule. Risk may be characterized in terms of a level (e.g., high, medium, low). Risk may be inherited from a parent requirement.
- *Key driving requirement (KDR)*—A KDR is a requirement that, to implement, can have a large impact on cost or schedule. A KDR can be of any priority or criticality—knowing the impact a KDR has on the design allows better management of requirements. When under schedule or budget pressure, a KDR that is low priority or low criticality may be a candidate for deletion.
- *Owner*—This is the person or element of the organization that has the right to say something about this requirement. The owner could be the source but they are two different attributes.
- *Rationale*—Rationale defines why the requirement is needed and other information relevant to better understand the reason for and intent of the requirement. Rationale can also define any assumptions that were made when writing the requirement, what design effort drove the requirement, the source of any numbers in the requirement, and note how and why a requirement is constrained (if indeed it is so).
- *Applicability*—This field may be used by an organization that has a family of similar product lines to identify the applicability of the requirement (e.g., to product line, region, or country).
- *Type*—It is often useful to attach to each requirement an attribute of type. While each organization will define types based on how they may wish to organize their requirements, examples of type include input, output, external interfaces, reliability, availability, maintainability, accessibility, environmental conditions, ergonomic, safety, security, facility, transportability, training, documentation, testing, quality provision, policy and regulatory, compatibility with existing systems, standards and technical policies, conversion, growth capacity, and installation. The type field is most useful because it allows the requirements database to be viewed by a large number of designers and stakeholders for a wide range of uses.

More detail on writing text-based requirements can be found in the *INCOSE Guide for Writing Requirements* (INCOSE RWG, 2012), which focuses on the writing of requirements and addresses the characteristics of individual requirement statements, the characteristics of sets of requirements, the attributes of individual requirement statements, and the rules for individual requirement statements.

4.3.2.3 Define, Derive, and Refine Functional/Performance Requirements At the beginning of the project, SE is concerned primarily with user requirements analysis—leading to the translation of user needs into basic functions and a quantifiable set of performance requirements that can be translated into design requirements.

Defining, deriving, and refining functional and performance requirements apply to the total system over its life cycle, including its support requirements. These requirements need to be formally captured in a manner that defines the functions and interfaces and characterizes system performance such that they can be flowed down to hardware and software designers. This is a key SE activity and is the primary focus through System Requirements Review (SRR). During the requirements analysis, the support from most other disciplines (e.g., software, hardware, manufacturing, quality, verification, specialty) is necessary to ensure a complete, feasible, and accurate set of requirements that consider all necessary life cycle factors of the system definition. The customer is also a key stakeholder and validates the work as it progresses.

Establishing a total set of system requirements is a complex, time-consuming task involving nearly all project areas in an interactive effort. It must be done early, since it forms the basis for all design, manufacturing, verification, operations, maintenance, and

retirement efforts and therefore determines the cost and schedule of the project. The activity is iterative for each stage, with continuous feedback as the level of design detail increases, and flows from the life cycle concepts, particularly the OpsCon.

The result of the system requirements definition process should be a baseline set of complete, accurate, nonambiguous system requirements, recorded in the requirements database, accessible to all parties, and captured in an approved, released SyRS.

4.3.2.4 Define Other Nonfunctional Requirements The life cycle concepts will also suggest requirements that are those dealing with operational conditions (e.g., safety; system security; reliability, availability, and maintainability; human systems integration, environmental engineering—see Chapter 10), as well as life cycle constraints (e.g., maintenance, disposal) that will strongly influence the definition of the solution elements.

4.3.2.5 Generate the SyRS The SyRS—which is often called the System Specification—is a baseline set of complete, accurate, nonambiguous system requirements, recorded in the requirements database and accessible to all parties. To be nonambiguous, requirements must be broken down into constituent parts in a traceable hierarchy such that each individual requirement statement is:

- Clear, unique, consistent, stand-alone (not grouped), and verifiable
- Traceable to an identified source requirement
- Not redundant, nor in conflict with, any other known requirement
- Not biased by any particular implementation

These objectives may not be achievable using source requirements. Often, requirements analysis is required to resolve potential conflicts and redundancies and to further decompose requirements so that each applies only to a single system function. Use of an automated requirements database will greatly facilitate this effort, but is not explicitly required.

During the system requirements definition process, it is often necessary to generate a "snapshot" report of clarified system requirements. To aid this process, it may be desirable to create a set of clarified requirement objects in the requirements database with information providing traceability from their corresponding originating requirement. Clarified requirements may be grouped as functional, performance, constraining, and nonfunctional.

4.4 ARCHITECTURE DEFINITION PROCESS

4.4.1 Overview

4.4.1.1 Purpose As stated in ISO/IEC/IEEE 15288,

> [6.4.4.1] The purpose of the Architecture Definition process is to generate system architecture alternatives, to select one or more alternative(s) that frame stakeholder concerns and meet system requirements, and to express this in a set of consistent views.

4.4.1.2 Description System architecture and design activities enable the creation of a global solution based on principles, concepts, and properties logically related and consistent with each other. The solution architecture and design have features, properties, and characteristics satisfying, as far as possible, the problem or opportunity expressed by a set of system requirements (traceable to mission/business and stakeholder requirements) and life cycle concepts (e.g., operational, support) and are implementable through technologies (e.g., mechanics, electronics, hydraulics, software, services, procedures). In this handbook, architecture and design activities are described as two separate processes to show that they are based on different and complementary notions. System architecture is more abstract, conceptualization oriented, global, focused to achieve the mission and OpsCon of the system, and focused on high-level structure in systems and system elements. It addresses the architectural principles, concepts, properties, and characteristics of the SOI. System design is more technology oriented through physical, structural, environmental, and operational properties forcing decisions for implementation by focusing on compatibility with technologies and other design elements and feasibility of construction and integration.

The architecture definition process aggregates and deals with incremental insights obtained about the specified requirements and emergent properties and behaviors of the system and system elements while managing suitability, viability, and affordability. The design definition

process uses the artifacts of the architecture definition process (e.g., architecture description, feasibility analyses, system balance trades, vetted requirements). For information about the uses of architecture definition, refer to ISO/IEC/IEEE 42010 (2011).

The architecture definition process is used to create and establish alternative architectures through several views and models, to assess the properties of these alternatives (supported by the system analysis process), and to select appropriate technological or technical system elements that compose the system:

> An effective architecture is as design-agnostic as possible to allow for maximum flexibility in the design trade space. An effective architecture also highlights and supports trade-offs for the Design Definition process and possibly other processes such as Portfolio Management, Project Planning, System Requirements Definition, Verification, etc. (ISO/IEC/IEEE 15288:2015)

This process is iterative and requires the participation of systems engineers or architects supported by relevant designers and specialists in the system domain. Also, iteration between this process and others is expected as more information is available and analysis is performed. This process also continues to be recursively applied to define the requirements for each system element. Iteration and recursion are further described in Section 4.3.1.2. Figure 4.6 is the IPO diagram for the architecture definition process.

4.4.1.3 Inputs/Outputs Inputs and outputs for the architecture definition process are listed in Figure 4.6. Descriptions of each input and output are provided in Appendix E.

4.4.1.4 Process Activities The architecture definition process includes the following activities:

- Prepare for architecture definition.
  - Identify and analyze relevant market, industry, stakeholder, organizational, business, operations, mission, legal, and other information that will help to understand the perspectives that will

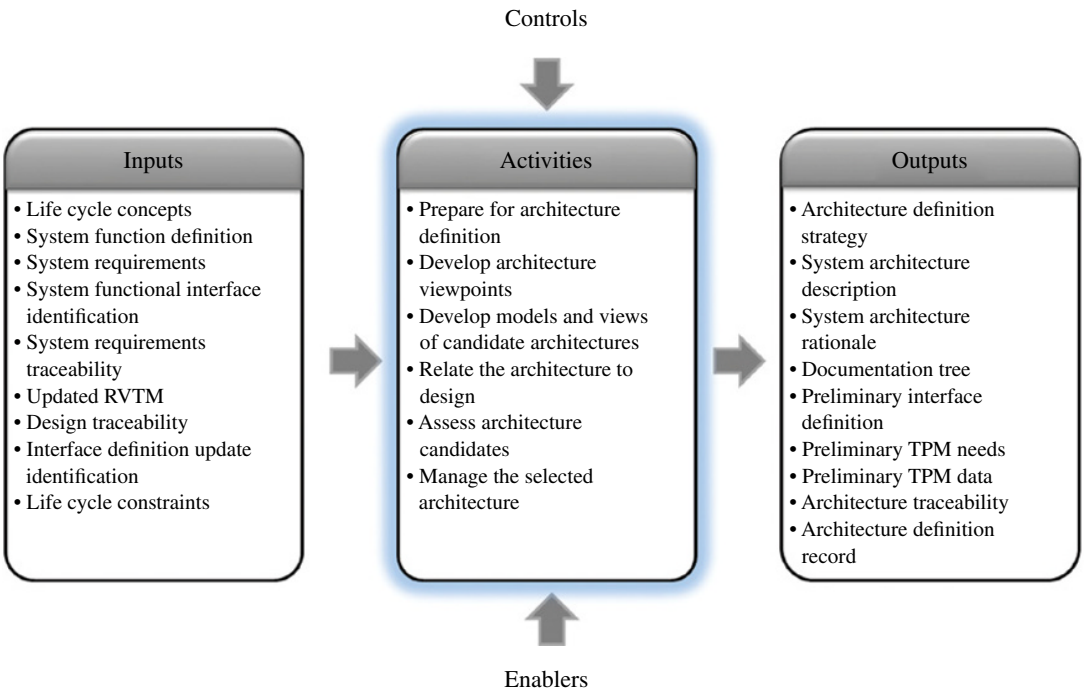


FIGURE 4.6 IPO diagram for the architecture definition process. INCOSE SEH original figure created by Shortell and Walden. Usage per the INCOSE Notices page. All other rights reserved.

guide the development of the architecture views and models. This information is intended to help build an understanding of the environment for which the solution is needed in order to establish better insight into the stakeholder concerns.
  - In particular, analyze the system requirements and tag nonfunctional requirements, that is, those dealing with operational conditions (e.g., safety, security, dependability, human factors, simplicity of interfaces, environmental conditions), as well as life cycle constraints (e.g., maintenance, disposal, deployment) that will strongly influence the definition of the solution elements.
  - Capture stakeholder concerns related to architecture. Usually, the stakeholder concerns focus on expectations or constraints that span one or more system life cycle stages. The concerns are often related to critical quality characteristics to the system that relate to those stages.
  - Establish the approach for defining the architecture. This includes an architecture roadmap and strategy, as well as methods, modeling techniques, tools, and the need for any enabling systems, products, or services. The approach should also include the process requirements (e.g., measurement approach and methods), evaluation (e.g., reviews and criteria), and necessary coordination. Capture the evaluation criteria.
  - Ensure the enabling elements or services will be available. As part of this task, plan for the need and identify the requirements for the enabling items.
- *Develop architecture viewpoints.*
  - Based on the identified stakeholder concerns, establish or identify the associated architecture viewpoints, the supporting kinds of models that facilitate the analysis and understanding of the viewpoint, and relevant architecture frameworks to support the development of the models and views.
- *Develop models and views of candidate architectures.*
  - Select or develop supporting modeling techniques and tools.
  - In conjunction with the system requirements definition process, determine the system context (i.e., how the SOI fits into the external environment) and boundary, including the interfaces, that reflect the operational scenarios and expected system behaviors. This task includes identification of expected interactions of the system with systems or other entities external to the system (control) boundary as defined in negotiated ICDs.
  - Determine which architectural entities (e.g., functions, input/output flows, system elements, physical interfaces, architectural characteristics, information/data elements, containers, nodes, links, communication resources, etc.) address the highest priority requirements (i.e., most important stakeholder concerns, critical quality characteristics, and other critical needs).
  - Allocate concepts, properties, characteristics, behaviors, functions, and/or constraints that are significant to architecture decisions of the system to architectural entities.
  - Select, adapt, or develop models of the candidate architectures of the system, such as logical and physical models. It is sometimes neither necessary nor sufficient to use logical and physical models. The models to be used are those that best address key stakeholder concerns. Logical models may include functional, behavioral, or temporal models; physical models may include structural blocks, mass, layout, and other physical models (see Section 9.1 for more information about models).
  - Determine need for derived system requirements induced by necessary added architectural entities (e.g., functions, interfaces) and by structural dispositions (e.g., constraints, operational conditions). Use the system requirements definition process to define and formalize them.
  - Compose views from the models of the candidate architectures. The views are intended to ensure that the stakeholder concerns and critical requirements have been addressed.
  - For each system element that composes the system, develop requirements corresponding to allocation, alignment, and partitioning of architectural entities and system requirements to system elements. To do this, invoke the stakeholder needs and requirements definition process and the system requirements definition process.

  - Analyze the architecture models and views for consistency and resolve any issues identified. Correspondence rules from frameworks can be useful in this analysis (ISO/IEC/IEEE 42010, 2011).
  - Verify and validate the models by execution or simulation, if modeling techniques and tools permit, and with traceability matrix of OpsCon. Where possible, use design tools to check their feasibility and validity. As needed, implement partial mock-ups or prototypes, or use executable architecture prototypes or simulators.
- *Relate the architecture to design.*
  - Determine the system elements that reflect the architectural entities. Since the architecture is intended to be design-agnostic, these system elements may be notional until the design evolves. To do this, partition, align, and allocate architectural entities and system requirements to system elements. Establish guiding principles for the system design and evolution. Sometimes, a "reference architecture" is created using these notional system elements as a means to convey architectural intent and to check for design feasibility.
  - Establish allocation matrices between architectural entities using their relationships.
  - Perform interface definition for interfaces that are necessary for the level of detail and understanding of the architecture. The definition includes the internal interfaces between the system elements and the external interfaces with other systems.
  - Determine the design characteristics that relate to the system elements and their architectural entities, such as by mapping (see Section 4.5).
  - Determine need for derived system requirements induced by necessary added architectural entities (e.g., functions, interfaces) and by structural dispositions (e.g., constraints, operational conditions). Use the system requirements definition process to formalize them.
  - For each system element that composes the parent system, develop requirements corresponding to allocation, alignment, and partitioning of architectural entities and system requirements to system elements. To do this, invoke the stakeholder needs and requirements definition process and the system requirements definition process.
- *Assess architecture candidates.*
  - Using the architecture evaluation criteria, assess the candidate architectures by applying the system analysis, measurement, and risk management processes.
  - Select the preferred architecture(s). This is done by applying the decision management process.
- *Manage the s*elected *architecture.*
  - Capture and maintain the rationale for all selections among alternatives and decision for the architecture, architecture framework(s), viewpoints, kinds of models, and models of the architecture.
  - Manage the maintenance and evolution of the architecture, including the architectural entities, their characteristics (e.g., technical, legal, economical, organizational, and operational entities), models, and views. This includes concordance, completeness, and changes due to environment or context changes, technological, implementation, and operational experiences. Allocation and traceability matrices are used to analyze impacts onto the architecture. The present process is performed at any time evolutions of the system occur.
  - Establish a means for the governance of the architecture. Governance includes the roles, responsibilities, authorities, and other control functions.
  - Coordinate review of the architecture to achieve stakeholder agreement. The stakeholder requirements and system requirements can serve as references.

Common approaches and tips:

- A function (e.g., to move) and its state of execution/operational mode (e.g., moving) are similar but two complementary views. Consider a behavioral model of the system that transits from an operational mode to another one.

4.4.2 Elaboration

4.4.2.1 Architecture Representation The notion of system is abstract, but it is a practical means to create, design, or redesign products, services, or enterprises.

A system is one solution that could address/answer a problem or an opportunity; there may be several solutions to address the same problem or opportunity. The solution may be more or less complex, and the notion of system is useful to engineer complex solutions. A complex solution cannot be apprehended with a single view or model because of the number of characteristics or properties. These last are grouped reflecting typologies of data, and each type of data/characteristics is structured. The set of different types and interrelated "structures" can be understood as THE architecture of the system. The majority of interpretations of system architecture are based on the fairly intangible notion of structure.

Therefore, the system architecture is formally represented with sets of architectural entities such as functions, function flows, interfaces, resource flow items, information/data elements, physical elements, containers, nodes, links, communication resources, etc. These architectural entities may possess architectural characteristics such as dimensions, environmental resilience, availability, robustness, learnability, execution efficiency, mission effectiveness, etc. The entities are not independent but interrelated by the means of relationships.

4.4.2.2 Architecture Description of the System ISO/IEC/IEEE 42010 specifies the normative features of architecture frameworks, viewpoints, and views as they pertain to architecture description. Viewpoints and views are sometimes specified in architecture frameworks such as Zachman (1987), DoDAF (2010), MoDAF (n.d.), The Open Group Architecture Framework (TOGAF), etc. Views are usually generated from models. Many SE practices use logical and physical models (or views) for modeling the system architecture. The architecture definition process includes also the possible usage of other viewpoints and views to represent how the system architecture addresses stakeholder concerns, for example, cost models, process models, rule models, ontological models, belief models, project models, capability models, data models, etc. Maier and Rechtin (2009) provide another view of the system architecture development process. Refer to Chapter 9 for a more detailed treatment of models.

A viewpoint is intended to address a particular stakeholder concern (or set of closely related concerns). The viewpoint will specify the model kinds to be used in developing an architectural view that depicts how the architecture addresses that concern (or set of concerns). The viewpoint also specifies the ways in which the model(s) should be generated and how the models are used to compose the view.

An architecture framework contains standardized viewpoints, view templates, metamodels, model templates, etc. that facilitate the development of the views contained in an architecture description. ISO/IEC/IEEE 42010 specifies the necessary features of an architecture framework.

4.4.2.3 Emergent Properties Emergence is the principle that whole entities exhibit properties, which are meaningful only when attributed to the whole, not to its parts. Every model of a human activity system exhibits properties as a whole entity that derive from its component activities and their structure, but cannot be reduced to them (Checkland, 1998).

System elements interact between themselves and can create desirable or undesirable phenomena called "emergent properties," such as inhibition, interference, resonance, or reinforcement of any property. Definition of the architecture of the system includes an analysis of interactions between system elements in order to prevent undesirable properties and reinforce desirable ones.

The notion of emergent property is used during architecture and design to highlight necessary derived functions and internal physical or environmental constraints. Corresponding derived requirements should be added to system requirements baseline when they impact the SOI.

4.4.2.4 Architecture in Product Lines The architecture plays a very important role in a product line. The architecture in a product line spans across several design variants, providing a cohesive basis for the product line designs by ensuring compatibility and interoperability across the product line.

4.4.2.5 Notion of Interface The notion of interface is one of the most important items to consider when defining the architecture of a system. The term "interface" comes from Latin words "inter" and "facere" and means "to *do* something *between* things." Therefore, the fundamental aspect of an interface is functional and is defined as inputs and outputs of functions. As functions are performed by physical elements, inputs/outputs of functions are also carried by physical elements, called

TABLE 4.1 Examples of system elements and physical interfaces

Element	Product system	Service system	Enterprise system
System element	Hardware parts (mechanics, electronics, electrical, plastic, chemical, etc.) Operator roles Software pieces	Processes, databases, procedures, etc. Operator roles Software applications	Corporate, direction, division, department, project, technical team, leader, etc. IT components
Physical interface	Hardware parts, protocols, procedures, etc.	Protocols, documents, etc.	Protocols, procedures, documents, etc.

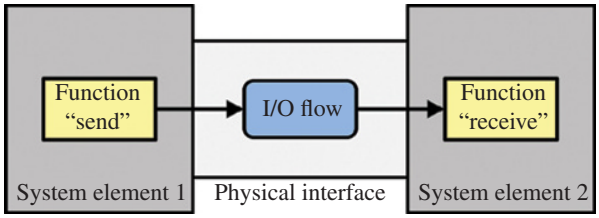


FIGURE 4.7 Interface representation. Reprinted with permission from Alain Faisandier. All other rights reserved.

physical interfaces. Consequentially, both functional and physical aspects are considered in the notion of interface. Examples of system elements and physical interfaces are shown in Table 4.1.

A representation of an interface in Figure 4.7 shows the function "send," located in one system element; the function "receive," located in the other one; and the physical interface that supports the input/output flow. In the context of complex exchanges between system elements in information technology (IT) systems, a protocol is seen as a physical interface that carries exchanges of data.

4.4.2.6 Coupling Matrix Coupling matrices (also called N^2 diagrams) are a basic method to define the aggregates and the order of integration (Grady, 1994). They are used during architecture definition, with the goal of keeping the interfaces as simple as possible (see Fig. 4.8). Simplicity of interfaces can be a distinguishing characteristic and a selection criterion between alternate architectural candidates. The coupling matrices are also useful for optimizing the aggregate definition and the verification of interfaces (see Section 4.8.2.3).

4.4.2.7 Allocation and Partitioning of Logical Entities to Physical Entities Defining a physical structural model of the architecture of a system consists of identifying system elements capable of performing the functions of logical models, identifying the physical interfaces capable to carry input/output flows and control flows, and taking into

(a)

C1		X			X			
	C2	X	X				X	
X		C3			X			
	X		C4	X			X	
	X			C5		X		X
X		X	X	X	C6			
				X		C7		X
	X		X				C8	
X				X		X		C9

A1	A2	A3

(b)

C1	X	X						
X	C3	X						
X	X	C6		X		X		
	X		C2	X	X			
			X	C4	X	X		
			X	X	C8			
			X			C5	X	X
						X	C7	X
X						X	X	C9

B1	B2	B3

FIGURE 4.8 (a) Initial arrangement of aggregates; (b) final arrangement after reorganization. Reprinted with permission from Alain Faisandier. All other rights reserved.

account architectural characteristics that characterize the system in which they are included.

The term allocation does not mean just to allocate logical entities to existing system elements. Partitioning and allocation means to separate, gather, or decompose logical entities into partitions and then to make the correspondence between these partitions and potential system elements. The system elements either exist (reusable, repurposed, or purchasable) can be developed and technically implemented.

Nonfunctional requirements and/or architectural characteristics are used as criteria to analyze, assess, and select candidate system elements and logical partitions. Examples of assessment criteria include similar transformations within the same technology, similar level of efficiency, exchange of same type of input/output flows (information, energy, materials), centralized or distributed controls, execution with close frequency level, dependability conditions, environment resistance level, and other enterprise constraints.

4.4.2.8 Defining Candidate Architectures and Selecting the Preferred One The goal of the architecture definition process is to provide the "best" possible architecture made of suitable system elements and interfaces, that is, the architecture that answers, at best, all the stakeholder and system requirements, depending on agreed limits or margins of each requirement. The preferred way to do this is by producing several candidate architectures; analyzing, assessing, and comparing them; and then selecting the most suitable one.

Candidate architectures are defined according to criteria or drivers in order to build up a set of system elements (e.g., separate, gather, connect, and disconnect the network of system elements and their physical interfaces). Criteria or drivers may include reduction of the number of interfaces, system elements that can be tested separately, modularity (i.e., low interdependence), replaceability of system elements during maintenance, compatible technology, proximity of elements in space, handling (e.g., weight, volume, transportation facilities), and optimization of resources and information shared between elements.

Viable candidate architectures have to satisfy all required features (e.g., functions, characteristics) after trade-offs are made. The preferred architecture represents an optimum such that the architecture and design match the complete set of stakeholder and system requirements. This proposition depends on stakeholder and system requirements being feasible and validated, and that feasibility and validation are demonstrated or proven. Assessments, studies, mock-ups, etc. are generally performed in parallel with architecture and design activities to obtain "proven" requirements.

Architecture definition activities include optimization to obtain a balance among architectural characteristics and acceptable risks. Certain analyses such as performance, efficiency, maintainability, and cost are required to get sufficient data that characterize the global or detailed behavior of the candidate architectures with respect to the stakeholder and system requirements. Those analyses are conducted with the system analysis process (see Section 4.6) and as specialty engineering activities (see Chapter 10).

4.4.2.9 Methods and Modeling Techniques Modeling, simulation, and prototyping used during architecture definition can significantly reduce the risk of failure in the finished system. Systems engineers use modeling techniques and simulation on large complex systems to manage the risk of failure to meet system mission and performance requirements. These are best performed by subject matter experts who develop and validate the models, conduct the simulations, and analyze the results. Refer to Chapter 9 for a more detailed treatment of models and simulations.

4.5 DESIGN DEFINITION PROCESS

4.5.1 Overview

4.5.1.1 Purpose As stated in ISO/IEC/IEEE 15288,

> [6.4.5.1] The purpose of the Design Definition process is to provide sufficient detailed data and information about the system and its elements to enable the implementation consistent with architectural entities as defined in models and views of the system architecture.

4.5.1.2 Description System architecture deals with high-level principles, concepts, and characteristics represented by general views or models excluding details (see Section 4.4). System design supplements the system architecture providing information and data useful and

necessary for implementation of the system elements. This information and data details the expected properties allocated to each system element and/or to enable the transition toward their implementation.

Design is the process of developing, expressing, documenting, and communicating the realization of the architecture of the system through a complete set of design characteristics described in a form suitable for implementation. Figure 4.9 is the IPO diagram for the design definition process.

Design concerns every system element (e.g., composed of implementation technologies such as mechanics, electronics, software, chemistry, human operations, and services) for which specific engineering processes are needed. The design definition process provides the detailed information and data that enable the implementation of a particular system element. This process provides feedback to the parent system architecture to consolidate or confirm the allocation and partitioning of architectural entities to system elements.

As a result, the design definition process provides the description of the design characteristics and design enablers necessary for implementation. Design characteristics include dimensions, shapes, materials, and data processing structures. Design enablers include formal expressions or equations, drawings, diagrams, tables of metrics with their values and margins, patterns, algorithms, and heuristics.

4.5.1.3 Inputs/Outputs Inputs and outputs for the design definition process are listed in Figure 4.9. Descriptions of each input and output are provided in Appendix E.

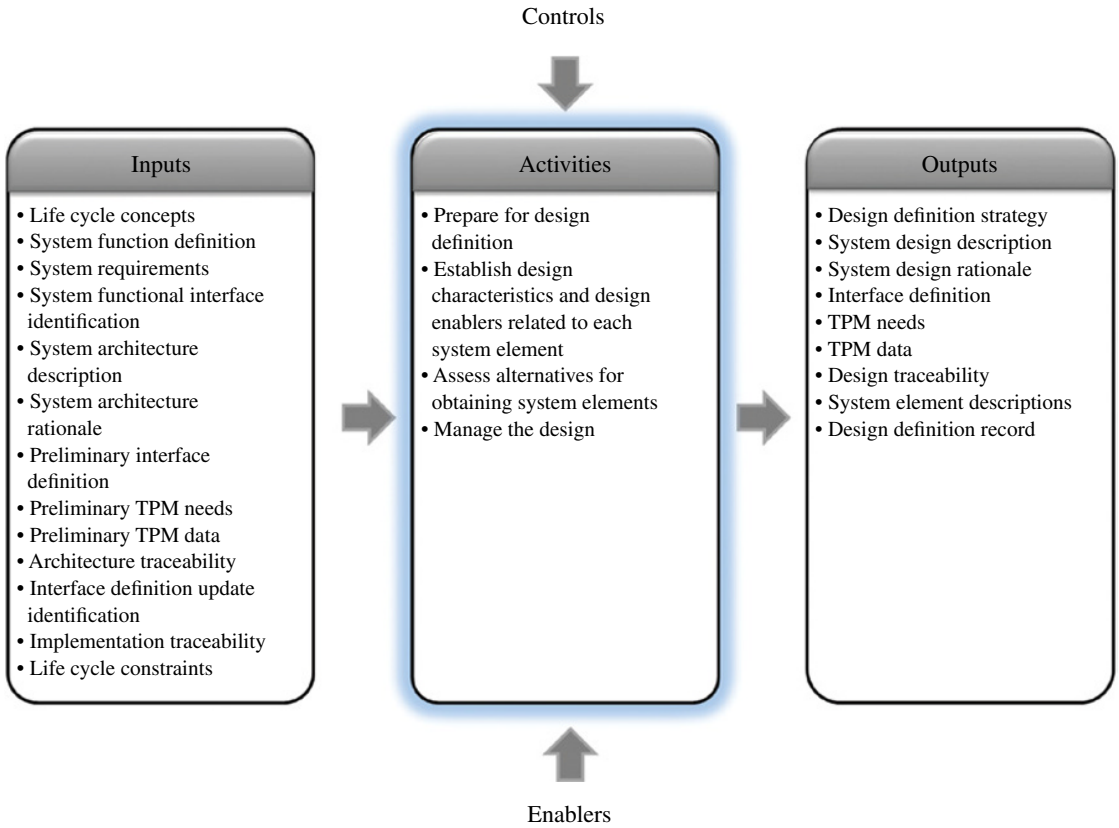


FIGURE 4.9 IPO diagram for the design definition process. INCOSE SEH original figure created by Shortell and Walden. Usage per the INCOSE Notices page. All other rights reserved.

4.5.1.4 Process Activities The design definition process includes the following activities:

- *Prepare for design definition.*
  - Plan for technology management. Identify the technologies needed to achieve the design objectives for the system and its system elements. The technology management includes obsolescence management. Determine which technologies and system elements have a risk of becoming obsolete. Plan for their potential replacement, including identification of potential evolving technologies.
  - Identify the applicable types of design characteristics for each system element considering the technologies that will be applied. Periodically assess the design characteristics and adjust as the system architecture evolves.
  - Define and document the design definition strategy, including the need for and requirements of any enabling systems, products, or services.
- *Establish design characteristics and design enablers related to each system element.*
  - Perform requirements allocation to system elements for all requirements and system elements not fully addressed in the architecture definition process.

    Note: Usually, every system requirement is transformed into architectural entities and architectural characteristics. Those entities or characteristics are then allocated to system elements, either by direct assignment or by some kind of partitioning. However, there will be some cases where it is impractical or impossible to transform a requirement into an architectural entity. In either case, it is important to do some degree of analysis or study to determine how best to flow down each requirement. It is good practice to try to do as much of the allocation as possible in the architecture definition process.

  - Define the design characteristics relating to the architectural characteristics for the architectural entities, and ensure that the design characteristics are feasible. Use design enablers such as models (physical and analytical), design heuristics, etc. If the design characteristics are determined infeasible, then assess other design alternatives or perform trades of other system definition elements.
  - Perform interface definition to define the interfaces that were not defined by the architecture definition process or that need to be refined as the design details evolve. This includes both internal interfaces between the system elements and the external interfaces with other systems.
  - Capture the design characteristics of each system element. The resulting artifacts will be dependent on the design methods and techniques used.
  - Provide rationale about selection of major implementation options and enablers.
- *Assess alternatives for obtaining system elements.*
  - Identify existing implemented elements. These include COTS, reused, or other nondeveloped system elements. Alternatives for new system elements to be developed may be studied.
  - Assess options for the system element, including the COTS system elements, the reused system elements, and the new system elements to be developed using selection criteria that is derived from the design characteristics.
  - Select the most appropriate alternatives.
  - If the decision is made to develop the system element, rest of the design definition process and the implementation process are used. If the decision is to buy or reuse a system element, the acquisition process may be used to obtain the system element.
- *Manage the design.*
  - Capture and maintain the rationale for all selections among alternatives and decisions for the design, architecture characteristics, design enablers, and sources of system elements.
  - Manage the maintenance and evolution of the design, including the alignment with the architecture. Assess and control evolution of the design characteristics.
  - Establish and maintain bidirectional traceability between the architecture entities (including views, models, and viewpoints) to the stakeholder requirements and concerns; system requirements and constraints; system analysis, trades, and rationale; verification criteria and results; and design elements. The traceability between the design characteristics and the architectural entities also helps ensure architectural compliance.

– Provide baseline information for configuration management.
– Maintain the design baseline and the design definition strategy.

Common approaches and tips:

- Discipline engineers, or designers, perform the design definition of each concerned system element; they provide strong support (knowledge and competencies) to systems engineers, or architects, in the evaluation and selection of candidate system architectures and system elements. Inversely, systems engineers, or architects, must provide feedback to discipline engineers or designers to improve knowledge and know-how.

4.5.2 Elaboration

4.5.2.1 Architecture Definition versus Design Definition The architecture definition process focuses on the understanding and resolution of the stakeholder concerns. It develops insights into the relation between these concerns, the solution requirements, and the emergent properties and behaviors of the system. Architecture focuses on suitability, viability, and adaptability over the life cycle. An effective architecture is as design-agnostic as possible to allow for maximum flexibility in the design trade space. It focuses more on the "what" than the "how."

The design definition process, on the other hand, is driven by specified requirements, the architecture, and more detailed analysis of performance and feasibility. Design definition addresses the implementation technologies and their assimilation. Design provides the "how" or "implement-to" level of the definition.

4.5.2.2 Notions and Principles Used within Design The purpose of system design is to make the link between the architecture of the SOI and the implementation of technological system elements that compose it. So, system design is understood as the complete set of detailed models, properties, or characteristics of each system element, described into a form suitable for implementation.

Every technological domain or discipline owns its peculiar laws, rules, theories, and enablers concerning transformational, structural, behavioral, and temporal properties of its composing parts of materials, energy, or information. These specific parts and/or their compositions are described with typical design characteristics and enablers. These allow achieving the implementation of the system element of interest through various transformations, linkages, and exchanges required by design characteristics (e.g., operability level, reliability rate, speed, safeguard level) that have been assigned during the architecture definition process.

- *Examples of generic design characteristics in mechanics of solids*: Shape, geometrical pattern, dimension, volume, surface, curves, resistance to forces, distribution of forces, weight, velocity of motion, temporal persistence
- *Examples of generic design characteristics in software*: Distribution of processing, data structures, data persistence, procedural abstraction, data abstraction, control abstraction, encapsulation, creational patterns (e.g., builder, factory, prototype, singleton), and structural patterns (e.g., adapter, bridge, composite, decorator, proxy)

4.5.2.3 Design Descriptors Because it is sometimes difficult to define applicable requirements to a system element from the engineering data of the parent system (in particular from the expected architectural characteristics), it is possible to use the design descriptor technique as a complement. A design descriptor is the set of generic design characteristics and of their possible values. If similar, but not exact, system elements exist, it is possible to analyze these in order to identify their basic characteristics. Variations of the possible values of each characteristic determine potential candidate system elements.

4.5.2.4 Holistic Design It is important to understand that design definition starts with the system as a whole consisting of system elements and ends with a definition (i.e., design) for each of these system elements (not just one of them) and how they are designed to work together as a complete system. System elements are identified in the architecture, although the architecture might only identify those elements that are architecturally significant.

During the design definition process, it might be necessary to identify additional system elements to make the whole system work. This might entail embedding some enabling elements or services inside the system

boundary. There is usually a trade-off between having an enabling item inside or outside the system. The architecture definition process could make this decision, but it might be better to allow design definition to handle this since it is often dependent on other design trade-offs and on design decisions that are made along the way.

Some of these additional system elements might be necessary to account for "missing" functions that were not identified in the architecture. For example, it might be determined that the various system elements should not all produce their own power backup but instead there should be a separate system element that performs this function for all the other elements. This would be the result of a design analysis to determine the best place to put this function. Or it could be the result of applying a design pattern to this particular problem.

It might be necessary to provide feedback to the architecture definition process regarding these design decisions and trade-offs to ensure there are no negative impacts on the architecture as a whole. The architecture might or might not be updated to reflect these design details, since this depends on whether it is important to capture these features as architecturally significant or not.

It is this holistic approach to design of a system that distinguishes this from design of an individual product or service. Holistic design is an approach to design that considers the system being designed as an interconnected whole, which is also part of something larger. Holistic concepts can be applied to the system as a whole along with the system in its context (e.g., the enterprise or mission in which the system participates), as well as the design of mechanical devices, the layout of spaces, and so forth. This approach to design often incorporates concerns about the environment, with holistic designers considering how their design will impact the environment and attempting to reduce environmental impact in their designs. Holistic design is about more than merely trying to meet the system requirements.

4.6 SYSTEM ANALYSIS PROCESS

4.6.1 Overview

4.6.1.1 Purpose As stated in ISO/IEC/IEEE 15288,

> [6.4.6.1] The purpose of the System Analysis process is to provide a rigorous basis of data and information for technical understanding to aid decision-making across the life cycle.

4.6.1.2 Description This process performs quantitative assessments and estimations that are based on analyses such as cost analysis, affordability analysis, technical risk analysis, feasibility analysis, effectiveness analysis, and other critical quality characteristics. Those analyses use mainly quantitative modeling techniques, analytical models, and associated simulations, which are applied at varying levels of rigor and complexity depending on the level of fidelity needed. In some cases, it may be necessary to employ a variety of analytic functions or experimentation to obtain the necessary insight. The results serve as inputs into various technical decisions, providing confidence in the adequacy and integrity of the system definition toward achieving the appropriate system balance.

This process is used by (examples, but not limited to):

- Mission and business analysis process to analyze and estimate candidate OpsCon and/or candidate business models related to a potential SOI in terms of feasibility, costs, risks, and effectiveness
- Stakeholder requirements definition process and system requirements definition process to analyze issues relating to conflicts among the set of requirements, in particular those related to feasibility, costs, technical risks, and effectiveness (mainly performances, operational conditions, and constraints)
- Architectural definition process and design definition process to analyze and estimate architectural and design characteristics of candidate architectures and/or system elements, providing arguments in order to be able to select the most efficient ones in terms of costs, technical risks, effectiveness (e.g., performances, dependability, human factors), and other stakeholder concerns such as critical quality characteristics, affordability, maintenance, etc.
- Integration process, verification process, and validation process to estimate the related strategies
- Project assessment and control process to obtain estimates of the performance against established targets and thresholds, especially with respect to the technical measures (MOEs, MOSs, MOPs, and TPMs).

The results of analyses and estimations, as data, information, and arguments, are provided to the decision management process for selecting the most efficient alternative or candidate. In some cases, the

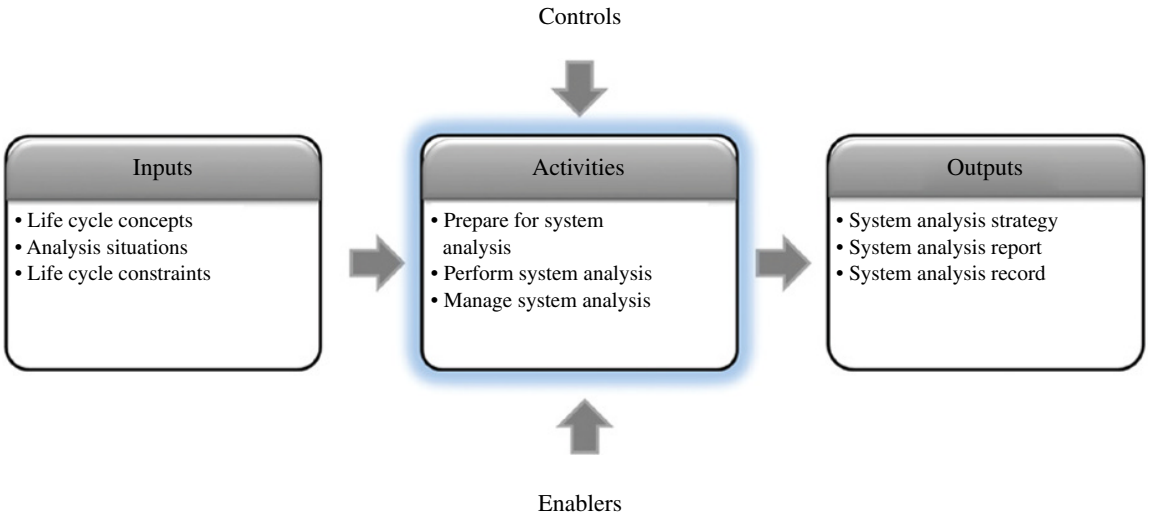


FIGURE 4.10 IPO diagram for the system analysis process. INCOSE SEH original figure created by Shortell and Walden. Usage per the INCOSE Notices page. All other rights reserved.

results may be provided to the project assessment and control process, if the information is needed to monitor the progress of the system against its system objectives, performance thresholds, or growth targets, such as the projected or modeled reliability early in the development of the system as compared to its reliability growth curve.

Figure 4.10 is the IPO diagram for the system analysis process.

4.6.1.3 Inputs/Outputs Inputs and outputs for the system analysis process are listed in Figure 4.10. Descriptions of each input and output are provided in Appendix E.

4.6.1.4 Process Activities The system analysis process includes the following activities:

- *Prepare for system analysis.*
  - Define the scope, types, objectives, and level of accuracy of required analyses and their level of importance to the system stakeholders.
  - Define or select evaluation criteria (e.g., operational conditions, environmental conditions, performance, dependability, costs types, risks types). The criteria mainly come from stakeholder needs, nonfunctional requirements, and design characteristics. The criteria must be consistent with the decision management strategy (see Section 5.3).
  - Determine the candidate elements to be analyzed, the methods and procedures to be used, and the needed justification items.
  - Determine the need and requirements for and obtain or acquire access to the enabling systems, products, or services necessary to perform analyses of the SOI.
  - Schedule the analyses according to the availability of models, engineering data (e.g., OpsCon, business models, stakeholder requirements, system requirements, design characteristics, verification actions, validation actions), skilled personnel, and procedures.
  - Document the corresponding system analysis strategy.
- *Perform system analysis.*
  - Collect the data and inputs needed for the analysis, highlighting any assumptions. Inputs can include models. Those models may be:
    - Physical models, specific to each discipline, allowing to simulate physical phenomena
    - Representation models mainly used to simulate the behavior of a system or system element
    - Analytical models (deterministic and stochastic) used to establish values of estimates to approach

the real operation of a system or system element
  - Carry out analyses as scheduled using defined methods and procedures for cost, risk, effectiveness, and validation of assumptions.
  - Conduct in-process peer reviews with appropriate subject matter experts to assess the validity, quality, and consistency of the evolving system with the stakeholder objectives and with previous analyses. Record and report in-process results.
- *Manage system analysis.*
  - Baseline the analysis results or reports using the configuration management process.
  - Maintain an engineering history of the system evolution from stakeholder needs definition to ultimate system retirement so that the project team can conduct bidirectional searches at any time during—or after—the system life cycle.

Common approaches and tips:

- The methods are chosen based on time, cost, accuracy, technical drivers, and criticality of analysis. Due to cost and schedule, most systems only perform system analysis for critical characteristics.
- Models can never simulate all the behavior of a system: they operate only in one limited field with a restricted number of variables. When a model is used, it is always necessary to make sure that the parameters and data inputs are part of the operation field; if not, irregular outputs are likely.
- Models evolve during the project: by modification of parameters, by entering new data, and by the use of new tools.
- It is recommended to concurrently use several types of models in order to compare the results and/or to take into account another characteristic or property of the system.
- Results of a simulation shall always be given in their modeling context: tool used, selected assumptions, parameters and data introduced, and variance of the outputs.

4.6.2 Elaboration

During a system's life cycle, assessments should be performed when technical choices have to be made or justified, and not just to compare different solutions. System analysis provides a rigorous approach to technical decision making. It is used to perform evaluations, including a set of analysis such as cost analysis, technical risk analysis, effectiveness analysis, and analysis of other properties.

4.6.2.1 Cost Analysis A cost analysis considers the full life cycle costs (LCC). The cost baseline can be adapted according to the project and the system. The full LCC may include labor and nonlabor cost items; it may include development, manufacturing, service realization, sales, customer utilization, supply chain, maintenance, and disposal costs (also see Section 10.1).

4.6.2.2 Technical Risk Analysis Technical risks should not be confused with project risks even if the method to manage them is the same. Technical risks address the system itself, not the project for its development. Of course, technical risks may interact with project risks. The system analysis process is often needed to perform the technical assessments that provide quantification and understanding of the probability or impact of a potential risk or opportunity (see Section 5.4 for risk management process details).

4.6.2.3 Effectiveness Analysis System effectiveness analysis is a term for a broad category of analyses that evaluate the degree or extent to which a system meets one or more criteria—the effectiveness of the system in meeting the criteria in its intended operational environment. The objective(s) and criteria may be derived from one or more desired system characteristics, such as MOEs, TPMs, or other attributes of the system, and influence the details of the analysis conducted. The analysis is more than simply determining if the criteria are met but also the degree to which they are met (or fall short or exceed), as this information is used to support trade-offs and evaluation of alternatives (such as which candidates to develop further, where improvements are needed or cost savings are possible). One of the challenges of effectiveness analysis is to prioritize and select the right set of effectiveness objectives and criteria; for example, if the product is made for a single use, the maintainability and the capability of evolution will not be relevant criteria.

4.6.2.4 Methods and Modeling Techniques Various types of models and modeling techniques can be used in the context of system analysis. These include, but are not

limited to, physical models, structural models, behavior models, functional models, temporal models, mass models, cost models, probabilistic models, parametric models, layout models, network models, visualizations, simulations, mathematical models, and prototypes. For more information on models and modeling methods, see Section 9.1.

4.7 IMPLEMENTATION PROCESS

4.7.1 Overview

4.7.1.1 Purpose As stated in ISO/IEC/IEEE 15288,

> [6.4.7.1] The purpose of the Implementation process is to realize a specified system element.

The implementation process creates or fabricates a system element conforming to that element's detailed description (requirements, architecture, design, including interfaces). The element is constructed employing appropriate technology and industry practices.

4.7.1.2 Description During the implementation process, engineers follow the requirements allocated to the system element to fabricate, code, or build each individual element using specified materials, processes, physical or logical arrangements, standards, technologies, and/or information flows outlined in detailed drawings or other design documentation. System requirements are verified and stakeholder requirements are validated. If subsequent configuration audits reveal discrepancies, recursive interactions occur with predecessor activities or processes, as required, to correct them. Figure 4.11 is the IPO diagram for the implementation process.

4.7.1.3 Inputs/Outputs Inputs and outputs for the implementation process are listed in Figure 4.11. Descriptions of each input and output are provided in Appendix E.

4.7.1.4 Process Activities Implementation process activities begin with detailed design and include the following:

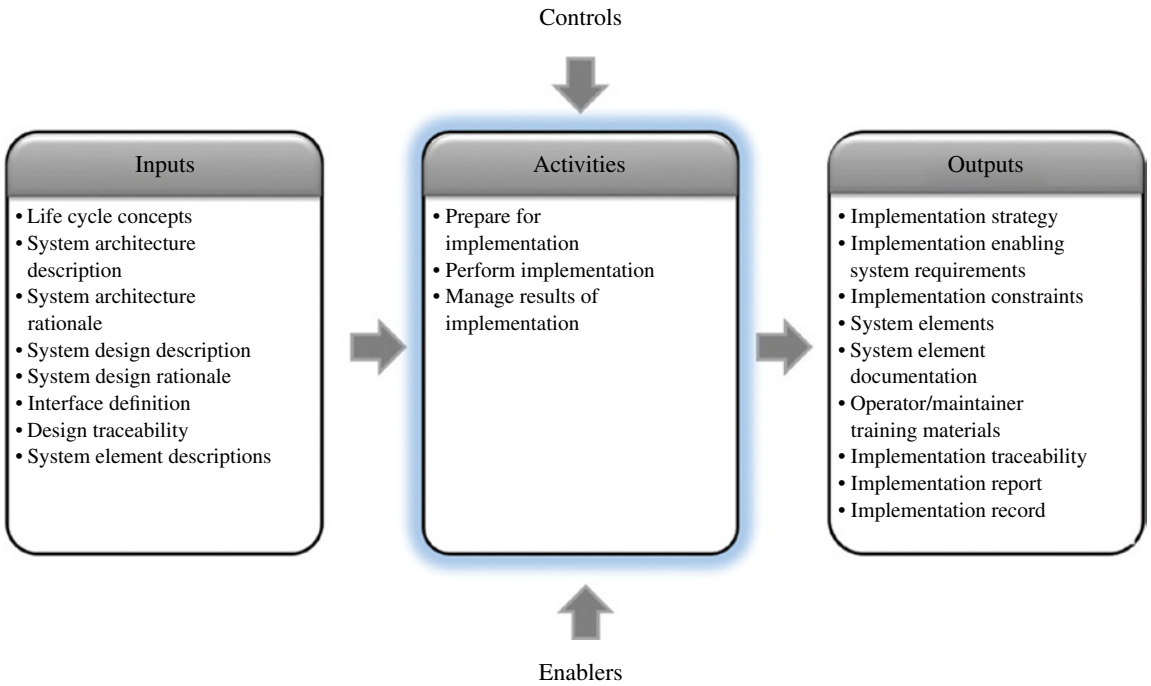


FIGURE 4.11 IPO diagram for the implementation process. INCOSE SEH original figure created by Shortell and Walden. Usage per the INCOSE Notices page. All other rights reserved.

- *Prepare for implementation.*
  - Define fabrication/coding procedures, tools and equipment to be used, implementation tolerances, and the means and criteria for auditing configuration of resulting elements to the detailed design documentation. In the case of repeated system element implementations (such as for mass manufacturing or replacement elements), the implementation strategy is defined/refined to achieve consistent and repeatable element production and retained in the project decision database for future use.
  - Elicit from stakeholders, developers, and teammates any constraints imposed by implementation technology, strategy, or implementation enabling systems. Record the constraints for consideration in the definition of the requirements, architecture, and design.
  - Document the plan for acquiring or gaining access to resources needed during implementation. The planning includes the identification of requirements and interfaces for the enabling system.
- *Perform implementation.*
  - Develop data for training users on correct and safe procedures for operating and maintaining that element, either as a stand-alone end item or as part of a larger system.
  - Complete detailed product, process, material specifications ("build-to" or "code-to" documents), and corresponding analyses.
  - Ensure the realization of the system elements per the detailed product, process, and material specifications and produce documented evidence of implementation compliance—specifically, these tasks are as follows:
    - Conduct peer reviews and testing—Inspect and verify software for correct functionality, white box testing, etc. in accordance with software/hardware best practices.
    - Conduct hardware conformation audits—Compare hardware elements to detailed drawings to ensure that each element meets its detailed specifications prior to integration with other elements.
  - Prepare initial training capability and draft training documentation—To be used to provide the user community with the ability to operate, conduct failure detection and isolation, conduct contingency scenarios, and maintain the system as appropriate.
  - Prepare a hazardous materials log, if applicable.
  - Determine the packaging and storage requirements for the system element and ensure initiation of the packaging and/or storage, at the appropriate time.
- *Manage results of implementation.*
  - Identify and record implementation results. Maintain the records per organizational policy.
  - Record any anomalies encountered during the implementation process, and analyze and resolve the anomalies (corrective actions or improvements) using the quality assurance process (see Section 5.8).
  - Establish and maintain traceability of the implemented system elements with the system architecture, design, and system and interface requirements that are needed for implementation.
  - Provide baseline information for configuration management.

Common approaches and tips:

- Keep the Integrated Product Development Team (IPDT) engaged to assist with configuration issues and redesign.
- Inspections are a proactive way to build in quality (Gilb and Graham, 1993).
- In anticipation of improving process control, reducing production inspections, and lowering maintenance activities, many manufacturing firms use Design for Six Sigma or lean manufacturing.
- Conduct hardware conformation audits or system element level hardware verification; ensure sufficient software unit verification prior to integration.
- Validate simulations; interface simulator drivers should be representative of tactical environments.

4.7.2 Elaboration

4.7.2.1 Implementation Concepts The implementation process typically focuses on the following four forms of system elements:

- *Hardware/physical*—Output is fabricated or adapted hardware or physical element. If the hardware element is being reused, it may require modification.
- *Software*—Output is software code and executable images
- *Operational resources*—Output includes procedures and training. These are verified to the system requirements and OpsCon.
- *Services*—Output includes specified services. These may be the result of one or more hardware, software, or operational elements resulting in the service.

The implementation process can support either the creation (fabrication or development) or adaptation of system elements. For system elements that are reused or acquired, such as COTS, the implementation process allows for adaption of the elements to satisfy the needs of the SOI. This is usually accomplished via configuration settings provided with the element (e.g., hardware configuration switches and software configuration tables). Newly created products have more flexibility to be designed and developed to meet the needs of the SOI without modification.

4.8 INTEGRATION PROCESS

4.8.1 Overview

4.8.1.1 Purpose As stated in ISO/IEC/IEEE 15288,

> [6.4.8.1] The purpose of the Integration process is to synthesize a set of system elements into a realized system (product or service) that satisfies system requirements, architecture, and design.

4.8.1.2 Description Integration consists of progressively assembling the implemented system elements (hardware, software, and operational resources) that compose the SOI as defined and verifying the correctness of the static and dynamic aspects of interfaces between the implemented system elements. There is a strong focus on the interfaces to ensure that the intended operation of the system elements and interoperation with other systems is achieved. Any integration constraints are identified and considered during the definition of the requirements, architecture, and design. The interaction of the integration process with the system definition processes (i.e., system requirements definition, architecture definition, and design definition) early in the development is essential for avoiding integration issues during the system realization.

The integration process works closely with the verification and validation (V&V) processes. This process is iterated with the V&V processes, as appropriate. As the integration of system elements occurs, the verification process is invoked to check the correct implementation of architectural characteristics and design properties. The validation process may be invoked to check that the individual system elements provide the function intended. The process checks that all boundaries between system elements have been correctly identified and described, including physical, logical, and human–system interfaces and interactions (physical, sensory, and cognitive), and that all system element functional, performance, and design requirements and constraints are satisfied.

Figure 4.12 is the IPO diagram for the integration process.

4.8.1.3 Inputs/Outputs Inputs and outputs for the integration process are listed in Figure 4.12. Descriptions of each input and output are provided in Appendix E.

4.8.1.4 Process Activities The integration process includes the following activities:

- *Prepare for integration.*
  - Define critical checkpoints to provide assurance of the correct behavior and operation of interfaces and functions of the system elements.
  - Establish the integration strategy that minimizes integration time, costs, and risks:
    - Define an optimized sequence order of assembly aggregates, composed of system elements, based on the system architecture definition,

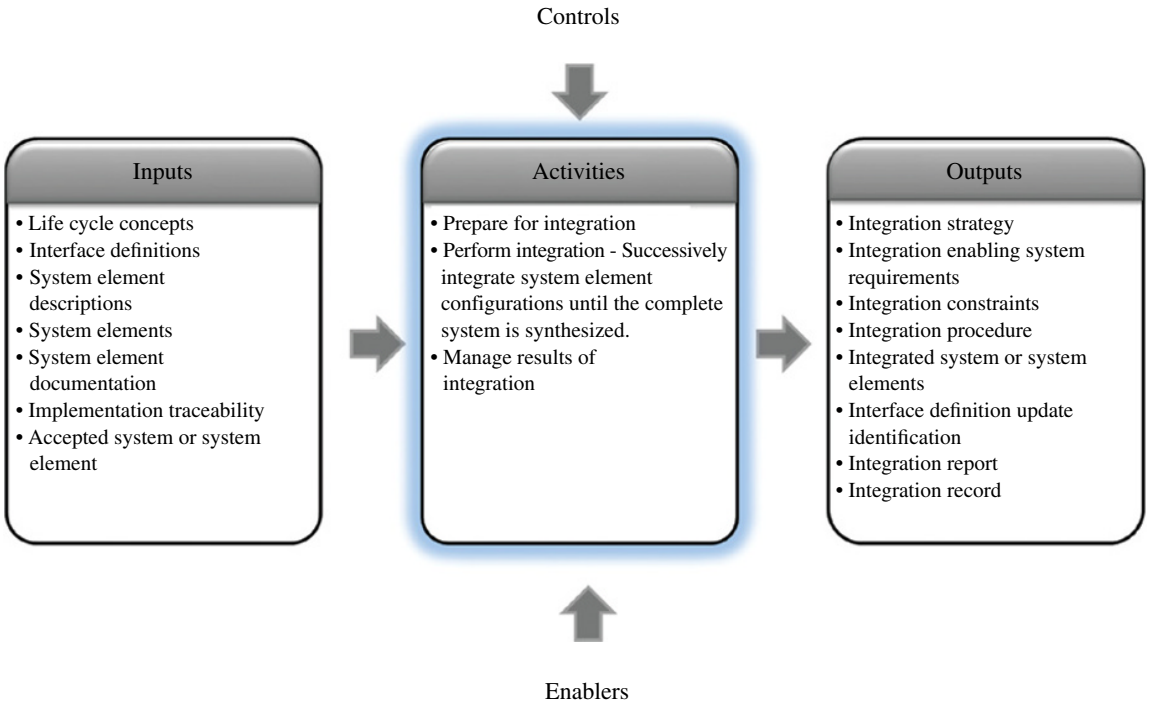


FIGURE 4.12 IPO diagram for the integration process. INCOSE SEH original figure created by Shortell and Walden. Usage per the INCOSE Notices page. All other rights reserved.

and on appropriate integration approaches and techniques.
    - Define the configurations of the aggregates to be built and verified (depending on sets of parameters).
    - Define the assembly procedures and related enablers.
  - Identify integration constraints on the SOI, arising from the integration strategy, to be incorporated in the system requirements, architecture, and design (this includes requirements such as accessibility, safety for integrators, and required interconnections for sets of implemented system elements and for enablers).
  - The acquisition of the enablers can be done through various ways such as rental, procurement, development, reuse, and subcontracting. An enabler may be a complete enabling system developed as a separate project from the project of the SOI.
- *Perform integration - Successively integrate system element configurations until the complete system is synthesized.*
  - Assemble the verified and validated system elements to form the incremental aggregate using the defined assembly procedures, the related integration enabling systems, and the interface control definitions.
  - Invoke the system V&V processes, as needed, to check the correct implementation of architectural characteristics and design properties and to check that the individual system elements provide the functions intended.
- *Manage results of integration.*
  - Identify and record the results of integration. Maintain bidirectional traceability of the updated integrated system elements with the updated system architecture, design, and system and interface requirements. Maintain the records, including configuration updates, per organizational policy.
  - Record anomalies observed during the integration process (identifying corrective actions or improvements), and resolve them using the quality assurance process (see Section 5.8).

– Update the integration strategy and schedule according to the progress of the project; in particular, the order of system elements assembly can be redefined or rescheduled because of unexpected events or unavailability of system elements as planned.
– Coordinate integration activities with the project manager (e.g., for scheduling, acquisition of enablers, hiring of qualified personnel and resources), the architects or designers (e.g., for understanding of the architecture, errors, defects, nonconformance reports), and the configuration manager (e.g., for versions of submitted elements, architecture and design baselines, enablers, assembly procedures).

Common approaches and tips:

- Define an integration strategy that accounts for the schedule of availability of system elements (including the personnel that will use, operate, maintain, and sustain the system) and is consistent with defect/fault isolation and diagnosis practices.
- Development of integration enablers such as tools and facilities can take as long as the system itself. Development should be started as early as possible and as soon as the preliminary architecture definition is frozen.
- The integration process of complex systems cannot be easily foreseen, and its progress may be difficult to control and observe. Therefore, it is recommended to plan integration with specific margins using flexible approaches and techniques, integrating sets by similar technologies, for example.
- Integrate aggregates in order to detect faults more easily. The use of the coupling matrix technique applies for all strategies and especially for the bottom-up integration strategy (see Section 4.8.2.3).

4.8.2 Elaboration

4.8.2.1 Concept of "Aggregate" The physical integration of a system is based on the notion of "aggregate." An aggregate is made up of several implemented system elements and their physical interfaces (system elements and connectors). Each aggregate is characterized by a configuration that specifies the implemented system elements to be physically assembled and their configuration status. A set of verification actions is applied on each aggregate. To perform these verification actions, a verification configuration that includes the aggregate plus verification tools is constituted. The verification tools are enabling elements and can be simulators (simulated system elements), stubs or caps, activators (launchers, drivers), harnesses, measuring devices, etc.

4.8.2.2 Integration by Level of System According to the Vee model, the system definition (top-down branch) is done by successive levels of decomposition; each level corresponds to physical architecture of systems and system elements. The integration (bottom-up branch) consists in following the opposite way of composition level by level.

On a given level, integration of implemented system elements is done on the basis of the physical architecture, using integration techniques or approaches as presented in the next section.

4.8.2.3 Integration Strategy and Approaches The integration of implemented system elements is performed according to a predefined strategy. The strategy relies on the way the architecture of the system has been defined. The strategy is described in an integration plan that defines the configuration of expected aggregates of implemented system elements, the order of assembly of these aggregates to carry out efficient verification actions and validation actions (e.g., inspections or tests). The integration strategy is thus elaborated in coordination with the selected verification strategy and validation strategy (see Sections 4.9 and 4.11).

To define an integration strategy, one can use one or several possible integration approaches and techniques. Any of these may be used individually or in combination. The selection of integration techniques depends on several factors, in particular the type of system element, delivery time, order of delivery of system elements, risks, constraints, etc. Each integration technique has strengths and weaknesses, which should be considered in the context of the SOI. Some integration techniques are summarized hereafter:

- *Global integration*—Also known as "big-bang integration"; all the delivered implemented system

elements are assembled in only one step. This technique is simple and does not require simulating the system elements not being available at that time. But it is difficult to detect and localize faults; interface faults are detected late. It should be reserved for simple systems, with few interactions and few system elements without technological risks.

- *Integration "with the stream"*—The delivered system elements are assembled as they become available. This technique allows starting the integration quickly. It is complex to implement because of the necessity to simulate the system elements not yet available. It is impossible to control the end-to-end "functional chains"; so global tests are postponed very late in the schedule. It should be reserved for well-known and controlled systems without technological risks.
- *Incremental integration*—In a predefined order, one or a very few system elements are added to an already integrated increment of system elements. This technique allows a fast localization of faults: a new fault is usually localized in lately integrated system elements or dependent on a faulty interface. It requires simulators for absent system elements and many test cases: each system element addition requires the verification of the new configuration and regression testing. This technique is applicable to any type of architecture.
- *Subset integration*—System elements are assembled by subsets (a subset is an aggregate), and then subsets are assembled together; could be called "functional chains integration." This technique saves time due to parallel integration of subsets; delivery of partial products is possible. It requires less means (enablers) and fewer test cases than integration by increments. The subsets may be defined during the architecture definition; they may correspond to subsystems/systems as defined in the architecture.
- *Top-down integration*—System elements or aggregates are integrated in their activation or utilization order. Availability of a skeleton of the system and early detection of architectural faults are possible; definition of test cases is close to reality; the reuse of test data sets is possible. But many stubs/caps need to be created; it is difficult to define test cases of the leaf system elements (lowest level). This technique is mainly used in intensive software systems. It starts from the system element of higher level; system elements of lower level are added until all leaf system elements are incorporated.
- *Bottom-up integration*—System elements or aggregates are integrated in the opposite order of their activation or utilization. The definition of test cases is easy; early detection of faults (usually localized in the leaf system element) is possible; the number of simulators to be used is reduced; an aggregate can be a subsystem. But the test cases shall be redefined for each step; drivers are difficult to define and realize; system elements of lower levels are "overtested"; this technique does not allow quick detection of the architectural faults. It is used for intensive software systems and for any kind of hardware system.
- *Criterion-driven integration*—The most critical system elements compared to the selected criterion are first integrated (e.g., dependability, complexity, technological innovation). The criteria are generally related to risks. This technique allows testing early and intensively critical system elements; early verification of architecture and design choices is possible. But the test cases and test data sets are difficult to define.
- *Reorganization of coupling matrices*—As noted in Section 4.4.2.6, coupling matrices are useful for highlighting interfaces during architecture definition as well as during integration. The integration strategy is defined and optimized by reorganizing the coupling matrix in order to group the system elements into aggregates and minimize the number of interfaces to be verified between aggregates. When verifying the interactions between aggregates, the matrix is an aid for fault detection. If by adding a system element to an aggregate, an error is detected, the fault can be either related to the system element, or to the aggregate, or to the interfaces. If the fault is related to the aggregate, it can relate to any system element or any interface between the system elements internal to the aggregate.

Usually, the integration strategy is defined as a combination of these approaches and techniques in order to optimize the integration work. The optimization takes into account the realization time of the system elements, their delivery scheduled order, their level of complexity, the technical risks, the availability of assembly tools, cost, deadlines, specific personnel capability, etc.

4.9 VERIFICATION PROCESS

4.9.1 Overview

4.9.1.1 Purpose As stated in ISO/IEC/IEEE 15288,

> [6.4.9.1] The purpose of the Verification process is to provide objective evidence that a system or system element fulfils its specified requirements and characteristics.

4.9.1.2 Content/Description As described here, this process is an instance of a verification process applied to a SOI, or any system or system element that compose it, to establish that it has been "built right."

The verification process can be applied to any engineering element that has contributed to the definition and realization of the system itself (e.g., verification of a system requirement, a function, an input/output flow, a system element, an interface, a design property, a verification procedure). The purpose of the verification process is to provide evidence that no error/defect/fault has been introduced at the time of any transformation of inputs into outputs; it is used to confirm that this transformation has been made "right" according to the requirements and selected methods, techniques, standards, or rules. As is often stated, verification is intended to ensure that the "product is built right," while validation is intended to ensure that the "right product is built." Verification is a transverse activity to every life cycle stage of the system. In particular during the development of the system, verification applies onto any activity and product resulting from the activity.

Figure 4.13 is the IPO diagram for the verification process.

4.9.1.3 Inputs/Outputs Inputs and outputs for the verification process are listed in Figure 4.13. Descriptions of each input and output are provided in Appendix E.

4.9.1.4 Process Activities The verification process includes the following activities:

- *Prepare for verification.*
  - Develop a strategy that prioritizes the verification actions to minimize costs and risks while maximizing operational coverage of system behaviors:
    - Establish a list of the items for verification, including requirements, architectural

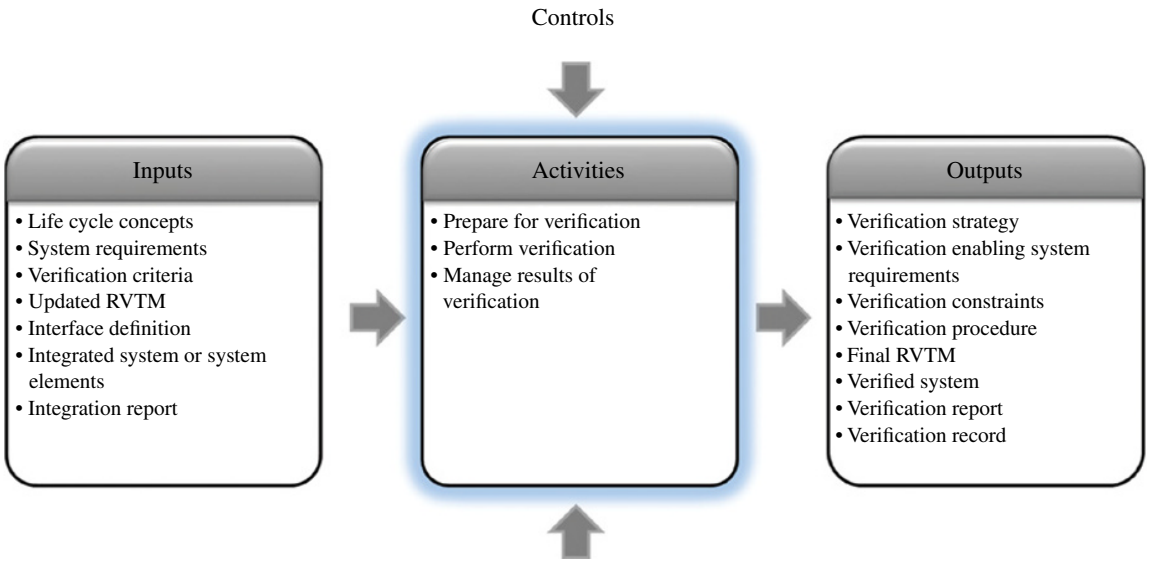


FIGURE 4.13 IPO diagram for the verification process. INCOSE SEH original figure created by Shortell and Walden. Usage per the INCOSE Notices page. All other rights reserved.

characteristics, or design properties, and define the corresponding verification actions. The approach to verification should be identified and documented at the time that a requirement is *first* documented to ensure that the requirement as written is indeed verifiable. This may require restating a requirement or decomposing it into several verifiable statements. (For example, how do you satisfy the legitimate requirement of a system being "user friendly"?)

- ◦ Establish a list of verification constraints that need to be considered. The constraints could impact the implementation of the verification actions and include contractual constraints, limitations due to regulatory requirements, cost, schedule, feasibility to exercise a function (such as in some ordinance), safety considerations, physical configurations, accessibility, etc.
- ◦ Considering the constraints, plan for the methods or techniques that will be applied for each verification action. The methods or techniques generally include inspection, analysis, demonstration, or test (see Section 4.9.2.2). Note that analysis often is considered to include modeling, simulation, and analogy when identifying the verification methods or techniques and success criteria are also defined that indicates successful verification.
- ◦ Establish the scope of the verification. Verification consumes resources: time, labor, facilities, and funds. The selection of what must be verified should be made according to the type of system, the objectives of the project, and the acceptable risks regarding the withdrawal of a verification action.

– Develop the verification procedures that support the verification actions.

- ◦ Schedule the execution of verification actions in the project steps and define the configuration of submitted items to verification actions.

– Identify verification constraints on the system or system elements, arising from the verification strategy, that relate to specific system requirements, architecture elements, or design elements. Typical constraints include performance characteristics, accessibility, and interface characteristics. Provide the constraint information for consideration in the system requirements definition, architecture definition, and design definition processes.

– Ensure that the necessary enabling systems, products, or services required for the verification actions are available, when needed. The planning includes the identification of requirements and interfaces for the enablers. The acquisition of the enablers can be done through various ways such as rental, procurement, development, reuse, and subcontracting. An enabler may be a complete enabling system developed as a separate project from the project of the SOI.

- *Perform verification.*
  - Implement the verification plan developed in the preceding subsection. That plan includes detailed descriptions for the selected verification actions:
    - ◦ Item to be verified
    - ◦ Expected results and success criteria
    - ◦ Selected verification method or technique
    - ◦ The data needed
    - ◦ The corresponding enabling systems, products, or services
  - Using the verification procedures, execute the verification actions and record the results.
  - Analyze the verification results against any established expectations and success criteria to determine whether the element being verified indicates conformance.
- *Manage results of verification.*
  - Identify and record verification results and enter data in the Requirements Verification and Traceability Matrix (RVTM). Maintain the records per organizational policy.
  - Record anomalies observed during the verification process, and analyze and resolve the anomalies (corrective actions or improvements) using the quality assurance process (see Section 5.8).
  - Establish and maintain bidirectional traceability of the verified system elements with the system architecture, design, and system and interface requirements that are needed for verification.
  - Provide baseline information for configuration management.

– Update the verification strategy and schedule according to the progress of the project; in particular, planned verification actions can be redefined or rescheduled as necessary.
– Coordinate verification activities with the project manager (e.g., for scheduling, acquisition of enablers, hiring of qualified personnel and resources), the architects or designers (e.g., for errors, defects, nonconformance reports), and the configuration manager (e.g., for versions of submitted items, requirements, architecture and design baselines, enablers, verification procedures).

Common approaches and tips:

- Beware the temptation to reduce the number of verification actions due to budget or schedule overruns. Remember that discrepancies and errors are more costly to correct later in the system life cycle.
- In the progress of the project, it is important to know, at any time, what has not been verified in order to estimate the risks about possibly dropping out some verification actions.
- Each system requirement should be quantitative, measurable, unambiguous, understandable, and testable. It is generally much easier and more cost-effective to ensure that requirements meet these criteria while they are being written. Requirements adjustments made after implementation and/or integration are generally much more costly and may have wide-reaching redesign implications. There are several resources that provide guidance on creating appropriate requirements (see the System Requirements Definition Process, Section 4.3).
- Avoid conducting verification only late in the schedule when there is less time to handle discrepancies.
- Testing the actual system is expensive and is not the only verification technique. Other techniques such as simulation, analysis, review, etc. can be used on other engineering elements representing the SOI such as models, mock-ups, or partial prototypes.

4.9.2 Elaboration

4.9.2.1 Notion of Verification Action A verification action describes what must be verified (e.g., a requirement, a characteristic, or a property as reference), on which item (e.g., requirement, function, interface, system element, system), the expected result (deduced from the reference), the verification technique to apply (e.g., inspection, analysis, demonstration, test), and on which level of decomposition of the system (e.g., SOI, intermediate level system element, leaf level system element).

The definition of a verification action applied to an engineering item (e.g., stakeholder requirement, system requirement, function, interface, system element, procedure, and document) includes the identification of the item on which the verification action will be performed, the reference used to define the expected result, and the appropriate verification technique.

The performance of a verification action onto the submitted item provides an obtained result which is compared with the expected result. The comparison enables the correctness of the item to be determined (see Fig. 4.14).

Examples of verification actions:

- *Verification of a stakeholder requirement or a system requirement*—To check the application of syntactic and grammatical rules and characteristics defined in the stakeholder needs and requirements definition process and the system requirements definition process such as necessity, implementation-free, unambiguous, consistent, complete, singular, feasible, traceable, and verifiable.
- *Verification of the architecture of a system*—To check the correct application of the appropriate patterns and heuristics used and the correct usage of modeling techniques or methods.
- *Verification of the design of a system element*—To check the correct usage of patterns, trade rules, or state of the art related to the concerned technology (e.g., software, mechanics, electronics, chemistry).
- *Verification of a system (product, service, or enterprise) or system element*—To check its realized characteristics or properties (e.g., as measured) against its specified requirements, expected architectural characteristics, and design properties (as described in the requirements, architecture, and design documents).

Considerations in selecting a verification approach include practical limitations of accuracy, uncertainty, repeatability that are imposed by the verification

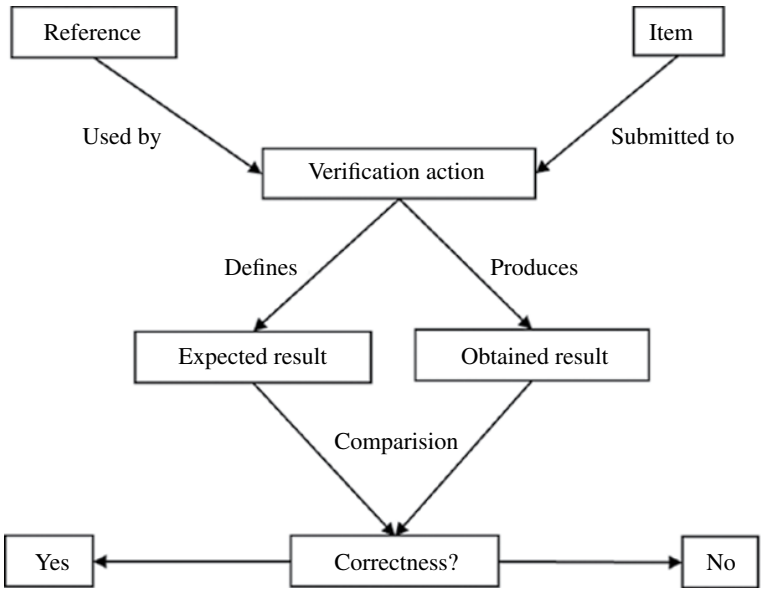


FIGURE 4.14 Definition and usage of a verification action. Reprinted with permission from Alain Faisandier. All other rights reserved.

enablers, the associated measurement methods, and the availability, accessibility, and interconnection with the enablers.

4.9.2.2 Verification Techniques Basic verification techniques are as follows (IEEE 1012, 2012; ISO/IEC/IEEE 29119, 2013; ISO/IEC/IEEE 29148, 2011):

- *Inspection*—This technique is based on visual or dimensional examination of an element; the verification relies on the human senses or uses simple methods of measurement and handling. Inspection is generally nondestructive and typically includes the use of sight, hearing, smell, touch, and taste; simple physical manipulation; mechanical and electrical gauging; and measurement. No stimuli (tests) are necessary. The technique is used to check properties best determined by observation (e.g., paint color, weight, documentation, listing of code). Peer reviews of process artifacts are also considered a type of inspection.
- *Analysis*—This technique is based on analytical evidence obtained without any intervention on the submitted element using mathematical or probabilistic calculation, logical reasoning (including the theory of predicates), modeling, and/or simulation under defined conditions to show theoretical compliance. Mainly used where testing to realistic conditions cannot be achieved or is not cost-effective.
- *Demonstration*—This technique is used to show correct operation of the submitted element against operational and observable characteristics without using physical measurements (no or minimal instrumentation or test equipment). It uses generally a set of actions selected to show that the element response to stimuli is suitable or to show that operators can perform their assigned tasks when using the element. Observations are made and compared with predetermined/expected responses.
- *Test*—This technique is performed onto the submitted element by which functional, measurable characteristics, operability, supportability, or performance capability is quantitatively verified when subjected to controlled conditions that are real or simulated. Testing often uses special test equipment or instrumentation to obtain accurate quantitative data to be analyzed.
- *Analogy or similarity*—This technique (often considered as a type of analysis technique) is based on evidence of similar elements to the submitted element or on experience feedback. It is absolutely necessary to show by prediction that the context is

invariant and that the outcomes are transposable (e.g., models, investigations, experience feedback). Analogy or similarity can only be used if the submitted element is similar in design, manufacture, and use; equivalent or more stringent verification actions were used for the similar element; and the intended operational environment is identical to or less rigorous than the similar element.

- *Simulation*—This technique (often considered as a type of analysis technique) is performed on models or mock-ups (not on the actual/physical elements) for verifying features and performance as designed.
- *Sampling*—This technique is based on verification of characteristics using samples. The number, tolerance, and other characteristics must be specified and be in agreement with the experience feedback.

Note: For techniques that do not include stimuli of the system element, no characteristics (exogenous attributes) can be observed only properties (endogenous attributes).

4.9.2.3 Integration, Verification, and Validation of the System There is sometimes a misconception that verification occurs after integration and before validation. In most of the cases, it is more appropriate to begin verification activities during development and to continue them into deployment and use.

Once system elements have been realized, their integration to form the whole system is performed. Integration assembles developed capabilities (via system elements) in preparation for verification actions as stated in the integration process (see Section 4.8).

4.9.2.4 Verification Level per Level Generally, the SOI has a number of layers of systems (made up of system elements at the next lower level). Thus, every system and system element is verified, and any findings possibly corrected before being integrated into the system of the higher level, as shown in Figure 4.15. In this figure, every time the term verify is used means that the corresponding verification process is invoked.

As necessary, systems and system elements are partially integrated in subsets (aggregates) in order to limit the number of properties to be verified within a single step. For each level, it is necessary to make sure by a set of verification actions that features stated at preceding level are not adversely affected. Moreover, a compliant result obtained in a given environment can turn noncompliant if the environment changes. So, as long as the system is not completely integrated and/or doesn't operate in the real operational environment, no result must be regarded as definitive.

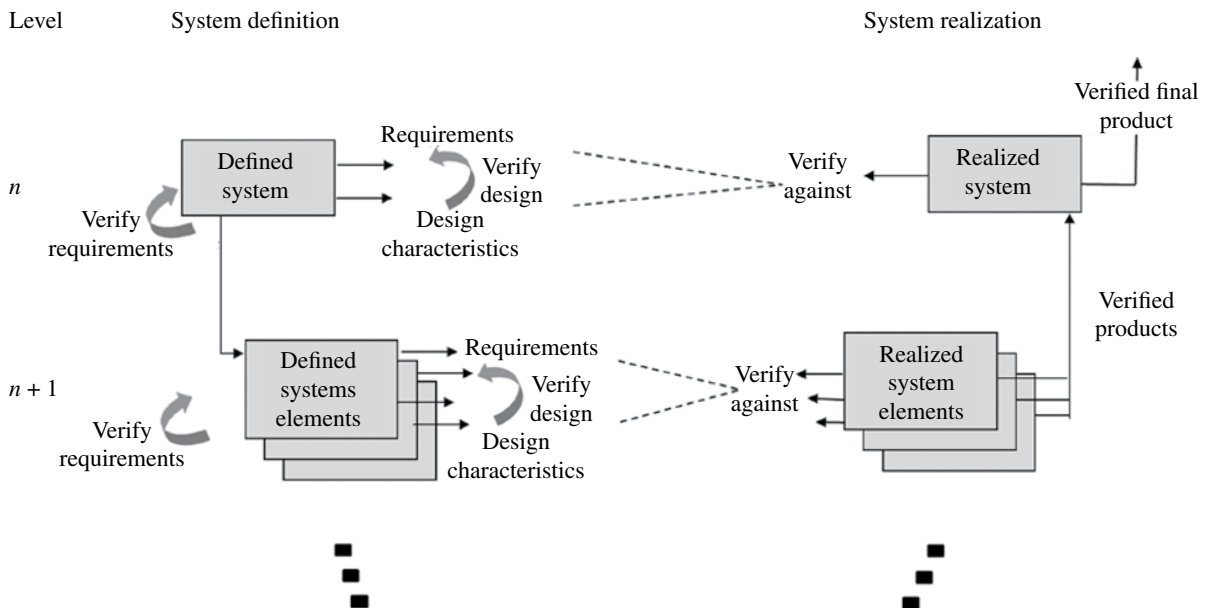


FIGURE 4.15 Verification level per level. Reprinted with permission from Alain Faisandier. All other rights reserved.

4.10 TRANSITION PROCESS

4.10.1 Overview

4.10.1.1 Purpose As stated in ISO/IEC/IEEE 15288,

> [6.4.10.1] The purpose of the Transition process is to establish a capability for a system to provide services specified by stakeholder requirements in the operational environment.

Ultimately, the transition process enables the transfer of custody of the system and responsibility for system support from one organizational entity to another. This includes, but is not limited to, transfer of custody from the development team to the organizations that will subsequently operate and support the system. Successful conclusion of the transition process typically marks the beginning of the utilization stage of the SOI.

4.10.1.2 Description The transition process installs a verified system in the operational environment along with relevant enabling systems, products, or services, such as operator training systems, as defined in the agreement. Using successful results from the verification process, the acquirer accepts that the system meets the specified system requirements in the intended operational environment prior to allowing a change in control, ownership, and/or custody. While this is a relatively short process, it should be carefully planned to avoid surprises and recrimination on either side of the agreement. Additionally, transition plans should be tracked and monitored to ensure all activities are completed to both parties' satisfaction, including resolution of any issues arising during transition. Figure 4.16 is the IPO diagram for the transition process.

4.10.1.3 Inputs/Outputs Inputs and outputs for the transition process are listed in Figure 4.16. Descriptions of each input and output are provided in Appendix E.

4.10.1.4 Process Activities The transition process includes the following activities:

- *Prepare for the transition.*
  - Plan for the transition of the system. The strategy should include operator training, logistics support, delivery strategy, and problem rectification/resolution strategy.
  - Develop installation procedures.

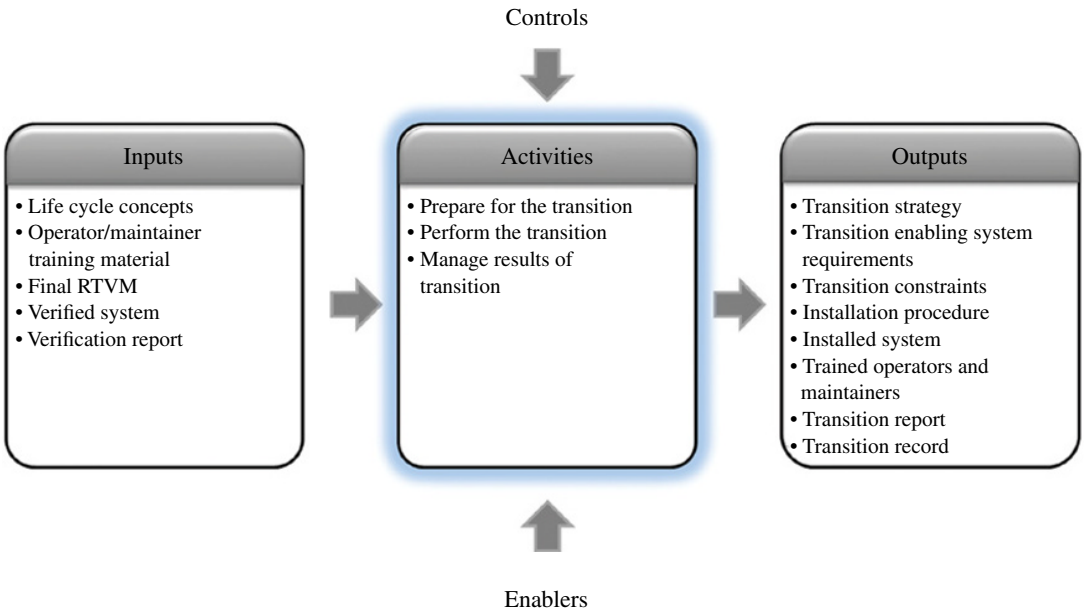


FIGURE 4.16 IPO diagram for the transition process. INCOSE SEH original figure created by Shortell and Walden. Usage per the INCOSE Notices page. All other rights reserved.

  - Ensure that the necessary enabling systems, products, or services required for transition are available, when needed. The planning includes the identification of requirements and interfaces for the enablers. The acquisition of the enablers can be done through various ways such as rental, procurement, development, reuse, and subcontracting. An enabler may be a complete enabling system developed as a separate project from the project of the SOI.
- *Perform the transition.*
  - Using the installation procedures, install the system.
  - Train the users in the proper use of the system and affirm users have the knowledge and skill levels necessary to perform operation and maintenance activities. This includes a complete review and handoff of operator and maintenance manuals, as applicable.
  - Receive final confirmation that the installed system can provide its required functions and can be sustained by the enabling systems and services. This process typically ends with a formal, written acknowledgement that the system has been properly installed and verified, that all issues and action items have been resolved, and that all agreements pertaining to development and delivery of a fully supportable system have been fully satisfied or adjudicated.
  - After the demonstration of functionality in the operational site and any review for operational readiness, the system can be placed into service.
- *Manage results of transition.*
  - Postimplementation incidents and problems are captured and may lead to corrective actions or changes to the requirements. The quality assurance process is used for the incident and problem resolution that is reported during performance of the transition process.
  - Record anomalies observed during the transition process. These provide awareness of the results, information needed to address anomalies, and a historical record. Anomalies may stem from the transition strategy, supporting enabling systems, interfaces, etc. The project assessment and control process is used to analyze the anomalies and determine what, if any, action is needed.
  - Maintain bidirectional traceability of the transitioned system elements with the transition strategy, system architecture, design, and system requirements.
  - Provide baseline information for configuration management.

Common approaches and tips:
- When acceptance activities cannot be conducted within the operational environment, a representative locale is selected.
- This process relies heavily on quality assurance and configuration management documentation.

4.11 VALIDATION PROCESS

4.11.1 Overview

4.11.1.1 Purpose As stated in ISO/IEC/IEEE 15288,

> [6.4.11.1] The purpose of the Validation process is to provide objective evidence that the system, when in use, fulfills its business or mission objectives and stakeholder requirements, achieving its intended use in its intended operational environment.

4.11.1.2 Description The validation process is applied to a SOI, or any system or system element that composes it, at the appropriate points in the life cycle stages to provide confidence that the right system (or system element) has been built.

The validation process can be applied to any system element or engineering item of the system or its definition that has been defined or realized (e.g., validation of a stakeholder requirement, a system requirement, a function, an input/output flow, a system element, an interface, a design property, an integration aggregate, a validation procedure). Thus, the validation process is performed to help ensure that the system or any system element meets the need of its stakeholder in the life cycle (i.e., the engineering processes and the transformation of inputs produced what was intended—the "right" result).

Validation is a transverse activity to every life cycle stage of the system. In particular, during the development of the system, validation applies to any process/activity and product resulting from the process/activity.

The validation process works closely with other life cycle processes. For example, the business or mission analysis process establishes a targeted operational capability. The operational capability (e.g., mission or business profile and operational scenarios) is transformed by the stakeholder needs and requirements definition process into stakeholder needs and requirements. The validation process works concurrently with these processes to define the applicable validation actions and validation procedures through the life cycle to ensure the system evolves in a way that there is a high level of confidence that the operational capability will be provided, as refined in the stakeholder needs and requirements.

Figure 4.17 is the IPO diagram for the validation process.

4.11.1.3 Inputs/Outputs Inputs and outputs for the validation process are listed in Figure 4.17. Descriptions of each input and output are provided in Appendix E.

4.11.1.4 Process Activities The validation process includes the following activities:

- *Prepare for validation.*
  - Establish the validation strategy, which is often part of a validation plan, that optimizes the number and type of validation actions while minimizing costs and risks:
    - Identify the stakeholders who will be involved in the validation activities and define their roles and responsibilities. This can include the acquirer, the supplier, and a third-party representative.
    - The scope of the validation plan is dependent on the life cycle stage and the progress within it. Validation may be appropriate for the full system, a system element, or an artifact, such as the ConOps or a prototype, in addition to the delivered system.
    - Establish a list of validation constraints that need to be considered. The constraints could impact the implementation of the validation actions and include contractual constraints, limitations due to regulatory requirements, cost, schedule, feasibility to exercise a function (such as in some ordinance), safety considerations, physical configurations, accessibility, etc.
    - With appropriate consideration to the constraints, select suitable validation approach to be applied, such as inspection, analysis,

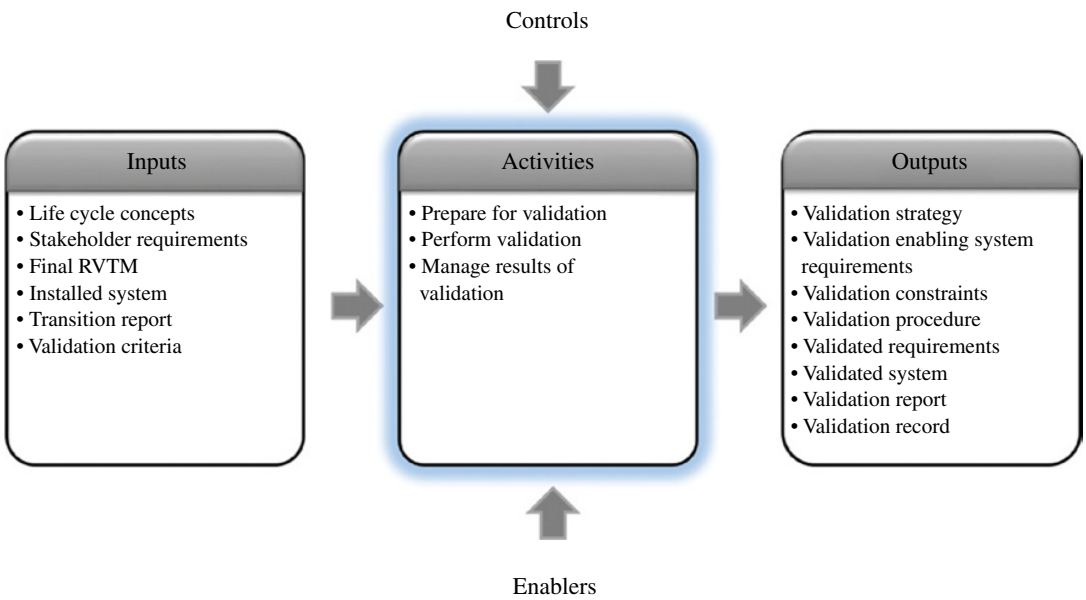


FIGURE 4.17 IPO diagram for the validation process. INCOSE SEH original figure created by Shortell and Walden. Usage per the INCOSE Notices page. All other rights reserved.

demonstration, or test, depending on the life cycle stage. Identify any enablers needed.
    - It may be necessary to prioritize the validation actions. The validation actions should be prioritized and evaluated against constraints, risks, type of system, project objectives, and other relevant criteria.
    - Determine if there are any validation gaps and that the resulting validation actions will provide an acceptable level of confidence that the system or system element will meet the identified needs.
    - Ensure appropriate scheduling via the project planning process to meet the requirements for the execution of the validation actions in the applicable project steps.
    - Define the configuration of submitted items to validation actions.
  - Identify validation constraints on the system, arising from the validation strategy, to be incorporated in the stakeholder requirements. This includes practical limitations of accuracy, uncertainty, repeatability that are imposed by the validation enablers, the associated measurement methods, and the availability, accessibility, and interconnection with enablers.
  - Ensure that the necessary enabling systems, products, or services required for the validation actions are available, when needed. The planning includes the identification of requirements and interfaces for the enablers. An enabler may be a complete enabling system developed as a separate project from the project of the SOI.
- *Perform validation.*
  - Develop the validation procedures that support the validation actions.
  - Ensure readiness to conduct validation: availability and configuration status of the system/item, the availability of the validation enablers, qualified personnel or operators, resources, etc.
  - Conduct validation actions in accordance with the procedures. This should include performing the actions in the operational environment or one as close to it as possible. During the conduct of the validation actions, record the results of the actions, as they are performed.
- *Manage results of validation.*
  - Identify and record validation results and enter data in the validation report (including any necessary updates to the RVTM). Maintain the records per organizational policy.
  - Record anomalies observed during the validation process, and analyze and resolve the anomalies (corrective actions or improvements) using the quality assurance process (see Section 5.8).
    - Ensure the results and anomalies and/or nonconformances are analyzed using the project assessment and control process.
    - Compare the obtained results with the expected results; deduce the degree of conformance of the submitted item (i.e., provides the services expected by the stakeholder); decide whether the degree of conformance is acceptable.
    - Problem resolution is handled through the quality assurance and project assessment and control processes. Changes to the system or system element definition (i.e., requirements, architecture, design, or interfaces) and associated engineering artifacts are performed within other technical processes.
  - Obtain acquirer (or other authorized stakeholders) acceptance of validation results.
  - Maintain bidirectional traceability of the validated system elements with the validation strategy, business/mission analysis, stakeholder requirements, system architecture, design, and system requirements.
  - Provide baseline information for configuration management.
  - Update the validation strategy and schedule according to the progress of the project; in particular, planned validation actions can be redefined or rescheduled as necessary.

Common approaches and tips:

- Validation methods during the business and mission analysis process include assessment of OpsCon through operational scenarios that exercise all system operational modes and demonstrating system-level performance over the entire operating regime. The architects and designers use the results of this activity

to forecast success in meeting the expectations of users and the acquirer, as well as to provide feedback to identify and correct performance deficiencies before implementation (Engel, 2010).

- It is recommended to start the drafting of the validation plan as soon as the first OpsCon and scenarios and stakeholder requirements are known. Early consideration of potential validation methods or techniques allows the project to anticipate constraints, estimate cost, and start the acquisition of validation enablers such as test facilities, simulators, etc., avoiding cost overruns and schedule slippages.
- Validation is applied to the operational system, but is most effective if it is also applied earlier by analysis, simulation, emulation, etc. of anticipated operational characteristics.
- A key output of validation is the assurance provided of the loci of system dynamic and integrity limits. These envelopes provide actionable knowledge for users to determine system suitability, anticipated effectiveness, survivability, and refurbishment.
- Validation also reveals the effects the SOI may have on collateral, enabling, or interoperating systems. Validation actions and analysis should include these system interactions in the scope.
- Involve the broadest range of stakeholders with validation. Often, the end users and other relevant stakeholders are involved in the validation activities.
- If possible, involve users/operators with validation. Validation will often involve going back directly to the users to have them perform some sort of acceptance test under their own local conditions.

4.11.2 Elaboration

4.11.2.1 Notion of Validation Action A validation action describes what must be validated (e.g., an operational scenario, a requirement, or a set of requirements as reference), on which item (e.g., requirement, function, interface, system element, system), the expected result (deduced from the reference), the validation technique to apply (e.g., inspection, analysis, demonstration, test), and on which level of the system hierarchy (e.g., SOI, intermediate level system element, leaf level system element).

The definition of a validation action applied to an engineering item (e.g., stakeholder requirement, system requirement, function, interface, system element, procedure, document) includes the identification of the item on which the validation action will be performed, the reference used to define the expected result, and the appropriate validation technique.

The performance of a validation action onto the submitted item provides an obtained result which is compared with the expected result. The comparison enables the project to judge the element's acceptability regarding the relevance in the context of use (see Fig. 4.18).

Examples of validation actions:

- *Validation of a requirement*—To make sure its content is justified and relevant to stakeholder needs or expectations.
- *Validation of an engineering artifact (architecture, design, etc.)*—To make sure its content is justified and relevant to stakeholder needs or expectations and contributes to achieving the mission or business profile and operational scenarios.
- *Validation of a system (product, service, or enterprise)*—To demonstrate that the product, service, or enterprise satisfies its stakeholder requirements, mission or business profile, and operational scenarios.

4.11.2.2 Validation Techniques The validation techniques are the same as those used for verification (see Section 4.9.2.2), but the purposes are different; verification is used to show compliance with the specified system requirements and to detect errors/defects/faults, whereas validation is to prove satisfaction of the desired operational capability through showing operational scenarios and stakeholder requirements can be met.

A Requirements and Validation Traceability Matrix may be used to record data such as validation action list, selected validation method/technique to validate implementation of every engineering item (in particular, operational scenarios and stakeholder requirements), the expected results, and the obtained results when a validation action has been performed. The use of such a matrix enables the project team to ensure that selected operational scenarios and stakeholder requirements have been validated, or to evaluate the percentage of validation actions completed.

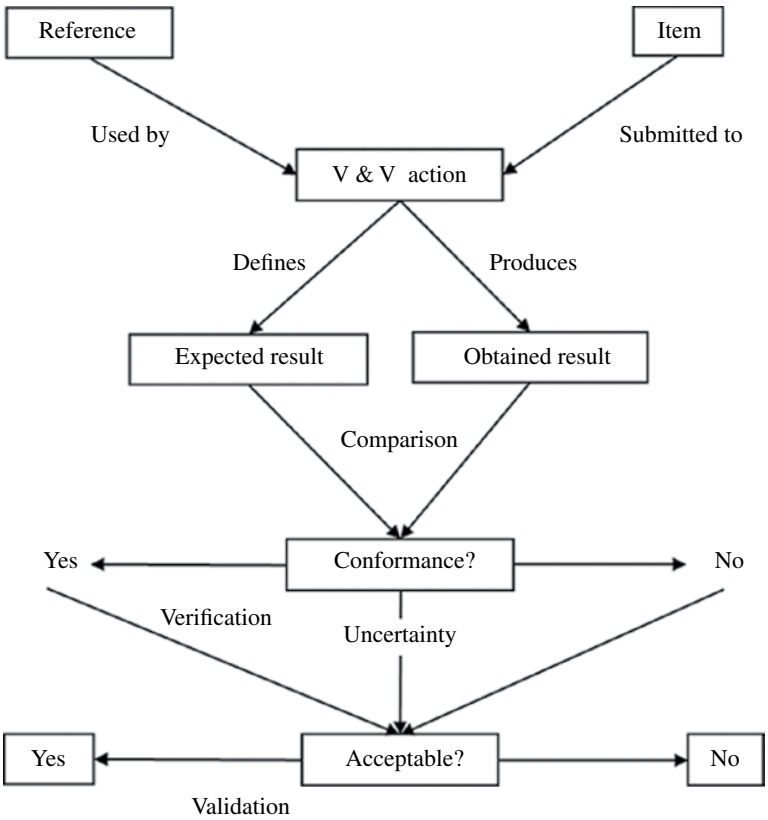


FIGURE 4.18 Definition and usage of a validation action. Reprinted with permission from Alain Faisandier. All other rights reserved.

4.11.2.3 Validation, Operational Validation, Acceptance, and Certification

Validation and Operational Validation Validation concerns the global system seen as a whole and is based on the totality of requirements (system requirements, stakeholder requirements). It is obtained gradually throughout the development stage by pursuing several nonexclusive ways:

- Cumulating V&V actions' results provided by application of the corresponding processes to every engineering item
- Performing final validation actions onto the complete integrated system in an industrial environment (as close as possible to the future operational environment)
- Performing operational validation actions onto the complete system in its operational environment (context of use)

The goal is to completely validate the system capability to meet all requirements prior to the production and utilization stages. Problems uncovered in these stages are very costly to correct. As such, early discovery of deviations from requirements reduces overall project risk and helps the project deliver a successful, low-cost system. Validation results are an important element of decision gate reviews.

Acceptance Acceptance is an activity conducted prior to transition such that the acquirer can decide that the system is ready to change ownership from supplier to acquirer. A set of operational validation actions is often exercised, or a review of validation results is systematically performed.

Certification Certification is a written assurance that the product or article has been developed, and can perform its assigned functions, in accordance with legal or industrial standards (e.g., for aircraft). The development

reviews, verification results, and validation results form the basis for certification. However, certification is typically performed by outside authorities, without direction as to how the requirements are to be verified. For example, this method is used for electronics devices via Conformité Européenne (CE) certification in Europe and Underwriters Laboratories (UL) certification in the United States and Canada.

Readiness for Use As part of the analysis of the validation results, the project team needs to make a readiness for use assessment. This may occur many times in the life cycle, including the first article delivery, completion of production (if more than a single system is produced), and following maintenance actions. In the field, particularly after maintenance, it is necessary to establish whether the system is ready for use.

Qualification The system qualification requires that all V&V actions have been successfully performed. These V&V actions cover not only the SOI itself but also all the interfaces with its environment (e.g., for a space system, the validation of the interface between space segment and ground segment). The qualification process has to demonstrate that the characteristics or properties of the realized system, including margins, meet the applicable system requirements and/or stakeholder requirements. The qualification is concluded by an acceptance review and/or an operational readiness review.

As illustration of this, for a space system, the last step of the qualification is covered by the first launch or the first flight. This first flight needs to be milestoned by a flight readiness review that will verify that the flight and ground segments including all supporting systems such as tracking systems, communication systems, and safety systems are ready for launch. A complementary review can be held just before launch (launch readiness review) to authorize the launch. A successful launch participates to the qualification process, but the final system qualification is achieved only after in-orbit tests for a spacecraft or even several flights for a launcher in order to cover the different missions for which the system has been developed.

4.11.2.4 *Validation Level per Level*

Generally, the SOI has been decomposed during architecture definition

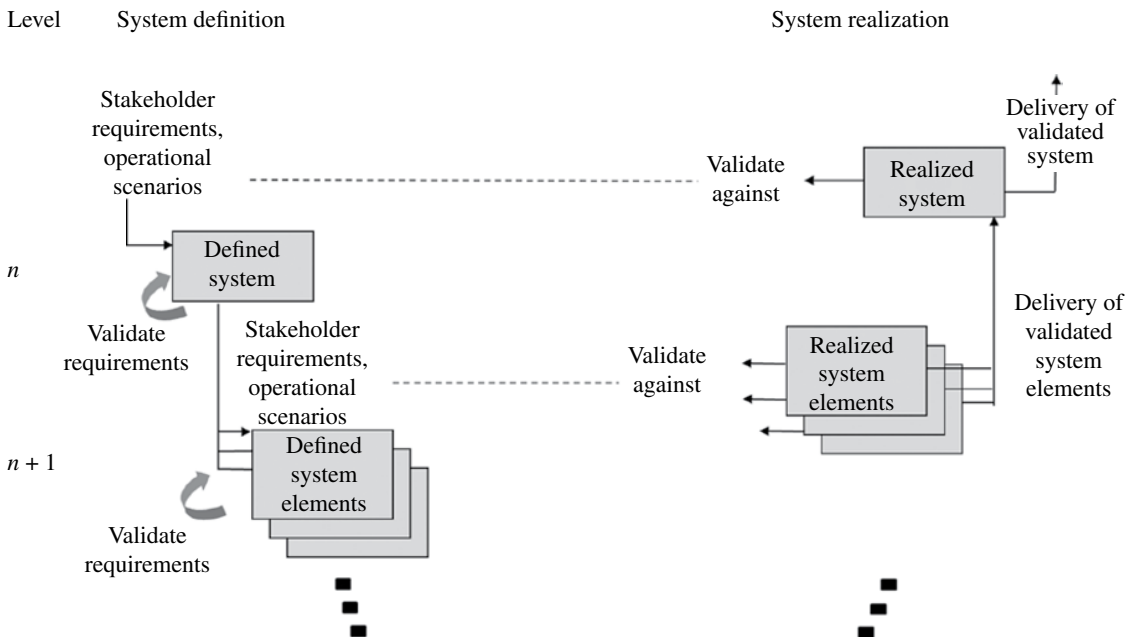


FIGURE 4.19 Validation level per level. Note: System requirements are validated against the stakeholder requirements. Reprinted with permission from Alain Faisandier. All other rights reserved.

in a set of layers of systems; thus, every system and system element is validated and possibly corrected before being integrated into the parent system of the higher level, as shown in Figure 4.19. In this figure, every time the term validate is used means that the corresponding validation process is invoked.

As necessary, systems and system elements are partially integrated in subsets (aggregates) in order to limit the number of properties to be validated within a single step. For each level, it is necessary to make sure by a set of final validation actions that features stated at preceding level are not adversely affected. Moreover, a compliant result obtained in a given environment can turn noncompliant if the environment changes. So, as long as the system is not completely integrated and/or doesn't operate in the real operational environment, no result must be regarded as definitive.

4.12 OPERATION PROCESS

4.12.1 Overview

4.12.1.1 Purpose As stated in ISO/IEC/IEEE 15288,

> [6.4.12.1] The purpose of the Operation process is to use the system to deliver its services.

This process is often executed concurrent with the maintenance process.

4.12.1.2 Description The operation process sustains system services by preparing for the operation of the system, supplying personnel to operate the system, monitoring operator–system performance, and monitoring the system performance. When the system replaces an existing system, it may be necessary to manage the migration between systems such that persistent stakeholders do not experience a breakdown in services.

The utilization and support stages of a system usually account for the largest portion of the total LCC. If system performance falls outside acceptable parameters, this may indicate the need for corrective actions in accordance with the support concept and any associated agreements. When the system or any of its constituent elements reach the end of their planned or useful life, the system may enter the disposal process (see Section 4.14). Figure 4.20 is the IPO diagram for the operation process.

4.12.1.3 Inputs/Outputs Inputs and outputs for the operation process are listed in Figure 4.20. Descriptions of each input and output are provided in Appendix E.

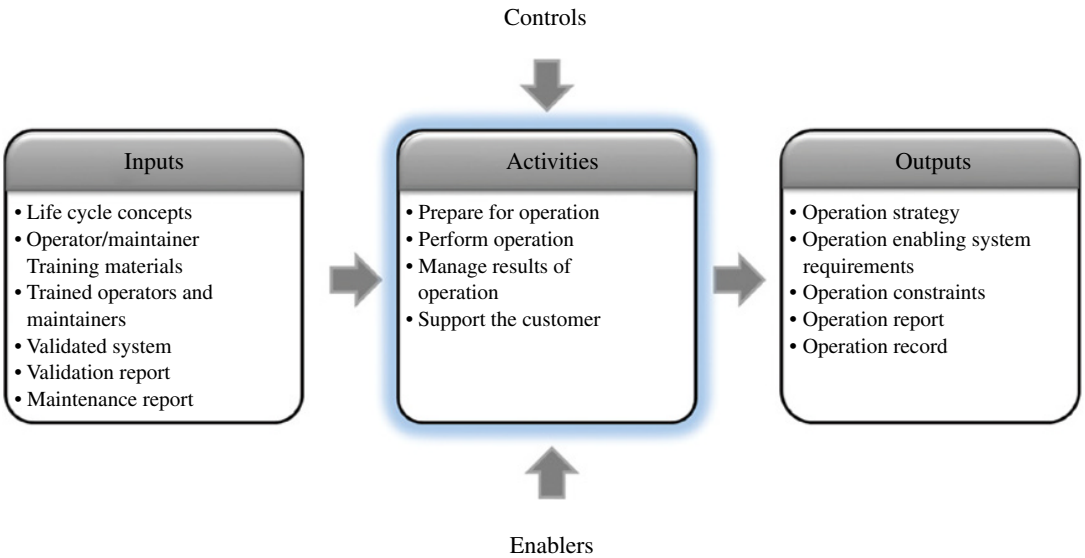


FIGURE 4.20 IPO diagram for the operation process. INCOSE SEH original figure created by Shortell and Walden. Usage per the INCOSE Notices page. All other rights reserved.

4.12.1.4 Process Activities The operation process includes the following activities:

- *Prepare for operation.*
  - Plan for operation, including the development of the strategy:
    - Establish availability of equipment, services, and personnel and performance tracking system.
    - Verify schedules for personnel and facilities (on a multishift basis if appropriate).
    - Define business rules related to modifications that sustain existing or enhanced services.
    - Implement the OpsCon and environmental strategies.
    - Review operational performance measures, thresholds, and criteria.
    - Verify all personnel have had the applicable system safety training.
  - Feed back any operation constraints on the system or system elements to the system requirements definition, architecture definition, or design definition processes.
  - Ensure that the necessary enabling systems, products, or services required for operation are available, when needed. The planning includes the identification of requirements and interfaces for the enablers. The acquisition of the enablers can be done through various ways such as rental, procurement, development, reuse, and subcontracting. An enabler may be a complete enabling system developed as a separate project from the project of the SOI.
  - Identify operator skill sets and train operators to operate the system.
- *Perform operation.*
  - Operate the system according to the OpsCon.
  - Track system performance and account for operational availability. This includes operating the system in a safe manner and performing operational analysis to determine system noncompliance.
  - When abnormal operational conditions warrant, conduct planned contingency actions. Perform system contingency operations, if necessary.
- *Manage results of operation.*
  - Document the results of operations.
  - Record anomalies observed during the operation process, and analyze and resolve the anomalies (corrective actions or improvements) using the quality assurance process (see Section 5.8). Implementation of procedures for restoring operations to a safe state should be followed. Maintain bidirectional traceability of the operations elements with the operation strategy, business/mission analysis, ConOps, OpsCon, and stakeholder requirements.
- *Support the customer.*
  - Perform tasks needed to address customer requests.

4.12.2 Elaboration

4.12.2.1 Operation Enabling Systems The operation process includes other enabling systems that need to be considered that may be different from other processes. The following amplifies on these:

- *Operational environment*—The circumstances surrounding and potentially affecting something that is operating. For example, electronic or mechanical equipment may be affected by high temperatures, vibration, dust, and other parameters that constitute the operating environment.
- *Training systems*—Provides operators with knowledge and skills required for proper system operations.
- *Technical data*—Procedures, guidelines, and checklists needed for proper operation of the system, including prerequisites for operation, procedures for activation and checkout of the system, operating procedures, procedures for monitoring system performance, procedures for problem resolution, and procedures for shutting the system down.
- *Facilities and infrastructure*—Facilities (e.g., buildings, airfields, ports, roadways) and infrastructure (e.g., IT services, fuel, water, electrical service) required for system operation.
- *Sustaining engineering*—Monitors system performance, conducts failure analysis, and proposes corrective actions to sustain required operational capabilities.

- *Maintenance planning and management*—With the goal of minimizing system downtime, planned and/or preventive maintenance is performed to sustain system operations. The maintenance system responds to operator trouble/problem reports, conducts corrective maintenance, and restores system for operations.

4.13 MAINTENANCE PROCESS

4.13.1 Overview

4.13.1.1 Purpose As stated in ISO/IEC/IEEE 15288,

> [6.4.13] The purpose of the Maintenance process is to sustain the capability of the system to provide a service.

4.13.1.2 Description Maintenance includes the activities to provide operations support, logistics, and material management. Based on feedback from ongoing monitoring of the operational environment, problems are identified, and corrective, remedial, or preventive actions are taken to restore full system capability. This process contributes to the system requirements definition process, architecture definition process, and design definition process when considerations of constraints imposed in later life cycle stages are used to influence the system requirements, architecture, and design. Figure 4.21 is the IPO diagram for the maintenance process.

4.13.1.3 Inputs/Outputs Inputs and outputs for the maintenance process are listed in Figure 4.21. Descriptions of each input and output are provided in Appendix E.

4.13.1.4 Process Activities The maintenance process includes the following activities:

- *Prepare for maintenance.*
  - Plan for maintenance, including the development of the strategy, that includes the following:
    - Sustained service across the life cycle in order to meet customer requirements and achieve customer satisfaction. The strategy should define the types of maintenance actions (preventive, corrective, modification) and levels of maintenance (operator, *in situ*, factory).
    - The various types of maintenance actions, including corrective maintenance (addressing failures or problems), adaptive maintenance (addressing changes needed to accommodate system evolution), perfective maintenance (addressing enhancements), and scheduled

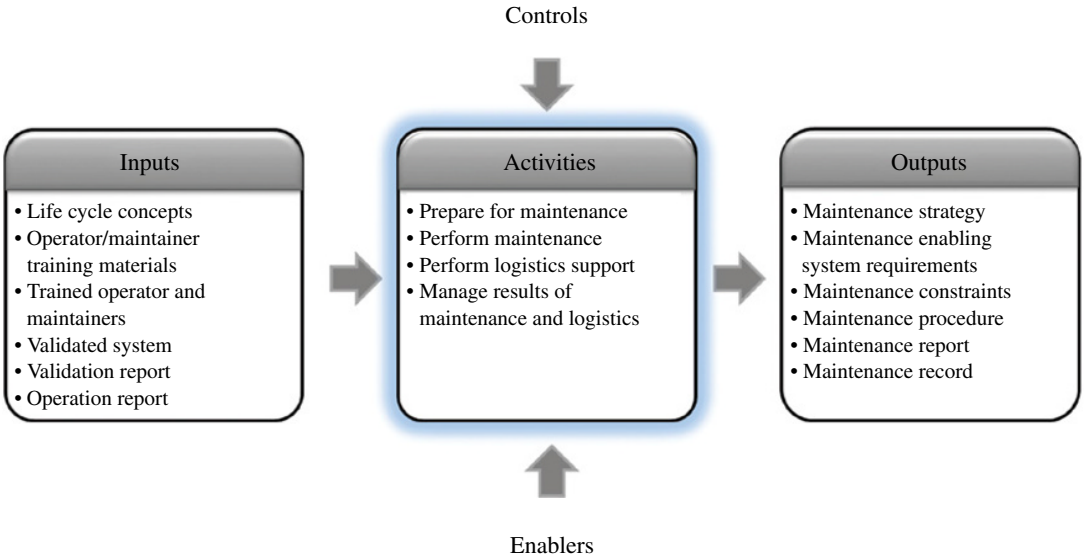


FIGURE 4.21 IPO diagram for the maintenance process. INCOSE SEH original figure created by Shortell and Walden. Usage per the INCOSE Notices page. All other rights reserved.

preventive maintenance (addressing routine servicing to prevent failures) while minimizing operational downtime.

- The approach to address logistics needs across the life cycle, considering the following:
  - Approach to manage spare parts or system elements (e.g., number, type, storage, location, shelf life, conditions, renewal frequency). Analysis includes requirements and plans for packaging, handling, storage, and transportation (PHS&T).
  - Anticounterfeit approach (i.e., prevention of counterfeit system elements, especially parts, in the supply chain).
  - Identify or define sources of technical data and training needed to support maintenance.
  - Identify or define the skills, training, qualifications, and number of personnel for maintenance. Consider any regulatory requirements that drive the need for specific skills, such as safety or security.

- Feed back any maintenance constraints on the system or system elements to the system requirements definition, architecture definition, or design definition processes.
- Use the system analysis process to support any trades needed of the maintenance strategy and approach to ensure affordability, feasibility, supportability, and sustainability of the system maintenance.
  - Ensure that the necessary enabling systems, products, or services required for maintenance are available, when needed. The planning includes the identification of requirements and interfaces for the enablers. The acquisition of the enablers can be done through various ways such as rental, procurement, development, reuse, and subcontracting. An enabler may be a complete enabling system developed as a separate project from the project of the SOI.
  - Assign trained, qualified personnel to be maintainers.

- *Perform maintenance.*
  - Develop maintenance procedures for preventive and corrective maintenance.
  - Identify, record, and resolve system anomalies.
  - Restore system operation after a failure.
  - Plan future maintenance by reviewing and analyzing anomaly reports.
  - Initiate analysis and corrective action to remedy previously undetected design errors.
  - Per the identified schedule, perform preventive maintenance actions using defined maintenance procedures.
  - Determine the need for adaptive or perfective.
- *Perform logistics support.*
  - Conduct acquisition logistics actions—Includes performing trade studies and analysis to determine the most cost-effective means to support the system across the life cycle. Design influence considers features that impact the inherent reliability and maintainability of system services contrasted with the affordability of other support options including the use of spare parts. Design considerations are often constrained by availability requirements, the impact of supply chain management, manpower restrictions, and system affordability. Acquisition logistics plans for and develops strategies for system life cycle supportability (maintenance, supply support, support equipment, staffing, and enabling systems) concurrently with the definition of the system requirements.
  - Conduct operational logistics actions—Operational logistics is the concurrent tuning of both the SOI and enabling systems throughout the operational life to ensure effective and efficient delivery of system capabilities. Operational logistics actions enable the system to achieve required operational readiness. The actions include staffing, maintenance management, supply support, support equipment, technical data needs (e.g., manuals, instructions, lists), training support, sustaining engineering, computing resources, and facilities. Operational logistics provides a rich source of data concerning the operational performance of the system. This data should be used to support trend analysis and provides a direct feedback loop on system effectiveness and efficiency and insight into customer satisfaction.
- *Manage results of maintenance and logistics.*
  - Document the results of the maintenance process.

– Record anomalies observed during the maintenance process, and analyze and resolve the anomalies (corrective actions or improvements) using the quality assurance process (see Section 5.8). Identify and record trends of maintenance and logistics actions.
– Maintain bidirectional traceability of the maintenance actions and the applicable system elements and system definition artifacts.
– Obtain feedback from customers to understand the level of satisfaction with the maintenance and logistics support.

4.13.2 Elaboration

Maintenance is a term often used to refer to a phase during which the system is already operational and the primary activities are related to sustaining the capability of the system, system enhancement, upgrades, and modernization (Patanakul and Shenhar, 2010). In this context, the objective is to design the system from the start to allow for the possibility of maintaining value over the entire system life cycle, thereby promoting value sustainment (Ross et al., 2008). Since operations and maintenance costs generally comprise a significant percent of total LCC, maintenance is an important part of the system definition, often linked to logistics engineering, disposal, and environmental impact analysis.

Maintenance helps ensure that a system continues to satisfy its objective over its intended lifetime. In that timeframe, system expectations will expand, environments in which the system is operated will change, technology will evolve, and elements of the system may become unsupportable and need to be replaced. The COTS desktop computing environment is a case in point. Since the introduction of USB printers, it is difficult to find cables to support parallel port printers.

Maintenance is an integrated effort designed to address aging systems and a need to maintain those systems in operation. A maintenance program may include reengineering electronic and mechanical elements to cope with obsolescence, developing automated test equipment, and extending the life of aging systems through technology insertion enhancements and proactive maintenance (Herald et al., 2009). There are two main mitigation strategies that help users cope with the performance capability gap: a performance-based logistics (PBL) approach known as performance-based life cycle product support and technology refreshment programs (TRPs) (Sols et al., 2012).

An open system architecture approach has been found to improve the ability to insert new capabilities (threat evolution), reduce development time, reduce maintenance cost, generate affordable COTS obsolescence management, and increase human systems integration. Open architectures enable a more capable, reliable, adaptable, and resilient system that accommodates continuous organizational and technology changes (Jackson and Ferris, 2013).

4.13.2.1 Maintenance Concept The maintenance concept is an important life cycle concept document. The following are recommended details that should be considered:

- *Types of maintenance*
  – Corrective maintenance—Processes and procedures for restoring system services to normal operations (e.g., remove and replace hardware, reload software, apply a software patch). Includes postmaintenance test procedures to verify that the system is ready for operations. Time spent troubleshooting can be greatly reduced by well-engineered diagnostic capabilities such as built-in test (BIT).
  – Preventive maintenance—Processes and procedures for scheduled/routine maintenance actions (e.g., cleaning, filter changes, visual inspections) needed to sustain optimal system operational performance.
  – System modifications—Processes and procedures intended to extend (sustain) the useful life of the system or to provide new (upgrade) system capabilities.
- *Levels of maintenance/repair*
  – User/operator maintenance—Some routine and simple maintenance tasks are able to be performed by the system operators or users. Operator maintenance includes routine (e.g., filter changes, recording data) and corrective (e.g., "software" resets, install a ready spare, tire change) tasks.
  – *In situ* maintenance and repair (sometimes referred to as "field" maintenance)—Maintenance

tasks performed by a trained maintainer at, or near, the operational location.

– Factory (maintenance, repair, and overhaul) (sometimes referred to as "depot" maintenance)—Major maintenance tasks that require advanced maintenance skills and special tooling/equipment not available at the operational location.

4.13.2.2 Maintenance Enabling Systems The maintenance process includes other enabling systems that need to be considered that can be different from other processes. The following amplifies on these:

- *Operational environment*—The circumstances surrounding and potentially affecting something that is operating. For example, electronic or mechanical equipment may be affected by high temperatures, vibration, dust, and other parameters that constitute the operating environment.
- *Supply support/PHS&T*—Consists of all actions, necessary to determine requirements and to acquire, catalog, receive, store, transfer, package, transport, issue, and dispose of spares, repair parts, and supplies needed to sustain required level of operations (e.g., system availability). The reliability of the system is impacted when a maintenance action is delayed by the supply system.
- *Training systems*—Provides maintainers with knowledge and skills required for proper system maintenance.
- *Technical data*—Procedures, guidelines, and checklists needed for proper maintenance of the system, including preventive actions (clean and adjust), analysis and diagnostics, fault isolation, fault localization, parts lists, corrective maintenance (remove and replace), calibration, modification and upgrade instructions, postmaintenance validation, etc. For systems with condition-based maintenance (CBM) capabilities, the technical documentation will include information on how those capabilities are used to support maintenance.
- *Facilities and infrastructure*—Facilities (e.g., buildings, warehouses, hangars, waterways) and infrastructure (e.g., IT services, fuel, water, electrical service, machine shops, dry docks, test ranges) required for system maintenance.
- *Tools and support equipment*—Common and special purpose tools (e.g., hand tools, meters) and support equipment (e.g., test sets, cranes) used to support system maintenance.
- *Design interface/sustaining engineering*—Design interface attempts to "design in" supportability features of the system. Supportability considerations minimize the logistics footprint, maximize reliability, ensure effective maintainability, and address the long-term issues related to obsolescence management, technology refreshment, modifications and upgrades, and overall usage under all operating conditions. Sustaining engineering ensures the continued operation and maintenance of a system with managed (i.e., known) risk. Activities include collection and analysis of use and maintenance data; analysis of safety hazards, failure causes and effects, reliability and maintainability trends, and operational usage profile changes; root cause analysis of in-service problems (including operational hazards, problem reports, parts obsolescence, corrosion effects, and reliability degradation); development of required design changes to resolve operational issues; and other activities necessary to ensure cost-effective support to achieve required levels of readiness and performance over the system's life cycle.
- *Maintenance planning and management*—The focus of the maintenance planning process is to define accessibility, diagnostics, repair, and sparing requirements; identify factors that impact the system's designed utilization rates (e.g., maintenance man-hours per maintenance action, maintenance ratios); identify life cycle supportability design, installation, maintenance, and operating constraints and guidelines; and provide Level of Repair Analysis (LORA).

4.13.2.3 Maintenance Techniques The following are some maintenance techniques that can be used for the SOI. This is by no means an exhaustive list:

- *Condition-Based Maintenance (CBM)* is a strategy to improve system reliability (by reducing the amount of time the system is unavailable while conducting routine or corrective maintenance). Well-designed systems include the cost-effective use of sensors (e.g., airflow, vibration,

thermal, viscosity) and integrated, analysis-based decision support capabilities to determine/predict the need for maintenance actions required to prevent a failure. System technical documents will include a description of any embedded condition based maintenance capabilities and how resultant maintenance actions will be performed.

- *Reliability-centered maintenance (RCM)* is a cost-effective maintenance strategy to address dominant causes of equipment failures [supported by failure modes, effects, and criticality analysis (FMECA) and fault tree analysis (FTA)]. It provides a systematic approach to defining a routine maintenance program composed of cost-effective tasks that preserve important functions. SAE International (SAE) JA1011:2009 provides detailed information on the RCM process. RCM is a strategy to improve system reliability (by reducing the amount of time the system is unavailable while conducting routine or preventive maintenance).
- *Performance-based life cycle product support* is a strategy for cost-effective system support. Rather than contracting for maintenance (goods and services) needed to sustain operations, the customer and the service provider(s) agree on the delivery of performance outcomes (defined by performance metric(s) for a system or product). When properly implemented, the provider is incentivized (increased profits or contract extensions) to develop sustainment strategies (e.g., system reliability investment, inventory management practices, logistics arrangements) necessary to meet the required performance outcomes at reduced costs. PBL approaches usually result in sustained performance outcomes at a lower cost than those achieved under a non-PBL or transactional product support arrangements for system material and maintenance.

4.14 DISPOSAL PROCESS

4.14.1 Overview

4.14.1.1 Purpose As stated in ISO/IEC/IEEE 15288,

> [6.4.14.1] The purpose of the Disposal process is to end the existence of a system element or system for a specified intended use, appropriately handle replaced or retired elements, and to properly attend to identified critical disposal needs.

The disposal process is conducted in accordance with applicable guidance, policy, regulations, and statutes throughout the system life cycle.

4.14.1.2 Description Disposal is a life cycle support process because concurrent consideration of disposal during the development stage generates requirements and constraints that must be balanced with defined stakeholders' requirements and other design considerations. Further, environmental concerns drive the designer to consider reclaiming the materials or recycling them into new systems. This process can be applied for the incremental disposal requirements at any point in the life cycle, for example, prototypes that are not to be reused or evolved, waste materials during manufacturing, or parts that are replaced during maintenance. Figure 4.22 is the IPO diagram for the disposal process.

4.14.1.3 Inputs/Outputs Inputs and outputs for the disposal process are listed in Figure 4.22. Descriptions of each input and output are provided in Appendix E.

4.14.1.4 Process Activities The disposal process includes any steps necessary to return the environment to an acceptable condition; handle all system elements and waste products in an environmentally sound manner in accordance with applicable legislation, organizational constraints, and stakeholder agreements; and document and retain records of disposal activities, as required for monitoring by external oversight or regulatory agencies. In general, disposal process includes the following activities:

- *Prepare for disposal.*
  - Review the retirement concept (may be called concept of disposal), including any hazardous materials and other environmental impacts to be encountered during disposal.
  - Plan for disposal, including the development of the strategy.
  - Impose associated constraints on the system requirements.
  - Ensure that the necessary enabling systems, products, or services required for disposal are

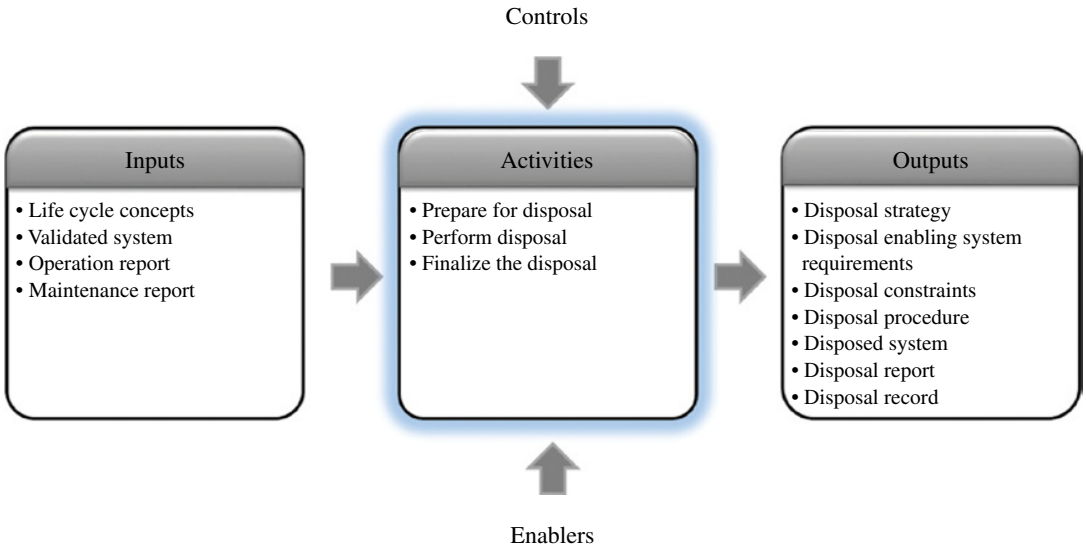


FIGURE 4.22 IPO diagram for the disposal process. INCOSE SEH original figure created by Shortell and Walden. Usage per the INCOSE Notices page. All other rights reserved.

available, when needed. The planning includes the identification of requirements and interfaces for the enablers. The acquisition of the enablers can be done through various ways such as rental, procurement, development, reuse, and subcontracting. An enabler may be a complete enabling system developed as a separate project from the project of the SOI.

- Identify elements that can be reused and that cannot be reused. Methods need to be implemented to prevent hazardous materials from reuse in the supply chain.
- Specify containment facilities, storage locations, inspection criteria, and storage periods, if the system is to be stored.

- *Perform disposal.*
  - Decommission the system elements to be terminated.
  - Disassemble the elements for ease of handling. Include identification and processing of reusable elements.
  - Extract all elements and waste materials that are no longer needed—this includes removing materials from storage sites, consigning the elements and waste products for destruction or permanent storage, and ensuring that the waste products cannot get back into the supply chain.
  - Dispose of deactivated system element per disposal procedure.
  - Remove affected staff and capture the tacit knowledge for future needs.
- *Finalize the disposal.*
  - Confirm no adverse effects from the disposal activities and return the environment to its original state.
  - Maintain documentation of all disposal activities and residual hazards.

Common approaches and tips:

- The project team conducts analyses to develop solutions for ultimate disposition of the system, constituent elements, and waste products based on evaluation of alternative disposal methods available. Methods addressed should include storing, dismantling, reusing, recycling, reprocessing, and destroying end products, enabling systems, system elements, and materials.
- Disposal analyses include consideration of costs, disposal sites, environmental impacts, health and

safety issues, responsible agencies, handling and shipping, supporting items, and applicable federal, state, local, and host-nation regulations.

- Disposal analyses support selection of system elements and materials that will be used in the system design and should be readdressed to consider design and project impacts from changing laws and regulations throughout the project life cycle.
- Disposal strategy and design considerations are updated throughout the system life cycle in response to changes in applicable laws, regulations, and policy.
- Consider donating an obsolete system—Many items, both systems and information, of cultural and historical value have been lost to posterity because museums and conservatories were not considered as an option during the retirement stage.
- Concepts such as zero footprint and zero emissions drive current trends toward corporate social responsibility that influence decision making regarding cleaner production and operational environments and eventual disposal of depleted materials and systems.
- The ISO 14000 series includes standards for environmental management systems and life cycle assessment (ISO 14001, 2004).
- Instead of designing cradle-to-grave products, dumped in landfills at the end of their "life," a new concept is transforming industry by creating products for cradle-to-cradle cycles, whose materials are perpetually circulated in closed loops. Maintaining materials in closed loops maximizes material value without damaging ecosystems (McDonough, 2013).

5

TECHNICAL MANAGEMENT PROCESSES

Within the system life cycle, the creation or upgrade of products and services is managed by the conduct of projects. For this reason, it is important to understand the contribution of systems engineering (SE) to the management of the project. Technical management processes are defined in ISO/IEC/IEEE 15288 as follows:

> [6.3] The Technical Management Processes are used to establish and evolve plans, to execute the plans, to assess actual achievement and progress against the plans and to control execution through to fulfillment. Individual Technical Management Processes may be invoked at any time in the life cycle and at any level …

Systems engineers continually interact with project management. Systems engineers and project managers bring unique skills and experiences to the program on which they work. A life cycle from project manager's point of view (project start–project end) is defined differently than from the systems engineer's point of view (product idea–product disposal). But there is a "shared space" where systems engineers and project managers have to collaborate to drive the team's performance and success (Langley et al., 2011).

Technical management processes include project planning, project assessment and control, decision management, risk management, configuration management, information management, measurement, and quality assurance (QA). These processes are found throughout an organization as they are essential to generic management practices and apply both inside and outside the project context. This chapter of the handbook focuses on processes relevant to the technical coordination of a project.

5.1 PROJECT PLANNING PROCESS

5.1.1 Overview

5.1.1.1 Purpose As stated in ISO/IEC/IEEE 15288,

> [6.3.1.1] The purpose of the Project Planning process is to produce and coordinate effective and workable plans.

5.1.1.2 Description Project planning starts with the identification of a new potential project and continues after the authorization and activation of the project until its termination. The project planning process is performed in the context of the organization. The life cycle model

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.

management process (see Section 7.1) establishes and identifies relevant policies and procedures for managing and executing a technical effort; identifying the technical tasks, their interdependencies, risks, and opportunities; and providing estimates of needed resources and budgets. The planning includes the determination of the need for specialized equipment, facilities, and specialists during the project to improve efficiency and effectiveness and decrease cost overruns. This requires coordination across the set of processes. For example, different disciplines work together in the performance of system requirements definition, architecture definition, and design definition processes to evaluate the parameters such as manufacturability, testability, operability, maintainability, and sustainability against product performance. Project tasking may be concurrent to achieve the best results.

Project planning establishes the direction and infrastructure necessary to enable the assessment and control of the project progress and identifies the details of the work and the right set of personnel, skills, and facilities with a schedule for needed resources from within and outside the organization. Figure 5.1 is the IPO diagram for the project planning process.

5.1.1.3 Inputs/Outputs Inputs and outputs for the project planning process are listed in Figure 5.1. Descriptions of each input and output are provided in Appendix E.

5.1.1.4 Process Activities The project planning process includes the following activities:

- *Define the project.*
  - Analyze the project proposal and related agreements to define the project objectives, scope, and constraints.
  - Establish tailoring of organization procedures and practices to carry out planned effort. Chapter 8 contains a detailed discussion on tailoring.
  - Establish a work breakdown structure (WBS) based on the evolving system architecture.
  - Define and maintain a life cycle model that is tailored from the defined life cycle models of the

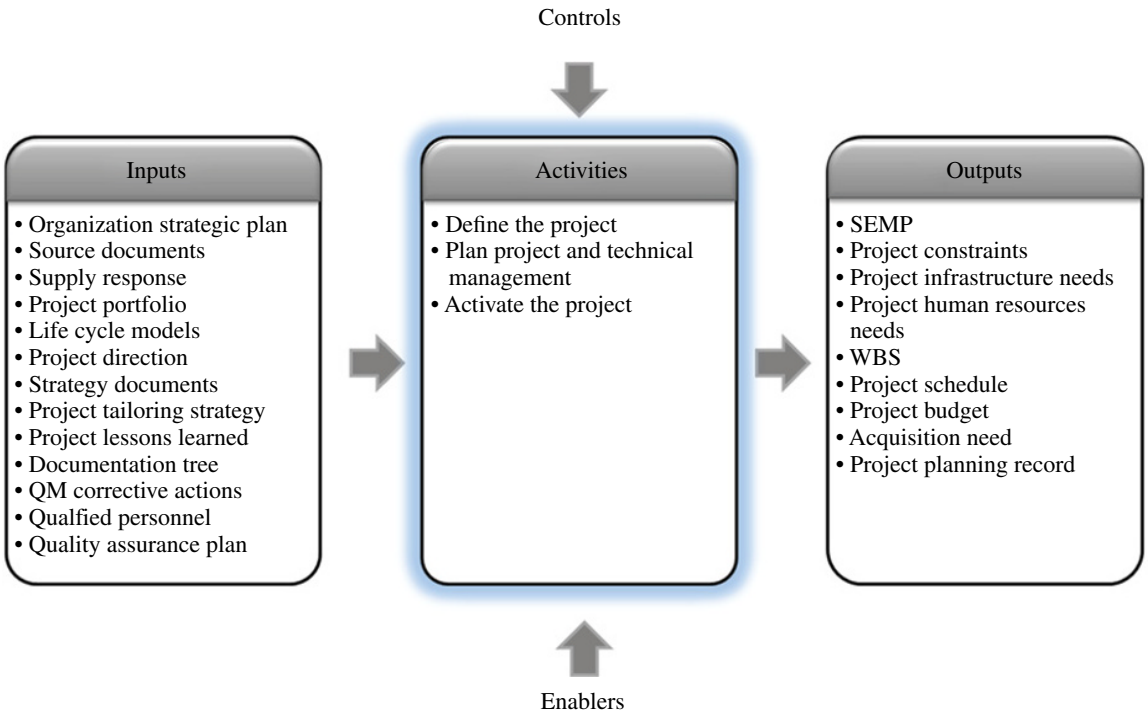


FIGURE 5.1 IPO diagram for the project planning process. INCOSE SEH original figure created by Shortell and Walden. Usage per the INCOSE Notices page. All other rights reserved.

organization. This includes the identification of major milestones, decision gates, and project reviews.

- *Plan project and technical management.*
  - Establish the roles and responsibilities for project authority.
  - Define top-level work packages for each task and activity identified. Each work package should be tied to required resources including procurement strategies.
  - Develop a project schedule based on objectives and work estimates.
  - Define the infrastructure and services required.
  - Define costs and estimate project budget.
  - Plan the acquisition of materials, goods, and enabling system services.
  - Prepare a systems engineering management plan (SEMP) or systems engineering plan (SEP), including the reviews that will be performed across the life cycle.
  - Generate or tailor quality management, configuration management, risk management, information management, and measurement plans to meet the needs of the project (may be the SEMP or SEP for smaller projects).
  - Establish the criteria to be used for major milestones, decision gates, and internal reviews.
- *Activate the project.*

Common approaches and tips:

- The SEMP is an important outcome of planning that identifies activities, key events, work packages, and resources. It references other planning documents that are tailored for use on the project as discussed in later sections of this chapter.
- IPDTs are frequently used to break down communications and knowledge stovepipes within organizations (Martin, 1996).
- The creation of the WBS is an activity where SE and project management intersect (Forsberg et al., 2005). Sometimes, software engineers presume that these tools apply only to large hardware projects, and they avoid the project management tools needed for success. Dr. Richard Fairley has documented an excellent approach to dispel this notion (Fairley, 2009).
- Skipping or taking shortcuts in the planning process reduces the effectiveness of other technical management processes.
- Agile project management methods also include planning—the cycles may be shorter and more frequent, but planning is an essential process.
- Defining project objectives and the criteria for success are critical to successful projects. The project value to the stakeholders should be clearly defined to guide project decision making. The project value should be expressed in technical performance measures (TPMs) (Roedler and Jones, 2006).
- Incorporate risk assessment early in the planning process to identify areas that need special attention or contingencies. Always attend to the technical risks (PMI, 2013).
- The Project Management Institute is a source of guidelines for project planning.
- The ISO/IEC/IEEE 16326 standard for project management also provides additional guidance on this subject (ISO/IEC/IEEE 16326, 2009).

5.1.2 Elaboration

5.1.2.1 Project Planning Concepts Project planning estimates the project budget and schedule against which project progress will be assessed and controlled. Systems engineers and project managers must collaborate in project planning. Systems engineers perform technical management activities consistent with project objectives. Technical management activities include planning, scheduling, reviewing, and auditing the SE process as defined in the SEMP and the SE Master Schedule (SEMS).

5.1.2.2 SEMP The SEMP is the top-level plan for managing the SE effort. It defines how the project will be organized, structured, and conducted and how the total engineering process will be controlled to provide a product that satisfies stakeholder requirements. A well-written SEMP provides guidance to a project and helps the organization avoid unnecessary discussions about how to perform SE. Organizations generally maintain a template of the SEMP suitable for tailoring and reuse. Effective project control requires that there be a SEMP, which the systems engineer keeps current and uses on a daily basis to manage the team's actions.

The SEMS is an essential part of the SEMP and a tool for project control because it identifies the critical path of technical activities in the project. Verification activities may also receive special attention in the SEMS. In addition, the schedule of tasks and dependencies helps justify requests for personnel and resources needed throughout the development life cycle.

The SEMP and SEMS are supported by a project or contract WBS that defines a project task hierarchy. Work authorization is the process by which the project is baselined and financially controlled. A description of the organization procedures for starting work on a part of the WBS may be defined in the SEMP.

TPMs (see Section 5.7.2.8) are a tool used for project control, and the extent to which TPMs will be employed should be defined in the SEMP (Roedler and Jones, 2006).

A SEMP should be prepared early in the project, submitted to the customer (or to management for in-house projects), and used in technical management for the concept and development stages of the project or the equivalent in commercial practice. The creation of the SEMP involves defining the SE processes, functional analysis approaches, what trade studies will be included in the project, schedule, and organizational roles and responsibilities, to name a few of the more important aspects of the plan. The SEMP also reports the results of the effort undertaken to form a project team and outlines the major deliverables of the project, including a decision database, specifications, and baselines. Participants in the creation of the SEMP should include senior systems engineers, representative subject matter experts, project management, and often the customer.

The format of the SEMP can be tailored to fit project, customer, or company standards. To maximize reuse of the SEMP for multiple projects, project-specific appendices are often used to capture detailed and dynamic information, such as the decision database, a schedule of milestones and decision gate reviews, and the methodology to be used in resolving problems uncovered in reviews.

The process inputs portion of the SEMP identifies the applicable source documents (e.g., customer specifications from the RFP, SOW, standards, etc.) to be used in the performance of the project and in developing associated deliverables (e.g., the system specification and technical requirements document). It also may include previously developed specifications for similar systems and company procedures affecting performance specifications. A technical objectives document should be developed and may be one of the source documents for the decision database. The document may also be part of the ConOps for the system (see Section 4.2.2.4).

The SEMP should include information about the project organization, technical management, and technical activities. A complete outline of a SEMP is available in ISO/IEC/IEEE 24748-4 (2014), which is aligned with ISO/IEC/IEEE 15288 and this handbook. As a high-level overview, the SEMP should include the following:

- Organization of the project and how SE interfaces with the other parts of the organization
- Responsibilities and authority of the key positions
- Clear system boundaries and scope of the project
- Project assumptions and constraints
- Key technical objectives
- Infrastructure support and resource management (i.e., facilities, tools, IT, personnel, etc.)
- Approach and methods used for planning and executing the technical processes described in this handbook (see Chapter 4)
- Approach and methods used for planning and executing the technical management processes described in this handbook (see Chapter 5)
- Approach and methods used for planning and executing the applicable specialty engineering processes described in this handbook (see Chapter 10)

The SEMP may sometimes address affordability/cost-effectiveness/life cycle cost (LCC) analysis (see Section 10.1) and value engineering (see Section 10.14) practices to provide insight into system/cost-effectiveness. For example, can the project be engineered to have significantly more value with minimal additional cost? If so, does the customer have the resources for even the modest cost increase for the improvement? Can the solutions be achieved within their budgets and schedules? This assures the customer that obvious cost-effective alternatives have been considered (ISO/IEC/IEEE 24748-4, 2014).

Technical reviews are essential to ensure that the system will meet the requirements and that the requirements are understood by the development team. Formal reviews are essential to determine readiness to proceed

to the next stage of the system life cycle. The number and frequency of these reviews and their associated decision gates must be tailored for specific projects. The SEMP should list what technical reviews will be conducted and the methodology to be used in solving problems uncovered during those reviews.

The system life cycles shown in Figure 3.3 illustrate the appropriate time for reviews and decision gates. They may not fit all projects, and some projects may need more or fewer reviews. Additionally, formal, documented decision gates, with the customer in attendance, can impose significant cost on the project. Projects should plan to use more frequent, informal, in-house reviews to resolve most issues and strive to exit decision gates with no major customer-imposed action items.

Transitioning critical technologies should be done as a part of risk management (see Section 5.4) but is called out separately here for special emphasis. Critical technologies should be identified, and the steps outlined for risk management should be followed. Additionally, completed and planned risk management work should be explicitly referenced in the SEMP.

The system being proposed may be complex enough that the customer will require training to use it. During the project, it may be necessary to train those who will develop, manufacture, verify, deploy, operate, support, conduct training, or dispose of the system. A plan for this training is required in the SEMP and should include the following:

1. Analysis of performance
2. Behavior deficiencies or shortfalls
3. Required training to remedy deficiencies or shortfalls
4. Schedules to achieve required proficiencies

Verification is usually planned using a verification matrix that lists all the requirements and anticipated verification methods. The possible methods of verification include inspection, analysis, demonstration, and test. The SEMP should state, at least in preliminary general terms, that a verification plan will be written to define the items to be verified and which methods will be used to verify performance. The plan should also define who is to perform and witness the verification of each item. This should also relate to the SEMS for time phasing of the verification process. Detailed procedures are usually not written for inspection, analysis, and demonstration methods. Simulations may be used for testing, when quantifiable results are needed, or for demonstration, when qualitative results are satisfactory.

A well-written SEMP provides guidance to a project and helps the organization avoid unnecessary discussions about how to perform SE. In addition, a schedule and organization are defined that help the project procure the personnel needed throughout the development life cycle and assess progress.

5.2 PROJECT ASSESSMENT AND CONTROL PROCESS

5.2.1 Overview

5.2.1.1 Purpose As stated in ISO/IEC/IEEE 15288,

> [6.3.2.1] The purpose of the Project Assessment and Control process is to assess if the plans are aligned and feasible; determine the status of the project, technical and process performance; and direct execution to ensure that the performance is according to plans and schedules, within projected budgets, to satisfy technical objectives.

Assessments are scheduled periodically and for all milestones and decision gates. The intention is to maintain good communications within the project team and with the stakeholders, especially when deviations are encountered.

The process uses these assessments to direct the efforts of the project, including redirecting the project when the project does not reflect the anticipated maturity.

5.2.1.2 Description The project planning process (see Section 5.1) identified details of the work effort and expected results. The project assessment and control process collects data to evaluate the adequacy of the project infrastructure, the availability of necessary resources, and the compliance with project performance measures. Assessments also monitor the technical progress of the project and may identify new risks or areas that require additional investigation. A discussion of the creation and assessment of TPMs is found in Section 5.7.2.8.

The rigor of the project assessment and control process is directly dependent on the complexity of the system of interest (SOI). Project control involves

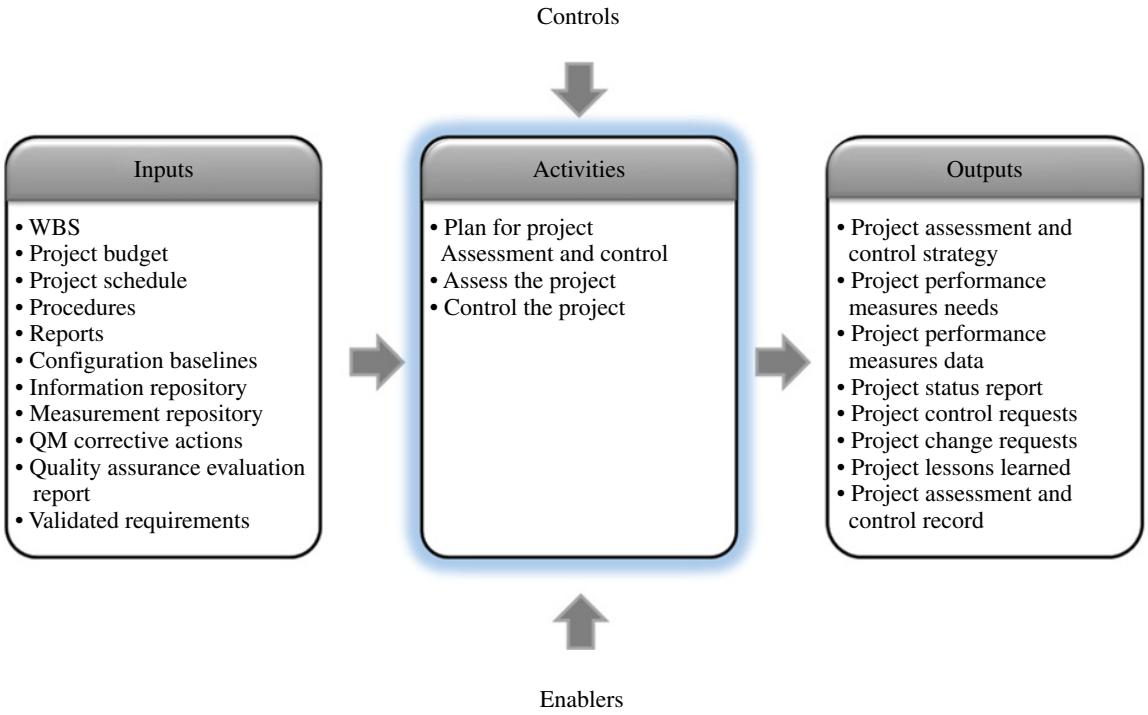


FIGURE 5.2 IPO diagram for the project assessment and control process. INCOSE SEH original figure created by Shortell and Walden. Usage per the INCOSE Notices page. All other rights reserved.

both corrective and preventive actions taken to ensure that the project is performing according to plans and schedules and within projected budgets. The project assessment and control process may trigger activities within the other process areas in this chapter. Figure 5.2 is the IPO diagram for the project assessment and control process.

5.2.1.3 Inputs/Outputs Inputs and outputs for the project assessment and control process are listed in Figure 5.2. Descriptions of each input and output are provided in Appendix E.

5.2.1.4 Process Activities The project assessment and control process includes the following activities:

- *Plan for project assessment and control.*
  - *Develop a strategy for project assessment and control for the system.*
- *Assess the project.*
  - Review measurement results associated with the project.
  - Determine actual and projected cost against budget, actual and projected time against schedule, and deviations in project quality.
  - Evaluate the effectiveness and efficiency of the performance of project activities.
  - Evaluate the adequacy and the availability of the project infrastructure and resources.
  - Evaluate project progress against established criteria and milestones.
  - Conduct required reviews, audits, and inspections to determine readiness to proceed to the next milestone.
  - Monitor critical tasks and new technologies (see Section 5.4).
  - Analyze assessment results.
  - Make recommendations for adjustments to project plans—these are input to the project control process and other decision-making processes.

  - Communicate status as designated in agreements, policies, and procedures.
- *Control the project*.
  - Initiate preventive actions when assessments indicate a trend toward deviation.
  - Initiate problem resolution when assessments indicate nonconformance with performance success criteria.
  - Initiate corrective actions when assessments indicate deviation from approved plans.
  - Establish work items and changes to schedule to reflect the actions taken.
  - Negotiate with suppliers for any goods or services acquired from outside the organization.
  - Make the decision to proceed, or not to proceed, when assessments support a decision gate or milestone event.

Common approaches and tips:

- One way for project management to remain updated on project status is to conduct regular team meetings. Short stand-up meetings on a daily or weekly schedule are effective for smaller groups.
- Prevailing wisdom suggests that "what gets measured gets done," but projects should avoid the collection of measures that are not used in decision making.
- The Project Management Institute provides industry-wide guidelines for project assessment, including Earned Value Management techniques.
- Project teams need to identify critical areas and control them through monitoring, risk management, or configuration management.
- An effective feedback control process is an essential element to enable the improvement of project performance.
- Agile project management techniques schedule frequent assessments and make project control adjustments on tighter feedback cycles than other plan-driven development models.
- Tailoring of organization processes and procedures (see Chapter 8) should not jeopardize any certifications. Processes must be established with effective review, assessment, audit, and upgrade.

5.3 DECISION MANAGEMENT PROCESS

5.3.1 Overview

5.3.1.1 Purpose As defined by ISO/IEC/IEEE 15288,

> [6.3.3.1] The purpose of the Decision Management process is to provide a structured, analytical framework for objectively identifying, characterizing and evaluating a set of alternatives for a decision at any point in the life cycle and select the most beneficial course of action.

Table 5.1 provides a partial list of decision situations (opportunities) that are commonly encountered throughout a system's life cycle. Buede (2009) provides a much larger list.

Consider the number of decisions involved in crafting a technology development strategy, generating an initial capability document, selecting a system architecture, converging on a detailed design, constructing test and evaluation plans, making production make-or-buy decisions, creating production ramp-up plans, crafting maintenance plans, and defining disposal approaches. New product developments entail an array of interrelated decisions that require the holistic perspective of the SE discipline. In fact, it can be argued that all SE activities

TABLE 5.1 Partial list of decision situations (opportunities) throughout the life cycle

Life cycle stage	Decision situation (opportunity)
Concept	Assess technology opportunity/initial business case
	Craft a technology development strategy
	Inform, generate, and refine an initial capability document
	Inform, generate, and refine a capability development document
	Conduct analysis of alternatives supporting program initiation decision
	Select system architecture
Development	Select system element
	Select lower-level elements
	Select test and evaluation methods
Production	Perform make-or-buy decision
	Select production process and location
Utilization, support	Select maintenance approach
Retirement	Select disposal approach

should be conducted within the context of supporting good decision making. If an SE activity cannot point to at least one of the many decisions embedded in a system life cycle, one must wonder why the activity is being conducted at all. Positioning decision management as a critical SE activity will ensure the efforts are rightfully interpreted as relevant and meaningful and thus maximize the discipline's value proposition to new product developers and their leadership.

5.3.1.2 Description A formal decision management process is the transformation of a broadly stated decision situation into a recommended course of action and associated implementation plan. The process can be executed by a resourced decision team that consists of a decision maker with full responsibility, authority, and accountability for the decision at hand, a decision analyst with a suite of reasoning tools, subject matter experts with performance models, and a representative set of end users and other stakeholders (Parnell et al., 2013). The decision process is executed within the policy and guidelines established by the sponsoring agent. The decision process realizes this transformation through a structured process. It must be recognized that this process, as with most any SE process, contains subjective elements, and two equally qualified teams can come to different conclusions and recommendations. A well-structured trade study process, however, will be able to capture and communicate the impact that different value judgments have on the overall decision and even facilitate the search for alternatives that remain attractive across a wide range of value schemes. Figure 5.3 is the IPO diagram for the decision management process.

5.3.1.3 Inputs/Outputs Inputs and outputs for the decision management process are listed in Figure 5.3. Descriptions of each input and output are provided in Appendix E.

5.3.1.4 Process Activities The decision management process includes the following activities:

- *Prepare for decisions.*
  - Define the decision management strategy for the system.
  - Establish and challenge the decision statement, and clarify the decision to be made. This is one of the most important steps since an incomplete or incorrect decision statement will inappropriately constrain the options considered or even lead the team down the wrong path. Consider the differences that would result from the following decision statements:
    - What car should I buy?
    - What vehicle should I buy?
    - What vehicle should I acquire (buy or lease)?
    - Do I need to travel?
- *Analyze the decision information.*
  - Frame, tailor, and structure decision.
  - Develop objectives and measures.

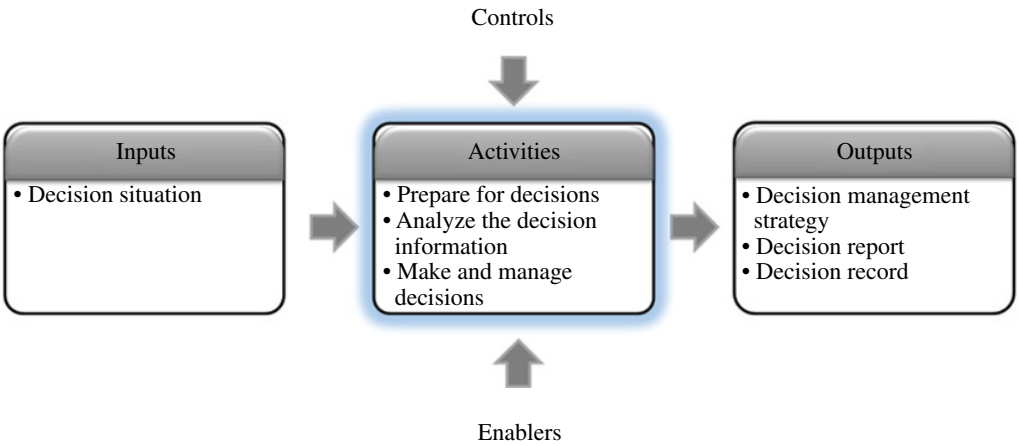


FIGURE 5.3 IPO diagram for the decision management process. INCOSE SEH original figure created by Shortell and Walden. Usage per the INCOSE Notices page. All other rights reserved.

  - Generate creative alternatives.
  - Assess alternatives via deterministic analysis.
  - Synthesize results.
  - Identify uncertainty and conduct probabilistic analysis, as appropriate.
  - Assess impact of uncertainty.
  - Improve alternatives.
  - Communicate trade-offs.
  - Present recommendation and implementation plan.
- *Make and manage decisions*.
  - Record the decision, with the relevant data and supporting documentation.
  - Communicate new directions from the decision.

5.3.2 Elaboration

Systems engineers will likely face many types of decision situations throughout the life cycle of a project. Systems engineers must choose the analytical approach that best fits the frame and structure of the decision problem at hand. For instance, when there are "… clear, important, and discrete events that stand between the implementation of the alternatives and the eventual consequences…" (Edwards et al., 2007), a decision tree is often a well-suited analytical approach, especially when the decision structure has only a few decision nodes and chance nodes. As the number of decision nodes and chance nodes grows, the decision tree quickly becomes unwieldy and loses some of its communicative power. Furthermore, decision trees require end node consequences be expressed in terms of a single number. This is commonly accomplished for decision situations where the value proposition of alternatives can be readily monetized and end state consequences can be expressed in dollars, euros, yen, etc. When the value proposition of alternatives within a decision problem cannot be easily monetized, an objective function can often be formulated to synthesize an alternative's response across multiple, often competing, objectives. This type of problem requires the use of a multiple objective decision analysis (MODA) approach.

The decision management method most commonly employed by systems engineers is the trade study and more often than not employs some form of MODA approach. The aim is to define, measure, and assess shareholder and stakeholder value and then synthesize this information to facilitate the decision maker's search for an alternative that represents the optimally balanced response to often competing objectives. MODA approaches generally differ in the degree to which an alternative's response to objectives (and subobjectives) is aggregated, the mathematics used to aggregate such responses, the techniques used to elicit value statements from stakeholders, the treatment of uncertainty, the robustness of sensitivity analysis, the use of screening techniques, and the versatility and quality of trade space visualization outputs. If time and funding allow, systems engineers may want to conduct trade studies using several techniques, compare and contrast results, and reconcile any differences to ensure findings are robust. Although there are many possible ways to specifically implement MODA, the discussion contained in the balance of this section represents a short summary of best practices regardless of the specific implementation technique employed.

5.3.2.1 Framing and Tailoring Decisions To help ensure the decision makers and stakeholders fully understand the decision context and to enhance the overall traceability of the decision, the decision analyst should capture a description of the system baseline as well as a notion for how the envisioned system will be used along with some indication of system boundaries and anticipated interfaces. Decision context includes such details as the time frame allotted for the decisions, an explicit list of decision makers and stakeholders, a discussion regarding available resources, and expectations regarding the type of action to be taken as a result of the decision at hand as well as decisions anticipated in the future (Edwards et al., 2007).

5.3.2.2 Developing Objectives and Measures Defining how a decision will be made may seem like a straightforward assignment but often becomes an arduous task of seeking clarity amidst a large number of ambiguous stakeholder need statements, engaging in uncomfortable discussions regarding the relative priority of each requirement, and establishing walkaway points and stretch goals. As Keeney puts it,

> Most important decisions involve multiple objectives, and usually with multiple-objective decisions, you can't have it all. You will have to accept less achievement in terms of some objectives in order to achieve more on

other objectives. But how much less would you accept to achieve how much more? (Keeney, 2002)

The first step is to use the information obtained from the stakeholder requirements definition process, requirements analysis process, and requirements management processes to develop objectives and measures.

For each fundamental objective, a measure must be established so that alternatives that more fully satisfy the objective receive a better score on the measure than those alternatives that satisfy the objective to a lesser degree. A measure (also known as attribute, criterion, and metric) must be unambiguous, comprehensive, direct, operational, and understandable (Keeney and Gregory, 2005).

5.3.2.3 Generating Creative Alternatives For many trade studies, the alternatives will be systems composed of many interrelated system elements. It is important to establish a meaningful product structure for the SOI and to apply this product structure consistently throughout the decision analysis effort in order to aid effectiveness and efficiency of communications about alternatives. The product structure should be a useful decomposition of the physical elements of the SOI. Each alternative is composed of specific design choices for each physical element. The ability to quickly communicate the differentiating design features of given alternatives is a core element of the decision-making exercise.

5.3.2.4 Assessing Alternatives via Deterministic Analysis With objectives and measures established and alternatives identified and defined, the decision team should engage subject matter experts, ideally equipped with operational data, test data, models, simulation, and expert knowledge. The decision team can best prepare for subject matter expert engagement by creating structured scoring sheets. Assessments of each concept against each criterion are best captured on separate structured scoring sheets for each alternative/objective combination. Each score sheet contains a summary description of the alternative under examination and a summary of the scoring criteria to which it is being measured.

5.3.2.5 Synthesizing Results At this point in the process, the decision team has generated a large amount of data as summarized in the objective measure consequence table created as the final task of the process step. Now, it is time to explore the data, make sense of the data, and display results in a way that facilitates understanding.

5.3.2.6 Identifying Uncertainty and Conducting Probabilistic Analysis As part of the assessment, it is important for the subject matter expert to explicitly discuss potential uncertainty surrounding the assessed score and variables that could impact one or more scores. One source of uncertainty that is common within SE trade-off analysis exploring various system architectures is that system concepts are generally described as a collection of system element design choices but lack discussion of the component-level design decisions that are normally made downstream during detailed design. Many times, the subject matter expert can assess an upper, nominal, and lower bound measure response by making three separate assessments (i) assuming a low performance, (ii) assuming moderate performance, and (iii) assuming high performance. Once all scores and their associated uncertainty are appropriately captured, Monte Carlo simulations can be executed to identify uncertainty that impacts the decision findings and identify areas of uncertainty that are inconsequential to decision findings.

5.3.2.7 Accessing Impact of Uncertainty: Analyzing Risk and Sensitivity Decision analysis uses many forms of sensitivity analysis including line diagrams, tornado diagrams, waterfall diagrams, and several uncertainty analyses including Monte Carlo simulation, decision trees, and influence diagrams (Parnell et al., 2013).

5.3.2.8 Improving Alternatives One could be tempted to end the decision analysis here, highlight the alternative that has the highest total value, and claim success. Such a premature ending however would not be considered best practice. Mining the data generated for the first set of alternatives will likely reveal opportunities to modify some system element design choices to claim untapped value and reduce risk.

5.3.2.9 Communicating Trade-Offs This is the point in the process where the decision team identifies key observations regarding what stakeholders seem to want and what they must be willing to give up in order to achieve it. It is here where the decision team can highlight the design decisions that are least significant and/or most influential and provide the best stakeholder value. In addition, the important uncertainties and risks should also be identified. Observations regarding combinatorial effects of various design decisions are also important

products of this process step. Finally, competing objectives that are driving the trade should be explicitly highlighted as well.

5.3.2.10 Presenting Recommendations and Implementing Action Plan It is often helpful to describe the recommendation in the form of clearly worded, actionable task list to increase the likelihood of the decision analysis leading to some form of action, thus delivering some tangible value to the sponsor. Reports are important for historical traceability and future decisions. Take the time and effort to create a comprehensive, high-quality report detailing study findings and supporting rationale. Consider static paper reports augmented with dynamic hyperlinked electronic reports.

5.4 RISK MANAGEMENT PROCESS

5.4.1 Overview

5.4.1.1 Purpose As stated in ISO/IEC/IEEE 15288,

> [6.3.4.1] The purpose of the Risk Management process is to identify, analyze, treat and monitor the risks continually.

5.4.1.2 Description Numerous standards, guidelines, and informational publications address the subjects of risk and risk management. In some cases, the application of specific standards may be mandated by industry regulations or customer contractual agreement.

Because there is significant variation, and even contradiction, in the risk and risk management concepts and practices presented in these publications, it is important that process owners, implementers, and users of risk management processes ensure that their understanding of, and approach to, risk management is sufficient, consistent, and appropriate for their specific context, scope, and objectives.

Definitions of Risk According to E. H. Conrow, "Traditionally, risk has been defined as the likelihood of an event occurring coupled with a negative consequence of the event occurring. In other words, a risk is a potential problem—something to be avoided if possible, or its likelihood and/or consequences reduced if not" (2003).

Below are several definitions consistent with this traditional concept of risk:

- ISO/IEC/IEEE 16085:2006, *Systems and software engineering—Life cycle processes—Risk management*, defines risk as the "combination of the probability of an event and its consequence," with the following three notes:
  - The term risk is generally used only when there is at least the possibility of negative consequences.
  - In some situations, risk arises from the possibility of deviation from the expected outcome or event.
  - See ISO/IEC Guide 51 (1999) for issues related to safety.

  Comments: (i) This ISO/IEC/IEEE 16085 definition is taken from ISO Guide 73: 2002, definition 3.1.1 (since replaced with the revised definition in ISO Guide 73:2009). (ii) This is the definition referenced in ISO/IEC/IEEE 15288.

As a corollary to risk, Conrow defines opportunity as "the potential for the realization of wanted, positive consequences of an event" (Conrow, 2003). The idea of considering opportunities and positive outcomes (in addition to negative outcomes) as an integral part of a risk management process has gained favor with some experts and practitioners. New risk and risk management concepts intended to support this broadened scope for risk management are evolving. Notable definitions of risk that reflect this broaden scope are:

- ISO Guide 73:2009, *Risk management—Vocabulary*, defines risk as the "effect of uncertainty on objectives" with the following five notes:
  - An effect is a deviation from the expected—positive and/or negative.
  - Objectives can have different aspects (such as financial, health and safety, and environmental goals) and can apply at different levels (such as strategic, organization-wide, project, product, and process).
  - Risk is often characterized by reference to potential events and consequences or a combination of these.
  - Risk is often expressed in terms of a combination of the consequences of an event (including changes in circumstances) and the associated likelihood of occurrence.

– Uncertainty is the state, even partial, of deficiency of information related to, understanding or knowledge of, an event, its consequence, or likelihood.

According to the *Practice Standard for Project Risk Management* (PMI, 2009), when assessing the importance of a project risk, consider the two key dimensions of risk: uncertainty and the effect on the project's objectives. The uncertainty dimension may be described using the term "probability," and the effect dimension may be called "impact" (though other descriptors are possible, such as "likelihood" and "consequence").

Risk includes both distinct events, which are uncertain but can be clearly described, and general conditions, which are less specific but may give rise to uncertainty. The two types of project risk, those with negative and those with positive effects, are called, respectively, "threats" and "opportunities."

Process Enablers It has been found that an organization's structure and culture can have a significant effect on the performance of the risk management process. ISO 31000, *Risk management—Principles and guidelines* (2009), outlines a model that advocates the establishment of principles for managing risk and a framework for managing risk that work in concert with the process for managing risk.

Risk Management Process Risk management is a disciplined approach to dealing with the uncertainty that is present throughout the entire system life cycle. A primary objective of risk management is to identify and manage (take proactive steps) to handle the uncertainties that threaten or reduce the value provided by a business enterprise or organization. Since risk cannot be reduced to zero, another objective is to achieve a proper balance between risk and opportunity.

This process is used to understand and avoid the potential cost, schedule, and performance (i.e., technical) risks to a system and to take a proactive and structured approach to anticipate negative outcomes and respond to them before they occur. Organizations manage many forms of risk, and the risk associated with system development is managed in a manner that is consistent with the organization strategy.

Every new system or modification of an existing system is based on the pursuit of an opportunity. Risk is always present in the life cycle of systems, and the risk management actions are assessed in terms of the opportunity being pursued.

External risks are often neglected in project management. External risks are risks caused by or originating from the surrounding environment of the project (Fossnes, 2005). Project participants often have no control or influence over external risk factors, but they can learn to observe the external environment and eventually take proactive steps to minimize the impact of external risks on the project. The typical issues are time-dependent processes, rigid sequence of activities, one dominant path for success, and little slack.

Typical strategies for coping with risk include transference, avoidance, acceptance, or taking action to reduce the anticipated negative effects of the situation. Most risk management processes include a prioritization scheme whereby risks with the greatest negative effect and the highest likelihood are treated before those deemed to have lower negative consequences and lower likelihood. The objective of risk management is to balance the allocation of resources such that the minimum amount of resources achieves the greatest risk mitigation (or opportunity realization) benefits. Figure 5.4 is the IPO diagram for the risk management process.

5.4.1.3 Inputs/Outputs Inputs and outputs for the risk management process are listed in Figure 5.4. Descriptions of each input and output are provided in Appendix E.

5.4.1.4 Process Activities The risk management process includes the following activities:

- *Plan risk management*.
  - Define and document the risk strategy.
- *Manage the risk profile*.
  - Establish and maintain a risk profile to include context of the risk and its probability, consequences, risk thresholds, and priority and the risk action requests along with the status of their treatment.
  - Define and document risk thresholds and acceptable and unacceptable risk conditions.
  - Periodically communicate the risks with the appropriate stakeholders.

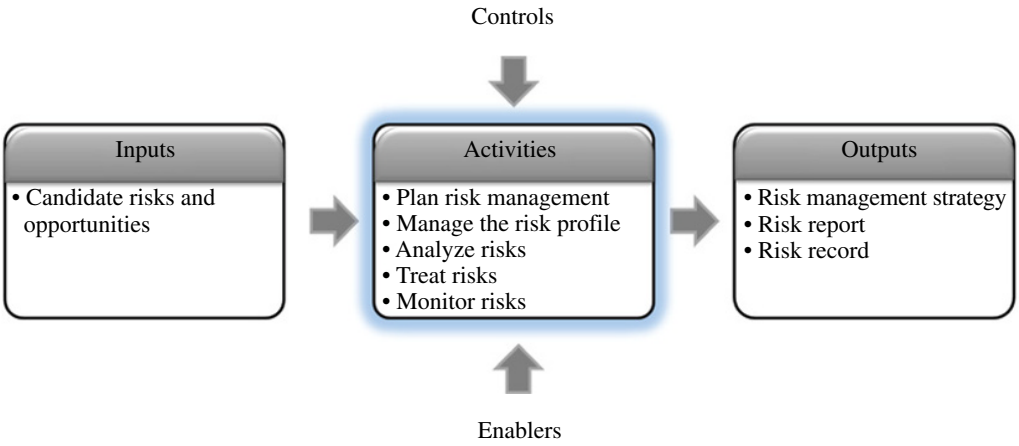


FIGURE 5.4 IPO diagram for the risk management process. INCOSE SEH original figure created by Shortell and Walden. Usage per the INCOSE Notices page. All other rights reserved.

- *Analyze risks*.
  - Define risk situations and identify the risks.
  - Analyze risks for likelihood and consequence to determine the magnitude of the risk and its priority for treatment.
  - Define a treatment scheme and resources for each risk, including identification of a person who will be responsible for continuous assessment of the status of the situation.
- *Treat risks*.
  - Using the criteria for acceptable and unacceptable risk, consider the risk treatment alternative, and generate a plan of action when the risk threshold exceeds acceptable levels.
- *Monitor risks*.
  - Maintain a record of risk items and how they were treated.
  - Maintain transparent risk management communications.

Common approaches and tips:

- In the project planning process (see Section 5.1), a risk management plan (RMP) is tailored to satisfy the individual project procedures for risk management.
- A risk management process establishes documentation, maintained as the risk profile, that includes a description, priority, treatment, responsible person, and status of each risk item.
- One rule of thumb for identifying risks is to pose each risk candidate in an "if <situation>, then <consequence>, for <stakeholder >" format. This form helps to determine the validity of a risk and assess its magnitude or importance. If the statement does not make sense or cannot be put in this format, then the candidate is probably not a valid risk. For example, a statement that describes a situation but not a consequence impacting a specific stakeholder implies that the potential event will not affect the project. Similarly, a statement of potential consequence to a stakeholder without a clear situation or event chain scenario description is probably not adequately understood and requires more analysis.
- All personnel are responsible for using the procedure to identify risks.
- Risks can be identified based on prior experience, brainstorming, lessons learned from similar programs, and checklists.
- Risk identification activity should be applied early and continuously throughout the life cycle of the project.
- Document everything so if unforeseen issues and challenges arise during execution, the project can recreate the environment within which the planning decisions were made and know where to update the information to correct the problem.
- Negative feedback toward personnel who identify a potential problem will discourage the full cooperation of engaged stakeholders and could result in failure to

address serious risk-laden situations. Conduct a transparent risk management process to encourage suppliers and other stakeholders to assist in risk mitigation efforts. Some situations can be difficult to categorize in terms of probability and consequences; involve all relevant stakeholders in this evaluation to capture the maximum variety in viewpoints.

- Many analyses completed throughout the technical processes, such as FMECA, may identify candidate risk elements.
- The measures for risk management vary by organization and by project. As with any measure, use measurement analysis or statistics that help manage the risk.
- Experience has shown that terms such as "positive risk" and concept models that define opportunity as a subset of risks serve only to confuse. Projects that are subject to regulatory standards or customer requirements regarding risk management process and output must use extreme care when integrating opportunity management with risk management. The Project Management Institute is a good source for more information for project risk management (PMI, 2013).

5.4.2 Elaboration

5.4.2.1 Risk Management Concepts Most projects are executed in an environment of uncertainty. Risks (also called threats) are events that if they occur can influence the ability of the project team to achieve project objectives and jeopardize the successful completion of the project (Wideman, 2002). Well-established techniques exist for managing threats, but there is some debate over whether the same techniques are applicable to recognizing opportunities. In an optimal situation, opportunities are maximized at the same time as threats are minimized, resulting in the best chance to meet project objectives (PMI, 2000). The Øresund Bridge case (see Section 3.6.2) illustrates this in that the man-made Peberholm Island was created from the materials dredged from the Strait to meet environmental requirements and is now a sanctuary for a rare species of tern.

The measurement of risk has two components (see Figure 5.5):

- The likelihood that an event will occur
- The undesirable consequence of the event if it does occur

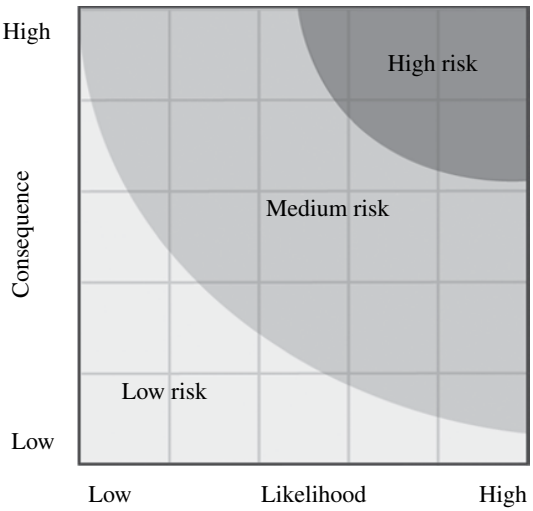


FIGURE 5.5 Level of risk depends upon both likelihood and consequences. INCOSE SEH v1 Figures 4.5 and 4.6. Usage per the INCOSE Notices page. All other rights reserved.

The likelihood that an undesirable event will occur is often expressed as a probability. The consequence of the event is expressed in terms that depend on the nature of the event (e.g., lost investment, inadequate performance, etc.). The combination of low likelihood and low undesirable consequences gives low risk, while high risk is produced by high likelihood and highly undesirable consequences.

By changing the adjective from undesirable to desirable, the noun changes from risk to opportunity, but the diagram remains the same. As suggested by the shading, most projects experience a comparatively small number of high-risk or high-opportunity events.

There is no alternative to the presence of risk in system development. The only way to remove risk is to set technical goals very low, to stretch the schedule, and to supply unlimited funds. None of these events happen in the real world, and no realistic project can be planned without risk. The challenge is to define the system and the project that best meet overall requirements, allow for risk, and achieve the highest chances of project success. Figure 5.6 illustrates the major interactions between the four risk categories: technical, cost, schedule, and programmatic. The arrow labels indicate typical risk relationships, though others certainly are possible.

- *Technical risk*—The possibility that a technical requirement of the system may not be achieved in

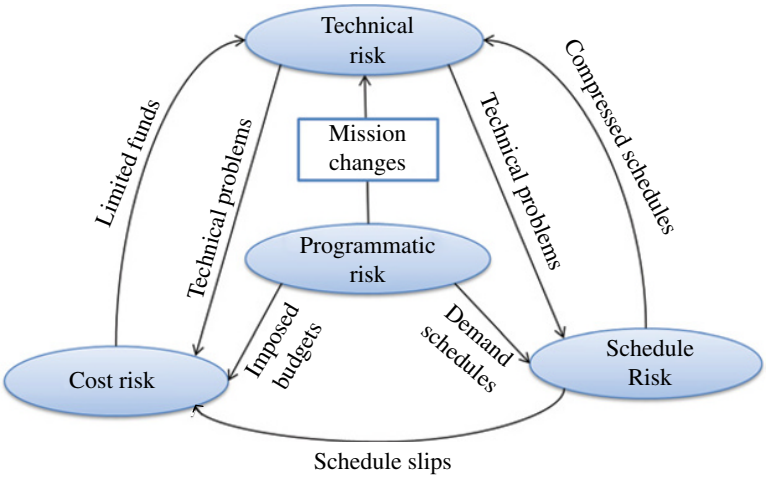


FIGURE 5.6 Typical relationship among the risk categories. INCOSE SEH v1 Figures 4.5, 4.6, and 4.7. Usage per the INCOSE Notices page. All other rights reserved.

the system life cycle. Technical risk exists if the system may fail to achieve performance requirements; to meet operability, producibility, testability, or integration requirements; or to meet environmental protection requirements. A potential failure to meet any requirement that can be expressed in technical terms is a source of technical risk.

- *Cost risk*—The possibility that available budget will be exceeded. Cost risk exists if the project must devote more resources than planned to achieve technical requirements, if the project must add resources to support slipped schedules due to any reason, if changes must be made to the number of items to be produced, or if changes occur in the organization or national economy. Cost risk can be predicted at the total project level or for a system element. The collective effects of element-level cost risks can produce cost risk for the total project.
- *Schedule risk*—The possibility that the project will fail to meet scheduled milestones. Schedule risk exists if there is inadequate allowance for acquisition delays. Schedule risk exists if difficulty is experienced in achieving scheduled technical accomplishments, such as the development of software. Schedule risk can be incurred at the total project level for milestones such as deployment of the first system element. The cascading effects of element-level schedule risks can produce schedule risk for the total project.
- *Programmatic risk*—Produced by events that are beyond the control of the project manager. These events are often produced by decisions made by personnel at higher levels of authority, such as reductions in project priority, delays in receiving authorization to proceed with a project, reduced or delayed funding, changes in organizational or national objectives, etc. Programmatic risk can be a source of risk in any of the other three risk categories.

5.4.2.2 Risk Management Approach Once a risk management strategy and risk profile have been established, the three key risk management process activities are analyze risks, treat risks, and monitor risks.

Analyze Risks Analyzing risks involves identifying risks and evaluating their relative likelihood and consequence. The basis for this evaluation may be qualitative or quantitative; regardless, the objective is to set priorities and focus attention on areas of risk with the greatest consequences to the success of the project. All stakeholders and project personnel should feel welcome to contribute to identifying and analyzing risks.

If a project is unprecedented, brainstorming using strength–weakness–opportunity–threat (SWOT) or Delphi techniques may be appropriate. However, most projects represent a new combination of existing systems or system elements or represent the insertion of incremental advances in technology. This means that key insights can be gained

concerning a current project's risk by examining the successes, failures, problems, and solutions of similar prior projects. The experience and knowledge gained, or lessons learned, can be applied to identify potential risk in a new project and to develop risk-specific management strategies.

The first step is to determine the information needs. This could vary from assessing the risk in development of a custom computer chip to identifying the risks associated with a major system development. Next, systems engineers define the basic characteristics of the new system as a basis for identifying past projects that are similar in technology, function, design, etc. Based on the availability of data, analogous systems or system elements are selected and data gathered. Often, the data collection process and initial assessment lead to a further definition of the system for the purposes of comparison. Comparisons to prior systems may not be exact or the data may need to be adjusted to be used as a basis for estimating the future. The desired output is insight into cost, schedule, and technical risks of a project based on observations of similar past projects.

Uncertainty is characterized by a distribution of outcomes based on the likelihood of occurrence and severity of consequences. As noted previously, risk involves both the likelihood and consequences of the possible outcomes. In its most general form, risk analysis should capture the spectrum of outcomes relative to the desired project technical performance, cost, and schedule requirements. Risk generally needs to be analyzed subjectively because adequate statistical data are rarely available. Expert interviews and models are common techniques for conducting risk analysis.

Risk Perception It is important to recognize that the severity of consequences (or impact) associated with a risk is an attribute of the person or group potentially affected by the risk, rather than an attribute of the "risk event" per se. In other words, the occurrence of a risk event will have different effects on different people depending on (i) their specific situation and perspective at the time of the occurrence and (ii) their unique personal values, perceptions, and sensitivities. It is possible, for example, for one person (or group) to be negatively impacted by an event or situation, while another person (or group) is positively impacted by the same event or situation. This is to be expected in win–lose and competitive event scenarios. Likewise, when two or more people (or groups) are affected in a uniformly negative or positive manner, at least some variation in their assessment of the level of negative or positive impact can be expected. In general, a risk event having significant variation in effect for different individuals or groups should be separated into an appropriate number of uniquely identified individual risk statements and detailed further to include reference to the effected person or group and the specific effects that are unique to them.

To achieve accuracy of risk estimation and overall effectiveness of the risk management effort, it is very important that risk estimates are based on clear, unambiguous risk descriptions and an adequate understanding of the values and situations of those potentially affected by the occurrence of the risk event. When possible, direct communication with the affected individuals and groups is preferred. Variation in perceived risk levels can often be reduced through clarification of risk event scenarios and better definition of risk assessment criteria and rating scales.

ISO Guide 73:2009, *Risk management—Vocabulary*, defines risk perception as a "stakeholder's view on a risk" noting that "risk perception reflects the stakeholder's needs, issues, knowledge, belief and values." A stakeholder is defined as a "person or organization that can affect, be affected by, or perceive themselves to be affected by a decision or activity."

Generally, the stakeholders consulted with to define system needs, expectations, and requirements (see Section 4.2) should also be consulted with to identify and assess risks. It is good practice to establish and maintain traceability between stakeholders and risks, as well as stakeholders and requirements. Risk statements and estimates that do not reference the stakeholder(s) affected by the risk event or situation should be viewed as vague and incomplete or at least potentially inaccurate.

Expert Interviews Efficient acquisition of expert judgments is extremely important to the overall accuracy of the risk management effort. The expert interview technique consists of identifying the appropriate experts, questioning them about the risks in their area of expertise, and quantifying these subjective judgments. One result is the formulation of a range of uncertainty or a probability density (with respect to cost, schedule, or performance) for use in any of several risk analysis tools.

Since expert interviews result in a collection of subjective judgments, the only real "error" can be in the

methodology for collecting the data. If it can be shown that the techniques for collecting the data are not adequate, then the entire risk assessment can become questionable. For this reason, the methodology used to collect the data must be thoroughly documented and defensible. Experience and skill are required to encourage the expert to divulge information in the right format. Typical problems encountered include identification of the wrong expert, obtaining poor quality information, unwillingness of the expert to share information, changing opinions, getting biased viewpoints, obtaining only one perspective, and conflicting judgments. When conducted properly, expert interviews provide reliable qualitative information. However, the transformation of that qualitative information into quantitative distributions or other measures depends on the skill of the analyst.

Risk Assessment Techniques ISO 31010, *Risk management—Risk assessment techniques* (2009), provides detailed descriptions and application guidance for approximately 30 assessment techniques ranging from brainstorming and checklists to Failure Mode and Effects Analysis (FMEA), fault tree analysis (FTA), Monte Carlo simulation, and Bayesian statistics and Bayes nets.

Treat Risks Risk treatment approaches (also referred to as risk handling approaches) need to be established for the moderate and high-risk items identified in the risk analysis effort. These activities are formalized in the RMP. There are four basic approaches to treat risk:

1. Avoid the risk through change of requirements or redesign.
2. Accept the risk and do no more.
3. Control the risk by expending budget and other resources to reduce likelihood and/or consequence.
4. Transfer the risk by agreement with another party that it is in their scope to mitigate. Look for a partner that has experience in the dedicated risk area.

The following steps can be taken to avoid or control unnecessary risks:

- *Requirements scrubbing*—Requirements that significantly complicate the system can be scrutinized to ensure that they deliver value equivalent to their investment. Find alternative solutions that deliver the same or comparable capability.
- *Selection of most promising options*—In most situations, several options are available. A trade study can include project risk as a criterion when selecting the most promising alternative.
- *Staffing and team building*—Projects accomplish work through people. Attention to training, teamwork, and employee morale can help avoid risks introduced by human errors.

For high-risk technical tasks, risk avoidance is insufficient and can be supplemented by the following approaches:

- Early procurement
- Initiation of parallel developments
- Implementation of extensive analysis and testing
- Contingency planning

The high-risk technical tasks generally imply high schedule and cost risks. Cost and schedule are impacted adversely if technical difficulties arise and the tasks are not achieved as planned. Schedule risk is controlled by early procurement of long-lead items and provisions for parallel-path developments. However, these activities also result in increased early costs. Testing and analysis can provide useful data in support of key decision points. Finally, contingency planning involves weighing alternative risk mitigation options.

For each risk that is determined credible after analysis, a Risk Treatment Plan (also referred to as a Risk Mitigation Action Plan) should be created that identifies the risk treatment strategy, the trigger points for action, and any other information to ensuring the treatment is effectively executed. The Risk Treatment Plan can be part of the risk record on the risk profile. For risks that have significant consequences, a contingency plan should be created in case the risk treatment is not successful. It should include the triggers for enacting a contingency plan.

In China, the authorities built the short maglev train line in Shanghai (see Section 3.6.3) as a proof of concept. In spite of the high investment, this represented lower risk to the project than attempting a longer line with an unproven technology. The results collected from this project are inspiring others to consider maglev alternatives for greater distances.

Monitor Risks Project management uses measures to simplify and illuminate the risk management process.

Each risk category has certain indicators that may be used to monitor project status for signs of risk. Tracking the progress of key system technical parameters can be used as an indicator of technical risk.

The typical format in tracking technical performance is a graph of a planned value of a key parameter plotted against calendar time. A second contour showing the actual value achieved is included in the same graph for comparative purposes. Cost and schedule risk are monitored using the products of the cost/schedule control system or some equivalent technique. Normally, cost and schedule variances are used along with a comparison of tasks planned to tasks accomplished. A number of additional references exist on the topic of risk management (AT&T, 1993; Barton et al., 2002; Michel and Galai, 2001; Shaw and Lake, 1993; Wideman, 2004).

Risk management process scope, context, and objectives The risk management process described in this section is generic and may be applied at any stage of the SE life cycle (see Section 3.3), at any level in a system hierarchy (see Section 2.3), or to an SoS (see Section 2.4). In addition, an organization may decide to include opportunity management (i.e., risk and opportunity management) or management of positive consequences (in addition to negative consequences) as part of one or more risk management processes. As a foundation for efficiency and effectiveness, the scope and context of the risk management process should be clearly defined and consistent with requirements and expectations for the process.

Defining the System and Its Boundaries ISO 31000, *Risk management—Principles and guidelines on implementation* (2009), provides guidance and rationale for establishing the external and internal context of a risk management process.

System models (see Section 9.1) describing the system to which the risk management process applies (whether enterprise, product, or service) can facilitate the risk management process by providing about what the system "is" and "does," how it behaves in different scenarios, the location of its boundaries, and the definition of internal and external interfaces. System models can greatly enhance communication and can help ensure the comprehensiveness needed for full risk identification.

The scope of a risk management process also includes a time dimension. It is rare that a single risk management process is used throughout the lifetime of a system for all risks in all life cycle stages. For example, a product development organization might utilize a project risk management process during the development stage, while separate risk management processes performed by different organizations may be used years later for the utilization and support stages. By defining the calendar time and life cycle stage scopes of each risk management process, the likelihood of gaps and overlap is reduced.

Risk Management and the System Life Cycle Once the scope and context of a system have been established from a hierarchical standpoint, it is possible to define and model the system (and its associated risks) in relation to its life cycle. Risks at the early stages of the life cycle (exploratory research and concept definition) are quite different than the risks at the final stage (retirement). It is often necessary to consider risks in other stages while performing activities in the current stage. For example, risks associated with the retirement stage (e.g., human exposure to hazard waste during disposal) should be considered as part of risk management performed to evaluate concept options for disposability.

The performance and output of risk management activities related to safety risks encountered during the development, production, utilization, support, and retirement stages may be governed by regulations and standards (e.g., ISO 14971, *Medical devices—Application of risk management to medical devices*) or by customer contractual agreement. As necessary, specialty engineering activities such as safety analysis (see Section 10.10), usability analysis/human systems integration (Section 10.13), and system security engineering (see Section 10.11) should be utilized and coordinated through a risk management process that complies with regulatory and/or customer requirements.

5.4.2.3 Opportunity Management Concepts SE and project management are all about pursuing an opportunity to solve a problem or fulfill a need. Opportunities enable creativity in resolving concepts, architectures, designs, and strategic and tactical approaches, as well as the many administrative issues within the project. It is the selection and pursuit of these strategic and tactical opportunities that determine just how successful the project and system will be. Of course, opportunities usually carry risks, and each opportunity will have its own

set of risks that must be intelligently judged and properly managed to achieve the full value (Forsberg et al., 2005).

Opportunities represent the potential for improving the value of the project results. The project champions (e.g., the creators, designers, integrators, and implementers) apply their "best-in-class" practices in the pursuit of opportunities. After all, the fun of working on projects is doing something new and innovative. It is these opportunities that create the project's value. *Risks* are defined as chances of injury, damage, or loss. Risks are the chances of not achieving the results as planned. Each of the strategic and tactical opportunities pursued has associated risks that undermine and detract from the opportunity's value. These are the risks that must be managed to enhance the opportunity value and the overall value of the project (see Figure 5.7). Opportunity management and risk management are, therefore, essential to—and performed concurrently with—the planning process but require the application of separate and unique techniques that justify this distinct technical management element.

There are two levels of opportunities and risks. Because a project is the pursuit of an opportunity, the *macro* level is the project opportunity itself. The approach to achieving the macro opportunity and the mitigation of associated project-level risks are structured into the strategy and tactics of the project cycle, the selected decision gates, the teaming arrangements, key personnel selected, and so on. The *element* level encompasses the tactical opportunities and risks within the project that become apparent at lower levels of decomposition and as project life cycle stages are planned and executed. This can include emerging, unproven technology; incremental and evolutionary methods that promise high returns; and the temptation to circumvent proven practices to deliver products better, faster, and cheaper.

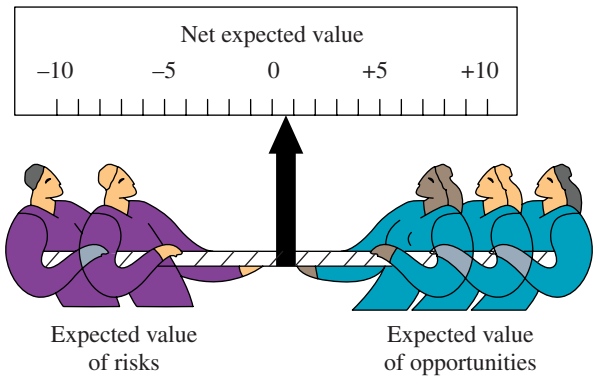


FIGURE 5.7 Intelligent management of risks and opportunities. Derived from (Forsberg et al., 2005) page 224. Reprinted with permission from Kevin Forsberg. All other rights reserved.

Overall project value can be expressed as benefit divided by cost. Opportunities and their risks should be managed jointly to enhance project value. This is based on the relative merits of exploiting each opportunity and mitigating each risk. In the context of the opportunity and the resultant value, we carry a spare tire to mitigate the risk of a flat tire by reducing the probability and impact of having a delayed trip. The high value we place on getting where we want to go far exceeds the small expense of a spare. When deciding to pursue the opportunity of a long automobile trip, we may take extra risk management precautions, such as preventive maintenance and spares for hard-to-find parts.

The assessment of opportunity and risk balance is situational. For example, few today have a car with more than one spare tire (multiple spares were common practice in the early 1900s). However, a few years ago, an individual decided to spend a full month driving across the Australian Outback in late spring. He was looking for solitude in the wilderness (his opportunity). On advice from experienced friends, he took four spare tires and wheels. They also advised him that the risk of mechanical breakdown was very high on a 30-day trip, and the consequence would almost certainly be fatal. The risk of two vehicles breaking down at the same time was acceptably low. So, he adjusted the opportunity for absolute solitude by joining two other adventurers. They set out in three cars. Everyone survived in good health, but only two cars returned, and two of his "spare" tires were shredded by the rough terrain. The "balanced" mitigation approach proved effective.

5.5 CONFIGURATION MANAGEMENT PROCESS

5.5.1 Overview

5.5.1.1 Purpose As stated in ISO/IEC/IEEE 15288,

> [6.3.5.1] The purpose of the Configuration Management process is to manage and control system elements and configurations over the life cycle. CM also manages consistency between a product and its associated configuration definition.

This is accomplished by ensuring the effective management of the evolving configuration of a system, both hardware and software, during its life cycle. Fundamental to this objective is the establishment, control, and maintenance of software and hardware baselines. Baselines are business, budget, functional, performance, and physical reference points for maintaining development and control. These baselines, or reference points, are established by review and acceptance of requirements, design, and product specification documents. The creation of a baseline may coincide with a project milestone or decision gate. As the system matures and moves through the life cycle stages, the software or hardware baseline is maintained under configuration control.

5.5.1.2 Description Evolving the concept definition and system definition is a reality that must be addressed over the life of a system development effort and throughout the utilization and support stages of the system. Configuration management ensures that product functional, performance, and physical characteristics are properly identified, documented, validated, and verified to establish product integrity; that changes to these product characteristics are properly identified, reviewed, approved, documented, and implemented; and that the products produced against a given set of documentation are known. Figure 5.8 is the IPO diagram for the configuration management process.

5.5.1.3 Inputs/Outputs Inputs and outputs for the configuration management process are listed in Figure 5.8. Descriptions of each input and output are provided in Appendix E.

5.5.1.4 Process Activities The configuration management process includes the following activities:

- *Plan configuration management*.
  - Create a configuration management strategy.
  - Implement a configuration control cycle that incorporates evaluation, approval, validation, and verification of engineering change requests (ECRs).
- *Perform configuration identification*.
  - Identify system elements and information items to be maintained under configuration control as configuration items (CIs).
  - Establish unique identifiers for the CIs.
  - Establish baselines for the CIs at appropriate points through the life cycle, including agreement of the baselines by the acquirer and supplier.

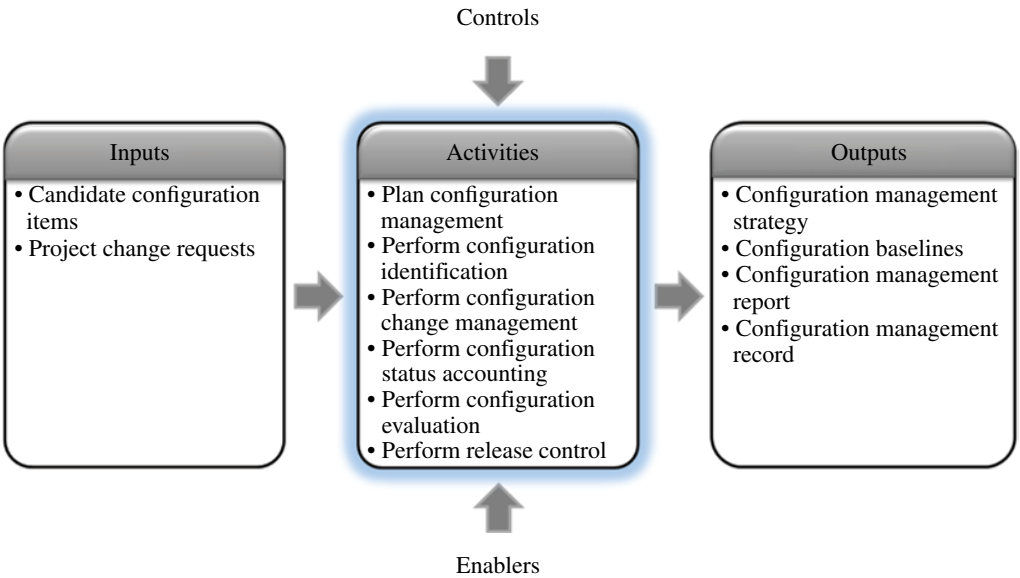


FIGURE 5.8 IPO diagram for the configuration management process. INCOSE SEH original figure created by Shortell and Walden. Usage per the INCOSE Notices page. All other rights reserved.

- *Perform configuration change management*.
  - Control baseline changes throughout the system life cycle. This includes the identification, recording, review, approval, tracking, and processing of requests for change (RFCs) and requests for variance (RFVs) (also known as deviations).
- *Perform configuration status accounting*.
  - Develop and maintain configuration control documentation and configuration management data, and communicate the status of controlled items to the project team.
- *Perform configuration evaluation*.
  - Perform configuration audits and configuration management surveillance reviews associated with milestones and decision gates to validate the baselines.
- *Perform release control*.
  - Perform prioritization, tracking, scheduling, and closing changes, including relevant supporting documentation.

Common approaches and tips:

- In the project planning process (see Section 5.1), a configuration management plan (CMP) is tailored to satisfy the individual project procedures for configuration management.
- The primary output of the configuration management process is the maintenance of the configuration baseline for the system and system elements wherein items are placed under formal control as part of the decision-making process.
- Establish a Configuration Control Board (CCB) with representation from all relevant stakeholders and engineering disciplines participating on the project.
- Begin the configuration management process in the infancy stages of the system and continue through until disposal of the system.
- Configuration management documentation is maintained throughout the life of the system.
- Additional guidance regarding configuration management activities can be found in ISO 10007 (2003), IEEE Std 828 (2012), and ANSI/EIA 649B (2011).
- Domain-specific practices, such as SAE Aerospace Recommended Practice (ARP) 4754A, *Guidelines for Development of Civil Aircraft and Systems* (2010), provide additional application detail for the domain.

5.5.2 Elaboration

5.5.2.1 Configuration Management Concepts The purpose of configuration management is to establish and maintain the control of requirements, documentation, and artifacts produced throughout the system's life cycle and to manage the impact of change on a project. Baselines consolidate the evolving configuration states of system elements to form documented baselines at designated times or under defined circumstances. Baselines form the basis for the next change. Selected baselines typically become formalized between acquirer and supplier, depending on the practices of the industry and the contractual involvement of the acquirer in the configuration change process. There are generally three major types of baselines at the system level: functional baseline, allocated baseline, and product baseline. These may vary by domain or local strategy (ISO/IEC/IEEE 15288:2015).

Change is inevitable, as indicated in Figure 5.9. Systems engineers ensure that the change is necessary and that the most cost-effective recommendation has been proposed.

Initial planning efforts for configuration management are performed at the onset of the project and defined in the CMP, which establishes the scope of items that are covered by the plan; identifies the resources and personnel skill level required; defines the tasks to be performed; describes organizational roles and responsibilities; and identifies configuration management tools and processes, as well as methodologies, standards, and procedures that will be used on the project. Configuration control maintains integrity by facilitating approved changes and preventing the incorporation of unapproved changes into the items under configuration control. Such activities as check-in and checkout of source code, versions of system elements, and deviations of manufactured items are part of configuration management. Independent configuration audits assess the evolution of a product to ensure compliance to specifications, policies, and contractual agreements. Formal audits may be performed in support of decision gate review.

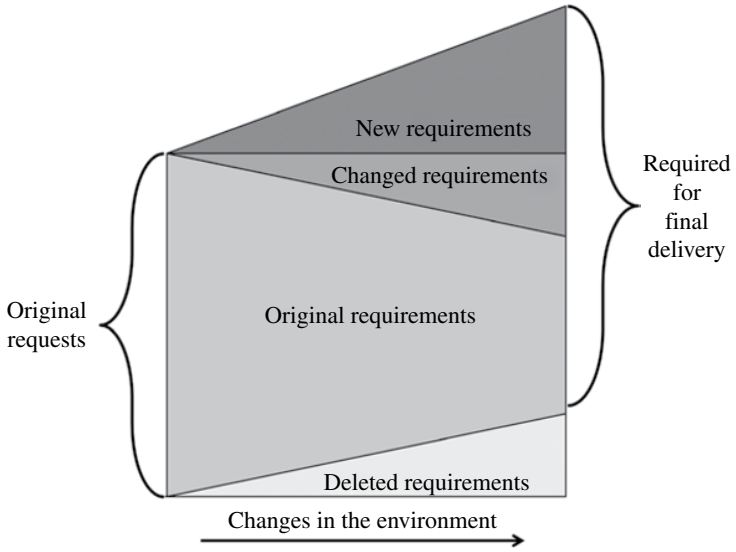


FIGURE 5.9 Requirements changes are inevitable. Derived from (Forsberg et al., 2005) Figure 9.3. Reprinted with permission from Kevin Forsberg. All other rights reserved.

A request to change the current configuration of a system is typically made using an Engineering Change Proposal (ECP). An ECP may originate in a number of ways. The customer may request an ECP to address a change in requirements or a change in scope; an unexpected breakthrough in technology may result in the supplier of a system element proposing an ECP; or a supplier may identify a need for changes in the system under development. Circumstances like these that will potentially change the scope or the requirements are appropriate reasons to propose an ECP and to conduct an analysis to understand the effect of the change on existing plans, costs, and schedules. The ECP must be approved before the change is put into effect. It is never appropriate to propose an ECP to correct cost or schedule variances absent of change in scope. A minor change that falls within the current project scope usually does not require an ECP but should be approved and result in the generation of an engineering notice (EN). It is also important to ask, "What is the impact of not making the change?" especially as the system matures, since changes made later in the life cycle have an increasing risk of hidden impacts, which can adversely affect system cost, schedule, and technical performance.

The most desirable outcomes of an ECP cycle are:

1. System functionality is altered to meet a changing requirement.
2. New technology or a new product extends the capabilities of the system beyond those initially required in ways that the customer desires.
3. The costs of development, or of utilization, or of support are reduced.
4. The reliability and availability of the system are improved.

Outcomes three and four reduce LCC and potentially save more money than is invested to fund the proposed change.

ECPs and ENs help ensure that a system evolves in ways that allow it to continue to satisfy its operational requirements and its objectives and that any modification is known to all relevant personnel. The airplane system illustrated in Figure 2.2 is an example of a product family that depends on accurate identification of system elements and characteristics to support the mix and match consumer market.

5.5.2.2 Configuration Management Approach Configuration management establishes and maintains control over requirements, specifications, configuration definition documentation, and design changes. Configuration identification, configuration control, configuration status accounting, and configuration audits of the functional and physical configuration (i.e., validation

and distribution) are the primary focus of configuration management.

There will always be a need to make changes; however, SE must ensure (i) that the change is necessary and (ii) that the most cost-effective solution has been proposed. Configuration management must, therefore, apply technical and administrative direction, surveillance, and services to do the following:

- Identify and document the functional and physical characteristics of individual CIs such that they are unique and accessible in some form.
- Assign a unique identifier to each version of each CI.
- Establish controls to allow changes in those characteristics.
- Concur in product release and ensure consistent products via the creation of baseline products.
- Record, track, and report change processing and implementation status and collect measures pertaining to change requests or problems with the product baseline.
- Maintain comprehensive traceability of all transactions.

Configuration Identification Configuration identification uniquely identifies the elements within a baseline configuration. This unique identification promotes the ability to create and maintain master inventory lists of baselines. As part of the SE effort, the system is decomposed into CIs, which serve as the critical elements subjected to rigorous formal control. The compilation of all the CIs is called the CI list. This list may reflect items that are developed, vendor produced, or provided by the customer for integration into the final system. These items may be deliverable items under the contract or used to produce the deliverable items.

Change Management Configuration change management, or change control, manages the collection of the items to be baselined and maintains the integrity of the CIs identified by facilitating approved changes (e.g., via ECRs) and preventing the incorporation of unapproved changes into the baseline. Change control should be in effect beginning at project initiation.

Change Classification Effective configuration control requires that the extent of analysis and approval action for a proposed engineering change be in concert with the nature of the change. The problem statement includes a description of the proposed change, the reason for the proposed change, the impacts of the change on cost and schedule, and all affected documentation. Change classification is a primary basis of configuration control. All changes to baselined items are classified as outside of the scope of the requirements or within the scope of the requirements. A change outside the scope of project requirements is a change to a project baseline item that affects the form, fit, specification, function, reliability, or safety. The coordinating review board determines if this proposed change requires a change notice for review and approval.

Changes are sometimes categorized into two main classes: Class I and Class II. A Class I change is a major or significant change that affects cost, schedule, or technical performance. Normally, Class I changes require customer approval prior to being implemented. A Class II change is a minor change that often affects documentation errors or internal design details. Generally, Class II changes do not require customer approval.

CCB An overall CCB is implemented at project initiation to provide a central point to coordinate, review, evaluate, and approve all proposed changes to baselined documentation and configurations, including hardware, software, and firmware. The review board is composed of members from the various disciplines, including SE, software and hardware engineering, project management, product assurance, and configuration management. The chairperson is delegated the necessary authority to act on behalf of the project manager in all matters falling within the review board responsibilities. The configuration management organization is delegated responsibility for maintaining status of all proposed changes. Satellite or subordinate boards may be established for reviewing software or hardware proposed changes below the CI level. If those changes require a higher approval review, they are forwarded to the overall review board for adjudication.

Changes that fall within review board jurisdiction should be evaluated for technical necessity, compliance with project requirements, compatibility with associated documents, and project impact. As changes are written while the hardware and/or software products are in various stages of manufacture or verification, the review board should require specific instructions for identifying

the effectivity or impact of the proposed software or hardware change and disposition of the in-process or completed hardware and/or software product. The types of impacts the review board should assess typically include the following:

- All parts, materials, and processes are specifically approved for use on the project.
- The design depicted can be fabricated using the methods indicated.
- Project quality and reliability assurance requirements are met.
- The design is consistent with interfacing designs.

Methods and Techniques Change control forms provide a standard method of reporting problems and enhancements that lead to changes in formal baselines and internally controlled items. The following forms provide an organized approach to changing hardware, software, or documentation:

- *Problem/change reports*—Can be used for documenting problems and recommending enhancements to hardware/software or its complementary documentation. These forms can be used to identify problems during design, development, integration, verification, and validation.
- *Specification change notice (SCN)*—Used to propose, transmit, and record changes to baselined specifications.
- *ECPs*—Used to propose Class I changes to the customer. These proposals describe the advantages of the proposed change and available alternatives and identify funding needed to proceed.
- *ECRs*—Used to propose Class II changes.
- *Request for deviation/waiver*—Used to request and document temporary deviations from configuration identification requirements when permanent changes to provide conformity to an established baseline are not acceptable.

Configuration Status Accounting Status accounting provides the data on the status of controlled products needed to make decisions regarding system elements throughout the product life cycle. Configuration management maintains a status of approved documentation that identifies and defines the functional and physical characteristics, status of proposed changes, and status of approved changes. This subprocess synthesizes the output of the identification and control subprocesses. All changes authorized by the configuration review boards (both overall and subordinate) culminate in a comprehensive traceability of all transactions. Such activities as check-in and checkout of source code, builds of CIs, deviations of manufactured items, and waiver status are part of the status tracking. By statusing and tracking project changes, a gradual change from the *build-to* to the *as-built* configuration is captured. Suggested measures for consideration include the following:

- Number of changes processed, adopted, rejected, and open
- Status of open change requests
- Classification of change requests summary
- Number of deviations or waivers by CI
- Number of problem reports open, closed, and in-process
- Complexity of problem reports and root cause
- Labor associated with problem resolution and verification stage when problem was identified
- Processing times and effort for deviations, waivers, ECPs, SCNs, ECRs, and problem reports
- Activities causing a significant number of change requests and rate of baseline changes

Configuration Evaluations Configuration evaluations, or audits, are performed independently by configuration management and product assurance to evaluate the evolution of a product and ensure compliance to specifications, policies, and contractual agreements. Formal audits, or functional and physical configuration audits, are performed at the completion of a product development cycle.

The functional configuration audit is intended to validate that the development of a CI has been completed and has achieved the performance and functional characteristics specified in the system specification (functional baseline). The physical configuration audit is a technical review of the CI to verify the as-built maps to the technical documentation (physical baseline). Finally, configuration management performs periodic in-process audits to ensure that the configuration management process is followed.

5.6 INFORMATION MANAGEMENT PROCESS

5.6.1 Overview

5.6.1.1 Purpose As stated in ISO/IEC/IEEE 15288,

> [6.3.6.1] The purpose of the Information Management process is to generate, obtain, confirm, transform, retain, retrieve, disseminate and dispose of information, to designated stakeholders.
>
> Information management plans, executes, and controls the provision of information to designated stakeholders that is unambiguous, complete, verifiable, consistent, modifiable, traceable, and presentable. Information includes technical, project, organizational, agreement, and user information. Information is often derived from data records of the organization, system, process, or project.

Information management needs to provide relevant, timely, complete, valid, and, if required, confidential information to designed parties during and, as appropriate, after the system life cycle. It manages designed information, including technical, project, organizational, agreement, and user information.

Information management ensures that information is properly stored, maintained, secured, and accessible to those who need it, thereby establishing/maintaining integrity of relevant system life cycle artifacts.

5.6.1.2 Description Information exists in many forms, and different types of information have different values within an organization. Information assets, whether tangible or intangible, have become so pervasive in contemporary organizations that they are indispensable. The impact of threats to secure access, confidentiality, integrity, and availability of information can cripple the ability to get work done. As information systems become increasingly interconnected, the opportunities for compromise increase (Brykczynski and Small, 2003). The following are important terms in information management:

- Information is what an organization has compiled or its employees know. It can be stored and communicated, and it might include customer information, proprietary information, and/or protected (e.g., by copyright, trademark, or patent) and unprotected (e.g., business intelligence) intellectual property.
- Information assets are intangible information and any tangible form of its representation, including drawings, memos, email, computer files, and databases.
- Information security generally refers to the confidentiality, integrity, and availability of the information assets (ISO 17799, 2005).
- Information security management includes the controls used to achieve information security and is accomplished by implementing a suitable set of controls, which could be policies, practices, procedures, organizational structures, and software.
- Information Security Management System is the life cycle approach to implementing, maintaining, and improving the interrelated set of policies, controls, and procedures that ensure the security of an organization's information assets in a manner appropriate for its strategic objectives.

Information management provides the basis for the management of and access to information throughout the system life cycle, including after disposal if required. Designated information may include organizational, project, agreement, technical, and user information. The mechanisms for maintaining historical knowledge in the prior processes—decision making, risk, and configuration management—are under the responsibility of information management. Figure 5.10 is the IPO diagram for the information management process.

5.6.1.3 Inputs/Outputs Inputs and outputs for the information management process are listed in Figure 5.10. Descriptions of each input and output are provided in Appendix E.

5.6.1.4 Process Activities The information management process includes the following activities:

- *Prepare for information management.*
  - Support establishing and maintaining a system data dictionary—see project planning outputs.
  - Define system-relevant information, storage requirements, access privileges, and the duration of maintenance.
  - Define formats and media for capture, retention, transmission, and retrieval of information.
  - Identify valid sources of information and designate authorities and responsibilities regarding the origination, generation, capture, archival,

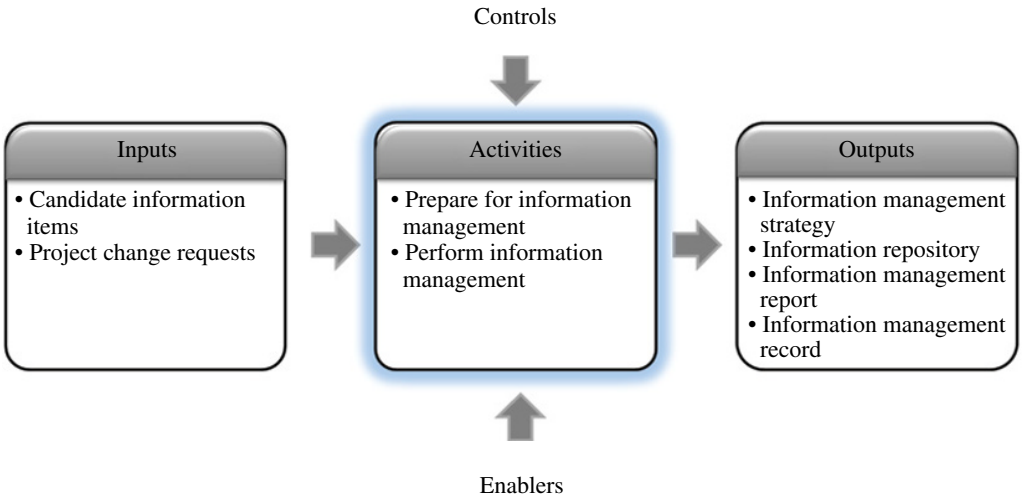


FIGURE 5.10 IPO diagram for the information management process. INCOSE SEH original figure created by Shortell and Walden. Usage per the INCOSE Notices page. All other rights reserved.

and disposal of information in accordance with the configuration management process.

- *Perform information management*.
  - Periodically obtain or transform artifacts of information.
  - Maintain information according to integrity, security, and privacy requirements.
  - Retrieve and distribute information in an appropriate form to designated parties, as required by agreed schedules or defined circumstances.
  - Archive designated information for compliance with legal, audit, knowledge retention, and project closure requirements.
  - Dispose of unwanted, invalid, or unverifiable information according to organizational policy, security, and privacy requirements.

Common approaches and tips:

- In the project planning process (see Section 5.1), an information management plan is tailored to satisfy the individual project procedures for information management. An information management plan identifies the system-relevant information to be collected, retained, secured, and disseminated, with a schedule for disposal.
- Identify information-rich artifacts and store them for later use even if the information is informal, such as a design engineer's notebook.
- Information management delivers value to the organization and the project by using a variety of mechanisms to provide access to the contents of data repositories. Email, web-based access through intranets, and database queries are a few examples.
- ISO 17799, *Code of Practice for Information Security Management*, is an international standard that provides a best practice framework for implementing security controls.
- ISO 10303, *Automation Systems and Integration—Product Data Representation and Exchange*—informally referred to as "Standard for the Exchange of Product Model Data" *(STEP)*. It includes Application Protocol (AP) 239, *Product Life Cycle Support* (PLCS), which addresses information requirements for complex systems.

5.6.2 Elaboration

5.6.2.1 Information Management Concepts The purpose of information management is to maintain an archive of information produced throughout the

system's life cycle. The initial planning efforts for information management are defined in the information management plan, which establishes the scope of project information that is maintained; identifies the resources and personnel skill level required; defines the tasks to be performed; defines the rights, obligations, and commitments of parties for generation, management, and access; and identifies information management tools and processes, as well as methodologies, standards, and procedures that will be used on the project. Typical information includes source documents from stakeholders, contracts, project planning documents, verification documentation, engineering analysis reports, and the files maintained by configuration management. Today, information management is increasingly concerned with the integration of information via databases, such as the decision database, and the ability to access the results from decision gate reviews and other decisions taken on the project; requirements management tools and databases; computer-based training and electronic interactive user manuals; websites; and shared information spaces over the Internet, such as INCOSE Connect. The STEP—ISO 10303 standard provides a neutral computer-interpretable representation of product data throughout the life cycle. ISO 10303-239 (AP 239), *PLCS*, is an international standard that specifies an information model that defines what information can be exchanged and represented to support a product through life (PLCS, 2013). INCOSE is a cosponsor of ISO 10303-233, *Application Protocol: Systems Engineering* (2012). AP 233 is used to exchange data between a SysML™ and other SE application and then to applications in the larger life cycle of systems potentially using related ISO STEP data exchange capabilities.

With effective information management, information is readily accessible to authorized project and organization personnel. Challenges related to maintaining databases, security of data, sharing data across multiple platforms and organizations, and transitioning when technology is updated are all handled by information management. With all the emphasis on knowledge management, organizational learning, and information as competitive advantage, these activities are gaining increased attention.

5.7 MEASUREMENT PROCESS

5.7.1 Overview

5.7.1.1 Purpose As stated in ISO/IEC/IEEE 15288,

> [6.3.7.1] The purpose of the Measurement process is to collect, analyze, and report objective data and information to support effective management and *demonstrate* the quality of the products, services, and processes.

5.7.1.2 Description The SE measurement process will help define the types of information needed to support program management decisions and implement SE best practices to improve performance. The key SE measurement objective is to measure the SE process and work products with respect to program/project and organization needs, including timeliness, meeting performance requirements and quality attributes, product conformance to standards, effective use of resources, and continuous process improvement in reducing cost and cycle time.

The *Practical Software and Systems Measurement Guide* (DoD and US Army, 2003), Section 1.1, states:

Measurement provides objective information to help the project manager:

- *Communicate effectively throughout the project organization*
- *Identify and correct problems early*
- *Make key trade-offs*
- *Track specific project objectives*
- *Defend and justify decisions*

Specific measures are based on information needs and how that information will be used to make decisions and take action. Measurement thus exists as part of a larger management process and includes not just the project manager but also systems engineers, analysts, designers, developers, integrators, logisticians, etc. The decisions to be made motivate the kinds of information to be generated and, therefore, the measurements to be made.

Another concept of successful measurement is the communication of meaningful information to the decision makers. It is important that the people who use

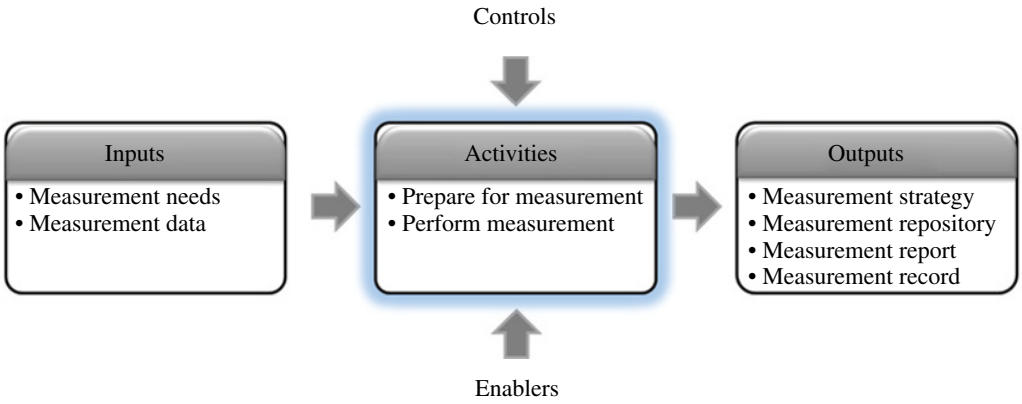


FIGURE 5.11 IPO diagram for the measurement process. INCOSE SEH original figure created by Shortell and Walden. Usage per the INCOSE Notices page. All other rights reserved.

the measurement information understand what is being measured and how it is to be interpreted. Figure 5.11 is the IPO diagram for the measurement process.

5.7.1.3 Inputs/Outputs Inputs and outputs for the measurement process are listed in Figure 5.11. Descriptions of each input and output are provided in Appendix E.

5.7.1.4 Process Activities The measurement process includes the following activities:

- *Prepare for measurement*.
  - Identify the measurement stakeholders and their measurement needs, and develop a strategy to meet them.
  - Identify and select relevant prioritized measures that aid with the management and technical performance of the program.
  - Define the base measures, derived measures, indicators, data collection, measurement frequency, measurement repository, reporting method and frequency, trigger points or thresholds, and review authority.
- *Perform measurement*.
  - Gather, process, store, verify, and analyze the data to obtain measurement results (information products).
  - Document and review the measurement information products with the measurement stakeholders and recommend action, as warranted by the results.

Further detail of the measurement process and activities can be found in the *INCOSE Systems Engineering Measurement Primer*, a useful guide for those new to measures as well as for experienced practitioners.

Common approaches and tips:

- Collection of measures for collection sake is a waste of time and effort.
- Each measure collected should be regularly reviewed by the measurement stakeholders. At a minimum, key measures should be reviewed monthly and weekly for the more mature organizations.
- Some contracts identify measures of effectiveness (MOEs) that must be met. The derived measures of performance (MOPs) and TPMs that provide the necessary insight into meeting the MOEs are automatic measures to be included within the measurement plan. Other measures to consider should provide insight into technical and programmatic execution of the program (Roedler and Jones, 2006).
- The best measures require minimal effort to collect and are repeatable, straightforward to understand, and presented in a format on a regular (weekly or monthly) basis with trend data.
- Many methods are available to present the data to the measurement stakeholders. Line graphs and control charts are two of the more frequently used. Tools are available to help with measurement.
- If a need for corrective action is perceived, further investigation into the measures may be necessary to

identify the root cause of the issue to ensure that corrective actions address the cause instead of a symptom.

- Measurement by itself does not control or improve process performance. Measurement results must be provided to decision makers in a manner that provides the needed insight for the right decisions to be made.

5.7.2 Elaboration

5.7.2.1 Measurement Concepts Measurement concepts have been expanded upon in the previous works that the SE measurement practitioner should reference for further insights:

- *INCOSE Systems Engineering Measurement Primer*, *v2.0* (INCOSE, 2010b)
- *Technical Measurement Guide*, Version 1.0 (INCOSE and PSM, 2005)
- *Systems Engineering Leading Indicators Guide*, Version 2.0 (LAI, INCOSE, PSM, and SEARI, 2010)
- ISO/IEC/IEEE 15939, *Systems and Software Engineering—Measurement Process* (2007)
- PSM Guide V4.0c, *Practical Software and Systems Measurement* (DoD and US Army, 2003)
- *CMMI® (Measurement and Quantitative Management Process Areas)*, Version 1.3 (Software Engineering Institute, 2010)
- *Practical Software Measurement: Objective Information for Decision Makers* (McGarry et al., 2001)
- *System Development Performance Measurement Report* (NDIA, 2011)
- SEBoK Part 3: SE and Management/Systems Engineering Management/Measurement (SEBoK, 2014)

5.7.2.2 Measurement Approach As discussed in the *Systems Engineering Measurement Primer* referenced earlier, measurement may be thought of as a feedback control system. The graphic below shows that measurements may be taken at three primary points in the system. However, the value in measurement comes not from the act of measurement but rather from the eventual analysis of the data and the implementation of action to either correct a variance from a target value or to improve current performance to a more desirable level. The decision as to when to act comes from comparison of the actual measured value against a target value for that measure. The target value and the allowable difference between the target and actual before action is taken are based upon evaluation of risk to the project or product performance meeting their required goals.

Figure 5.12 further indicates that there might be advantage to separating the planning of the measurement process into measures related to the development process (staffing, requirements approved, etc.) and measures related to the product performance (product weight, product speed, etc.). The advantage in this separation would be the recognition that the stakeholders and data collectors for process-related measures will likely be a different population than stakeholders and data collectors for product-related measures.

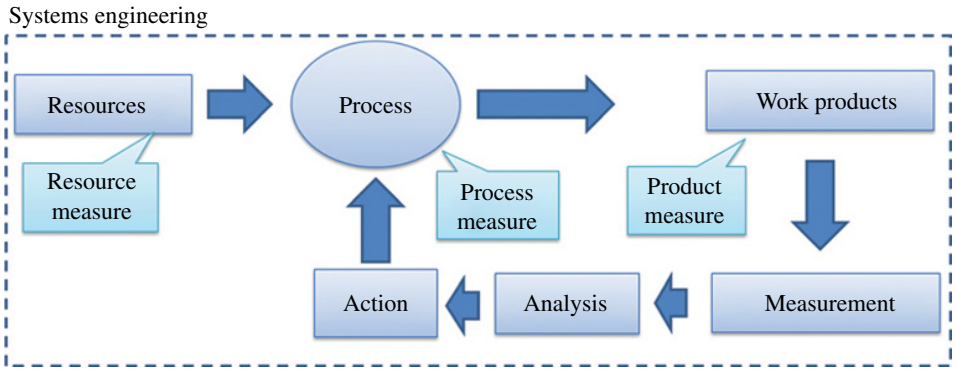


FIGURE 5.12 Measurement as a feedback control system. From INCOSE-TP-2010-005-02 Figure 2.1. Usage per the INCOSE Notices page. All other rights reserved.

5.7.2.3 Process-Oriented Measures A reasonable approach in organizing process-oriented measures is to categorize them based upon the organizational goal supported. Organizational goals may be broken into the following categories:

- Cost (development)
- Schedule (development)
- Quality (process)

The specific measures selected under each goal would be those necessary to track process activities thought to produce the greatest risk to meeting the goals of the product or organization.

5.7.2.4 Leading Indicators An important subgroup of process-related measures is leading indicators. A leading indicator is a measure for evaluating the effectiveness of how a specific activity is applied on a program in a manner that provides information about impacts that are likely to affect the system performance or SE effectiveness objectives.

A leading indicator may be an individual measure, or collection of measures, that is predictive of future system performance before the performance is realized. Leading indicators aid leadership in delivering value to customers and end users while assisting in taking interventions and actions to avoid rework and wasted effort.

Leading indicators differ from conventional SE measures in that conventional measures provide status and historical information, while leading indicators use an approach that draws on trend information to allow for predictive analysis (forward looking). By analyzing the trends, predictions can be forecast on the outcomes of certain activities. Trends are analyzed for insight into both the entity being measured and potential impacts to other entities. This provides leaders with the data they need to make informed decisions and, where necessary, take preventive or corrective action during the program in a proactive manner. While the leading indicators appear similar to existing measures and often use the same base information, the difference lies in how the information is gathered, evaluated, interpreted, and used to provide a forward-looking perspective. Examples of leading indicator measures include the following:

- *Requirements trends*—Rate of maturity of the system definition against the plan. Requirements trends characterize the stability and completeness of the system requirements that could potentially impact design and production.
- *Interface trends*—Interface specification closure against plan. Lack of timely closure could pose adverse impact to system architecture, design, implementation, and/or verification and validation (V&V), any of which could pose technical, cost, and schedule impact.
- *Requirements validation trends*—Progress against the plan in ensuring that the customer requirements are valid and properly understood. Adverse trends would pose impacts to system design activity with corresponding impacts to technical, cost, and schedule baselines and customer satisfaction.

For a more detailed treatment of this topic, including measurement examples, please consult the *Systems Engineering Leading Indicators Guide* (Roedler et al., 2010).

5.7.2.5 Product-Oriented Measures As shown in referenced document Technical Measurement (Roedler and Jones, 2006), product measures can be thought of as an interdependent hierarchy (see Figure 5.13).

5.7.2.6 MOEs and MOPs MOEs and MOPs are two concepts that represent types of measures typically collected. MOEs are defined in the *INCOSE-TP-2003-020-01, Technical Measurement Guide* (Roedler and Jones, 2006), as follows:

> The "operational" measures of success that are closely related to the achievement of the mission or operational objective being evaluated, in the intended operational environment under a specified set of conditions; i.e., how well the solution achieves the intended purpose. (Adapted from DoD 5000.2, DAU, and INCOSE)

MOPs are defined as follows:

> The measures that characterize physical or functional attributes relating to the system operation, measured or estimated under specified testing and/or operational environment conditions. (Adapted from DoD 5000.2, DAU, INCOSE, and EPI 280-04, LM Integrated Measurement Guidebook)

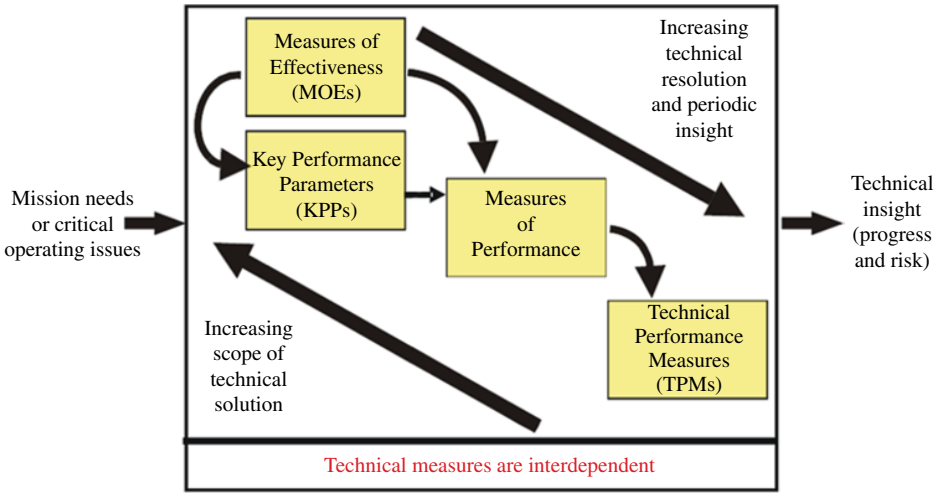


FIGURE 5.13 Relationship of technical measures. From INCOSE-TP-2003-020-01 Figure 1.1. Usage per the INCOSE Notices page. All other rights reserved.

MOEs, which are stated from the acquirer (customer/user) viewpoint, are the acquirer's key indicators of achieving the mission needs for performance, suitability, and affordability across the life cycle.

Although they are independent of any particular solution, MOEs are the overall operational success criteria (e.g., mission performance, safety, operability, operational availability, etc.) to be used by the acquirer for the delivered system, services, and/or processes. INCOSE-TP-2003-020-01, *Technical Measurement Guide* (Roedler and Jones, 2006), Section 3.2.1, provides further discussion on this topic.

MOPs measure attributes considered as important to ensure that the system has the capability to achieve operational objectives. MOPs are used to assess whether the system meets design or performance requirements that are necessary to satisfy the MOEs. MOPs should be derived from or provide insight for MOEs or other user needs. INCOSE-TP-2003-020-01, *Technical Measurement Guide*, Section 3.2.2, provides further discussion on this topic.

5.7.2.7 Key Performance Parameters Key Performance Parameters (KPPs) are defined in the INCOSE-TP-2003-020-01, *Technical Measurement Guide* (Roedler and Jones, 2006), as follows:

> Key Performance Parameters are a critical subset of the performance parameters representing those capabilities and characteristics so significant that failure to meet the threshold value of performance can be cause for concept, or system selected to be reevaluated or the project to be reassessed, or terminated.

Each KPP has a threshold and objective value. KPPs are the minimum number of performance parameters needed to characterize the major drivers of operational performance, supportability, and interoperability. The acquirer defines the KPPs at the time the operational concepts and requirements are defined.

5.7.2.8 TPMs TPMs are defined in INCOSE-TP-2003-020-01, *Technical Measurement Guide* (Roedler and Jones, 2006), as follows:

> TPMs measure attributes of a system element to determine how well a system or system element is satisfying or expected to satisfy a technical requirement or goal.

TPMs are used to assess design progress, compliance to performance requirements, or technical risks and provide visibility into the status of important project technical parameters to enable effective management, thus enhancing the likelihood of achieving the technical objectives of the project. TPMs are derived from or provide insight for the MOPs focusing on the critical technical parameters of specific architectural elements of the system as it is designed and implemented.

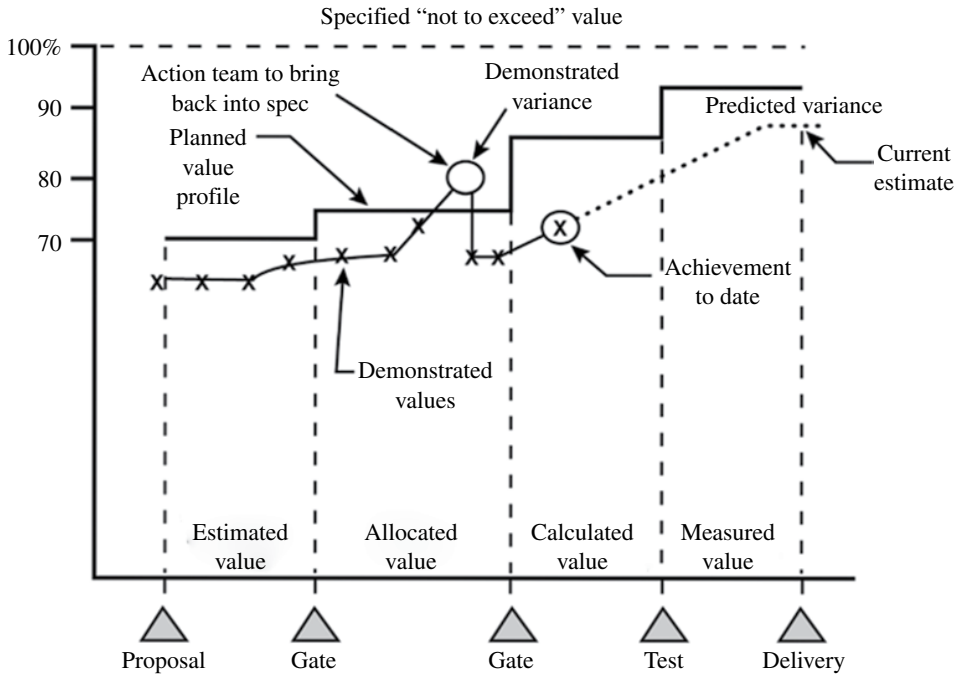


FIGURE 5.14 TPM monitoring. Reprinted with permission from Kevin Forsberg. All other rights reserved.

Selection of TPMs should be limited to critical technical thresholds or parameters that, if not met, put the project at cost, schedule, or performance risk. The TPMs are not a full listing of the requirements of the system or system element. The SEMP should define the approach to TPMs (Roedler and Jones, 2006).

Without TPMs, a project manager could fall into the trap of relying on cost and schedule status alone, with perhaps the verbal assurances of technical staff to assess project progress. This can lead to a product developed on schedule and within cost that does not meet all key requirements. Values are established to provide limits that give early indications if a TPM is out of tolerance, as illustrated in Figure 5.14.

Periodic recording of the status of each TPM provides the continuing verification of the degree of anticipated and actual achievement of technical parameters. Measured values that fall outside an established tolerance band alert management to take corrective action. INCOSE-TP-2003-020-01, *Technical Measurement Guide*, Section 3.2.3, provides further discussion on this topic.

5.8 QUALITY ASSURANCE PROCESS

5.8.1 Overview

5.8.1.1 Purpose As stated in ISO/IEC/IEEE 15288,

> [6.3.8.1] The purpose of the Quality Assurance process is to help ensure the effective application of the organization's Quality Management process to the project.

5.8.1.2 Description Quality assurance (QA) is broadly defined as the set of activities throughout the entire project life cycle necessary to provide adequate confidence that a product or service conforms to stakeholder requirements or that a process adheres to established methodology (ASQ, 2007). A subset of the quality management process, the quality assurance process activities are defined to provide an independent assessment of whether development and SE processes are capable of outcomes that meet requirements and that those processes are performed accurately, precisely, and consistently with all applicable prescriptions and documentation.

QA provides confidence that the developing organization, including subcontractors, adheres to established procedural requirements. Controlling variation in the development process is key to reducing variation in development outcomes. Thus, the quality assurance process offers a means of introducing checks and balances into the development process to ensure that errors or cost or schedule pressures do not result in uncontrolled process or procedural changes.

The term "quality assurance" (or QA) is often used interchangeably with the term "quality control." However, the focus of QA is during development activities (proactive), while "quality control" is typically associated with "inspection" after development activities (reactive).

QA is implemented through procedures for monitoring development and production processes and verifying that QA activities are effective in reducing defects in product or service outcomes. Additionally, QA is responsible for identification, analysis, and control of anomalies or errors discovered during life cycle activities. The level of rigor in QA must be appropriate to product or service requirements for the system under development.

Figure 5.15 is the IPO diagram for the quality assurance process.

5.8.1.3 Inputs/Outputs Inputs and outputs for the quality assurance process are listed in Figure 5.15. Descriptions of each input and output are provided in Appendix E.

5.8.1.4 Process Activities The quality assurance process includes the following activities:

- *Prepare for quality assurance*.
  - Establish and maintain the QA strategy (often captured in a QA plan).
  - Establish and maintain QA guidelines—policies, standards, and procedures.
  - Define responsibilities and authorities.
- *Perform product or service evaluations*.
  - Perform the evaluations at appropriate times in the life cycle as defined by the QA plan, ensuring V&V of the outputs of the life cycle processes. Ensure that QA perspectives are appropriately represented during design, development, verification, validation, and production activities.
  - Evaluate product verification results as evidence of QA effectiveness.

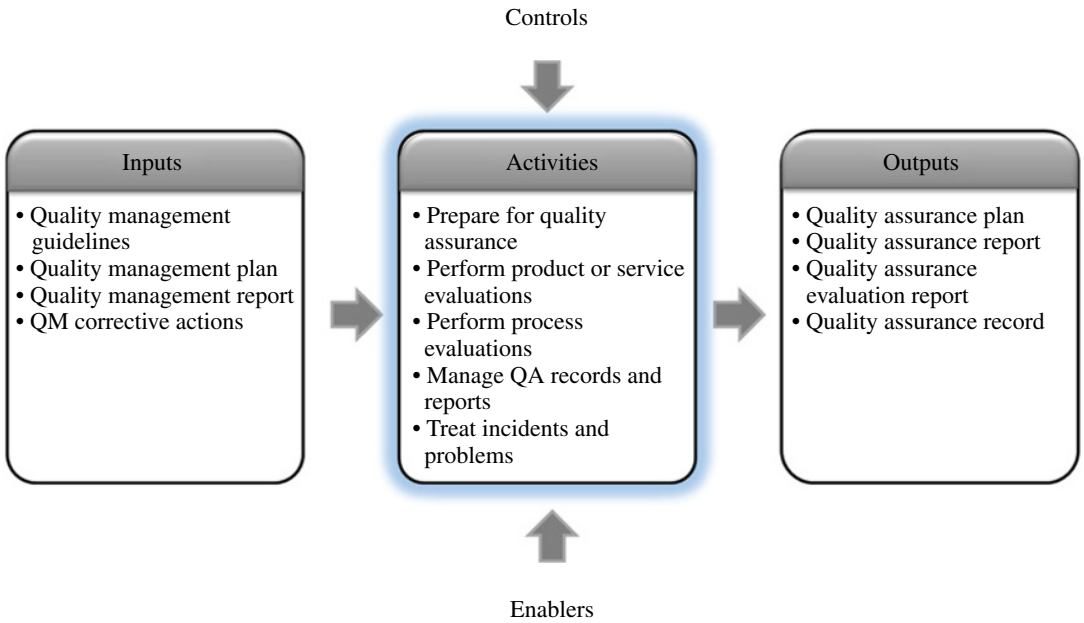


FIGURE 5.15 IPO diagram for the quality assurance process. INCOSE SEH original figure created by Shortell and Walden. Usage per the INCOSE Notices page. All other rights reserved.

- *Perform process evaluations*.
  - Implement prescribed surveillance on processes to provide an independent evaluation of whether the developing organization is in compliance with established procedures.
  - Evaluate enabling tools and environments for conformance and effectiveness.
  - Flow applicable procedural and surveillance requirements throughout the project supply chain and evaluate subcontractor processes for conformance to allocated requirements.
- *Manage QA records and reports*.
  - Create, maintain, and store records and reports in accordance with applicable requirements.
  - Identify incidents and problems associated with product and process evaluations.
- *Treat incidents and problems*.

  Note: Incidents are short-term anomalies or observations that require immediate attention, and problems are confirmed nonconformities that would cause the project to fail to meet requirements.
  - Document, classify, report, and analyze all anomalies.
  - Perform root cause analysis and note trends.
  - Recommend appropriate actions to resolve anomalies and errors, when indicated.
  - Track all incidents and problems to closure.

Common approaches and tips:

- Use existing agreements and applicable quality certifications or registrations (e.g., ISO 9001, CMMI, etc.), together with the organization's overarching quality management policy to provide essential guidance for QA approaches.
- Analyze statistics from process audits, verification results, product discrepancy reports, customer satisfaction monitoring, and accident and incident reporting to verify whether QA activities are effective.
- Continually demonstrate uncompromising integrity in monitoring development and SE processes. There must not be a perception that the development organization or project management is inappropriately influencing the judgment of QA personnel.
- Establish organizational independence and consistent support from senior leadership. The QA team should not be completely beholden to the project manager. Implement an escalation process so that QA issues not addressed by the project can be escalated to organizational leadership, as appropriate.

5.8.2 Elaboration

5.8.2.1 QA Concepts

Quality cannot be "inspected into" products or services after they are developed. QA performs an important role in ensuring that all elements of the developing organization execute activities in accordance with approved plans and procedures as a means of building quality into products or services. Through this process control role, QA enables systematic process improvement. Deming described this relationship between quality and process improvement: "Quality comes not from inspection, but from improvement of the production process" (Deming, 1986).

QA applies the policies and standards that govern development activities and procurement of raw materials in support of quality goals and objectives of the project. For example, the NASA has adopted the SAE Aerospace Standard (AS) 9100, Quality Systems—Aerospace—Model for Quality Assurance in Design, Development, Production, Installation and Servicing, as a means of building quality into its systems and controlling the statistical variation in all system elements by requiring adherence to a common quality standard (SAE Aerospace Quality Standard AS9100:C, 2009). Similarly, an acquiring organization may mandate that suppliers achieve a given CMMI level as a means of assuring that a given supplier is capable of delivering a consistent level of quality in its development processes.

QA also plays a key role during the verification activities themselves. The presence of independent QA personnel during verification activities provides an unbiased perspective on the integrity of verification procedures and the appropriate calibration of verification equipment and facilities. QA personnel also provide an independent assessment that verification results are accurately recorded. For example, it is not uncommon to require a QA signature on verification results reports

to attest that the verification procedures were followed and the results were accurately reported.

5.8.2.2 QA Methods Common QA techniques include:

- *Checklist*—A tool for ensuring all important steps or actions in an operation have been taken (ASQ, 2007).
- *Quality audit*—An independent review to determine whether project activities comply with established policies, processes, or procedures.
- *Root cause analysis*—A method of problem solving using specific techniques designed to address the underlying (root) causes of defects or anomalies. Popular root cause analysis techniques include Ishikawa (fishbone) diagrams, FTA, failure mode effects and criticality analysis, and the five why's technique.

6

AGREEMENT PROCESSES

The initiation of a project begins with user need. Once a need is perceived and resources are committed to establish a project, it is possible to define the parameters of an acquisition and supply relationship. One instance for which this relationship exists is whenever an organization with a need does not have the ability to satisfy that need without assistance. Agreement processes are defined in ISO/IEC/IEEE 15288 as follows:

> [6.1(b)] [Agreement] processes define the activities necessary to establish an agreement between two organizations.

Acquisition is also an alternative for optimizing investment when a supplier can meet the need in a more economical or timely manner. The acquisition and supply processes are the subject of Sections 6.1 and 6.2, respectively.

Virtually all organizations interface with one or more organizations from industry, academia, government, customers, partners, etc. An overall objective of agreement processes is to identify these external interfaces and establish the parameters of these relationships, including identifying the inputs required from the external entities and the outputs that will be provided to them. This network of relationships provides the context of the business environment of the organization and access to future trends and research. Some relationships are defined by the exchange of products or services.

The acquisition and supply processes are two sides of the same coin. Each process establishes the context and constraints of the agreement under which the other system life cycle processes are performed, regardless of whether the agreement is formal (as in a contract) or informal. The unique activities for the agreement processes are related to contracts and managing business relationships. An important contribution of ISO/IEC/IEEE 15288 is the recognition that systems engineers are relevant contributors in this domain (Arnold and Lawson, 2003). The maglev train case (see Section 3.6.3) is an example where the government representatives of China and Germany participated in the relationship.

Agreement negotiations are handled in various ways depending on the specific organizations and the formality of the agreement. For example, in formal contract situations with government agencies, there is usually a contract negotiation activity that may include multiple roles for both the acquirer and supplier to refine the contract terms and conditions. The systems engineer is usually in a supporting role to the project manager during

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.

negotiations and is responsible for impact assessments for changes, trade studies on alternatives, risk assessments, and other technical input needed for decisions.

A critical element to each party is the definition of acceptance criteria, such as:

1. Percent completion of the SyRS
2. Requirements stability and growth measures, such as the number of requirements added, modified, or deleted during the preceding time interval (e.g., month, quarter, etc.)
3. Percent completion of each contract requirements document: SOW, RFP, etc.

These criteria protect both sides of the business relationship—the acquirer from being coerced into accepting a product with poor quality and the supplier from the unpredictable actions of a fickle or indecisive buyer.

It is important to note that the previous criteria are negotiated for use during the agreement. During negotiations, it is also critical that both parties are able to track progress toward an agreement. Identifying where there are agreements or disagreements for documents and clauses is vital.

Note also that the agreement processes can be used for coordinating within an organization between different business units or functions. In this case, the agreement will usually be more informal, not requiring a contract or other legally binding set of documentation.

Two agreement processes are identified by ISO/IEC/IEEE 15288: the acquisition process and the supply process. They are included in this handbook because they conduct the essential business of the organization related to the system of interest (SOI) and establish the relationships between organizations relevant to the acquisition and supply (i.e., buying and selling) of products and services.

6.1 ACQUISITION PROCESS

6.1.1 Overview

6.1.1.1 Purpose As stated in ISO/IEC/IEEE 15288,

> [6.1.1.1] The purpose of the Acquisition process is to obtain a product or service in accordance with the acquirer's requirements.

The acquisition process is invoked to establish an agreement between two organizations under which one party acquires products or services from the other. The acquirer experiences a need for an operational system, for services in support of an operational system, for elements of a system being developed by a project, or for services in support of project activities. Often, our experience with the acquisition process is typified by the purchase of commodities or commercial products, such as telephones or automobiles. SE is often required to facilitate the procurement of more complex services and products. The start of an acquisition/supply process begins with the determination of, and agreement on, user needs. The goal is to find a supplier that can meet those needs.

6.1.1.2 Description The role of the acquirer demands familiarity with the technical, technical management, and organizational project-enabling processes as it is through them that the supplier will execute the agreement. An acquirer organization applies due diligence in the selection of a supplier to avoid costly failures and impacts to the organization budgets and schedules. This section is written from the perspective of the acquirer organization. Figure 6.1 is the IPO diagram for the acquisition process.

6.1.1.3 Inputs/Outputs Inputs and outputs for the acquisition process are listed in Figure 6.1. Descriptions of each input and output are provided in Appendix E.

6.1.1.4 Process Activities The acquisition process includes the following activities:

- *Prepare for the acquisition.*
  - Develop and maintain acquisition plans, policies, and procedures to meet the organization strategies, goals, and objectives as well as the needs of the project management and technical SE organizations.
  - Identify needs in a request for supply—such as an RFP or request for quotation (RFQ) or some other mechanism to obtain the supply of the system, service, or product. Through the use of the technical processes, including system requirements definition, the acquiring organization produces a set of requirements that will form the basis for the technical information of the agreement.
  - Identify a list of potential suppliers—suppliers may be internal or external to the acquirer organization.

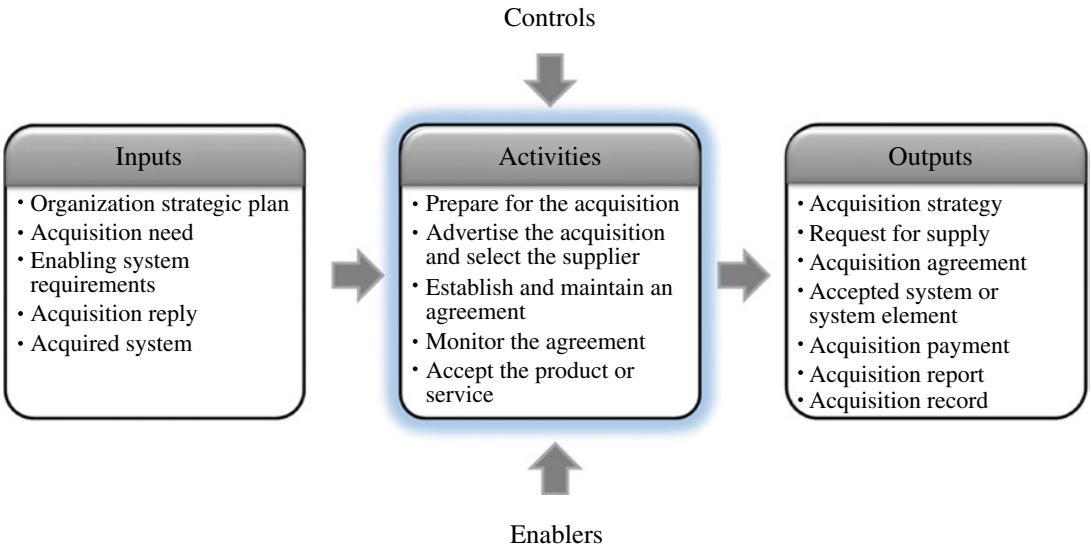


FIGURE 6.1 IPO diagram for the acquisition process. INCOSE SEH original figure created by Shortell and Walden. Usage per the INCOSE Notices page. All other rights reserved.

- *Advertise the acquisition and select the supplier.*
  - Distribute the RFP, RFQ, or other documented request for supply and select appropriate suppliers—using selection criteria, rank suppliers by their suitability to meet the overall need and establish supplier preferences and corresponding justifications. Viable suppliers should be willing to conduct ethical negotiations, able to meet technical obligations, and willing to maintain open communications throughout the acquisition process.
  - Evaluate supplier responses to the RFP or supply request—ensure the (SOI) meets acquirer needs and complies with industry and other standards. Assessments from the project portfolio management and quality management processes and recommendations from the requesting organization are necessary to determine the suitability of each response and the ability of the supplier to meet the stated commitments. Record recommendations from evaluation of responses to the RFP. This can range from formal documentation to less formal interorganizational interactions (e.g., between design engineering and marketing).
  - Select the preferred supplier based on acquisition criteria.
- *Establish and maintain an agreement.*
  - Negotiate agreement—the supplier commits to provide a product or service that satisfies the specified requirements and acceptance criteria for the system of interest. Both supplier and acquirer agree to participate in verification, validation, and acceptance activities; the acquirer agrees to render payment according to the schedule. Both agree to participate in exception and change control procedures and contribute to transparent risk management procedures. The agreement will establish criteria for assessing progress toward final delivery.
  - Establish delivery acceptance criteria—the procurement specification, in the context of the overall agreement, should clearly state the criteria by which the acquirer will accept delivery from the supplier. A verification matrix can be used to clarify these criteria.
- *Monitor the agreement.*
  - Manage acquisition process activities, including decision making for agreements, relationship building and maintenance, interaction with organization management, responsibility for the development of plans and schedules, and final approval authority for deliveries accepted from the supplier.

  - Maintain communications with supplier, stakeholders, and other organizations regarding the project.
  - Status progress against the agreed-to schedule to identify risks and issues, to measure progress toward mitigation of risks and adequacy of progress toward delivery and cost and schedule performance, and to determine potential undesirable outcomes for the organization. The project assessment and control process provides necessary evaluation information regarding cost, schedule, and performance.
  - Amend agreements when impacts on schedule, budget, or performance are identified.
- *Accept the product or service.*
  - Accept delivery of products and services—in accordance with all agreements and relevant laws and regulations.
  - Render payment—or other agreed consideration in accordance with agreed payment schedules.
  - Accept responsibility in accordance with all agreements and relevant laws and regulations.
  - When an acquisition process cycle concludes, a final review of performance is conducted to extract lessons learned for continued process performance.

Note: The agreement is closed through the portfolio management process, which manages the full set of systems and projects of the organization.

Common approaches and tips:

- Establish acquisition guidance and procedures that inform acquisition planning, including recommended milestones, standards, assessment criteria, and decision gates. Include approaches for identifying, evaluating, choosing, negotiating, managing, and terminating suppliers.
- Establish a technical point of responsibility within the organization for monitoring and controlling individual agreements. This person maintains communication with the supplier and is part of the decision-making team to assess progress in the execution of the agreement. The possibility of late delivery or cost overruns should be identified and communicated to the organization as early as noted.

Note: There can be multiple points of responsibility for an agreement that focus on technical, programmatic, marketing, etc.

- Define and track measures that indicate progress on agreements. Appropriate measures require the development of tailored measures that do not drive unnecessary and costly efforts but do provide the information needed to ensure the progress is satisfactory and that key issues and problems are identified early to allow time for resolution with minimal impact to the delivery and quality of the product and service.
- Include technical representation in the selection of the suppliers to critically assess the capability of the supplier to perform the required task. This helps reduce the risk of contract failure and its associated costs, delivery delays, and increased resource commitment needs. Past performance is highly important, but changes to key personnel should be identified and evaluated.
- Communicate clearly with the supplier about the real needs and avoid conflicting statements or making frequent changes in the statement of need that introduce risk into the process.
- Maintain traceability between the supplier's responses to the acquirer's solicitation. This can reduce the risk of contract modifications, cancellations, or follow-on contracts to fix the product or service.
- Institute for supply management has useful guidance for purchasing and marketing (ISM, n.d.).

6.2 SUPPLY PROCESS

6.2.1 Overview

6.2.1.1 Purpose As stated in ISO/IEC/IEEE 15288,

> [6.1.2.1] The purpose of the Supply process is to provide an acquirer with a product or service that meets agreed requirements.

The supply process is invoked to establish an agreement between two organizations under which one party supplies products or services to the other. Within the supplier organization, a project is conducted according to the recommendations of this handbook with the objective

of providing a product or service to the acquirer that meets the contracted requirements. In the case of a mass-produced commercial product or service, a marketing function may represent the acquirer and establish customer expectations.

6.2.1.2 Description The supply process is highly dependent upon the technical, technical management, and organizational project-enabling processes as it is through them that the work of executing the agreement is accomplished. This means that the supply process is the larger context in which the other processes are applied under the agreement. This section is written from the perspective of the supplier organization. Figure 6.2 is the IPO diagram for the supply process.

6.2.1.3 Inputs/Outputs Inputs and outputs for the supply process are listed in Figure 6.2. Descriptions of each input and output are provided in Appendix E.

6.2.1.4 Process Activities The supply process includes the following activities:

- *Prepare for the supply.*
  - Develop and maintain strategic plans, policies, and procedures to meet the needs of potential acquirer organizations, as well as internal organization goals and objectives including the needs of the project management and technical SE organizations.
  - Identify opportunities.
- *Respond to a tender.*
  - Select appropriate acquirers willing to conduct ethical negotiations, able to meet financial obligations, and willing to maintain open communications throughout the supply process.
  - Evaluate the acquirer requests and propose a SOI that meets acquirer needs and complies with industry and other standards. Assessments from the project portfolio management, human resource management, quality management, and business or mission analysis processes are necessary to determine the suitability of this response and the ability of the organization to meet these commitments.
- *Establish and maintain an agreement.*
  - Supplier commits to meet the negotiated requirements for the SOI; meet delivery milestones, verification, validation, and acceptance conditions; accept the payment schedule; execute exception and change control procedures; and maintain transparent risk management procedures. The agreement will establish criteria for assessing progress toward final delivery.

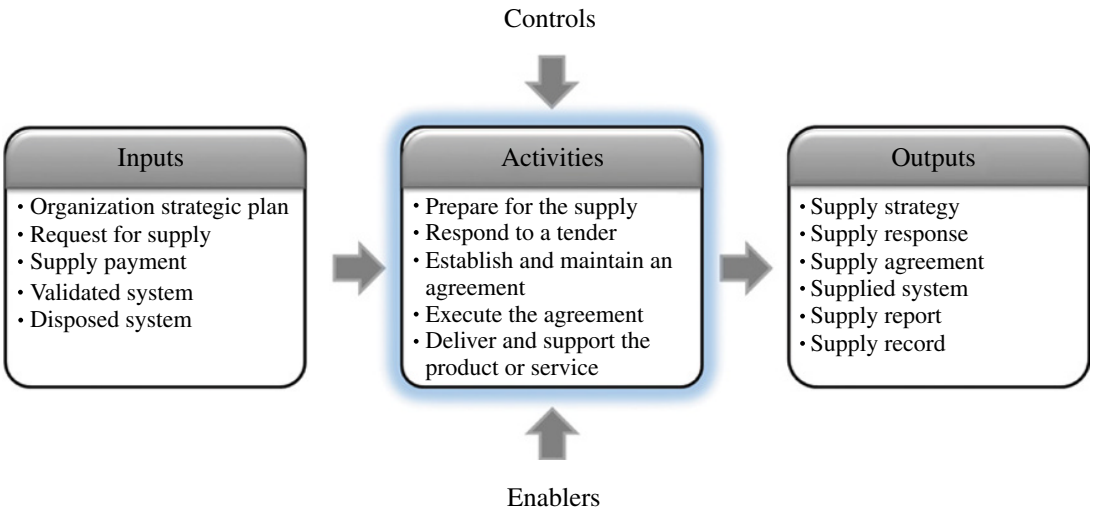


FIGURE 6.2 IPO diagram for the supply process. INCOSE SEH original figure created by Shortell and Walden. Usage per the INCOSE Notices page. All other rights reserved.

- *Execute the agreement.*
  - Start the project and invoke the other processes defined in this handbook.
  - Manage supply process activities, including decision making for agreements, relationship building and maintenance, interaction with organization management, responsibility for the development of plans and schedules, and final approval authority for deliveries made to acquirer.
  - Maintain communications with acquirers, subsuppliers, stakeholders, and other organizations regarding the project.
  - Evaluate carefully the terms of the agreement to identify risks and issues, progress toward mitigation of risks, and adequacy of progress toward delivery; evaluate cost and schedule performance; and determine potential undesirable outcomes for the organization.
- *Deliver and support the product or service.*
  - After acceptance and transfer of the final products and services, the acquirer will provide payment or other consideration in accordance with all agreements, schedules, and relevant laws and regulations.
  - When a supply process cycle concludes, a final review of performance is conducted to extract lessons learned for continued process performance.

Note: The agreement is closed through the portfolio management process, which manages the full set of systems and projects of the organization. When the project is closed, action is taken to close the agreement.

Common approaches and tips:

- Agreements fall into a large range from formal to very informal based on verbal understanding. Contracts may call for a fixed price, cost plus fixed fee, incentives for early delivery, penalties for late deliveries, and other financial motivators.
- Relationship building and trust between the parties is a nonquantifiable quality that, while not a substitute for good processes, makes the human interactions agreeable.
- Develop technology white papers or similar documents to demonstrate and describe to the (potential) acquirer the range of capabilities in areas of interest. Use traditional marketing approaches to encourage acquisition of mass-produced products.
- Maintain an up-to-date Internet presence, even if the organization does not engage in electronic commerce.
- When expertise is not available within the organization (e.g., legal and other governmental regulations, laws, etc.), retain subject matter experts to provide information and specify requirements related to agreements.
- Invest sufficient time and effort into understanding acquirer needs before the agreement. This can improve the estimations for cost and schedule and positively affect agreement execution. Evaluate any technical specifications for the product or service for clarity, completeness, and consistency.
- Involve personnel who will be responsible for agreement execution to participate in the evaluation of and response to the acquirer's request. This reduces the start-up time once the project is initiated, which in turn is one way to recapture the cost of writing the response.
- Make a critical assessment of the ability of the organization to execute the agreement; otherwise, the high risk of failure and its associated costs, delivery delays, and increased resource commitment needs will reflect negatively on the reputation of the entire organization.

7

ORGANIZATIONAL PROJECT-ENABLING PROCESSES

Organizational project-enabling processes are the purview of the organization (also known as enterprise) and are used to direct, enable, control, and support the system life cycle. Organizational project-enabling processes are defined in ISO/IEC/IEEE 15288 as follows:

> [6.2] The Organizational Project-Enabling Processes help ensure the organization's capability to acquire and supply products or services through the initiation, support and control of projects. They provide resources and infrastructure necessary to support projects ….

This chapter focuses on the capabilities of an organization relevant to the realization of a system; as stated above, they are not intended to address general business management objectives, although sometimes the two overlap.

Organizational units cooperate to develop, produce, deploy, utilize, support, and retire (including dispose of) the system of interest (SOI). Enabling systems may also need to be modified to meet the needs of new systems, developed, or acquired if they do not exist. Examples include development, manufacturing, training, verification, transport, logistics, maintenance, and disposal systems that support the SOI.

Six organizational project-enabling processes are identified by ISO/IEC/IEEE 15288. They are life cycle model management, infrastructure management, portfolio management, human resource management, quality management (QM), and knowledge management (KM). The organization will tailor these processes and their interfaces to meet specific strategic and communications objectives in support of the system projects.

7.1 LIFE CYCLE MODEL MANAGEMENT PROCESS

7.1.1 Overview

7.1.1.1 Purpose As stated in ISO/IEC/IEEE 15288,

> [6.2.1.1] The purpose of the Life Cycle Model Management process is to define, maintain, and assure availability of policies, life cycle processes, life cycle models, and procedures for use by the organization with respect to the scope of [ISO/IEC/IEEE 15288].

The value propositions to be achieved by instituting organization-wide processes for use by projects are as follows:

- Provide repeatable/predictable performance across the projects in the organization (this helps the

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.

organization in planning and estimating future projects and in demonstrating reliability to customers)
- Leverage practices that have been proven successful by certain projects and instill those in other projects across the organization (where applicable)
- Enable process improvement across the organization
- Improve ability to efficiently transfer staff across projects as roles are defined and performed consistently
- Enable leveraging lessons that are learned from one project for future projects to improve performance and avoid issues
- Improve startup of new projects (less reinventing the wheel)

In addition, the standardization across projects may enable cost savings through economies of scale for support activities (tool support, process documentation, etc.).

7.1.1.2 Description This process (i) establishes and maintains a set of policies and procedures at the organization level that support the organization's ability to acquire and supply products and services and (ii) provides integrated system life cycle models necessary to meet the organization's strategic plans, policies, goals, and objectives for all projects and all system life cycle stages. The processes are defined, adapted, and maintained to support the requirements of the organization, SE organizational units, individual projects, and personnel. Life cycle model management processes are supplemented by recommended methods and tools. The resulting guidelines in the form of organization policies and procedures are still subject to tailoring by projects, as discussed in Chapter 8. Figure 7.1 is the IPO diagram for the life cycle model management process.

7.1.1.3 Inputs/Outputs Inputs and outputs for the life cycle model management process are listed in Figure 7.1. Descriptions of each input and output are provided in Appendix E.

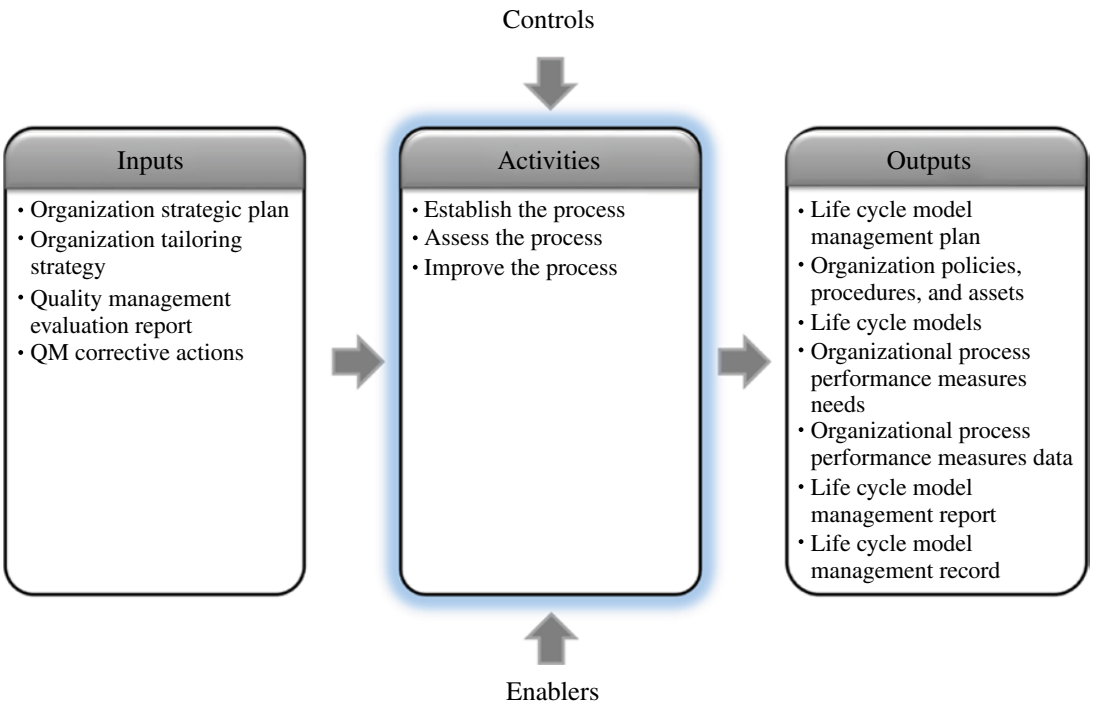


FIGURE 7.1 IPO diagram for the life cycle model management process. INCOSE SEH original figure created by Shortell and Walden. Usage per the INCOSE Notices page. All other rights reserved.

7.1.1.4 Process Activities The life cycle model management process includes the following activities:

- *Establish the process.*
  - Identify sources (organization, corporate, industry, academia, stakeholders, and customers) of life cycle model management process information.
  - Distill the information from multiple sources into an appropriate set of life cycle models that are aligned with the organization and business area plans and infrastructure.
  - Establish life cycle model management guidelines in the form of plans, policies, procedures, tailoring guidance, models, and methods and tools for controlling and directing the life cycle models.
  - Define, integrate, and communicate life cycle model roles, responsibilities, authorities, requirements, measures, and performance criteria based on the life cycle model management process guidelines.
  - Use business achievements to establish entrance and exit criteria for decision gates.
  - Disseminate policies, procedures, and directives throughout the organization.
- *Assess the process.*
  - Use assessments and reviews of the life cycle models to confirm the adequacy and effectiveness of the life cycle model management processes.
  - Identify opportunities to improve the organization life cycle model management guidelines on a continuing basis based on individual project assessments, individual feedback, and changes in the organization strategic plan.
  - Lessons learned and measurement results from process performance on projects should be used as significant sources from which to identify improvements.
- *Improve the process.*
  - Prioritize and implement the identified improvement opportunities.
  - Communicate with all relevant organizations regarding the creation of and changes in the life cycle model management guideline.

Common approaches and tips:

- Base the policies and procedures on an organization-level strategic and business area plan that provides a comprehensive understanding of the organization's goals, objectives, stakeholders, competitors, future business, and technology trends.
- Ensure that policy and procedure compliance review is included as part of the business decision gate criteria.
- Develop a life cycle model management process information database with essential information that provides an effective mechanism for disseminating consistent guidelines and providing announcements about organization-related topics, as well as industry trends, research findings, and other relevant information. This provides a single point of contact for continuous communication regarding the life cycle model management guidelines and encourages the collection of valuable feedback and the identification of organization trends.
- Establish an organization center of excellence for life cycle model management processes. This organization can become the focal point for the collection of relevant information, dissemination of guidelines, and analysis of assessments and feedback. They can also develop checklists and other templates to support project assessments to ensure that the predefined measures and criteria are used for evaluation.
- Manage the network of external relationships by assigning personnel to identify standards, industry and academia research, and other sources of organization management information and concepts needed by the organization.

  The network of relationships includes government, industry, and academia. Each of these external interfaces provides unique and essential information for the organization to succeed in business and meet the continued need and demand for improved and effective systems and products for its customers. It is up to the life cycle model management process to fully define and utilize these external entities and interfaces (i.e., their value, importance, and capabilities that are required by the organization):
  - Legislative, regulatory, and other government requirements
  - Industry SE and management-related standards, training, and capability maturity models
  - Academic education, research results, future concepts and perspectives, and requests for financial support

- Establish an organization communication plan for the policies and procedures. Most of the processes in this handbook include dissemination activities. An effective set of communication methods is needed to ensure that all stakeholders are well informed.
- Ensure the methods and tools for enabling the application of life cycle model management processes are effective and tailored to the implementation approach of the organization and its projects. A responsible organization can be created or designated to coordinate the identification and development of partnerships and/or relationships with tool vendors and working groups. They can recommend the use of methods and tools that are intended to help personnel avoid confusion, frustration, and wasting valuable time and money. These experts may also establish an integrated tool environment between interacting tools to avoid cumbersome (and inaccurate) data transfer.
- Include stakeholders, such as engineering and project management organizations, as participants in developing the life cycle model management guidelines. This increases their commitment to the recommendations and incorporates a valuable source of organization experience.
- Develop alternative life cycle models based on the type, scope, complexity, and risk of a project. This decreases the need for tailoring by engineering and project organizations.
- Provide clear guidelines for tailoring and adaptation.
- Work continually to improve the life cycle models and processes.

7.1.2 Elaboration

7.1.2.1 Standard SE Processes An organization engaged in SE provides the requirements for establishing, maintaining, and improving the standard SE process and the policies, practices, and supporting functional processes (see Fig. 7.2) necessary to meet customer needs throughout the organization. Further, it defines the process for tailoring the standard SE process for use on projects and for making improvements to the project-tailored SE processes.

Organizational management must review and approve the standard SE process and changes to it. Organizations should consider establishing an SE Process Group (SYSPG) to oversee SE process definition and implementation.

An organization defines or selects a standard set of SE processes for use as a reference SE process model, which is tailored by projects to meet specific customer and stakeholder needs. The reference model should tailor industry, government, or other agency "best practices" based on multiple government, industry, and organization reference SE process documents. The reference SE model includes an SE process improvement requirements, activity, or process to ensure ongoing evaluation and improvement

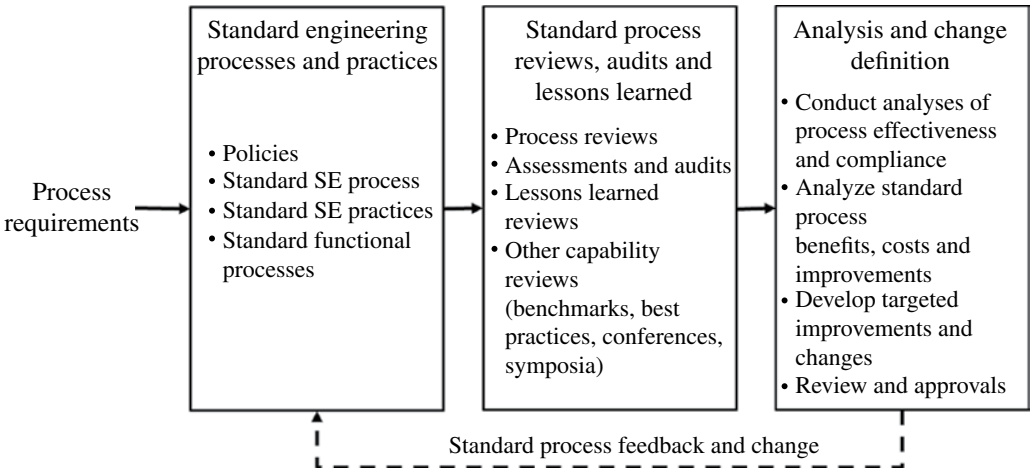


FIGURE 7.2 Standard SE process flow. INCOSE SEH v2 Figure 5.3. Usage per the INCOSE Notices page. All other rights reserved.

actions that take advantage of lessons learned. Projects are expected to follow this process, as tailored to meet project-specific SE process needs. The standard process must be tailorable, extensible, and scalable to meet a diverse range of projects, from small study projects to large projects requiring thousands of participants.

The standard SE process model can be established by selection of specific processes and practices from this handbook, industry SE process references (such as ISO/IEC/IEEE 15288 and ANSI/EIA-632), and government SE process references, as appropriate, and is applicable to every engineering capability maturity focus area or process area in the CMMI® approach (CMMI Product Team, 2010).

A high-performing organization also reviews the process (as well as work products), conducts assessments and audits (e.g., CMMI assessments and ISO audits), retains corporate memory through the understanding of lessons learned, and establishes how benchmarked processes and practices of related organizations can affect the organization. Successful organizations should analyze their process performance, its effectiveness and compliance to organizational and higher directed standards, and the associated benefits and costs and then develop targeted improvements.

The basic requirements for standard and project-tailored SE process control, based on CMMI and other resources, are as follows:

1. Process responsibilities for projects:
   i. Identify SE processes.
   ii. Document the implementation and maintenance of SE processes.
   iii. Use a defined set of standard methods and techniques to support the SE processes.
   iv. Apply accepted tailoring guidelines to the standard SE processes to meet project-specific needs.
2. Good process definition includes:
   i. Inputs and outputs
   ii. Entrance and exit criteria
3. Process responsibilities for organizations and projects:
   i. Assess strengths and weaknesses in the SE processes.
   ii. Compare the SE processes to benchmark processes used by other organizations.
   iii. Institute SE process reviews and audits of the SE processes.
   iv. Institute a means to capture and act on lessons learned from SE process implementation on projects.
   v. Institute a means to analyze potential changes for improvement to the SE processes.
   vi. Institute measures that provide insight into the performance and effectiveness of the SE processes.
   vii. Analyze the process measures and other information to determine the effectiveness of the SE processes.

Although it should be encouraged to identify and capture lessons learned throughout the performance of every project, the SE organization must plan and follow through to collect lessons learned at predefined milestones in the system life cycle. The SE organization should periodically review lessons learned together with the measures and other information to analyze and improve SE processes and practices. The results need to be communicated and incorporated into training. It should also establish best practices and capture them in an easy-to-retrieve form.

For more information on process definition, assessment, and improvement, see the resources in the bibliography, including the CMMI.

7.2 INFRASTRUCTURE MANAGEMENT PROCESS

7.2.1 Overview

7.2.1.1 Purpose As stated in ISO/IEC/IEEE 15288,

> [6.2.2.1] The purpose of the Infrastructure Management process is to provide the infrastructure and services to projects to support organization and project objectives throughout the life cycle.

7.2.1.2 Description The work of the organization is accomplished through projects, which are conducted within the context of the infrastructure environment. This infrastructure needs to be defined and understood within the organization and the project to ensure alignment of the working units and achievement of overall organization

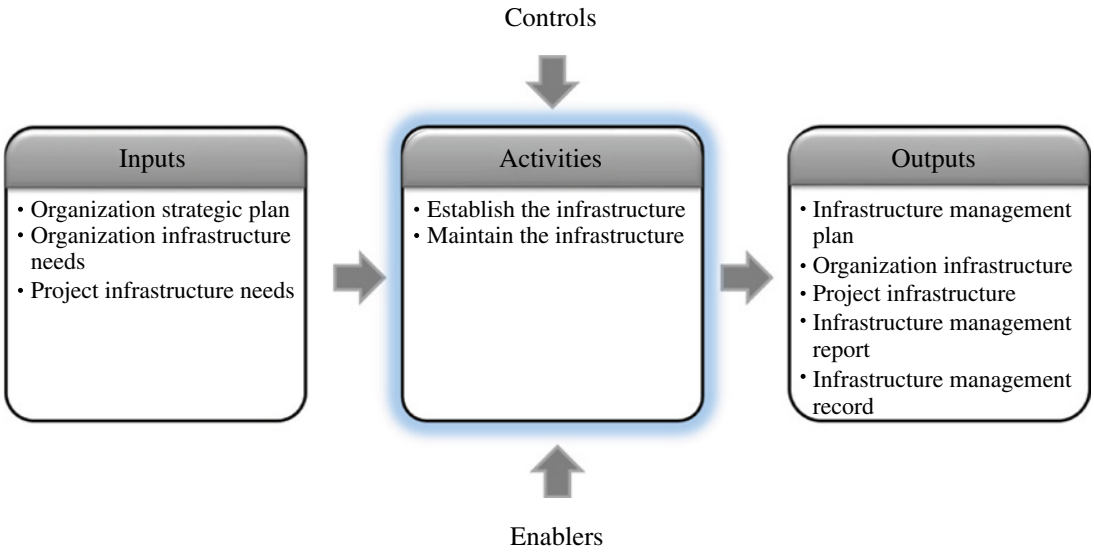


FIGURE 7.3 IPO diagram for the infrastructure management process. INCOSE SEH original figure created by Shortell and Walden. Usage per the INCOSE Notices page. All other rights reserved.

strategic objectives. This process exists to establish, communicate, and continuously improve the system life cycle process environment. Figure 7.3 is the IPO diagram for the infrastructure management process.

7.2.1.3 Inputs/Outputs Inputs and outputs for the infrastructure management process are listed in Figure 7.3. Descriptions of each input and output are provided in Appendix E.

7.2.1.4 Process Activities The infrastructure management process includes the following activities:

- *Establish the infrastructure.*
  - Gather and negotiate infrastructure resource needs with organization and projects.
  - Establish the infrastructure resources and services to ensure organization goals and objectives are met.
  - Manage resource and service conflicts and shortfalls with steps for resolution.
- *Maintain the infrastructure.*
  - Manage infrastructure resource availability to ensure organization goals and objectives are met. Conflicts and resource shortfalls are managed with steps for resolution.
  - Allocate infrastructure resources and services to support all projects.
  - Control multiproject infrastructure resource management communications to effectively allocate resources throughout the organization, and identify potential future or existing conflict issues and problems with recommendations for resolution.

Common approaches and tips:

- Qualified resources may be leased (insourced or outsourced) or licensed in accordance with the investment strategy.
- Establish an organization infrastructure architecture. Integrating the infrastructure of the organization can make the execution of routine business activities more efficient.
- Establish a resource management information system with enabling support systems and services to maintain, track, allocate, and improve the resources for present and future organization needs. Computer-based equipment tracking, facilities allocation, and other systems are recommended for organizations with over 50 people.
- Attend to physical factors, including facilities and human factors, such as ambient noise level and computer access to specific tools and applications.

- Begin planning in early life cycle stages of all system development efforts to address utilization and support resource requirements for system transition, facilities, infrastructure, information/data storage, and management. Enabling resources should also be identified and integrated into the organization's infrastructure.

7.2.2 Elaboration

7.2.2.1 Infrastructure Management Concepts Projects all need resources to meet their objectives. Project planners determine the resources needed by the project and attempt to anticipate both current and future needs. The infrastructure management process provides the mechanisms whereby the organization infrastructure is made aware of project needs and the resources are scheduled to be in place when requested. While this can be simply stated, it is less simply executed. Conflicts must be negotiated and resolved, equipment must be obtained and sometimes repaired, buildings need to be refurbished, and information technology services are in a state of constant change. The infrastructure management organization collects the needs, negotiates to remove conflicts, and is responsible for providing the enabling organization infrastructure without which nothing else can be accomplished. Since resources are not free, their costs are also factored into investment decisions. Financial resources are addressed under the portfolio management process, but all other resources, except for human resources (see Section 7.4), are addressed under this process.

Infrastructure management is complicated by the number of sources for requests, the need to balance the skills of the labor pool against the other infrastructure elements (e.g., computer-based tools), the need to maintain a balance between the budgets of individual projects and the cost of resources, the need to keep apprised of new or modified policies and procedures that might influence the skills inventory, and myriad unknowns.

Resources are allocated based on requests. Infrastructure management collects the needs of all the projects in the active portfolio and schedules or acquires nonhuman assets, as needed. Additionally, the infrastructure management process maintains and manages the facilities, hardware, and support tools required by the portfolio of organization projects. Infrastructure management is the efficient and effective deployment of an organization's resources when and where they are needed. Such resources may include inventory, production resources, or information technology. The goal is to provide materials and services to a project when they are needed to keep the project on target and on budget. A balance should be found between efficiency and robustness. Infrastructure management relies heavily on forecasts into the future of the demand and supply of various resources.

The organization environment and subsequent investment decisions are built on the existing organization infrastructure, including facilities, equipment, personnel, and knowledge. Efficient use of these resources is achieved by exploiting opportunities to share enabling systems or to use a common system element on more than one project. These opportunities are enabled by good communications within the organization. Integration and interoperability of supporting systems, such as financial, human resources (see Section 7.4), and training, is critically important to executing organization strategic objectives. Feedback from active projects is used to refine and continuously improve the infrastructure.

Further, trends in the market may suggest changes in the supporting environment. Assessment of the availability and suitability of the organization infrastructure and associated resources provides feedback for improvement and reward mechanisms. All organization processes require mandatory compliance with government and corporate laws and regulations. Decision making is governed by the organization strategic plan.

7.3 PORTFOLIO MANAGEMENT PROCESS

7.3.1 Overview

7.3.1.1 Purpose As stated in ISO/IEC/IEEE 15288,

> [6.2.3.1] The purpose of the Portfolio Management process is to initiate and sustain necessary, sufficient and suitable projects in order to meet the strategic objectives of the organization.

Portfolio management also provides organizational output regarding the set of projects, systems, and technical investments of the organization to external stakeholders, such as parent organizations, investors/funding sources, and governance bodies.

7.3.1.2 Description Projects create the products or services that generate income for an organization. Thus, the conduct of successful projects requires an adequate allocation of funding and resources and the authority to deploy them to meet project objectives. Most business entities manage the commitment of financial resources using well-defined and closely monitored processes.

The portfolio management process also performs ongoing evaluation of the projects and systems in its portfolio. Based on periodic assessments, projects are determined to justify continued investment if they have the following characteristics:

- Contribute to the organization strategy
- Progress toward achieving established goals
- Comply with project directives from the organization
- Are conducted according to an approved plan
- Provide a service or product that is still needed and providing acceptable investment returns

Otherwise, projects may be redirected or, in extreme instances, cancelled. Figure 7.4 is the IPO diagram for the portfolio management process.

7.3.1.3 Inputs/Outputs Inputs and outputs for the portfolio management process are listed in Figure 7.4. Descriptions of each input and output are provided in Appendix E.

7.3.1.4 Process Activities The portfolio management process includes the following activities:

- *Define and authorize projects.*
  - Identify, assess, and prioritize investment opportunities consistent with the organization strategic plan.
  - Establish business area plans—use the strategic objectives to identify candidate projects to fulfill them.
  - Establish project scope, define project management accountabilities and authorities, and identify expected project outcomes.
  - Establish the domain area of the product line defined by its main features and their suitable variability.
  - Allocate adequate funding and other resources.
  - Identify interfaces and opportunities for multi-project synergies.

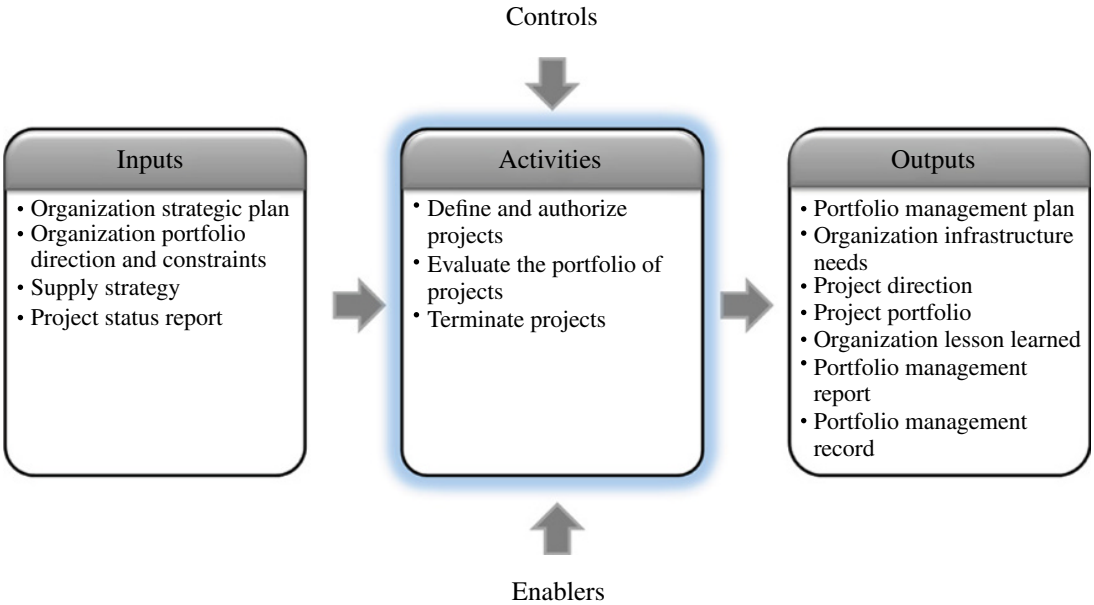


FIGURE 7.4 IPO diagram for the portfolio management process. INCOSE SEH original figure created by Shortell and Walden. Usage per the INCOSE Notices page. All other rights reserved.

  - Specify the project governance process including organizational status reporting and reviews.
  - Authorize project execution.
- *Evaluate the portfolio of projects.*
  - Evaluate ongoing projects to provide rationale for continuation, redirection, or termination.
- *Terminate projects.*
  - Close, cancel, or suspend projects that are completed or designated for termination.

Common approaches and tips:

- The process of developing the business area plans helps the organization assess where it needs to focus resources to meet present and future strategic objectives. Include representatives from relevant stakeholders in the organization community.
- When investment opportunities present themselves, prioritize them based on measurable criteria such that projects can be objectively evaluated against a threshold of acceptable performance.
- Expected project outcomes should be based on clearly defined, measurable criteria to ensure that an objective assessment of progress can be determined. Specify the investment information that will be assessed for each milestone. Initiation should be a formal milestone that does not occur until all resources are in place as identified in the project plan.
- Establish a program office or other coordination organization to manage the synergies between active projects in the organization portfolio. Complex and large organization architectures require the management and coordination of multiple interfaces and make additional demands on investment decisions. These interactions occur within and between the projects.
- Set a product line approach when different customers need the same or similar systems (i.e., common features), with some customizations (i.e., variants).
- Include risk assessments (see Section 5.4) in the evaluation of ongoing projects. Projects that contain risks that may pose a challenge in the future might require redirection. Cancel or suspend projects whose disadvantages or risks to the organization outweigh the investment.
- Include opportunity assessments in the evaluation of ongoing projects. Addressing project challenges may represent a positive investment opportunity for the organization. Avoid pursuing opportunities that are inconsistent with the capabilities of the organization and its strategic goals and objectives or contain unacceptably high technical risk, resource demands, or uncertainty.
- Allocate resources based on the requirements of the projects; otherwise, the risk of cost and schedule overruns may have a negative impact on quality and performance of the project.
- Establish effective governance processes that directly support investment decision making and communications with project management.

7.3.2 Elaboration

7.3.2.1 Define the Business Case and Develop Business Area Plans Portfolio management balances the use of financial assets within the organization. Organization management generally demands that there be some beneficial return for the effort expended in pursuing a project. The business case and associated business area plans establish the scope of required resources (e.g., people and money) and schedule and set reasonable expectations. An important element of each control design gate is a realistic review of the business case as the project matures. The result is reverification or perhaps restatement of the business case. The Iridium case, described in Section 3.2, illustrates the dangers of failing to keep a realistic perspective. Similarly, despite the technological triumph of implementing the world's first maglev train line (Section 3.6.3), the exorbitant initial cost and slow return on investment are causing authorities to question plans to build another line.

The business case may be validated in a variety of ways. For large projects, sophisticated engineering models, or even prototypes of key system elements, help prove that the objectives of the business case can be met and that the system will work as envisioned prior to committing large amounts of resources to full-scale engineering and manufacturing development. For very complex systems, such a demonstration can be conducted at perhaps 20% of development cost. For smaller projects, when the total investment is modest, proof-of-concept models may be constructed during the concept stage to prove the validity of business case assumptions.

Investment opportunities are not all equal, and organizations are limited in the number of projects that can be conducted concurrently. Further, some investments are not well aligned with the overall strategic plan of the organization. For these reasons, opportunities are evaluated against the portfolio of existing agreements and ongoing projects, taking into consideration the attainability of the stakeholders' requirements.

7.4 HUMAN RESOURCE MANAGEMENT PROCESS

7.4.1 Overview

7.4.1.1 Purpose As stated in ISO/IEC/IEEE 15288,

> [6.2.4.1] The purpose of the Human Resource Management process is to provide the organization with necessary human resources and to maintain their competencies, consistent with business needs.

7.4.1.2 Description Projects all need resources to meet their objectives. This process deals with human resources. Nonhuman resources, including tools, databases, communication systems, financial systems, and information technology, are addressed using the infrastructure management process (see Section 7.2).

Project planners determine the resources needed for the project by anticipating both current and future needs. The human resource management process provides the mechanisms whereby the organization management is made aware of project needs and personnel are scheduled to be in place when requested. While this can be simply stated, it is less simply executed. Conflicts must be resolved, personnel must be trained, and employees are entitled to vacations and time away from the job.

The human resource management organization collects the needs, negotiates to remove conflicts, and is responsible for providing the personnel without which nothing else can be accomplished. Since qualified personnel are not free, their costs are also factored into investment decisions. Figure 7.5 is the IPO diagram for the human resource management process.

7.4.1.3 Inputs/Outputs Inputs and outputs for the human resource management process are listed in Figure 7.5. Descriptions of each input and output are provided in Appendix E.

7.4.1.4 Processes Activities The human resource management process includes the following activities:

- *Identify skills.*
  - The skills of the existing personnel are identified to establish a "skills inventory."

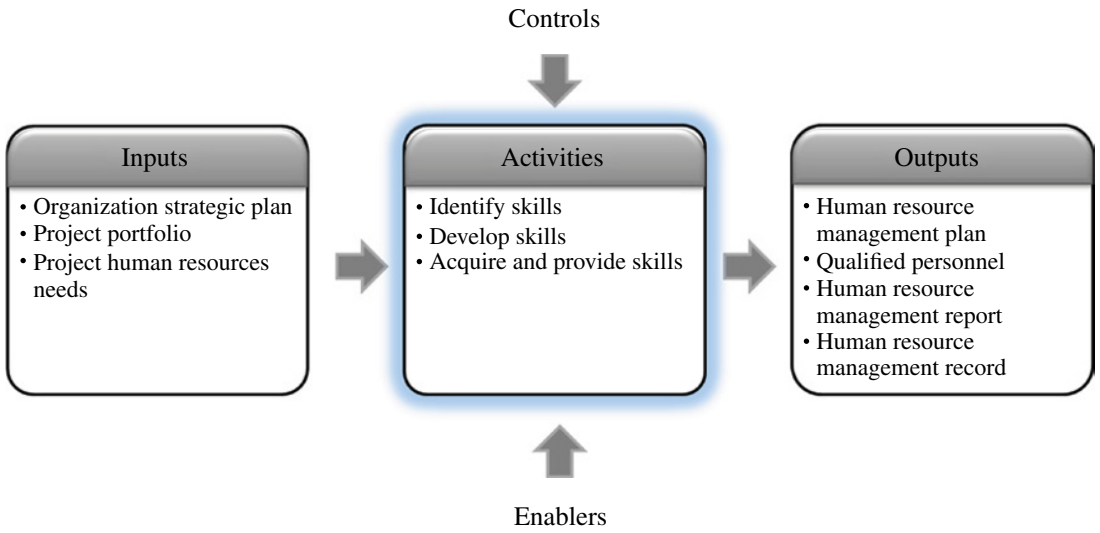


FIGURE 7.5 IPO diagram for the human resource management process. INCOSE SEH original figure created by Shortell and Walden. Usage per the INCOSE Notices page. All other rights reserved.

  - Review current and anticipated projects to determine the skills needed across the portfolio of projects.
  - Skill needs are evaluated against available people with the prerequisite skills to determine if training or hiring activities are indicated.
- *Develop skills.*
  - Obtain (or develop) and deliver training to close identified gaps of project personnel.
  - Identify assignments that lead toward career progression.
- *Acquire and provide skills.*
  - Provide human resources to support all projects.
  - Train or hire qualified personnel when gaps indicate that skill needs cannot be met with existing personnel.
  - Maintain communication across projects to effectively allocate human resources throughout the organization, and identify potential future or existing conflicts and problems with recommendations for resolution.
  - Other related assets are scheduled or, if necessary, acquired.

Common approaches and tips:

- The availability and suitability of personnel is one of the critical project assessments and provides feedback for improvement and reward mechanisms.
- Consider using an IPDT environment as a means to reduce the frequency of project rotation, recognize progress and accomplishments and reward success, and establish apprentice and mentoring programs for newly hired employees and students.
- Maintain a pipeline of qualified candidates that are interested in joining the organization as employees or temporary staff. Focus recruitment, training, and retention efforts on personnel with experience levels, skills, and subject matter expertise demanded by the projects. Personnel assessments should review proficiency, motivation, and ability to work in a team environment, as well as the need to be retrained, reassigned, or relocated.
- Personnel are allocated based on requests and conflicts are negotiated. The goal is to provide personnel to a project when they are needed to keep the project on target and on budget.
- A key concern is keeping project personnel from becoming overcommitted, especially persons with specialized skills.
- Skills inventory and career development plans are important documentation that can be validated by engineering and project management.
- Qualified personnel and other resources may be hired temporarily—insourced or outsourced in accordance with the organizational strategy.
- Encourage personnel to engage in external networks as a means of keeping abreast of new ideas and attracting new talent to the organization.
- Maintain an organization career development program that is not sidetracked by project demands. Develop a policy that all personnel receive training or educational benefits on a regular cycle. This includes both undergraduate and graduate studies, in-house training courses, certifications, tutorials, workshops, and conferences.
- Remember to provide training on organization policies and procedures and system life cycle processes.
- Establish a resource management information infrastructure with enabling support systems and services to maintain, track, allocate, and improve the resources for present and future organization needs. Computer-based human resource allocation and other systems are recommended for organizations over 50 people.
- Use the slack time in the beginning of a project to obtain and train the necessary people to avoid a shortfall of skilled engineers, technologists, managers, and operations experts.
- Trends in the market may suggest changes in the composition of project teams and the supporting IT environment.
- All organization processes require mandatory compliance with government and corporate laws and regulations.
- Employee performance reviews should be conducted regularly, and career development plans should be managed and aligned to the objectives of both the employee and the organization. Career development plans should be reviewed, tracked, and refined to provide a mechanism to help manage the employee's career within the organization.

7.4.2 Elaboration

7.4.2.1 Human Resource Management Concepts The human resource management process maintains and manages the people required by the portfolio of organization projects. Human resource management is the efficient and effective deployment of qualified personnel when and where they are needed. A balance should be found between efficiency and robustness. Human resource management relies heavily on forecasts into the future of the demand and supply of various resources.

The primary objective of this process is to provide a pool of qualified personnel to the organization. This is complicated by the number of sources for requests, the need to balance the skills of the labor pool against the other infrastructure elements (e.g., computer-based tools), the need to maintain a balance between the budgets of individual projects and the cost of resources, the need to keep apprised of new or modified policies and procedures that might influence the skills inventory, and myriad unknowns.

Project managers face their resource challenges competing for scarce talent in the larger organization pool. They must balance access to the experts they need for special studies with stability in the project team with its tacit knowledge and project memory. Today's projects depend on teamwork and optimally multidisciplinary teams. Such teams are able to resolve project issues quickly through direct communication between team members. Such intrateam communication shortens the decision-making cycle and is more likely to result in improved decisions because the multidisciplinary perspectives are captured early in the process.

7.5 QUALITY MANAGEMENT PROCESS

7.5.1 Overview

7.5.1.1 Purpose As stated in ISO/IEC/IEEE 15288,

> [6.2.5.1] The purpose of the Quality Management process is to assure that products, services and implementations of the quality management process meet organizational and project quality objectives and achieve customer satisfaction.

7.5.1.2 Description The quality management process makes visible the goals of the organization toward customer satisfaction. Since primary drivers in any project are time, cost, and quality, inclusion of a quality management process is essential to every organization. Many of the system life cycle processes are concerned with quality issues, and this forms some of the justification for exerting time, money, and energy into establishing these processes in the organization. Application of this handbook is one approach toward inserting a quality discipline into an organization.

Quality management (QM) process establishes, implements, and continuously improves the focus on customer satisfaction and organization goals and objectives. There is a cost to managing quality as well as a benefit. The effort and time required to manage quality should not exceed the overall value gained from the process. Figure 7.6 is the IPO diagram for the quality management process.

7.5.1.3 Inputs/Outputs Inputs and outputs for the quality management process are listed in Figure 7.6. Descriptions of each input and output are provided in Appendix E.

7.5.1.4 Process Activities The quality management process includes the following activities:

- *Plan quality management.*
  - Identify, assess, and prioritize quality guidelines consistent with the organization strategic plan. Establish QM guidelines—policies, standards, and procedures.
  - Establish organization and project QM goals and objectives.
  - Establish organization and project QM responsibilities and authorities.
- *Assess quality management.*
  - Evaluate project assessments.
  - Assess customer satisfaction against compliance with requirements and objectives.
  - Continuously improve the QM guidelines.
- *Perform quality management corrective action and preventive action.*
  - Recommend appropriate action, when indicated.
  - Maintain open communications—within the organization and with stakeholders.

Common approaches and tips:

- Quality is a daily focus—not an afterthought!

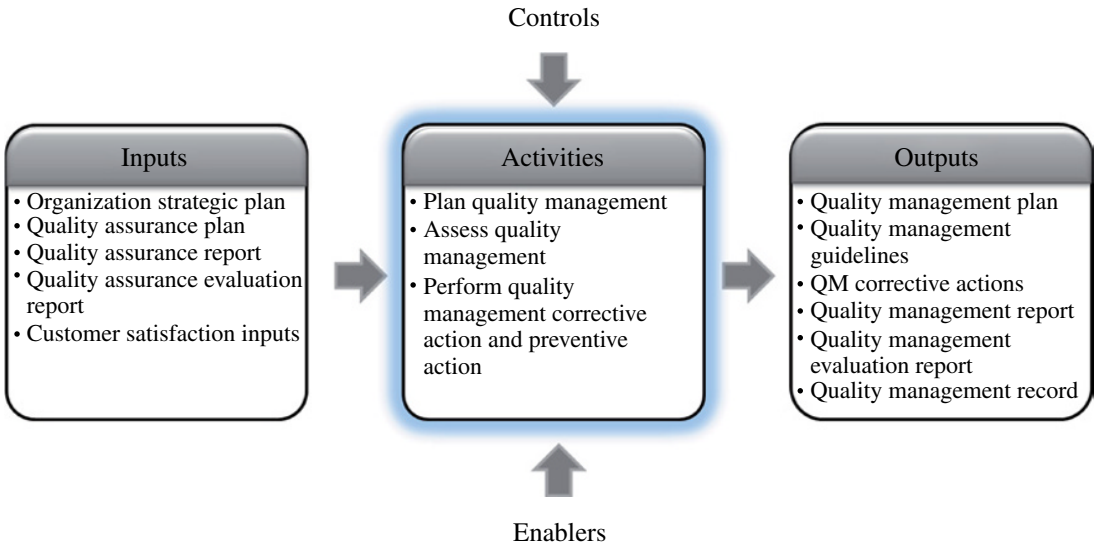


FIGURE 7.6 IPO diagram for the quality management process. INCOSE SEH original figure created by Shortell and Walden. Usage per the INCOSE Notices page. All other rights reserved.

- Strategic documentation, including quality policy, mission, strategies, goals, and objectives, provides essential inputs for analysis and synthesis of quality impacts, requirements, and solutions. Existing agreements also provide direction regarding the appropriate level of attention given to quality within the organization.
- Management commitment to quality is reflected in the strategic planning of the organization—the rest of the organization will follow. Everyone in the organization should know the quality policy.
- Development of a QM intranet and information database with essential information provides an effective mechanism for disseminating consistent guidelines and providing announcements about related topics, as well as industry trends, research findings, and other relevant information. This provides a single point of contact for continuous communication regarding the QM guidelines and encourages the collection of valuable feedback and the identification of organization trends.
- Analyze statistics from process audits, tests and evaluations, product discrepancy reports, customer satisfaction monitoring, accident and incident reporting, and the implementation of changes to items of a product (e.g., recalled product and/or production lines).
- QM is big business, and a plethora of standards, methods, and techniques exist to help an organization. A short list includes the ISO 9000 series, total quality management (TQM), and Six Sigma (statistical process control) (Brenner, 2006). According to ISO 9001 (2008), quality is the "Ability of a set of inherent characteristics of a product, system, or process to fulfill requirements of customers and other interested parties."
- A successful strategy is to aim at achieving customer satisfaction primarily by preventing nonfulfillment of requirements. Ideally, customer satisfaction is linked to compliance with requirements. Two indicators that the process is not working are situations where (i) the project is compliant but the customer is unhappy or (ii) the project is not compliant but the customer is happy.
- The consistent involvement and commitment of top management with timely decision making is mandatory for the quality program. This is reflected in staffing and training of project auditors.
- Project assessments include measurements that can be evaluated to determine the performance of a project team and the progress toward a quality outcome.

- Trends in tailoring of project-specific quality plans provide clear indications of potential improvements in the overall organization guidelines.
- The team of people working in this process will also find a wealth of material in ISO standards and other sources.
- The quality program should have an escalation mechanism so that findings/issues not addressed by the project can be raised to senior management. Projects are under schedule/resource constraints and may not always be responsive to quality findings. In these situations, there should be a mechanism to raise this to senior management to alert them to potential impacts and help them to determine how to proceed.

7.5.2 Elaboration

7.5.2.1 QM Concepts The purpose of QM is to outline the policies and procedures necessary to improve and control the various processes within the organization that ultimately lead to improved business performance.

The primary objective of QM is to produce an end result that meets or exceeds stakeholder expectations. For example, using a quality system program, manufacturers establish requirements for each type or family of product to achieve products that are safe and effective. To meet this objective, they establish methods and procedures to design, produce, distribute, service, and document devices that meet the quality system requirements. QM is closely related to the V&V processes.

Quality assurance (QA) is generally associated with activities such as failure testing, statistical control, and total quality control. Many organizations use statistical process control as a means to achieve Six Sigma levels of quality. Traditional statistical process controls use random sampling to test a fraction of the output for variances within critical tolerances. When these are found, the manufacturing processes are corrected before more bad parts can be produced.

Quality experts (Crosby, 1979; Juran, 1974) have determined that if quality cannot be measured, it cannot be systematically improved. Assessment provides the feedback needed to monitor performance, make midcourse corrections, diagnose difficulties, and pinpoint improvement opportunities. A widely used paradigm for QA management is the plan–do–check–act approach, also known as the Shewhart cycle (Shewhart, 1939).

Quality pioneer W. Edwards Deming stressed that meeting user needs represents the defining criterion for quality and that all members of an organization need to participate actively in "constant and continuous" quality improvement—to commit to the idea that "good enough isn't" (Deming, 1986). His advice marked a shift from inspecting for quality after production to building concern for quality into organization processes. As an example, in 1981, Ford launched a "Quality is Job 1" campaign that went beyond getting good workers and supporting them with high-quality training, facilities, equipment, and raw materials. By characterizing quality as a "job," everyone in the organization was motivated to concern themselves with quality and its improvement—for every product and customer (Scholtes, 1988).

Total quality control deals with understanding what the stakeholder/customer really wants. If the original need statement does not reflect the relevant quality requirements, then quality can be neither inspected nor manufactured into the product. For instance, the Øresund Bridge consortium included not only the bridge material and dimensions but also operating, environmental, safety, reliability, and maintainability requirements.

Product certification is the process of certifying that a certain product has passed performance or QA tests or qualification requirements stipulated in regulations, such as a building code or nationally accredited test standards, or that it complies with a set of regulations governing quality or minimum performance requirements. Today, medical device manufacturers are advised to use good judgment when developing their quality system and apply those sections of the Food and Drug Administration Quality System Regulation that are applicable to their specific products and operations. The regulation 21-CFR-820.5 is continuously updated since its release in 1996. As such, it ought not to be possible to repeat the errors of the Therac-25 project (see Section 3.6.1).

7.6 KNOWLEDGE MANAGEMENT PROCESS

7.6.1 Overview

7.6.1.1 Purpose As stated in ISO/IEC/IEEE 15288,

> The purpose of the Knowledge Management process is to create the capability and assets that enable the organization to exploit opportunities to re-apply existing knowledge.

7.6.1.2 Description KM is a broad area that transcends the bounds of SE and project management, and there are a number of professional societies that focus on it. KM includes the identification, capture, creation, representation, dissemination, and exchange of knowledge across targeted groups of stakeholders. It draws from the insights and experiences of individuals and/or organizational groups. The knowledge includes both explicit knowledge (conscious realization of the knowledge, often documented and easily communicated) and tacit knowledge (internalized in an individual without conscious realization) and can come from either individuals (through experience) or organizations (through processes, practices, and lessons learned) (Alavi and Leidner, 1999; Roedler, 2010).

Within an organization, explicit knowledge is usually captured in its training, processes, practices, methods, policies, and procedures. In contrast, tacit knowledge is embodied in the individuals of the organization and requires specialized techniques to identify and capture the knowledge, if it is to be passed along within the organization.

KM efforts typically focus on organizational objectives such as improved performance, competitive advantage, innovation, the sharing of lessons learned, integration, and continuous improvement of the organization (Gupta and Sharma, 2004). So it is generally advantageous for an organization to adopt a KM approach that includes building the framework, assets, and infrastructure to support the KM.

The motivation for putting KM in place includes:

- Information sharing across the organization
- Reducing redundant work due to not having the information needed at the right time
- Avoiding "reinventing the wheel"
- Facilitating training, focusing on best practices
- Capturing knowledge that would "go out the door" with retirements and attrition

The last item in this list is a major concern as we see a negative slope in the supply of systems engineers. As the percentage of experienced systems engineers retiring is increasing, it becomes even more important to capture the tacit knowledge that otherwise could be lost and then make that knowledge available to the developing systems engineers.

In this handbook, KM is viewed from an organizational project-enabling perspective, that is, how the organization supports the program or project environment with the resources in its KM system. The support provided to the project can come in several ways, including:

- Knowledge captured from technical experts
- Lessons learned captured from previous similar projects
- Domain engineering information that is applicable for reuse on the project, such as part of a product line or family of systems
- Architecture or design patterns that are commonly encountered
- Other reusable assets that may be applicable to the SOI

Figure 7.7 is the IPO diagram for the knowledge management process.

7.6.1.3 Inputs/Outputs Inputs and outputs for the knowledge management process are listed in Figure 7.7. Descriptions of each input and output are provided in Appendix E.

7.6.1.4 Process Activities The knowledge management process includes the following activities:

- *Plan knowledge management.*
  - Establish a KM strategy that defines how the organization and projects within the organization will interact to ensure the right level of knowledge is captured to provide useful knowledge assets. This needs to be done in a cost-effective manner, so there is a need to prioritize the efforts.
  - Establish the scope of the KM strategy—the organization and projects need to identify the specific knowledge information to capture and manage.
  - Establish which projects will be subject to the knowledge management process. If the knowledge assets are not used, then the effort has been wasted. If there is no identified project that will benefit from the knowledge asset, then it probably should not be considered.
- *Share knowledge and skills throughout the organization.*
  - Capture, maintain, and share knowledge and skills per the strategy.

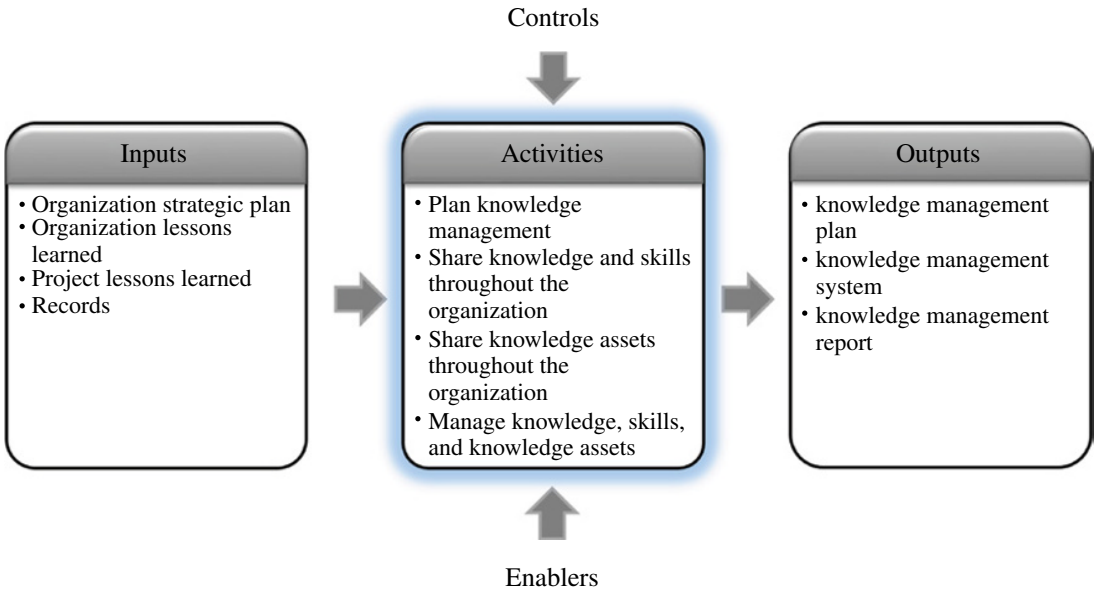


FIGURE 7.7 IPO diagram for the knowledge management process. INCOSE SEH original figure created by Shortell and Walden. Usage per the INCOSE Notices page. All other rights reserved.

  - The infrastructure should be established to include mechanisms to easily identify and access the assets and to determine the level of applicability for the project considering its use.
- *Share knowledge assets throughout the organization.*
  - Establish a taxonomy for the reapplication of knowledge.
  - Establish a representation for domain models and domain architectures. The intent of this is to ensure an understanding of the domain to help identify and manage opportunities for common system elements and their representations, such as architecture or design patterns, reference architectures, common requirements, etc.
  - Define or acquire the knowledge assets applicable to the domain, including system and software elements, and share them across the organization. As the system and system elements are defined in the technical processes, the information items that represent those definitions should be captured and included as knowledge assets for the domain.
- *Manage knowledge, skills, and knowledge assets.*
  - As the domain, family of systems, or product line changes, ensure the associated knowledge assets are revised or replaced to reflect the latest information. In addition, the associated domain models and architectures also may need to be revised.
  - Assess and track where the knowledge assets are being used. This can help understand the utility of specific assets, as well as determine whether they are being applied where they are applicable.
  - Determine whether the knowledge assets reflect the advances in technology, where applicable, and if they continue to evolve with the market trends and needs.

Common approaches and tips:

- The planning for KM may include:
  - Plans for obtaining and maintaining knowledge assets for their useful life
  - Characterization of the types of assets to be collected and maintained along with a scheme to classify them for the convenience of users
  - Criteria for accepting, qualifying, and retiring knowledge assets
  - Procedures for controlling changes to the knowledge assets
  - A mechanism for knowledge asset storage and retrieval

- In developing an understanding of the domain, it is important to identify and manage both the commonalities (such as features, capabilities, or functions) and the differences or variations of the system elements (including where a common system element has variations in parameters depending on the system instance). The domain representations should include:
  - Definition of the boundaries
  - Relationships of the domains to other domains
  - Domain models that incorporate the commonalities and differences allowing for sensitivity analysis across the range of variation
  - An architecture for a family of systems or product line within the domain, including their commonalities and variations

7.6.2 Elaboration

7.6.2.1 General KM Implementation Since KM focuses on capturing the organizational, project, and individual knowledge for use throughout the organization in the future, it is important to capture end-of-project lessons learned prior to the project personnel moving on to new assignment. An effective knowledge management process has the knowledge capture mechanisms in place to capture the relevant information throughout the life of the project rather than trying to piece it together at the end. This includes identification of systems that are part of a product line or family of systems and system elements that are designed for reuse. For the first instance of these systems and system elements, the KM system needs to capture the domain engineering artifacts in a way to facilitate their use in the future. For subsequent instances, the KM system needs to provide the domain engineering information and capture any variations, updates of technology, and lessons learned. Issues important to the organization:

- Definition and planning of KM activities for domain engineering and asset preservation, including tasks dedicated to domain engineering of product lines or families of systems and to the preservation of reusable assets.
- Integration of architecture management into the KM system including frameworks, architecture reuse, architecture reference models, architecture patterns, platform-based engineering, and product line architecture.
- Characterization of the types of assets to be collected and maintained including an effective means for users to find the applicable assets
- Determination of the quality and validity of the assets

7.6.2.2 Potential Reuse Issues There are serious traps in reuse, especially with respect to COTS and nondevelopmental item (NDI) elements:

- Are the new system or system element requirements and operational characteristics exactly like the prior one? Trap: the prior solution was intended for a different use, environment, or performance level, or it was a concept only that was never built.
- Did the prior system or system element work correctly? Trap: it worked perfectly, but the new application is outside the qualified range (e.g., using a standard car for a high-speed track race).
- Is the new system or system element to operate in the same environment as the prior one? Trap: it is not certain, but there is no time to study it. One NASA Mars probe was lost because the development team used a radiator design exactly as was used on a successful satellite in Earth orbit. When the Mars mission failed, the team then realized that Earth orbiting environment, while in space, is different from a deep space mission.
- Is the system/system element definition defined and understood (i.e., requirements, constraints, operating scenarios, etc.)? Trap: too often, the development team assumes that if a reuse solution will be applied (especially for COTS), there is no need for well-defined system definition. The issues may not show up until systems integration, causing major cost and schedule perturbations.
- Is the solution likely to have emergent requirements/behaviors where the reuse is being considered? Trap: a solution that worked in the past was used without consideration for the evolution of the solution. If COTS is used, there may be no way to adapt or modify it for the emergent requirements.

A properly functioning KM system paired with well-defined processes and engineering discipline can help avoid these problems.

8

TAILORING PROCESS AND APPLICATION OF SYSTEMS ENGINEERING

Standards and handbooks address life cycle models and systems engineering (SE) processes that may or may not fully apply to a given organization and/or project. Most are accompanied by a recommendation to adapt them to the situation at hand.

The principle behind tailoring is to ensure that the process meets the needs of the project while being scaled to the level of rigor that allows the system life cycle activities to be performed with an acceptable level of risk. Tailoring scales the rigorous application to an appropriate level based on need. The life cycle model may be tailored as described in Chapter 3. Processes may be tailored as described in this section. All processes apply to all stages, tailoring determines the process level that applies to each stage, and that level is never zero. There is always some activity in each process in each stage.

Figure 8.1 is a notional graph for balancing formal process against the risk of cost and schedule overruns (Salter, 2003). Insufficient SE effort is generally accompanied by high risk. However, as illustrated in Figure 8.1, too much formal process also introduces high risk. If too much rigor or unnecessary process activities or tasks are performed, increased cost and schedule impacts will occur with little or no added value. Tailoring occurs dynamically over the system life cycle depending on risk and the situational environment and should be continually monitored and adjusted as needed.

For ISO/IEC/IEEE 15288, process tailoring is the deleting or adapting the processes to satisfy particular circumstances or factors of the organization or project using the process. While ISO/IEC/IEEE 15288 tailoring focuses on the deletion of unnecessary or unwarranted process elements, it does allow for additions and modifications as well.

This chapter describes the process of tailoring the life cycle models and SE processes to meet organization and project needs. Refer to ISO/IEC TR 24748-1 (2010) and ISO/IEC TR 24748-2 (2010) for more information on adaptation and tailoring. This chapter also describes the application of the SE processes in various product sectors and domains, for product lines, for services, for enterprises, and in very small and micro enterprises (VSMEs).

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.

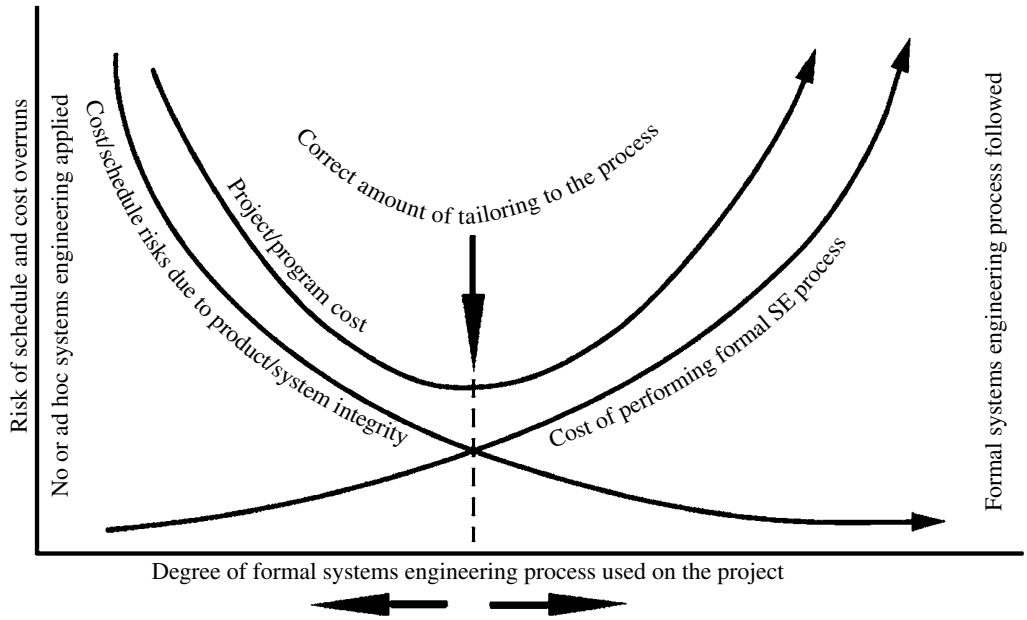


FIGURE 8.1 Tailoring requires balance between risk and process. INCOSE SEH original figure created by Michael Krueger, adapted from Ken Salter. Usage per the INCOSE Notices page. All other rights reserved.

8.1 TAILORING PROCESS

8.1.1 Overview

8.1.1.1 Purpose As stated in ISO/IEC/IEEE 15288,

> [A.2.1] The purpose of the Tailoring process is to adapt the processes of [ISO/IEC/IEEE 15288] to satisfy particular circumstances or factors.

8.1.1.2 Description At the organization level, the tailoring process adapts external standards in the context of the organizational processes to meet the needs of the organization. At the project level, the tailoring process adapts organizational processes for the unique needs of the project. Figure 8.2 is the IPO diagram for the tailoring process.

8.1.1.3 Inputs/Outputs Inputs and outputs for the tailoring process are listed in Figure 8.2. Descriptions of each input and output are provided in Appendix E.

8.1.1.4 Process Activities The tailoring process includes the following activities:

- *Identify and record the circumstances that influence tailoring.*
  - Identify tailoring criteria for each stage—Establish the criteria to determine the process level that applies to each stage.
- *Take due account of the life cycle structures recommended or mandated by standards.*
- *Obtain input from parties affected by the tailoring decisions.*
  - Determine process relevance to cost, schedule, and risks.
  - Determine process relevance to system integrity.
  - Determine quality of documentation needed.
  - Determine the extent of review, coordination, and decision methods.
- *Make tailoring decisions.*
- *Select the life cycle processes that require tailoring.*
  - Determine other changes needed for the process to meet organizational or project needs beyond the tailoring (e.g., additional outcomes, activities, or tasks).

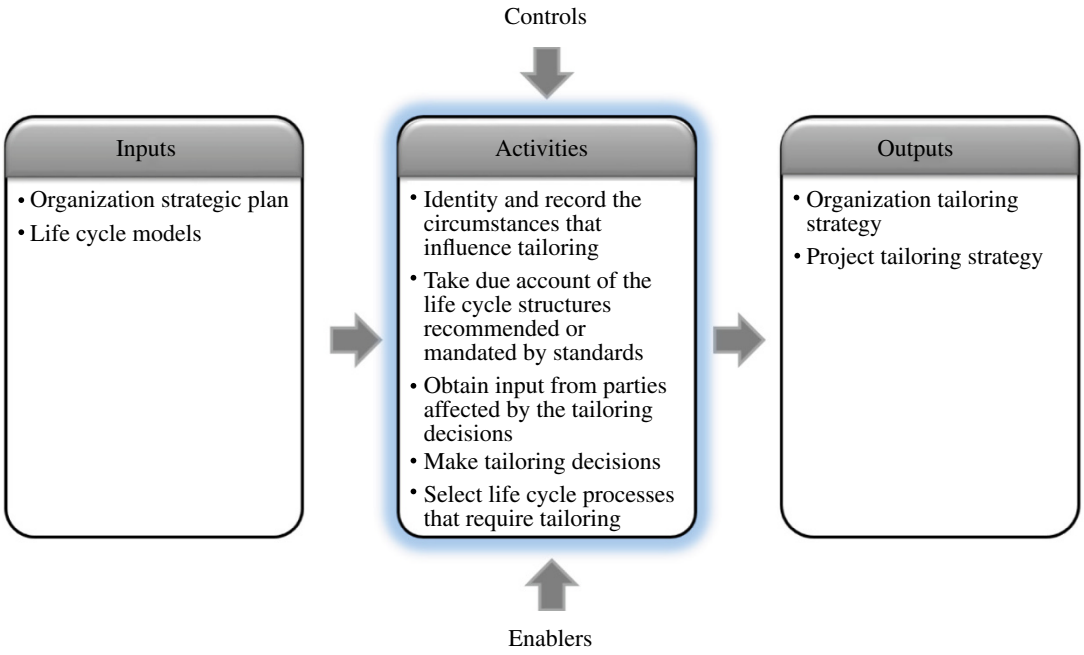


FIGURE 8.2 IPO diagram for the tailoring process. INCOSE SEH original figure created by Shortell and Walden. Usage per the INCOSE Notices page. All other rights reserved.

Common approaches and tips:

- Eliminate unnecessary outcomes, activities, and tasks, and add additional ones.
- Base decisions on facts and obtain approval from an independent authority.
- Use the decision management process to assist in tailoring decisions.
- Conduct tailoring at least once for each stage.
- Drive the tailoring based on the environment of the system life cycle stages.
- Constrain the tailoring based on agreements between organizations.
- Control the extent of tailoring based on issues of compliance to stakeholder, customer, and organization policies, objectives, and legal requirements.
- Influence the extent of tailoring of the agreement process activities based on the methods of procurement or intellectual property.
- Remove extra activities as the level of trust builds between parties.
- Identify a set of formal processes and outcomes/activities/tasks at the end of the tailoring process. This includes but is not limited to:
  - A documented set of tailored processes
  - Identification of the system documentation required
  - Identified reviews
  - Decision methods and criteria
  - The analysis approach to be used
- Identify the assumptions and criteria for tailoring throughout the life cycle to optimize the use of formal processes.

8.1.2 Elaboration

8.1.2.1 Organizational Tailoring

When contemplating if and how to incorporate a new or updated external standard into an organization, the following should be considered (Walden, 2007):

- Understand the organization.
- Understand the new standard.

- Adapt the standard to the organization (not vice versa).
- Institutionalize standards compliance at the "right" level.
- Allow for tailoring.

8.1.2.2 Project Tailoring Project tailoring applies specifically to the work executed through programs and projects. Factors that influence tailoring at the project level include:

- Stakeholders and customers (e.g., number of stakeholders, quality of working relationships, etc.)
- Project budget, schedule, and requirements
- Risk tolerance
- Complexity and precedence of the system

Today's systems are more often jointly developed by many different organizations. Cooperation must transcend the boundaries of any one organization. Harmony between multiple suppliers is often best maintained by agreeing to follow a set of consistent processes and standards. Consensus on a set of practices is helpful but adds complexity to the tailoring process.

8.1.2.3 Traps in Tailoring Common traps in the tailoring process include, but are not limited to, the following:

1. Reuse of a tailored baseline from another system without repeating the tailoring process
2. Using all processes and activities "just to be safe"
3. Using a preestablished tailored baseline
4. Failure to include relevant stakeholders

8.2 TAILORING FOR SPECIFIC PRODUCT SECTOR OR DOMAIN APPLICATION

The discipline of SE can be applied on any size and type of system. However, that does not mean that it should be blindly applied in the same fashion on every system. While the same SE fundamentals apply, different domains require different points of emphasis in order to be successful.

This section provides a starting point for people looking to apply SE in different domains. An objective of this section is to provide guidance on how to introduce the concepts of SE to a new field. As an example, the audience for this guidance might be SE champions that already work in the field and want to promote SE.

The domains are listed in alphabetical order and should not be considered to be exhaustive. In addition, one needs to recognize the evolving maturity of SE in these applications. The following descriptions capture the current state of the practice and may not necessarily be representative of the state of practice at the time this handbook is being read/used.

8.2.1 Automotive Systems

SE has been traditionally applied with success by industries that develop and sell large complex systems, with a relatively long life cycle and small production volumes. In contrast, the automotive industry has manufactured high volumes of consumer products with a great diversity for more than 120 years, with quite a success. A highly cost-driven industry, the automotive sector is characterized by a permanent quest for cost-efficiency and massive reuse of parts and engineering artifacts.

Modern commercial automobiles have become complex, high-technology products in a relatively short time span. The complexity drivers usually pointed out by the actors of the industry are:

- The increasing number of vehicle functionalities supported by software, electronics, and mechatronic technologies
- Increasing safety constraints
- Increasing environmental constraints, such as air pollution and decreasing fossil fuel stocks, leading to a partial or total electrification of the vehicle power train
- Globalization and the car market growth in emerging countries, which induce greater variations in the product definition and a different repartition of worldwide production of vehicles
- New trends such as "advanced mobility services," "smart cities and smart transportation systems," and autonomous vehicles, which induce changes in the purpose and scope of the traditional automobile product

Most engineering specialties and domain specific practices involved in the automotive industry follow international

TABLE 8.1 Standardization-related associations and automotive standards

Organization/standard	Description
SAE International, formerly the Society of Automotive Engineers	One of the main organizations that coordinate the development of technical standards for the automotive industry. Currently, SAE International is a globally active professional association and standards organization for engineering professionals in various industries, whose principal emphasis is placed on transport industries such as automotive, aerospace, and commercial vehicles
Japan Society of Automotive Engineers (JSAE)	An organization that sets automotive standards in Japan, analogous to the SAE International
Association for Standardization of Automation and Measuring Systems (ASAM)	An incorporated association under German law whose members are primarily international car manufacturers, suppliers, and engineering service providers from the automotive industry. The ASAM standards define protocols, data models, file formats, and application programming interfaces (APIs) for the use in the development and testing of automotive electronic control units
AUTomotive Open System ARchitecture (AUTOSAR)	An open and standardized automotive software architecture, jointly developed by automobile manufacturers, suppliers, and tool developers. Some of its key goals include the standardization of basic system functions, scalability to different vehicle and platform variants, transferability throughout the network, integration from multiple suppliers, maintainability
The GENIVI Alliance	A nonprofit consortium whose goal is to establish a globally competitive, Linux-based operating system, middleware, and platform for the automotive in-vehicle infotainment (IVI) industry. GENIVI specifications cover the entire product life cycle and software updates and upgrades over the vehicle's lifetime
ISO/TS 16949	An international standard for particular requirements for the application of ISO 9001 quality management systems for automotive production and relevant service part organizations
IEC 62196	An international standard for set of electrical connectors and charging modes for electric vehicles maintained by the International Electrotechnical Commission (IEC)

standards. Some important automotive-related standards, associations, and SE standards are shown in Table 8.1. These standards usually define characteristics, measurement procedures, or testing procedures for specific vehicle systems or specific aspects of the product and are often adapted to establish regulations by local authorities.

Automotive standards are normally used for product qualification or "certification" purposes, although it should be noted that certification procedures in the automotive industry differ greatly from those applied in other sectors, like aerospace. Usually, the scope of these standards is either the product or the manufacturing process, with specific regulations ruling each vehicle system. One remarkable exception to this type of standards is the functional safety standard ISO 26262 (2011), which defines activities and work products covering a full "safety life cycle" and allowing a systematic and traceable mastery of safety risks.

SE as a discipline started to develop rather recently in some automotive organizations, around the mid-1990s or early 2000s, mainly as a response to the aforementioned challenges. While the organizations that have shown interested in SE agree that most of these challenges could be leveraged by applying a systems approach, SE is not yet a standard practice in the industry.

The application of SE is not mandatory in the automotive industry to develop and sell products. The cultural and organizational changes that are required for an effective global industry-wide implementation of SE are beginning to take place.

The automotive industry is characterized by an extensive reuse of legacy engineering artifacts (requirements, architectures, validation plans) and system constituents (systems, system elements, parts) and by the management of huge product lines (at vehicle, system, and part levels), with high stakes associated to the efficient management of those product lines.

When applying SE in the automotive industry, one should take into consideration these specificities. An adapted SE process for the automotive domain should then be an ideal combination of product-driven and stakeholder-/requirement-driven approaches. The weight of the former type of approach will be more important for "classical" vehicle systems and based on typical variant management techniques (product derivation or configuring), while the relative weight of the second type of approach will be more important for innovative systems and "untraditional" vehicle scopes or usages.

In particular, when implementing the SE technical processes for the first time, the importance of upstream SE activities (e.g., operational analysis and requirement elaboration) must be underlined. Ideally, these activities should take place *off-line*, that is, "disconnected" from the development of the product, so that the produced SE artifacts baselines are properly defined and verified.

Since the involvement of carmakers in the development the different vehicle systems differs from one car maker to the other and from one vehicle domain to the other, care should also be taken in the tailoring of agreement processes, which will depend on the particular project context. The formalization of the exchanges between the Original Equipment Manufacturer (OEM) and their providers during the lifetime of a project should help tailoring these processes and could set the groundwork for their future standardization.

8.2.2 Biomedical and Healthcare Systems

SE has come to be recognized as a major contributor to biomedical and healthcare product and development, particularly due to the need for architectural soundness, requirements traceability, and subdisciplines such as safety, reliability, and human factors engineering. Biomedical and healthcare systems range from low complexity to high complexity. They also range from low risk to high risk. Many of these systems must work in harsh environments, including inside the human body. SE within the biomedical and healthcare environment may not be as mature as the defense and aerospace domain, for example, but the growing complexity of medical devices, their connectivity, and their use environment is quickly increasing the breadth and depth of the practice. Standards such as ISO 14971 (application of risk management to medical devices) and IEC 60601 (medical device safety) are driving organizations to take a deeper look into safety and the engineering practices behind it. Other government standards and regulations such as the US Code of Federal Regulations (CFR) Title 21 Part 820 (medical devices/quality system regulations) shape the development process for medical devices sold therein. Thus, systems engineers are increasingly being brought on board to leverage their life cycle management skills and validate that the final product does indeed meet the needs of its stakeholders.

In the biomedical and healthcare environment, it is important to understand that risk management is generally centered around user safety rather than technical or business risks and that traceability is often a key factor in regulatory audits. Organizations that have strong SE practices are therefore in a better position to avoid development pitfalls and to effectively defend their design decisions if a regulatory audit does occur. Going forward, biomedical and healthcare organizations that effectively leverage SE standards such as ISO/IEC/IEEE 15288 and ISO 29148 will be in a much better position to develop safe and effective products. In general, such standards do not need to be excessively tailored, although firms with new or maturing practices may want to focus on lean implementations in order to obtain early and effective adoption.

8.2.3 Defense and Aerospace Systems

While SE has been practiced in some form from antiquity, what has now become known as the modern definition of SE has its roots in defense and aerospace systems of the twentieth century. It has been recognized as a distinct activity in the late 1950s and early 1960s as a result of the technological advances taking place that led to increasing levels of system complexity and systems integration challenges. The need for SE increased with the large-scale introduction of digital computers and software. Defense and aerospace evolved to address systemic approaches to issues such as the widespread adaptation of COTS technologies and the use of system-of-system (SoS) approaches.

Defense and aerospace systems are characterized as complex technical systems with many stakeholders and compressed development timelines. The systems must also be highly available and work in extreme conditions all over the world—from deserts to rain forests and to arctic outposts. Defense and aerospace systems are also characterized by long system life cycles, so logistics is of

prime importance. Most defense and aerospace systems have a strong human interaction, so usability/human systems integration is critical for successful operations.

Since SE has a strong heritage in defense and aerospace, much of the SE processes in this handbook can be used as is in a straightforward manner, with just the normal project tailoring for the unique aspects of the project. It is important to note that as ISO/IEC/IEEE 15288 has evolved into a more domain- and country-neutral SE standard, some care must be taken to ensure that the defense and aerospace focus is reasserted upon application. The defense organizations of many countries have specific policies, standards, and guidebooks to guide the application of SE in their environment.

8.2.4 Infrastructure Systems

Infrastructure systems are significant physical structures used for commodity transfer (e.g., roads, bridges, railroads, mass transit, and electricity, water, wastewater, and oil and gas distribution) or are major industrial plants (e.g., oil and gas platforms, refineries, mines, smelters, nuclear reprocessing, water and wastewater treatment, and steelworks). The domain is multidisciplinary, vast and diverse, and characterized by large-scale projects often with loosely defined boundaries, evolving system architectures, long implementation (which can exceed a few decades) and asset life periods, and multiphase life cycles. Infrastructure systems are distinguished from manufacturing and production by most often being one-off, large objects and where construction takes place on a site rather than in a factory.

Infrastructure projects tend to exhibit a degree of uniqueness, complexity, and cost uncertainty. A significant proportion of time and cost is spent during the construction stage. These difficulties are exacerbated by changes in project environments (economical, political, legislative, technological, etc.) and hence to the stakeholders' expectations and design solutions that take place over an extended period of time.

Unlike other SE domains, most infrastructure projects cannot be standardized and do not involve a prototype. In addition, significant external interfaces exist, and failures in one element may cascade to other elements. For example, falling debris from the World Trade Center attack damaged the water system and flooded the New York Stock Exchange, resulting in severe impacts to an interconnected system of infrastructure systems.

Within infrastructure, many of the engineering disciplines (e.g., civil, structural, mechanical, chemical, process, and electrical) have well-established, traditional practices and are guided by industry codes and standards. SE practices are more developed in the high-technology subsystems that involve software development (e.g., modern communications-based train control systems, intelligent transportation systems, telemetry and process control) and particularly in industries where system safety is paramount.

The benefit of SE to the infrastructure domain lies in the structured approach to delivering and operating a multidisciplinary, integrated, and configurable system and needs to align with the associated project management and asset management practices. Many of the processes described in this handbook are being employed to handle that complexity but using a different terminology. The "Guide for the Application of Systems Engineering in Large Infrastructure Projects" (INCOSE-TP-2010-007-01, 2012) emphasizes the relationship between system, work, and organizational breakdown structures and the importance of good configuration management, not only through the project life cycle but during the whole asset lifetime. These items and the interactions between them are determined by a multitude of constraints, with the physical structure of the object and access being the most obvious but also many others, such as availability of suitable contractors and of manpower and machines, fabrication and shipping durations, and ones arising from the environment in which the construction takes place.

Infrastructure projects tend to quickly define the high-level design solution (e.g., a high-speed railway or nuclear power plant), and therefore, the greatest benefit of applying SE principles is gained in the systems integration and construction stage. Careful analysis to identify all the potentially affected organizations, structures, systems, people, and processes is required to ensure proposed changes will not adversely impact other areas and will lead to the required outcome.

The domain could benefit from better organization and integration of activities leading into the construction stage of the project life cycle. This would help manage the risks and uncertainty associated with cost estimating and changing scope and could improve construction productivity, hence making the industry more cost-effective. SE efforts tailored to meet the uniqueness characterized by infrastructure projects control the project's internal

and external dynamics (including uncertainty caused by natural events) and consider the project and postproject conditions will allow the domain to respond to the unique challenges it faces due to its broad, diverse, unique, and unpredictable makeup.

The documentation of infrastructure processes is largely company and project specific, although based on general standards, such as the ISO 9000/10000 series (quality management), ISO 10845 (construction procurement), and ISO 12006 (organization of information about construction works), as well as various national and international standards for the contractual (legal) framework in which a construction project is embedded. The choice of contractual framework and the allocation of liability and commercial risk is a major factor in designing the construction process.

8.2.5 Space Systems

Space systems are those systems that leave the Earth's atmosphere or are closely associated with their support and deployment. Due to the extremely high costs and physical exertion of deploying assets into Earth orbit or beyond, space systems typically require high reliability with no maintenance other than software changes. This makes it necessary for all system elements to work the first time or be compensated by complex operational workarounds.

SE was developed in large part due to the demands of the space race and associated defense technologies such as ballistic missiles. The discipline is very mature in this domain and does not require adaptation.

Key emphases of SE in the space domain are validation and verification, testing, and integration of highly reliable, well-characterized systems. Risk management is also key in determining when to incorporate new technologies and how to react to changing requirements through multiyear developments and programmatic challenges. The traditional SE Vee approach has its basis in space systems, as they are typically relatively new designs conceived, built, and deployed by agencies or prime contractors. Declining budgets are making the unity of vision less common as consortia and partnerships are made to pool resources.

A large number of standards are used in the space domain. Telecommunications is a major source, since spectra and noninterference must be negotiated on a global basis. Electrical and data standards are also used in many parts of both in-space and ground support systems. National space agencies and militaries often set standards as well (e.g., the European Cooperation for Space Standardization, US "MIL" Standards). An excellent example of a space-based SE handbook is freely available from the NASA (2007b). As space systems are deployed by more countries, standardization via ISO and IEEE is becoming more common for an increasing set of interoperability issues.

8.2.6 (Ground) Transportation Systems

The use of SE has emerged as the standard approach for complex capital programs in the fields of aerospace engineering, defense, and information technology. In the transportation industry, however, the assimilation of SE principles and methods has grown more slowly. The historical orientation of most transportation agencies has been to structure capital projects by engineering disciple and adopt low-bid procurement methods.

However, in the past few decades, the use of SE in transportation has been growing, with several progressive transit agencies (including London Underground and Network Rail in the United Kingdom, ProRail in the Netherlands, New York City Transit in the United States, and the Land Transport Authority in Singapore) having established SE departments and many others beginning to embed SE principles into their capital programs.

Emerging experience from these agencies suggests the following aspects are most relevant to consider when applying and tailoring SE principles to the transportation domain:

- *Recognize the single-discipline tendency*—the historical orientation in transportation agencies is to organize projects by engineering discipline. This can generate disciplinary silos that flow from early engineering through procurement and into system design, build, and test. SE must work across these silos, which can introduce tension to organizations used to working in the traditional way.
- *Working with in-service SoS*—many transportation systems are large, distributed, in-service SoS. This means that transportation SE work is often focused on systems upgrade and must work with and around the existing operation while still maintaining appropriate levels of public service.

- *Delivering socioeconomic benefits are often the key driver, yet the operator has an important role*—transportation systems primarily existing to provide socioeconomic benefits to the public. The key drivers for many transportation agencies are therefore often focused on improving customer experience and public safety. However, care should be taken to also focus on operational benefits, and operability and maintainability should be considered throughout.
- *Demonstrating the value of SE*—many transportation agencies have a historic divide between capital delivery and operations that persists into today's organizations. Project managers face cost and schedule pressure to deliver quickly, and this can sometimes generate opposition from those who perceive that the emphasis SE places on early-phase analysis injects delay into the project manager's schedule, while the benefits accrued from the analysis are felt most strongly by the operational stakeholders who are outside of the project manager's immediate organization.
- *Considering the commercial and public pressures*—there is increasing dependence on private investment to fund the larger transportation infrastructure projects, which increases the demand for a faster return on investment (ROI), which in turn places greater restrictions on time available to explore the problem space. Furthermore, while the desire to select to a jump to technology solution is endemic in many sectors, many transportation agencies face acute pressure from the public, the media, and politicians to demonstrate early signs of progress in terms that internal and external stakeholders understand—typically in specifics of technology. These pressures can sometimes negatively influence an agency's ability to apply SE to its projects.

8.3 APPLICATION OF SYSTEMS ENGINEERING FOR PRODUCT LINE MANAGEMENT

Product line management (PLM) is a combination of product, process, management, and organization to migrate from single-system engineering to a product line approach. PLM can support the goal of improved organizational competitiveness as it decreases the development cost, increases the quality, and enlarges the product catalog.

For the customer, a product line approach allows an organization to offer a family of products, which are adapted to the customer. This can lead to optimization of the product or service, the cost and time of acquisition, the quality of the system, and the cost of ownership. For the organization, PLM leads to optimization of its commercial position and its industrial loads, usually highlighting the identical functionality leading to standardization. These relationships are shown in Figure 8.3.

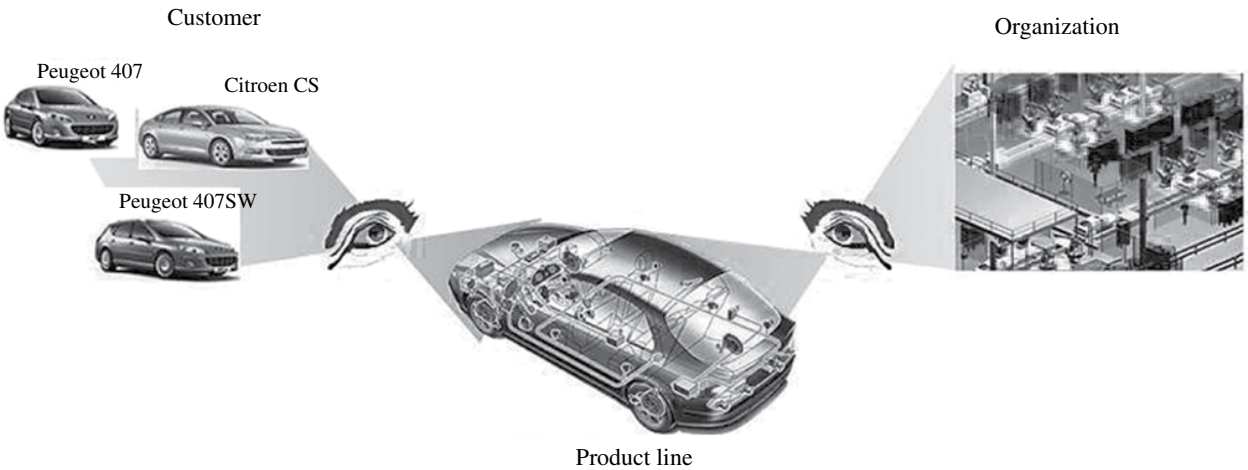


FIGURE 8.3 Product line viewpoints. Reprinted with permission from Alain Le Put. All other rights reserved.

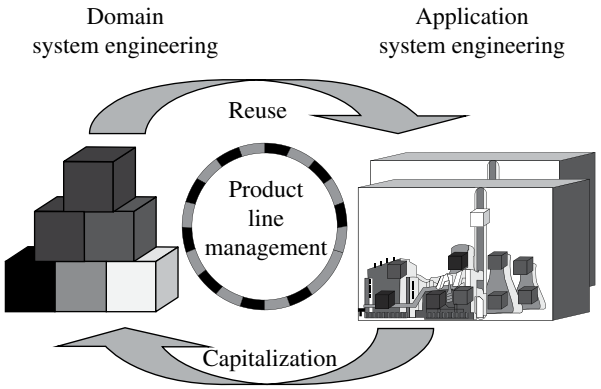


FIGURE 8.4 Capitalization and reuse in a product line. Reprinted with permission from Alain Le Put. All other rights reserved.

When implementing PLM, both the domain and the application SE processes must change as shown in Figure 8.4. The domain products (or generic products) address the product line requirements and are the results of domain SE. The generic products produced by the domain SE activities are capitalized from the application artifacts (e.g., generic requirements, generic architecture, and generic tests). These artifacts may be common or variable. The application products are the results of application SE. These products are the result of the instantiation of the generic products (i.e., reuse).

8.3.1 Product Line Scoping

Product line scoping defines the main features of the product line, which provide sufficient reuse potential to justify the setting up or the evolution of a product line organization. Product line scoping is mainly based on:

- The market analysis: target the perimeter of the concerned product line and the main external features of the product that provide enough commercial potential.
- The SE process assessment: process maturity, weaknesses source of errors, and repetitive work without greater added value. (For example, it is even possible that some teams already have a process akin to the product lines.)
- The products analysis: product tree and the presence of numerous common artifacts.
- The industrial process analysis: production and maintenance. (For example, the objective to be able to produce in several plants can be source of variability.)
- The acquisition strategies analysis: the desire to diversify suppliers and the need to integrate different system elements (in size, performance, etc.).
- The technology analysis: evaluation of technology readiness levels, design maturity, and domain applicability.

*8.3.1.1 **Return on Investment*** Developing a first system in a product line is an investment. The initial development is more expensive and longer than for a single-system development. But the following systems developed in the same product line are less expensive and may be delivered earlier (i.e., reduced time to market). In either case, the ROI must be evaluated, measured and managed.

As illustrated in Figure 8.5 three projects are developed either in a single-system engineering classical approach (top part) or in a product line SE approach (middle part). The ROI (bottom part) is initially negative (e.g., investment phase) and then positive (benefits of the product line approach). In this example, the break-even point is reached for the third project.

8.4 APPLICATION OF SYSTEMS ENGINEERING FOR SERVICES

This section introduces the concept of service SE, where SE methodologies are adapted to include a disciplined, systemic, and service-oriented, customer-centric approach among different stakeholders and resources for near-real-time value cocreation and service delivery. The twenty-first-century technology-intensive global services economy can be characterized as "information driven, customer-centric, e-oriented, and productivity focused." It requires transdisciplinary collaborations between society, science, enterprises, and engineering (Chang, 2010). Several researchers and businesses are utilizing a socioeconomic and technological perspective to investigate end-user (customer) interactions with enterprises by developing formal methodologies for value cocreation and productivity improvements. These methodologies have evolved into service SE, which mandates a disciplined and systemic approach (service

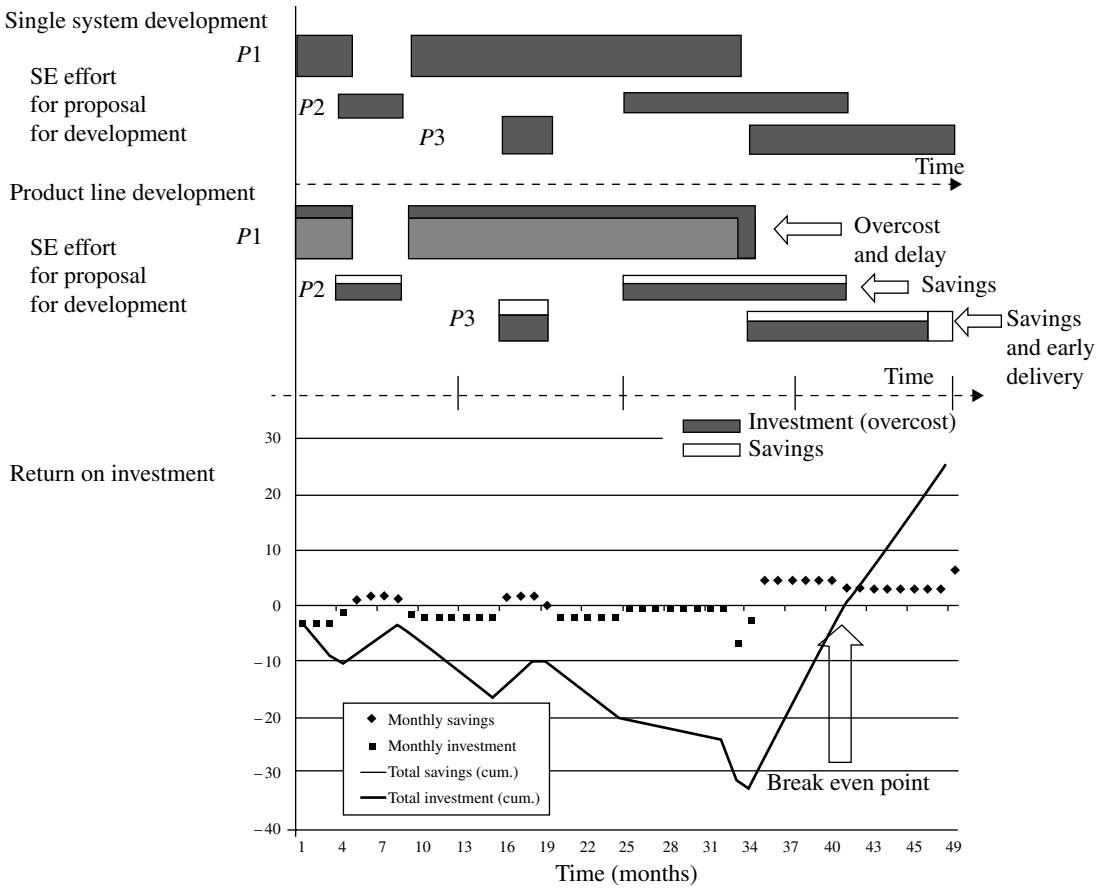


FIGURE 8.5 Product line return on investment. Reprinted with permission from Alain Le Put. All other rights reserved.

oriented, customer-centric) among different stakeholders and resources in the design and delivery of the service to help customize and personalize service transactions to meet particular customer needs (Hipel et al., 2007; Maglio and Spohrer, 2008; Pineda et al., 2014; Tien and Berg, 2003; Vargo and Akaka, 2009).

Service systems can be viewed as an SoS, where individual, heterogeneous, functional systems are linked together to realize new features/functionalities of a meta-system and to improve robustness, lower cost, and increase reliability. For service systems, understanding the integration needs among loosely coupled systems and system entities along with the information flows required for both governance and operations, administration, maintenance, and provisioning (OAM&P) of the service presents major challenges in the definition, design, and implementation of services (Domingue et al., 2009; Maier, 1998). Cloutier et al. (2009) presented the importance of Network-Centric Systems (NCS) for dynamically binding different system entities in engineered systems rapidly to realize adaptive SoS that, in the case of service systems, are capable of knowledge emergence and real-time behavior emergence for service discovery and delivery.

The typical industry example given of this progression toward services is the International Business Machines (IBM), which still produces some hardware but view their business as overwhelmingly service oriented wherein hardware plays only an incidental role in their business solutions services. Furthermore, Apple, Amazon, Facebook, Twitter, eBay, Google, and other application service providers have provided the integrated access to people, media, services, and things to enable new styles of societal and economic interactions

at unprecedented scales, flexibility, and quality to allow for mass collaboration and value cocreation. Several researchers have concluded that even manufacturing firms are more willing to produce "results," rather than solely products as specific artifacts by collaborating with end users that can potentially help define such results (Cook, 2004; Wild et al., 2007).

As the world becomes more widely interconnected and people become better educated, the services networks (created by the interaction of the system entities) will be accessible from anywhere, at any time, by anyone with the proper access rights.

8.4.1 Fundamentals of Service

Services are activities that cause a transformation of the state of an entity (a person, product, business, region, or nation) by mutually agreed terms between the service provider and the customer. Individual services are relatively simple, although they may require customization and significant backstage support (e.g., knowledge management, logistics, decision analysis, forecasting) to assure quality and timely delivery. A *service system* enables a service and/or set of services to be accessible to the customer (individual or enterprise) where stakeholders interact to create a particular service value chain to be developed and delivered with a specific objective (Spohrer and Maglio, 2010). In service systems practice, the *service value chain* is described in terms of links among the system entities connected via the NCS. A value proposition can be viewed as a request from one service system to another to run an algorithm (the value proposition) from the perspectives of multiple stakeholders according to culturally determined value principles (Spohrer, 2011). Spath and Fahnrich (2007) defined a service metamodel comprising nine types of system entities and their corresponding attributes as shown in Table 8.2.

Thus, a service or service offering is created by the relationships among service system entities (including information flows) through business processes into strategic capabilities that consistently provide superior value to the customer. From an SE perspective, participating system entities dynamically configure four types of resources: people, technology/environment infrastructure, organizations/institutions, and shared information/symbolic knowledge in the service delivery process.

Systems are part of other systems that are often expressed by *system hierarchies* (Skyttner, 2006) to create a multilevel hierarchy. Thus, the service system is composed of service system entities that interact through processes defined by governance and management rules to create different types of outcomes in the context of stakeholders with the purpose of providing improved customer interaction and value cocreation. This concept can be extended to create the service system hierarchy shown in Figure 8.6.

The *fundamental attributes* of a service system include togetherness, structure, behavior, and emergence. As mentioned earlier, today's global economy is very competitive, and the service system's trajectory should be well controlled as time goes by (Qiu, 2009) since services are "real time in nature and are consumed at the time they are co-produced" (Tien and Berg, 2003), that is, during service transactions. The service system should evolve and adapt to the conditions within the

TABLE 8.2 Attributes of system entities

Entity type	Attributes
Customer	Features, attitudes, preferences, requirements
Goals	Business, service, customer
Inputs	Physical, information, knowledge, constraints
Outputs	Physical, information, knowledge, waste, customer satisfaction
Processes	Service provision, service delivery, service operations, service support, customer relationships, planning and control
Human enablers	Service providers, support providers, management, owner organization, customer
Physical enablers	Enterprise, organizations, buildings, equipment, enabling technologies at customer premises (e.g., desktop 3D printers), furnishings, location, etc.
Informatics enablers	Information; knowledge; methods, processes, and tools (MPTs); decision support; skill acquisition
Environment	Political, economic, social, technological, environmental factors

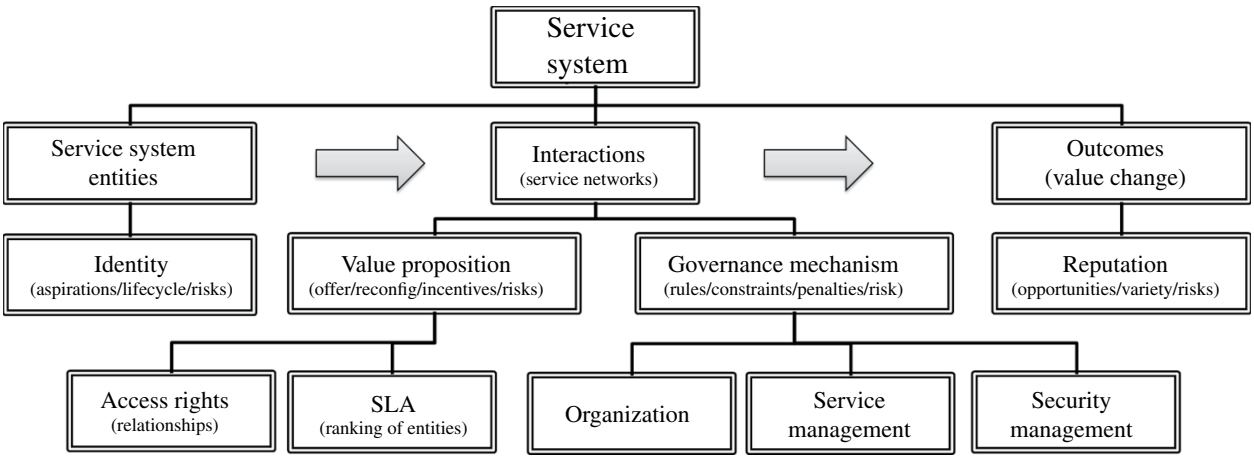


FIGURE 8.6 Service system conceptual framework. Reprinted with permission from Dr. James C. Spohrer. All other rights reserved.

business space in a manner to ensure that the customized service behaves as expected. This adaptive behavior of service system implies that its design must be truly transdisciplinary and must include methodologies from social science (e.g., sociology, psychology, and philosophy) to natural science (mathematics, biology, etc.) to management (e.g., organization, economics, and entrepreneurship) (Hipel et al., 2007).

8.4.2 Properties of Services

Services not only involve the interaction between the service provider and the consumer to produce value but have other intangible attributes like quality of service (e.g., ambulance service availability and response time to an emergency request). The demand for service may have loads dependent on time of day, day of week, season, or unexpected needs (e.g., natural disasters, product promotion campaigns, holiday season, etc.), and services are rendered at the time they are requested. Thus, the design and operations of service systems "is all about finding the appropriate balance between the resources devoted to the systems and the demands placed on the system, so that the quality of service to the customer is as good as possible" (Daskin, 2010).

A *service-level agreement* (SLA) represents the negotiated service-level requirements of the customer and establishes valid and reliable service performance measures to ensure that service providers meet and maintain the prescribed quality of service, user-perceived performance, and degree of satisfaction of the user. The service system's SLAs are then the composition of these categories evaluated on a systemic level to ensure consistency, equity, and sustainability of the service (Spohrer, 2011; Theilmann and Baresi, 2009; Tien and Berg, 2003).

The twenty-first century is witnessing accelerated technology development and global mass collaboration as an established mode of operation. Value cocreation is achieved as loosely entangled actors or entities come together in unprecedented ways to meet mutual and broader market requirements.

This transformation will radically reshape the nature of work, the boundaries of the enterprise, and the responsibilities of business leaders (McAfee, 2009). Spohrer (2011) has captured this trend in service by categorizing the different service sectors into three types of service systems:

- *Systems that focus on flow of things*: transportation and supply chain, water and waste recycling, food and products, energy and electric grid, information, and cloud.
- *Systems that focus on human activities and development*: buildings and construction, retail, hospitality/media, entertainment, banking and finance, business consulting, healthcare, family life, education, work life/jobs, and entrepreneurship.
- *Systems that focus on governing* (city, state, nation): the classification helps in identifying different objectives and constraints for the design and operations of the service system (e.g., strategic policies under

limited budget, education, strategic readiness for quick response, national defense, maximizing profit while minimizing cost) and determining the overlap and synergies required among different service entities and required science disciplines.

8.4.3 Scope of Service SE

Current enterprises must plan, develop, and manage the enhancements of their infrastructure, products, and services, including marketing strategies to include product and service offerings based on new, unexplored, or unforeseen customer needs with clearly differentiated value propositions. Taking the service SE approach is imperative for the service-oriented, customer-centric, holistic view to select and combine service system entities to define and discover relationships among service system entities to plan, design, adapt, or self-adapt to cocreate value. Major challenges faced by service SE include the dynamic nature of service systems evolving and adapting to constantly changing operations and/or business environments and the need to overcome silos of knowledge. Interoperability of service system entities through interface agreements must be at the forefront of the service SE design process for the harmonization of OAM&P procedures of the individual service system entities (Pineda, 2010). In addition, service systems require open collaboration among all stakeholders, but recent research on mental models of multidisciplinary teams shows integration and collaboration into cohesive teams has proven to be a major challenge (Carpenter et al., 2010).

Interoperability among the different service system entities becomes highly relevant in service SE since the constituent entities are designed according to stakeholder needs; the entity is usually managed and operated to satisfy its own objectives independently of other system entities. The objectives of individual service system entities may not necessarily converge with the overall objectives of the service system. Thus, the *service system design process* (SSDP) adapts traditional SE life cycle best practices, as illustrated by Luzeaux and Ruault (2010), Lin and Hsieh (2011), Lefever (2005), and the Office of Government Commerce (2009).

Another important role of service SE is the *management of the service design process* whose main focus is to provide the planning, organizational structure, collaboration environment, and program controls required to ensure that stakeholder's needs are met from an end-to-end customer perspective. The service design management process aligns business objectives and business operational plans with end-to-end service objectives including customer management plans, service management and operations plans, and operations technical plans (Hipel et al., 2007; Pineda et al., 2014).

8.4.4 Value of Service SE

Service SE brings a customer focus to promote service excellence and to facilitate service innovation through the use of emerging technologies to propose creation of new service systems and value cocreation. Service SE uses disciplined approaches to minimize risk by coordinating/orchestrating social aspects, governance (including security), environmental, human behavior, business, customer care, service management, operations, and technology development processes. Service systems engineers must play the role of an integrator, considering the interface requirements for the interoperability of service system entities—not only for technical integration but also for the processes and organization required for optimal customer experience during service operations. The service design definition process includes the definition of methods, processes, and procedures necessary to monitor and track service requirements verification and validation as they relate to the OAM&P procedures of the whole service system and its entities. These procedures ensure that failures by any entity are detected and do not propagate and disturb the operations of the service (Luzeaux and Ruault, 2010).

The world's economies continue to move toward the creation and delivery of more innovative services. To best prepare tomorrow's leaders, new disciplines are needed that include and ingrain different skills and create the knowledge to support such global services. Service systems engineers fit the T-shaped model of professionals (Maglio and Spohrer, 2008) who must have a deeply developed specialty area as well as a broad set of skills and capabilities in addition to the service system management and engineering skills, as summarized by Chang (2010).

8.5 APPLICATION OF SYSTEMS ENGINEERING FOR ENTERPRISES

This section illustrates the applications of SE principles in enterprise SE for the planning, design, improvement, and operation of an enterprise in order to transform and continuously improve the enterprise to survive in a

globally competitive environment. Enterprise SE is the application of SE principles, concepts, and methods to the planning, design, improvement, and operation of an enterprise. Enterprise SE is an emerging discipline that focuses on frameworks, tools, and problem-solving approaches for dealing with the inherent complexities of the enterprise. Furthermore, enterprise SE addresses more than just solving problems; it also deals with the exploitation of opportunities for better ways to achieve the enterprise goals. A good overall description of enterprise SE is provided in the book by Rebovich and White (2011). For more detailed information on this topic, please see the enterprise SE articles in Part 4 of SEBoK (2014).

8.5.1 Enterprise

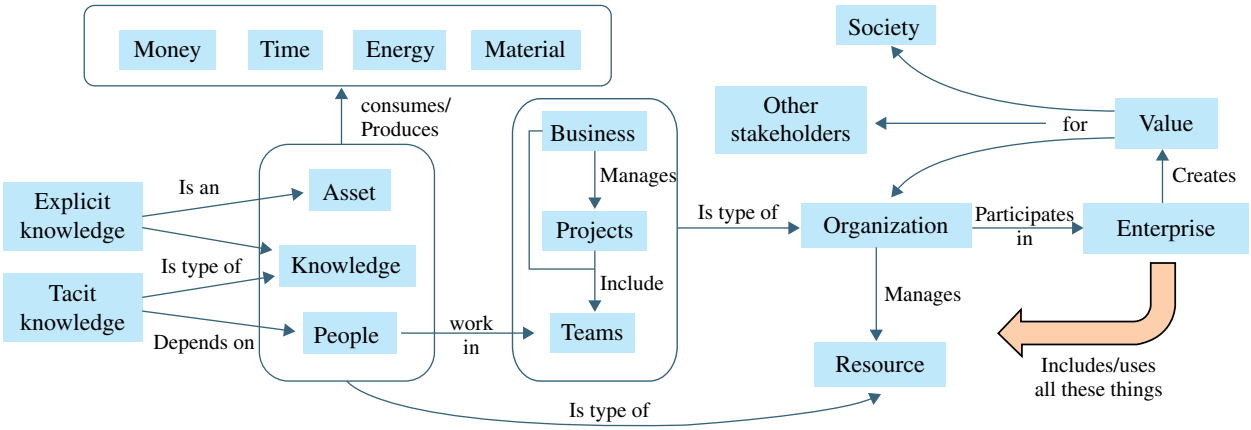
An enterprise consists of a purposeful combination (e.g., a network) of interdependent resources (e.g., people, processes, organizations, supporting technologies, and funding) that interact with each other to coordinate functions, share information, allocate funding, create workflows, and make decisions and their environment(s) to achieve business and operational goals through a complex web of interactions distributed across geography and time (Rebovich and White, 2011).

An enterprise must do two things: (1) develop things within the enterprise to serve as either external offerings or as internal mechanisms to enable achievement of enterprise operations and (2) transform the enterprise itself so that it can most effectively and efficiently perform its operations and survive in its competitive and constrained environment.

It is worth noting that an enterprise is not equivalent to an "organization" according to the definition earlier. This is a frequent misuse of the term enterprise. Figure 8.7 shows that an enterprise includes not only the organizations that participate in it but also people, knowledge, and other assets such as processes, principles, policies, practices, doctrine, theories, beliefs, facilities, land, intellectual property, and so on.

Giachetti (2010) distinguishes between enterprise and organization by saying that an organization is a view of the enterprise. The organization view defines the structure and relationships of the organizational units, people, and other actors in an enterprise. Using this definition, we would say that all enterprises have some type of organization, whether formal, informal, hierarchical, or self-organizing network.

To enable more efficient and effective enterprise transformation, the enterprise needs to be looked at "as a system," rather than merely as a collection of functions connected



Notes:
1. All entities shown are decomposable, except people. For example, a business can have sub businesses, a project can have subprojects, a resource can have sub resources, an enterprise can have sub enterprises.
2. All entities have ether names. For example, a program can be a project comprising server all subprojects. (Often called merely projects). Business can be an agency, team can be group, value can be utility, etc.
3. There is no attempt to be prescriptive in the names chosen for this diagram. The main goal of this is to show how this chapter uses these terms and how they are related to each other in a conceptual manner.

FIGURE 8.7 Organizations manage resources to create enterprise value. From SEBoK (2014). Reprinted with permission from BKCASE Editorial Board. All other rights reserved.

solely by information systems and shared facilities (Rouse, 2009). While a system perspective is required for dealing with the enterprise, this is rarely the task or responsibility of people who call themselves systems engineers.

8.5.2 Creating Value

The primary purpose of an enterprise is to create value for society, for other stakeholders, and for the organizations that participate in that enterprise. This is illustrated in Figure 8.7, which shows all the key elements that contribute to this value creation process.

There are three types of organizations of interest to an enterprise: businesses, projects, and teams. A typical business participates in multiple enterprises through its portfolio of projects. Large SE projects can be enterprises in their own right, with participation by many different businesses, and may be organized as a number of subprojects.

8.5.3 Capabilities in the Enterprise

The enterprise acquires or develops systems or individual elements of a system. The enterprise can also create, supply, use, and operate systems or system elements. Since there could possibly be several organizations involved in this enterprise venture, each organization could be responsible for particular systems or perhaps for certain kinds of elements. Each organization brings their own organizational capability with them, and the unique combination of these organizations leads to the overall operational capability of the whole enterprise. These concepts are also illustrated in Figure 8.7.

The word "capability" is used in SE in the sense of "the ability to do something useful under a particular set of conditions." This section discusses three different kinds of capabilities: organizational capability, system capability, and operational capability. It uses the word "competence" to refer to the ability of people relative to the SE task. Individual competence (sometimes called "competency") contributes to, but is not the sole determinant of, organizational capability. This competence is translated to organizational capabilities through the work practices that are adopted by the organizations. New systems (with new or enhanced system capabilities) are developed to enhance enterprise operational capability in response to stakeholder's concerns about a problem situation.

As shown in Figure 8.8, operational capabilities provide operational services that are enabled by system

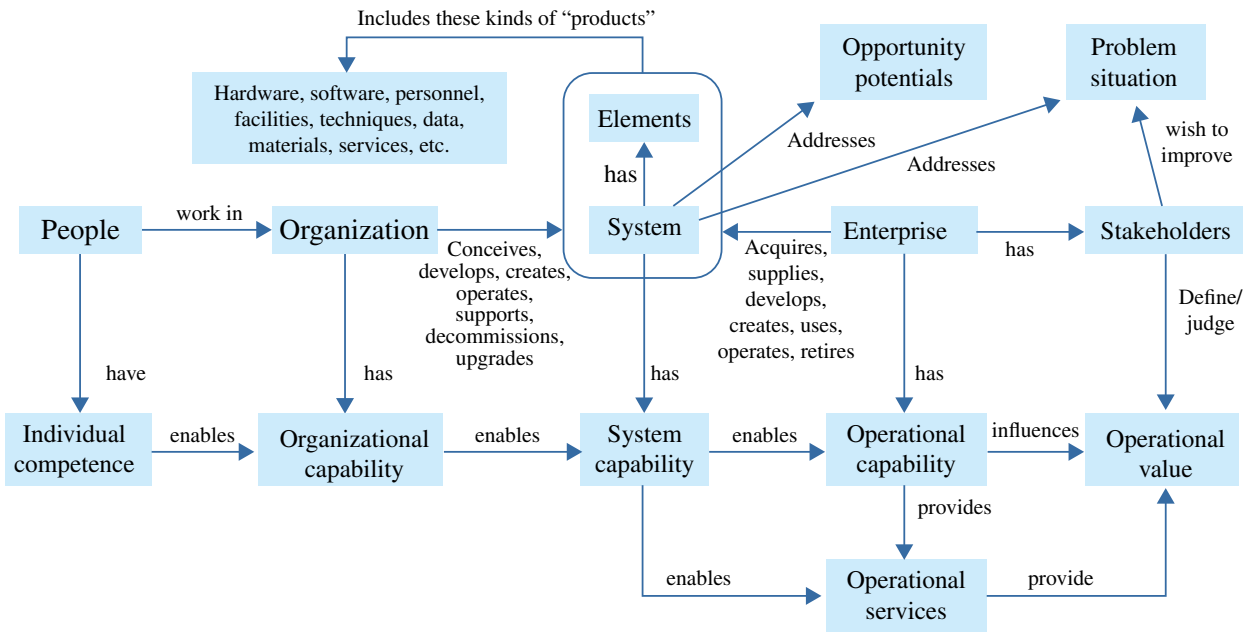


FIGURE 8.8 Individual competence leads to organizational, system, and operational capability. From SEBoK (2014). Reprinted with permission from BKCASE Editorial Board. All other rights reserved.

capabilities. These system capabilities are inherent in the system that is conceived, developed, created, and/or operated by an enterprise. Enterprise SE concentrates its efforts on maximizing operational value for various stakeholders, some of whom may be interested in the improvement of some problem situation.

Enterprise SE, however, addresses more than just solving problems; it also deals with the exploitation of opportunities for better ways to achieve the enterprise goals. These opportunities might involve lowering of operating costs, increasing market share, decreasing deployment risk, reducing time to market, and any number of other enterprise goals. The importance of addressing opportunity potentials should not be underestimated in the execution of enterprise SE practices.

The operational capabilities of an enterprise will have a contribution to operational value (as perceived by the stakeholders). Notice that the organization or enterprise can deal with either the system as a whole or with only one (or a few) of its elements. These elements are not necessarily hard items, like hardware and software, but can also include "soft" items, like people, processes, principles, policies, practices, organizations, doctrines, theories, beliefs, and so on.

8.5.4 Enterprise Transformation

Enterprises are constantly transforming, whether at the individual level (wherein individuals alter their work practices) or at the enterprise level (large-scale planned strategic changes) (Srinivansan, 2010). These changes are a response on the part of the enterprise to evolving opportunities and emerging threats. It is not merely a matter of doing work better, but doing different work, which is often a more important result. Value is created through the execution of business processes. However, not all processes necessarily contribute to overall value (Rouse, 2005). It is important to focus on process and how they contribute to the overall value stream.

Rebovich says there are "new and emerging modes of thought that are increasingly being recognized as essential to successful systems engineering in enterprises" (2006). For example, in addition to the traditional SE process areas, MITRE has included the following process areas in their enterprise SE process to close the gap between enterprise SE and traditional SE (DeRosa, 2005):

- Strategic technical planning
- Enterprise architecture
- Capabilities-based planning analysis
- Technology planning
- Enterprise analysis and assessment

These enterprise SE processes are shown in the context of the entire enterprise in Figure 8.9 (DeRosa, 2006). The enterprise SE processes are shown in the middle with business processes on the left and traditional SE processes on the right. Further information on using SE practices to transform an enterprise is provided in

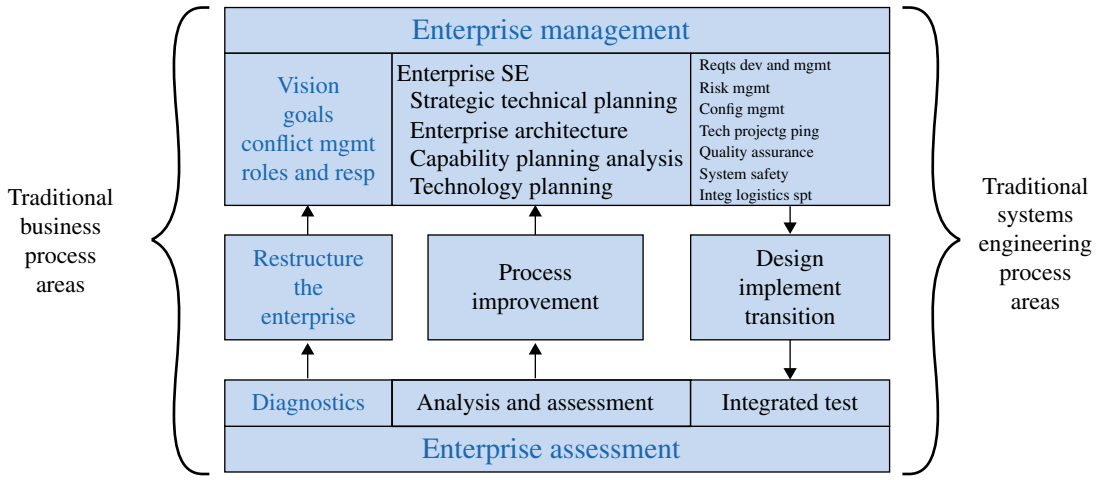


FIGURE 8.9 Enterprise SE process areas in the context of the entire enterprise. Reprinted with permission from the MITRE Corporation. All other rights reserved.

Transforming the Enterprise Using a Systems Approach (Martin, 2011).

8.6 APPLICATION OF SYSTEMS ENGINEERING FOR VERY SMALL AND MICRO ENTERPRISES

VSMEs are defined as organizations that have a small number of employees, many times less than 50 and as few as 1. The contributions of VSMEs to the global economy are well documented. By some estimates, over 98% of economic value is generated globally by enterprises with less than 25 people. In addition, VSMEs contribute to large enterprise systems and SoS and are important and essential to system success. VSME guidance is generic and applicable to SE functions in any domain.

Tailoring of SE processes in any life cycle stage or domain is typical for all projects but is crucial for small enterprises. Of course, as with any project tailoring for processes, the tailoring needs to be risk driven, accounting for things such as the criticality attributes of the project. Due to their small size, VSMEs often find it difficult to apply international standards to their business needs and to justify the application of standards to their business practices. The typical VSME does not have a comprehensive infrastructure, and the few personnel usually are performing multiple roles.

The ISO/IEC/IEEE 29110 standard series defines SE life cycle processes for very small entities (VSEs) derived from ISO/IEC/IEEE 15288 and consistent with this handbook. The ISO/IEC/IEEE 29110 Series (2014) standards define profiles to provide the VSME guidance for use of the standard on noncritical programs. The profiles are oriented on the level of involvement in a product development. A profile is a type of matrix that identifies which elements should be taken from existing standards. A collection of four profiles (entry, basic, intermediate, and advanced) provides a progressive approach to serving most VSMEs.

The entry profile focuses on start-up VSMEs and those working on small projects (i.e., project size of less than six person-months). The basic profile describes the system development practices of a single application by a single project team and with no special risk or situational factors. The intermediate profile is aimed at VSMEs developing multiple projects, while the advanced profile applies to VSMEs that want to grow as independent system development businesses.

For critical programs, such as mission critical or safety critical, this guidance does not apply, since the criticality of the programs would dictate a much greater level of rigor and comprehensive SE.

9

CROSS-CUTTING SYSTEMS ENGINEERING METHODS

The previous chapters provided a serial description of the systems engineering (SE) processes used across the system life cycle. This chapter provides insight into methods that cut across the SE processes, reflecting various aspects of the iterative and recursive nature of SE.

9.1 MODELING AND SIMULATION

Stakeholders of the SE life cycle have used models and simulations for some time both to check their own thinking and to communicate their concepts to others. The benefit is twofold: (i) models and simulations confirm the need for the systems and the anticipated system behaviors before proceeding with the development of an actual system, and (ii) models and simulations present a clear, coherent design to those who will develop, test, deploy, and evolve the system, thereby maximizing productivity and minimizing error. The ability to detect limitations and incompatibilities via system models and simulations early in a project helps avoid higher project cost and schedule overruns later in a project, especially during system operation. The value of modeling and simulation increases with the size, be it physical or complexity, of the system or system of systems (SoS) under development.

Early in the SE life cycle, the objective of modeling and simulation is to obtain information about the system before significant resources are committed to its design, development, construction, verification, or operation. To that end, modeling and simulation helps generate data in the domain of the analyst or reviewer, not available from existing sources, in a manner that is affordable and timely to support decision making. An adequate, accurate, and timely model and simulation informs stakeholders of the implications of their preferences, provides perspective for evaluating alternatives, and builds confidence in the capabilities the system will provide. They also help the development, deployment, and operational staffs comprehend the design requirements, appreciate imposed limits from technology and management, and ensure an adequate degree of sustainability. Finally, adequate, accurate, and timely models and simulations help the organization and its suppliers provide the necessary and sufficient personnel, methods, tools, and infrastructure for system realization.

The long-term benefits of modeling and simulation are commensurate with the gap between the extent, variety, and ambiguity of the problem and the competencies of downstream staffing. A relatively simple model of an intended system may be sufficient for a highly

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.

competent staff, whereas a much more elaborate simulation may be necessary for a less competent staff, especially one faced with producing a novel, large-scale system that is capable of autonomously coping with unpredictable mission situations. Ultimately, the benefit of modeling and simulation is proportional to the stakeholders' perception of the timeliness, trustworthiness, and ease of use and maintenance of the model or simulation. Consequently, the planned resources anticipated to be spent in development, verification, validation, accreditation, operation, and maintenance of the model must be consistent with the expected value of the information obtained through use of the model.

9.1.1 Models versus Simulations

The terms model and simulation are often mistakenly interchanged in discussions. Each term has its own specific meaning. The term "model" has many definitions but generally refers to an abstraction or representation of a system, entity, phenomenon, or process of interest (DoD 5000.59, 2007). The many other definitions of model generally refer to a model as a representation of some entity in the physical world. The representations are intended to describe selected aspects of the entity, such as its geometry, functions, or performance. In the context of SE, a model that represents a system and its environment is of particular importance to the systems engineer who must analyze, specify, design, and verify systems, as well as share information with other stakeholders. Different types of models are used to represent systems for different modeling purposes.

The term "simulation" is the implementation of a model (or models) in a specific environment that allows the model's execution (or use) over time. In general, simulations provide a means for analyzing complex dynamic behavior of systems, software, hardware, people, and physical phenomena.

A computer simulation includes the analytical model that is represented in executable code, the input conditions and other input data, and the computing infrastructure. The computing infrastructure includes the computational engine needed to execute the model, as well as input and output devices. The great variety of approaches to computer simulation is apparent from the choices that the designer of computer simulation must make.

In addition to representing the system and its environment, the simulation must provide efficient computational methods for solving the equations. Simulations may be required to operate in real time, particularly if there is an operator in the loop. Other simulations may be required to operate much faster than real time and perform thousands of simulation runs to provide statistically valid simulation results.

9.1.2 Purpose of Modeling

System models can be used for many purposes. One of the first principles of modeling is to clearly define the purpose of the model. Some of the purposes that models can serve throughout the system life cycle include:

- *Characterizing an existing system:* Many existing systems are poorly documented, and modeling the system can provide a concise way to capture the existing system architecture and design. This information can then be used to facilitate maintaining the system or to assess the system with the goal of improving it. This is analogous to creating an architectural model of an old building with overlays for electrical, plumbing, and structure before proceeding to upgrade it to new standards to withstand earthquakes.
- *Mission and system concept formulation and evaluation:* Models can be applied early in the system life cycle to synthesize and evaluate alternative mission and system concepts. This includes clearly and unambiguously defining the system's mission and the value it is expected to deliver to its beneficiaries. Models can be used to explore a trade space by modeling alternative system designs and assessing the impact of critical system parameters such as weight, speed, accuracy, reliability, and cost on the overall measures of merit. In addition to bounding the system design parameters, models can also be used to validate that the system requirements meet stakeholder needs before proceeding with later life cycle activities such as architecting and design.
- *System architecture design and requirements flowdown:* Models can be used to support architecting system solutions, as well as flow mission and system requirements down to system elements. Different models may be required to address different aspects of the system design and respond to the broad range of system requirements. This may include models that specify functional, interface, performance, and

physical requirements, as well as other nonfunctional requirements such as reliability, maintainability, safety, and security.

- *Support for systems integration and verification:* Models can be used to support integration of the hardware and software elements into a system, as well as to support verification that the system satisfies its requirements. This often involves integrating lower-level hardware and software design models with system-level design models, which verify that system requirements are satisfied. Systems integration and verification may also include replacing selected hardware and design models with actual hardware and software products in order to incrementally verify that system requirements are satisfied. This is referred to as hardware-in-the-loop testing and software-in-the-loop testing. Models can also be used to define the test cases and other aspects of the test program to assist in test planning and execution.
- *Support for training:* Models can be used to simulate various aspects of the system to help train users to interact with the system. Users may be operators, maintainers, or other stakeholders. Models may be a basis for developing a simulator of the system with varying degrees of fidelity to represent user interaction in different usage scenarios.
- *Knowledge capture and system design evolution:* Models can provide an effective means for capturing knowledge about the system and retaining it as part of organizational knowledge. This knowledge, which can be reused and evolved, provides a basis for supporting the evolution of the system, such as changing system requirements in the face of emerging, relevant technologies, new applications, and new customers. Models can also enable the capture of families of products.

Models represent the essential characteristics of the system of interest (SOI), the environment in which the system operates, and the interactions with enabling and interfacing systems and operators. Models and simulations can be used within most system life cycle processes, for example:

- *Business or mission analysis*—Descriptive model of the problematic situation ensures the right problem(s) is being addressed.
- *Requirements (stakeholder and system) definition*—Enables justification of requirements and avoids over-/underspecification.
- *Architecture definition*—Evaluate candidate options against selection criteria and enable active agents to discover the best architecture, including the integration to other systems.
- *Design definition*—Obtain needed design data, adjust parameters for optimization, and update system model fidelity as actual data for system elements become available.
- *Verification and validation*—Simulate the system's environment, evaluate verification and validation data (simulation uses observable data as inputs for computation of critical parameters that are not directly observable), and validate the fidelity of the simulation (false-positives/false-negatives).
- *Operations*—Simulations that reflect actual behavior and simulate operations in advance of execution for planning, validation, and operator training.

9.1.3 Model Scope

The model must be scoped to address its intended purpose. In particular, the types of models and associated modeling languages selected must support the specific needs to be met. For example, suppose models are constructed to support an aircraft's development. A system architecture model may describe the interconnection among the airplane parts, a trajectory analysis model may analyze the airplane trajectory, and a fault tree analysis model may assess potential causes of airplane failure. For each type of model, the appropriate breadth, depth, and fidelity should be determined to address the model's intended purpose.

The model breadth reflects the system requirements coverage in terms of the degree to which the model must address the functional, interface, performance, and physical requirements, as well as other nonfunctional requirements. For an airplane functional model, the model breadth may be required to address some or all of the functional requirements to power up, take off, fly, land, power down, and maintain the aircraft's environment.

The model's depth indicates the coverage of system decomposition from the system context down to the system elements. For the air transport SoS example shown in

Figure 2.2, a model's scope may require it to define the system context, ranging from the aircraft, the control tower, and the physical environment down to the navigation system and its system elements, such as the inertial measurement unit, and perhaps down to the lower-level parts of the inertial measurement unit.

The model's fidelity indicates the level of detail the model must represent for any given part of the model. For example, a model that specifies the system interfaces may be fairly abstract and represent only the logical information content, such as aircraft status data; or it may be much more detailed to support higher-fidelity information, such as the encoding of a message in terms of bits, bytes, and signal characteristics. Fidelity can also refer to the precision of a computational model, such as the time step required for a simulation.

9.1.4 Types of Models and Simulations

There are many different types of models and simulations to address different aspects of a system and different types of systems. The specific type of model or simulation selected for a phase of the system's life cycle depends on the intended use and particular characteristics of the system that are of interest and the level of model accuracy required, in other words its "fitness for purpose." Generally, a specific model or simulation focuses on some subset of the total system characteristics, such as timing, process behavior, or various performance measures.

9.1.4.1 Types of Models "The image of the world around us, which we carry in our head, is just a model" (Forrester, 1961). Most systems start as a mental model that is elaborated and translated in several stages to form a final model or simulation product. A model maybe a mental image of selected concepts, and relationships between them, that can be translated to sketches, textual specifications, graphics/images, mock-ups, scale models, prototypes, or emulations. Often, separate models are prepared for distinct viewpoints, such as functional, performance, reliability, survivability, operational availability, and cost.

It is useful to classify models to assist in selecting the right kind of model for the intended purpose and scope. Models can be classified in many different ways. The model taxonomy shown in Figure 9.1 is one such taxonomy and provides a useful classification for one instance as an illustration and not necessarily providing an exhaustive set of model classes; other classes may exist:

- *Physical mock-ups*—A model that represents an actual system, such as a model airplane or wind tunnel model, or a more abstract representation, such as a model that is often represented using a computer.
- *Abstract models*—An abstract model can have many different expressions to represent a system, entity, phenomena, or process, which vary in degrees of formalism. Therefore, the initial classification of models that distinguishes between informal models and formal models is noted, with this guidance focusing on formal models.

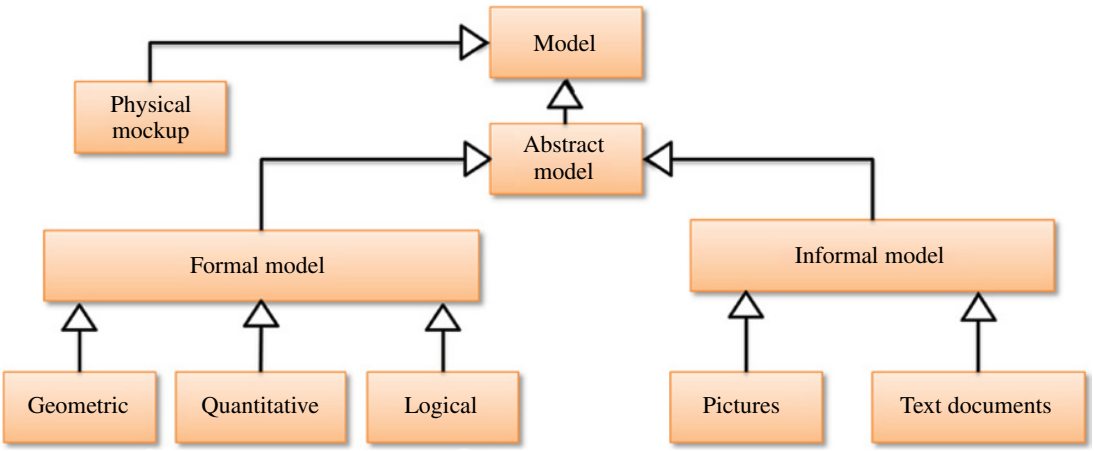


FIGURE 9.1 Sample model taxonomy. Reprinted with permission from Sandy Friedenthal. All other rights reserved.

- *Informal models*—One can represent a system using a simple drawing tool or with words. However, unless there is clear agreement on the meaning of the terms in the less formal representations, there is a potential lack of precision and the possibility of ambiguity in the representation. While such informal representations can be useful, a model must meet certain expectations for it to be considered within the scope of modeling and simulation for SE.
- *Formal models*—Formal models can be further classified as geometric, quantitative (i.e., mathematical), and/or logical models. Geometric model represents the geometric or spatial relationships of the system or entity. Quantitative models represent quantitative relationships (e.g., mathematical equations) about the system or entity that yield numerical results. Logical models, also referred to as conceptual models, represent logical relationships about the system such as a whole–part relationship, an interconnection relationship between parts, or a precedence relationship between activities, to name a few. The logical models are often depicted in graphs (nodes and arcs) or tables.

The following example illustrates the above taxonomy. An aircraft may be represented by a three-dimensional geometric model that specifies the detailed geometry of the aircraft. The aircraft may also be represented by a quantitative model that represents its possible flight trajectories in terms of its acceleration, speed, position, and orientation. The aircraft may be further represented by a logical model that describes the source and destination of signals across the aircraft or potential causes of airplane failure, such as how an engine failure can result in a loss of power and cause the airplane to lose altitude. It is apparent that many different models may be used to represent a SOI.

It should be noted that the semantics for each kind of formal model described above, including the geometric, quantitative, and logical models, can be defined using a mathematical formalism.

A system model is used to represent a system and its environment. A system model may comprise multiple views of the system to support planning, requirements, architecture, design, analysis, verification, and validation. System models can include a combination of geometric, quantitative, and logical models. They often span several modeling domains such as different systems (e.g., thermal, power), different technology domains (e.g., hardware, software), and different characteristics (e.g., physical, performance). Each of these models must be integrated to ensure a consistent and cohesive system representation. As such, the system model must enable representation of general-purpose system modeling concepts such as behavior and structure that can be shared across modeling domains.

A. Wayne Wymore is credited with one of the early efforts to formally define a system model using a mathematical framework in *A Mathematical Theory of Systems Engineering: The Elements* (1967). Wymore established a rigorous mathematical framework for designing systems in a model-based context.

Some examples of system models may include the following (from ISO/IEC/IEEE 15288):

- *A functional model* that captures the system functions and their functional interfaces
- *A behavioral model* that captures the overall behavior of the system functions
- *A temporal model* that captures the timing-related aspects of the architecture
- *A structural model* that captures the system elements and their physical interfaces
- *A mass model* that captures the mass-related aspects of the system
- *A layout model* that captures the absolute and relational spatial placements of the system elements
- *A network model* that captures the flow of resources among the applicable system functions or elements

9.1.4.2 Types of Simulations Simulations can be described under one or more of the following types:

- *Physical simulations* utilize physical models and aim to replicate a relatively small number of system attributes to a high degree of accuracy (fidelity). Often, such simulations require physical models of specific environmental attributes with similar levels of fidelity. Such simulations are often costly to construct and the limited number of system and environment attributes restricts the range of questions that can be answered. This kind of simulation is used when cheaper computer-based simulations cannot be constructed to answer questions. Examples of physical simulations are wind tunnels tests,

environmental tests, and mock-ups that elucidate manufacturing processes.

- *Computer-based simulations* can be divided into subtypes based on models of computation (MoC), for example, discrete event, continuous time solving, or finite element. Each requires the mathematical model to conform to a certain structure, and some can be combined to create a hybrid MoC. Where stochastic processes are being modeled or there is uncertainty in system inputs, Monte Carlo simulations can be performed in order to perform a statistical analysis on output of many simulation runs. There is a trend in simulation environments to separate the execution architecture (that in effect implements the MoC algorithm) and the models of the systems of interest, with the latter being implemented in a modular form. Doing this helps deal with complex models and improves the possibility of reusing models in different simulations. Computer-based simulations can be made to cover a broad scope of system attributes and indeed can become quite complex in their own right by including models of many types of systems interacting in many different ways. When the level of complexity becomes such that the expertise necessary to create fit-for-purpose models for different parts of the overall simulation is distributed between many subject matter experts, the construction of such simulations becomes in itself an exercise in SE.
- *Hardware and/or human-in-the-loop simulations* execute in real time and use computer-based models to close the loop on inputs and outputs with a hardware and or human element of the system. Such simulations have a high level of fidelity but can be costly, especially if physical stimulation is required, for example, motion or visual scene generation.

Within the US defense community, it is common to refer to simulations as live, virtual, or constructive, where:

- *Live simulation* refers to live operators operating real systems
- *Virtual simulation* refers to live operators operating simulated systems
- *Constructive simulation* refers to simulated operators operating with simulated systems

The virtual and constructive simulations may also include actual system hardware and software in the loop as well as stimulus from a real system environment.

9.1.5 Developing Models and Simulations

The completed model or simulation can be considered a system or a product in its own right. Therefore, the general steps in the development and application of a model or simulation are closely aligned to the SE processes described within this handbook. Models need to be planned and tracked, just like any other developmental effort.

Executed in parallel and critical to any model and simulation development effort is the verification, validation, and accreditation (VV&A) process that certifies that a model or simulation is acceptable for use for a specific purpose. Given the consequences of using the knowledge gained from the model or simulation, the user of that knowledge must be confident that the knowledge is of sufficient credibility (i.e., it is "fit for purpose"). This implies that the associated risks of employing the model or simulation in the decision process are minimized such that it is perceived that the model and simulation is of more use than not (i.e., the risks of not using the model or simulation are greater than the risks of using the model or simulation). The US DoD Modeling and Simulation Coordination Office has committed significant resources to produce comprehensive guidance on VV&A (M&SCO, 2013).

9.1.6 Model and Simulation Integration

Many different types of models and simulations may be used as part of a model-based approach. A key activity is to facilitate the integration of models and simulations across multiple domains and disciplines. As an example, system models can be used to specify the elements of the system. The logical model of the system architecture may be used to identify and partition the elements of the system and define their interconnection or other relationships among the elements. Quantitative models for performance, physical, and other quality characteristics, such as reliability, may be employed to determine the required values for specific element properties to satisfy the system requirements. An executable system model that represents the interaction of the system elements may be used to validate that the element requirements can satisfy

the system behavioral requirements. The aforementioned models each represent different facets of the same system. The different engineering disciplines for electrical, mechanical, and software each create their own models representing different facets of the same system. The different models must be sufficiently integrated to ensure a cohesive system solution.

To support the integration, the models and simulations must establish semantic interoperability to ensure that a construct in one model has the same meaning as a corresponding construct in another model. A simple example is the name of a particular element that may appear in a higher-level system element model, a reliability model, and electrical design model. This modeling information must be exchanged between modeling tools and consistently represented in the different models.

One approach to semantic interoperability is to use model transformations between different models. Transformations are defined which establish correspondence between the concepts in one model and the concepts in another. In addition to establishing correspondence, the tools must have a means to exchange the model data and share the information. There are multiple means for exchanging data between tools, including file exchange, use of API, and a shared repository.

The use of modeling standards for modeling languages, model transformations, and data exchange is an important enabler of integration across modeling domains.

9.1.7 Model Management

Since the system models and simulations are primary artifacts of the SE effort, their management is of particular importance. The management of models and simulations throughout the system life cycle includes configuration management concerns related to versioning and change control. These are complex processes in their own right, particularly when distributed teams may update different aspects of different pieces. Change management techniques such as branch and merge may be employed along with other integration approaches. Another important aspect of model and simulation management is the ongoing validation. As changes are introduced to the models and simulations, the team needs to ensure they remain a sufficient representation of the system for their intended purpose.

9.1.8 Modeling Standards

Different types of models are needed to support the analysis, specification, design, and verification of systems. Modeling standards play an important role in defining agreed-upon system modeling concepts that can be represented for a particular domain of interest and enable the integration of different types of models across domains of interest.

Standards for system modeling languages can enable cross-discipline, cross-project, and cross-organization communication. This communication offers the potential to reduce the training requirements for practitioners when transitioning from one project to another and enables the reuse of system artifacts within and across projects and organizations. Standard modeling languages also provide a common foundation for advancing the practice of SE, as do other SE standards.

Modeling standards include standards for modeling languages, data exchange between models, and the transformation of one model to another to achieve semantic interoperability. A partial list of representative modeling standards can be found under the modeling standards section of SEBoK (2014).

9.1.9 Modeling Languages

Modeling languages are generally intended to be both human interpretable and computer interpretable and are specified in terms of both syntax and semantics.

The abstract syntax specifies the model constructs and the rules for constructing the model from its constructs. In the case of a natural language like English, the constructs may include types of words such as verbs, nouns, adjectives, and prepositions, and the rules specify how these words can be used together to form proper sentences. The abstract syntax for a mathematical model may specify constructs to define mathematical functions, variables, and their relationship. The abstract syntax for a logical model may also specify constructs to define logical entities and their relationships such as the interconnection relationship between parts or the precedence relationship between actions. A well-formed model conforms to its rules, just as a well-formed sentence must conform to the grammatical rules of the natural language.

The concrete syntax specifies the symbols used to express the model constructs. The natural language, such

as English or German, can be expressed in text or Morse code. A modeling language may be expressed using graphical symbols and/or text statements. For example, a functional flow model may be expressed using graphical symbols consisting of a combination of graphical nodes and arcs annotated with text, while a simulation modeling language may be expressed using a text syntax in a programming language such as Fortran or C.

The semantics of a language define the meaning of the constructs. For example, an English word does not have explicit meaning until the word is defined. A sentence can be grammatically correct, but can still be gibberish if the words are not defined, or be misunderstood if the meaning of the words is ambiguous in the context of their use. The language must give meaning both to the concept of a verb or noun and to the meaning of a specific word that is a verb or noun. Similarly, a modeling construct that is expressed as a symbol, such as a box or arrow on a flowchart, does not have meaning until it is defined. The box and arrow each represent different concepts. These concepts must be defined, and the specific boxes and arrows should also be defined. The definitions can be expressed in natural language or other formalisms. For example, the symbols $\sin(x)$ and $\cos(x)$ represent the sine and cosine function, which are defined precisely in mathematics. If the position of a pendulum is defined in terms of $\sin(\theta)$ and $\cos(\theta)$, the meaning of the pendulum position is understood in terms of these formalisms.

The SysML™ from the OMG has emerged as an important modeling language for systems (OMG, 2013b). Summary descriptions of the SysML diagram types shown in Figure 9.2 are as follows:

- A *package diagram* (pkg) is used to organize the model into packages that contain other model elements. This facilitates model navigation and reuse, as well as access and change control.
- A *requirements diagram* (req) captures text-based requirements. Having requirements within the model enables fine-grained traceability from requirement to requirement and between requirements and design, analysis, and verification elements in the model.
- System structure is represented using block diagrams:
  - A *block definition diagram* (bdd) describes the system hierarchy and classifications of system elements.
  - An *internal block diagram* (ibd) depicts the internal structure of a system in terms of how its parts are interconnected using ports and connectors, describing how the parts within the system are interconnected.
- Behavior is captured in use case, activity, sequence, and state machine diagrams:
  - A *use case diagram* (uc) provides a high-level description of the system functionality in terms

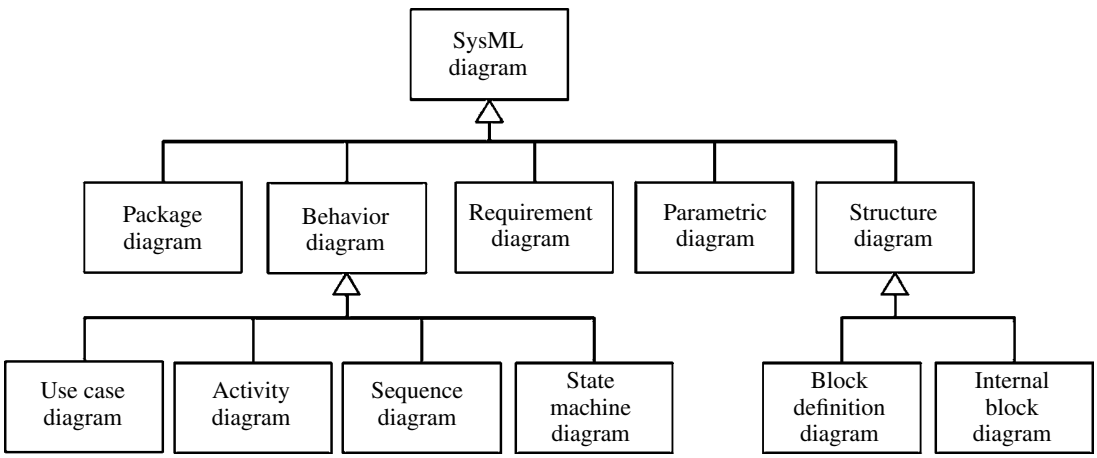


FIGURE 9.2 SysML diagram types. From Friedenthal et al. (2012), Figure 3.2. Reprinted with permission from Sandy Friedenthal. All other rights reserved.

of how users and external systems use the system to achieve their goals.
  - An *activity diagrams* (act) represents the transformation of inputs to outputs through a controlled sequence of actions.
  - A *sequence diagram* (sd) represents interaction in terms of the time-ordered exchange of messages between collaborating parts of a system.
  - A *state machine diagram* (stm) describes the states of a system or its parts; the transitions between the states; the actions that occur within states or upon transition, entry, or exit; and the events that trigger transitions.
- A *parametric diagram* (par) represents constraints on system property values as necessary to support detailed engineering analysis. These constraints may include performance, reliability, and mass properties, among others. SysML™ can be integrated with other engineering analysis models and tools to execute the analysis.

SysML includes an allocation relationship to represent allocation of functions to elements, allocation of logical to physical elements, and other types of allocations. SysML™ is a general-purpose modeling language that is intended to support many different model-based methods, such as structured analysis methods and object-oriented methods. A particular method may require only a subset of the diagrams. For example, a simplified functional analysis method may only require activity diagrams augmented by bdds, ibds, and perhaps requirements diagrams.

For general information on SysML™, along with links to tool vendors, articles, and books, see the official OMG SysML™ website at http://www.omgsysml.org.

9.1.10 Modeling and Simulation Tools

Models and simulations are created by a modeler using modeling and simulation tools. For physical models (e.g. physical mock-ups), the modeling tools may include drills, lathes, and hammers. For abstract models, the modeling tools are typically software programs running on a computer. These programs provide the ability to express modeling constructs using a particular modeling language. A word processor can be viewed as a tool used to build text descriptions using natural language. In a similar way, a modeling tool is used to build models using a modeling language. The tool often provides a tool palette to select symbols and a content area to construct the model from the graphical symbols or other concrete syntax. A modeling tool typically checks the model to evaluate whether it conforms to the rules of the language and enforces such rules to help the modeler create a well-formed model. This is similar to the way a word processor checks the text to see that it conforms to the grammar rules for the natural language.

Some modeling and simulation tools are commercially available products, while others may be created or customized to provide unique modeling solutions. Modeling and simulation tools are often used as part of a broader set of engineering tools, which constitute the system development environment. There is increased emphasis on tool support for standard modeling languages that enable models and modeling information to be interchanged among different tools.

9.1.11 Indicators of Model Quality

The quality of a model should not be confused with the quality of the design that the model represents. For example, one may have a high-quality, computer-aided design model of a chair that accurately represents the design of the chair, yet the design itself may be flawed such that when one sits in the chair, it falls apart. A high-quality model should provide a representation sufficient to assist the design team in assessing the quality of the design and uncovering design issues.

Model quality is often assessed in terms of the adherence of the model to modeling guidelines and the degree to which the model addresses its intended purpose. Typical examples of modeling guidelines include naming conventions, application of appropriate model annotations, proper use of modeling constructs, and applying model reuse considerations. Specific guidelines are different for different types of models. For example, the guidelines for developing a geometric model using a computer-aided design tool may include conventions for defining coordinate systems, dimensioning, and tolerances.

9.1.12 Model and Simulation-Based Metrics

Models and simulations can provide a wealth of information that can be used for both technical and management metrics to assess the modeling and simulation efforts

and, in many cases, the overall SE effort. Different types of models and simulations provide different types of information. In general, models and simulations provide information that enables one to:

- Assess progress
- Estimate effort and cost
- Assess technical quality and risk
- Assess model quality

A model's progress can be assessed in terms of the completeness of the modeling effort relative to the defined scope of the model. Models may also be used to assess progress in terms of the extent to which the requirements have been satisfied by the design or verified through testing. When augmented with productivity metrics, the model can be used to estimate the cost of performing the required SE effort to deliver the system.

Models and simulations can be used to identify critical system parameters and assess technical risks in terms of any uncertainty that lies in those parameters. The models and simulations can also be used to provide additional metrics that are associated with its purpose. For example, when the model's purpose is to support mission and system concept formulation and evaluation, then a key metric may be the number of alternative concepts that are explored over a specified period of time.

9.2 MODEL-BASED SYSTEMS ENGINEERING

This section provides an overview of the model-based SE (MBSE) methodology including a summary of benefits of a model-based approach over a more traditional document-based approach, the purpose and scope of an MBSE approach, references to a survey of MBSE methods used to perform MBSE, and a brief discussion on model management.

9.2.1 MBSE Overview

A number of model and simulation practices have been formalized into SE processes. These processes are the foundation of MBSE. The *INCOSE Systems Engineering Vision 2020* (2007) defines MBSE as "the formalized application of modeling to support system requirements, design, analysis, verification, and validation activities beginning in the conceptual design phase and continuing throughout development and later life cycle phases."

MBSE enhances the ability to capture, analyze, share, and manage the information associated with the specification of a product, resulting in the following benefits:

- *Improved communications* among the development stakeholders (e.g., the customer, program management, systems engineers, hardware and software developers, testers, and specialty engineering disciplines)
- *Increased ability to manage system complexity* by enabling a system model to be viewed from multiple perspectives and to analyze the impact of changes
- *Improved product quality* by providing an unambiguous and precise model of the system that can be evaluated for consistency, correctness, and completeness
- *Enhanced knowledge capture* and reuse of the information by capturing information in more standardized ways and leveraging built-in abstraction mechanisms inherent in model-driven approaches. This in turn can result in reduced cycle time and lower maintenance costs to modify the design
- *Improved ability to teach and learn SE fundamentals* by providing a clear and unambiguous representation of the concepts

MBSE is often contrasted with a traditional document-based approach to SE. In a document-based SE approach, there is often considerable information generated about the system that is contained in documents and other artifacts such as specifications, interface control documents, system description documents, trade studies, analysis reports, and verification plans, procedures, and reports. The information contained within these documents is often difficult to maintain and synchronize, and difficult to assess in terms of its quality (correctness, completeness, and consistency).

In an MBSE approach, much of this information is captured in a system model or set of models. The system model is a primary artifact of the SE process. MBSE formalizes the application of SE through the use of models. The degree to which this information is captured in models and maintained throughout the life cycle depends on the scope of the MBSE effort. Leveraging an MBSE

approach to SE is intended to result in significant improvements in system requirements, architecture, and design quality; lower the risk and cost of system development by surfacing issues early in the system definition; enhance productivity through reuse of system artifacts; and improve communications among the system development team.

9.2.1.1 Survey of MBSE Methodologies In general, a methodology can be defined as the collection of related processes, methods, and tools used to support a specific discipline (Martin, 1996). That more general notion of methodology can be specialized to *MBSE methodology*, which we characterize as the collection of related processes, methods, and tools used to support the discipline of SE in a "model-based" or "model-driven" context (Estefan, 2008).

In 2008, a survey of candidate MBSE methodologies was published under the auspices of an INCOSE technical publication (Estefan, 2008). Six (6) candidate MBSE methodologies were surveyed: INCOSE Object-Oriented Systems Engineering Method (OOSEM), IBM Rational Telelogic Harmony-SE, IBM Rational Unified Process for Systems Engineering (RUP-SE), Vitech MBSE Methodology, JPL State Analysis (SA), and Dori Object-Process Methodology (OPM). Additional information on these methodologies is available on the INCOSE MBSE Initiative Wiki (INCOSE, 2010a).

Two example methods that are included in the SE handbook are the functions-based SE (FBSE) method in Section 9.3 and OOSEM in Section 9.4. Although the functions-based method is not referred to as model based, there are other functions-based methods such as the Vitech MBSE Methodology that are explicitly model based. OOSEM is defined as an end-to-end MBSE method, where the artifacts of the method are modeling artifacts that are managed and controlled throughout the SE process.

9.3 FUNCTIONS-BASED SYSTEMS ENGINEERING METHOD

9.3.1 Introduction

FBSE is an approach to SE that focuses on the functional architecture of the system. A *function* is a characteristic task, action, or activity that must be performed to achieve a desired outcome. A function may be accomplished by one or more system elements comprising equipment (hardware), software, firmware, facilities, personnel, and procedural data.

The objective of FBSE is to create a functional architecture for which system products and processes can be designed and to provide the foundation for defining the system architecture through the allocation of functions and subfunctions to hardware/software, databases, facilities, and operations (e.g., personnel).

FBSE describes what the system will do, not how it will do it. Ideally, this process begins only after all of the system requirements have been fully identified. Often, this will not be possible, and these tasks will have to be done iteratively, with the functional architecture being further defined as the system requirements evolve.

9.3.1.1 Method Overview The FBSE process is iterative, even within a single stage in the system life cycle. The functional architecture begins at the top level as a set of functions that are defined in the applicable requirements document or specification, each with functional, performance, and limiting requirements allocated to it (in the extreme, top-level case, the only function is the system, and all requirements are allocated to it). As shown in Figure 9.3, the next lower level of the functional architecture is developed and evaluated to determine whether further decomposition is required. If it is, then the process is iterated through a series of levels until a functional architecture is complete.

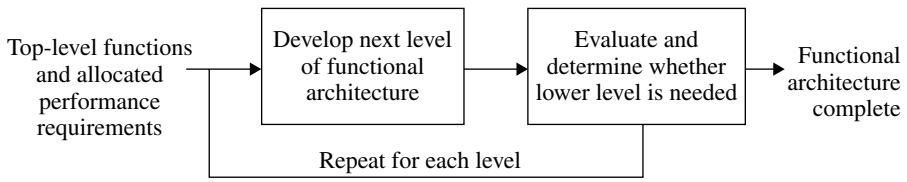


FIGURE 9.3 Functional analysis/allocation process. INCOSE SEH v1 Figure 4.3-1. Usage per the INCOSE Notices page. All other rights reserved.

FBSE should be conducted iteratively:

- To define successively lower-level functions required to satisfy higher-level functional requirements and to define alternative sets of functional requirements
- With requirements definition, to define mission- and environment-driven performance and to determine that higher-level requirements are satisfied
- To flow down performance requirements and design constraints
- With architecture and design, to refine the definition of product and process solutions

At each level of the process, alternative decompositions and allocations may be considered and evaluated for each function and a single version selected. After all of the functions have been identified, then all the internal and external interfaces to the decomposed subfunctions are established. These steps are shown in Figure 9.4.

FBSE examines a defined function to identify all the subfunctions necessary to accomplish that function; all usage modes must be included in the analysis. This activity is conducted to the level of depth needed to support required architecture and design efforts. Identified functional requirements are analyzed to determine the lower-level functions required to accomplish the parent requirement. Every function that must be performed by the system to meet the operational requirements is identified and defined in terms of allocated functional, performance, and other limiting requirements. Each function is then decomposed into subfunctions, and the requirements allocated to the function are each decomposed with it. This process is iterated until the system has been completely decomposed into basic subfunctions, and each subfunction at the lowest level is completely, simply, and uniquely defined by its requirements. In the process, the interfaces between each of the functions and subfunctions are fully defined, as are the interfaces to the external world.

Identified subfunctions are arranged in a functional architecture to show their relationships and interfaces (internal and external). Functional requirements should be arranged in their logical sequence so that lower-level functional requirements are recognized as part of higher-level requirements. Functions should have their input, output, and functional interface requirements (both internal and external) defined and be traceable from beginning to end conditions. Time critical requirements must also be analyzed.

Performance requirements should be successively established, from the highest to lowest level, for each functional requirement and interface. Upper-level performance requirements are then flowed down and allocated to lower-level subfunctions. Timing requirements that are prerequisite for a function or set of functions must be determined and allocated. The resulting set of requirements should be defined in measurable terms and in sufficient detail for use as design criteria. Performance requirements should be traceable from the lowest level of the current functional architecture, through the analysis by which they were allocated, to the higher-level requirement they are intended to support. All of these types of product requirements must also be verified.

Note that while performance requirements may be decomposed and allocated at each level of the functional decomposition, it is sometimes necessary to proceed through multiple levels before allocating the performance requirements. Also, sometimes, it is necessary to develop alternative functional architectures and conduct a trade study to determine a preferred one. With each iteration of

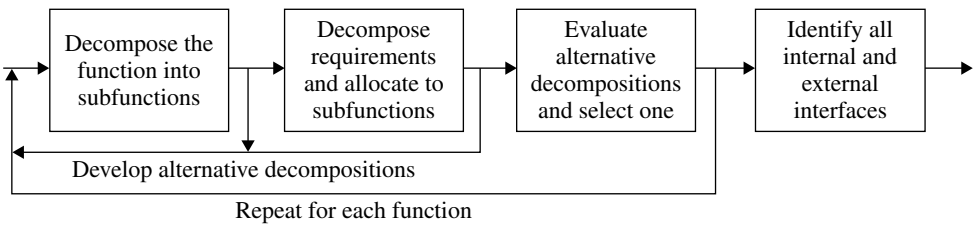


FIGURE 9.4 Alternative functional decomposition evaluation and definition. INCOSE SEH v1 Figure 4.3-2. Usage per the INCOSE Notices page. All other rights reserved.

FBSE, alternative decompositions are evaluated and all interfaces are defined.

The products of FBSE can take various formats depending on the specific stage of the project and on the specific technique used to develop the functional architecture. The following are some key outputs generated from FBSE:

1. *Input–process–output (IPO) diagrams*—Top-level diagram of a data flow that is related to a specific level of system decomposition. This diagram portrays all inputs and outputs of a system but shows no decomposition.
2. *Behavior diagrams*—Describe behavior that specifies system-level stimulus responses using constructs that specify time sequences, concurrencies, conditions, synchronization points, state information, and performance.
3. *Control flow diagrams*—Depict the set of all possible sequences in which operations may be performed by a system or a software program. There are several types of control flow diagrams, including box diagrams, flowcharts, and state transition diagrams.
4. *Data flow diagrams (DFDs)*—Provide an interconnection of each of the behaviors that the system must perform. All inputs to the behavior designator and all outputs that must be generated are identified along with each of the data stores that each must access. Each of the DFDs must be checked to verify consistency with the IPO diagram or higher-level DFD.
5. *Entity relationship (ER) diagrams*—Depict a set of entities (e.g., functions or architecture elements) and the logical relationships between them.
6. *Functional flow block diagrams (FFBDs)*—Relate the inputs and outputs and provide some insight into flow between the system functions.
7. *Integrated definition for functional modeling (IDEF) diagrams*—Show the relationship between functions by sequential input and output flows. Process controls enter the top of each represented function, and lines entering the bottom show the supporting mechanism needed by the function.
8. *Data dictionaries*—Documentation that provides a standard set of definitions of data flows, data elements, files, etc. as an aid to communications across the development organizations.
9. *Models*—Abstractions of relevant characteristics of a system used as a means to understand, communicate, design, and evaluate a system. They are used before the system is built and while it is being verified or in service.
10. *Simulation results*—Output from a simulation of the system that behaves or operates like the SOI when provided a set of controlled inputs.

The objective of the functional decomposition activity is to develop a hierarchy of FFBDs that meet all the functional requirements of the system. Note, however, that this hierarchy is only a portion of the functional architecture. The architecture is not complete until all of the performance and limiting requirements have been appropriately decomposed and allocated to the elements of the hierarchy, as described earlier.

A description of each function in the hierarchy should be developed to include the following:

1. Its place in a network (e.g., FFBD or IDEF0/1 diagrams) characterizing its interrelationship with the other functions at its level
2. The set of functional requirements that have been allocated to it and define what it does
3. Its inputs and outputs, both internal and external

These various outputs characterize the functional architecture. There is no one preferred output that will support this analysis. In many cases, several of these are necessary to understand the functional architecture and the risks that may be inherent in the system architecture. Using more than one of these formats allows for a "check and balance" of the analysis process and will aid in communication across the system design team.

9.3.2 FBSE Tools

Tools that can be used to perform FBSE include:

- Analysis tools
- Modeling and simulation tools
- Prototyping tools
- Requirements traceability tools

9.3.3 FBSE Measures

The following measures can be used to measure the overall process and products of FBSE:

- Number of allocation-related trade studies completed as a percent of the number identified
- Percent of analyses completed
- Number of functions without a requirements allocation
- Number of functions not decomposed
- Number of alternative decompositions
- Number of internal and external interfaces not completely defined
- Depth of the functional hierarchy as a percentage versus the target depth
- Percent of performance requirements that have been allocated at the lowest level of the functional hierarchy

9.4 OBJECT-ORIENTED SYSTEMS ENGINEERING METHOD

9.4.1 Introduction

OOSEM (Estefan, 2008) integrates object-oriented concepts with model-based and traditional SE methods to help architect flexible and extensible systems that accommodate evolving technologies and changing requirements. OOSEM supports the specification, analysis, design, and verification of systems. OOSEM can also facilitate integration with object-oriented software development, hardware development, and verification and validation methods.

Object-oriented SE evolved from work in the mid-1990s at the Software Productivity Consortium in collaboration with Lockheed Martin Corporation. The methodology was applied in part to a large, distributed information system development at Lockheed Martin that included hardware, software, database, and manual procedure elements. The INCOSE Chesapeake Chapter established the OOSEM Working Group in November 2000 to help evolve the methodology further. OOSEM is described in INCOSE and industry papers (Friedenthal, 1998; Lykins et al., 2000) and in *A Practical Guide to SysML: The Systems Modeling Language*, by Friedenthal et al. (2012).

The OOSEM objectives are as follows:

- Capture information throughout the life cycle sufficient to specify, analyze, design, verify, and validate systems
- Integrate MBSE methods with object-oriented software, hardware, and other engineering methods
- Support system-level reuse and design evolution

Figure 9.5 depicts the techniques and concepts that constitute OOSEM. OOSEM incorporates foundational SE practices, object-oriented concepts, and other unique techniques to deal with system complexity. Practices recognized as essential to SE are core tenets of OOSEM. These include requirements analysis, trade studies, and Integrated Product and Process Development (IPPD). See Section 9.7 for more about IPPD, which emphasizes multidisciplinary teamwork in the development process.

Object-oriented concepts that are leveraged in OOSEM include blocks (i.e., classes in UML) and objects, along with the concepts of encapsulation and inheritance. These concepts are supported directly by SysML. Techniques that are unique to OOSEM include parametric flow-down, system/logical decomposition, requirements variation analysis, and several others.

9.4.2 Method Overview

The OOSEM supports a development process as illustrated in Figure 9.6.

This development process includes subprocesses to:

- *Manage system development*—To plan and control the technical effort, including planning, risk management, configuration management, and measurement
- *Define system requirements and design*, including specifying the system requirements, developing the system architecture, and allocating the system requirements to system elements
- *Develop system elements*—To design, implement and test the element, which satisfies the allocated requirements
- *Integrate and test the system*—To integrate the system elements and verify that they satisfy the system requirements, individually and together

This process is consistent with a typical "Vee" process as described in Chapter 3. It can be applied recursively and iteratively at each level of the system hierarchy. For example, if the system hierarchy includes multiple system element levels, the process may be applied at the

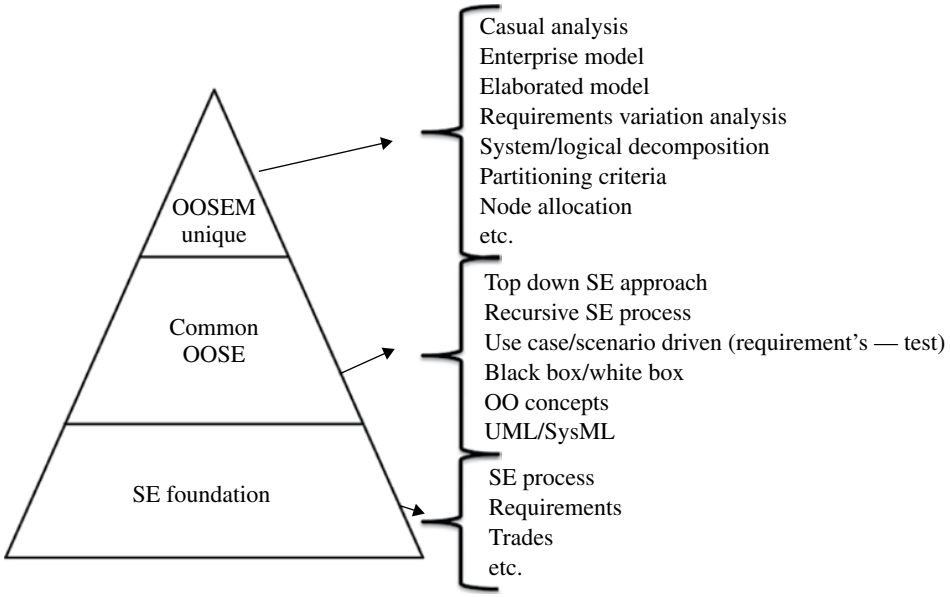


FIGURE 9.5 OOSEM builds on established SE foundations. Reprinted with permission from Howard Lykins. All other rights reserved.

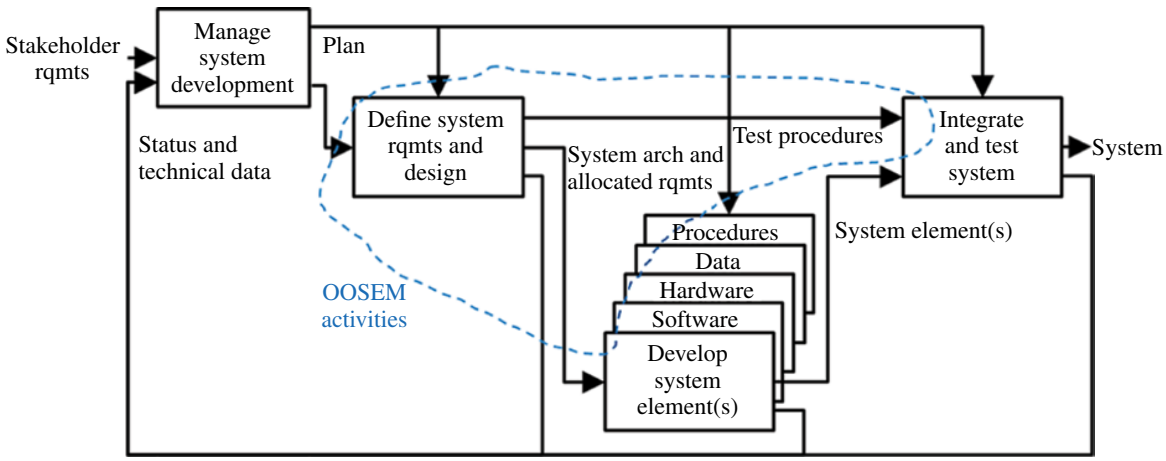


FIGURE 9.6 OOSEM activities in context of the system development process. Reprinted with permission from Howard Lykins. All other rights reserved.

system level to specify the first level of system element requirements. Then the process can be applied again for each system element at the first level to specify requirements for the system elements at the second level, and so forth.

To be effective, OOSEM development activities must be supported by systems engineers applying fundamental tenets of SE, including the use of multidisciplinary teams and disciplined management processes such as planning, risk management, configuration management, and measurement. OOSEM development activities and accompanying process flows are more fully described in *Object-Oriented Systems Engineering Method (OOSEM) Tutorial* (LMCO, 2008) and in *Tutorial Material—Model-Based Systems Engineering Using the OOSEM* (JHUAPL, 2011).

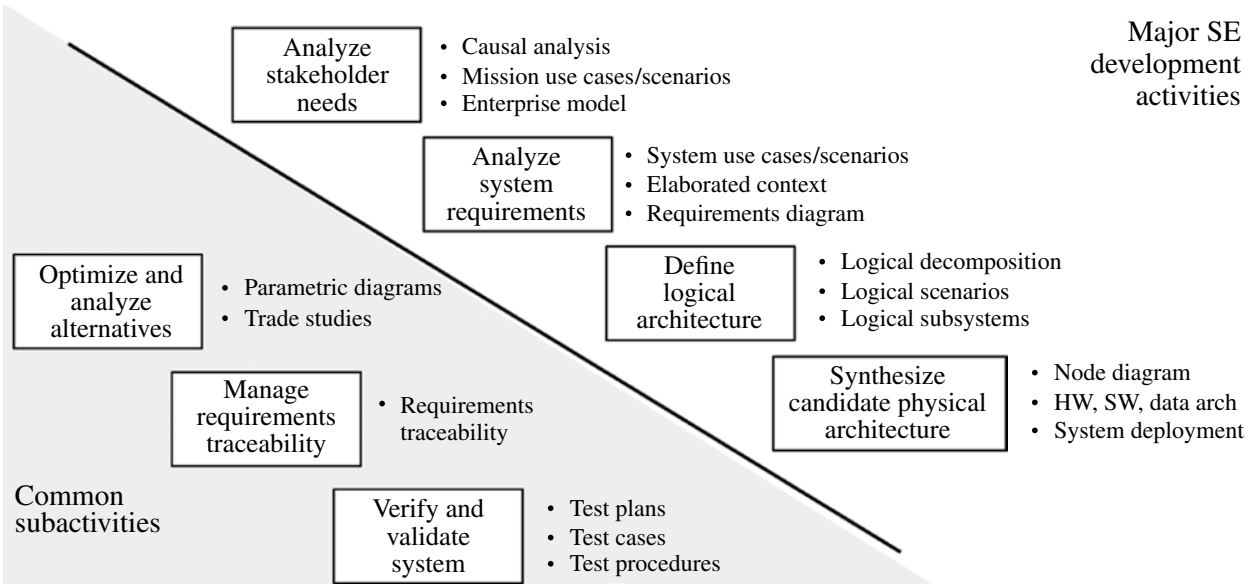


FIGURE 9.7 OOSEM activities and modeling artifacts. Reprinted with permission from Howard Lykins. All other rights reserved.

The system requirements and design process is decomposed into the following OOSEM high-level activities, as depicted in Figure 9.7.

9.4.2.1 Analyze Stakeholder Needs This activity supports analysis of both the "as-is" and the "to-be" enterprise. In OOSEM, an enterprise aggregates the system with other external systems that work together to accomplish the mission. The "as-is" systems and enterprise are captured in sufficient detail to understand their limitations and needed improvements. The limitations of the "as-is" enterprise, as determined through causal analysis techniques, are the basis for deriving the mission requirements for the "to-be" enterprise.

OOSEM specifies the mission requirements for the "to-be" enterprise to reflect customer and other stakeholder needs. The mission requirements include definition of new and improved capabilities to address the limitations identified in the causal analysis. The capabilities for the "to-be" enterprise are represented as use cases with corresponding MOEs. The "to-be" enterprise sets the context for the system or system(s) to be developed.

The modeling artifacts that support analysis, including use cases, scenario analysis, causal analysis, and context diagrams, can be captured in the customer's "as-is" and/or "to-be" concept document(s).

9.4.2.2 Analyze System Requirements This activity specifies the system requirements that support the mission requirements. The system is modeled as a black box that interacts with the external systems and users. The use cases and scenarios reflect the operational concept for how the system is used to support the mission. The scenarios are modeled using activity diagrams with swim lanes that represent the black box system, users, and external systems. The scenarios for each use case are used to derive functional, interface, data, and performance requirements for the black box system. The requirements management database is updated to trace system requirement to use cases and associated mission requirements.

Requirements may change as development proceeds. For example, a system's external interfaces may change, or its performance requirements may increase. Requirements variation is evaluated in terms of the probability that a requirement will change and the impact of such change on the mission. These factors are included in the risk assessments and later used to determine how to design the system to accommodate potential requirements changes.

9.4.2.3 Define Logical Architecture This activity includes decomposing and partitioning the system into logical elements, for example, a user interface that will

be realized by a web browser or an environmental monitor that will be realized by an infrared sensor. The elements interact to satisfy system requirements and capture system functionality. Having a logical architecture/design mitigates the impact of requirements and technology changes on system design.

OOSEM provides guidelines for decomposing the system into its logical elements. Functions for logical elements are derived from logical scenarios to support black box system functions. Logical element functionality and data may be repartitioned based on other criteria, such as cohesion, coupling, design for change, reliability, and performance.

9.4.2.4 Synthesize Candidate Physical Architectures The physical architecture describes relationships among physical system elements, including hardware, software, data, people, and procedures. Logical elements are allocated to physical elements. For distributed systems, OOSEM includes guidance for distributing the physical elements across the system nodes to address concerns, such as performance, reliability, and security. The system architecture continues to be refined to address concerns associated with the software, hardware, and data architectures. Requirements for each physical element are traced to the system requirements and maintained in the requirements management database.

9.4.2.5 Optimize and Evaluate Alternatives This activity is invoked throughout all other OOSEM activities to optimize the candidate architectures and conduct trade studies to select an architecture. Parametric models for modeling performance, reliability, availability, life cycle cost, human, and other specialty engineering concerns are used to analyze and optimize the candidate architectures to the level needed to compare the alternatives. Criteria and weighting factors used to perform the trade studies are traced to the system requirements and MOEs. TPMs are monitored, and potential risks are identified.

9.4.2.6 Manage Requirements Traceability This activity is performed throughout the other OOSEM activities to ensure traceability between requirements, architecture, design, analysis, and verification elements. Requirements relationships are established and maintained. Requirements in the system model are synchronized with the requirements management database. Traceability is continuously analyzed to assess and fill gaps or deficiencies. As requirements change, traceability is used to assess the impact of requirements changes on the system design, analysis, and verification elements.

9.4.2.7 Validate and Verify System This activity verifies that the system design satisfies its requirements and validates that those requirements meet the stakeholder needs. Verification plans, procedures, and methods (e.g., inspection, demonstration, analysis, and test) are developed. The primary inputs to the development of the test cases and associated verification procedures are system-level use cases, scenarios, and associated requirements. The verification system can be modeled using the same activities and artifacts described earlier for modeling the operational system. The requirements management database is updated during this activity to trace the system requirements and design information to the system verification methods, test cases, and results.

9.4.3 Applying OOSEM

OOSEM is an MBSE method used to specify and design systems. These include not only the operational system such as an aircraft or an automobile but also systems that enable the operational system throughout its life cycle, such as manufacturing, support, and verification systems. The method may also be applied to architect a system of systems or an enterprise, as well as to architect individual systems, or even system elements.

OOSEM should be tailored to support specific applications, project needs, and constraints. Tailoring may include varying the degree of emphasis on a particular activity and associated modeling artifacts and/or sequencing activities to suit a particular life cycle model.

The modeling artifacts can also be refined and reused in other applications to support product line and evolutionary development approaches. Product line modeling concerns the modeling of variability. Three refinements are to be considered in order to ensure what could be called as a “model-based product line SE”:

- The modeling of the variability and the constraints between these variabilities for each type of artifacts: Needs, requirements, architecture, tests, and others.

- The modeling of the variability for each type of SysML diagrams: Activity diagrams, use cases, and others.
- The modeling of the dependency links between these artifacts and the constraints between these variabilities to explain the relationship between them: As an example for a product line of water heaters, the variability of the electrical resistance component power depends, *or not*, on the variability of the water heater capacity.

OOSEM can be used in conjunction with principles and practices from various schools of thought, such as agile software development. Again, this requires adapting and tailoring how OOSEM is applied to integrate with the other approaches.

9.5 PROTOTYPING

Prototyping is a technique that can significantly enhance the likelihood of providing a system that will meet the user's need. In addition, a prototype can facilitate both the awareness and understanding of user needs and stakeholder requirements. Two types of prototyping are commonly used: rapid and traditional.

Rapid prototyping is probably the easiest and one of the fastest ways to get user performance data and evaluate alternate concepts. A rapid prototype is a particular type of simulation quickly assembled from a menu of existing physical, graphical, or mathematical elements. Examples include tools such as laser lithography or computer simulation shells. They are frequently used to investigate form and fit, human–system interface, operations, or producibility considerations. Rapid prototypes are widely used and are very useful; but except in rare cases, they are not truly "prototypes."

Traditional prototyping is a tool that can reduce risk or uncertainty. A partial prototype is used to verify critical elements of the SOI. A full prototype is a complete representation of the system. It must be complete and accurate in the aspects of concern. Objective and quantitative data on performance times and error rates can be obtained from these higher-fidelity interactive prototypes.

The original use of a prototype was as the first-of-a-kind product from which all others were replicated. However, prototypes are not "the first draft" of production entities. Prototypes are intended to enhance learning and should be set aside when this purpose is achieved. Once the prototype is functioning, changes will often be made to improve performance or reduce production costs. Thus, the production entity may require different behavior. The maglev train system (see Section 3.6.3) may be considered a prototype (in this case, proof of concept) for longer distance systems that will exhibit some but not all of the characteristics of the short line. Scientists and engineers are in a much better position to evaluate modifications that will be needed to create the next system because of the existence of a traditional prototype.

9.6 INTERFACE MANAGEMENT

Interface management is a proven set of activities that cut across the SE processes. Although some organizations treat interface management as a separate process, these are crosscutting activities of the technical and technical management processes that the project team should apply and track as a specific view of the system. When interface management is applied as a specific objective and focus of the technical processes, it will often help highlight underlying critical issues much earlier in the project than would otherwise be revealed. This would then impact upon project cost, schedule, and technical performance.

Interfaces are identified within the architecture definition process (Section 4.4), as the architecture models are developed. The interface requirements are defined through the system requirements definition process (Section 4.3). As the requirements are defined, the interface descriptions and definitions are defined to the extent needed for the architecture description within the architecture definition process (Section 4.4). Any further refinement and detail of the interface definition is provided by the design definition process (Section 4.5) as the details of the specific system implementation details are defined. The evolution of the system definition involves iteration between these processes, and the interface definition is an essential part of it. As the interface identification and definition evolve in the architecture definition process, there is an objective of keeping the interfaces as simple as possible (Fig. 4.8).

As part of the interface definition, many projects find the need or benefit to apply interface standards. In some cases, such as for plug-and-play elements or interfaces across open systems, it is necessary to strictly apply

interface standards to ensure the necessary interoperation with systems for which the project team does not control. Examples of these standards include Internet Protocol standards and Modular Open Systems Architecture standards. Interface standards can also be beneficial for systems that are likely to have emergent requirements by enabling the evolution of capabilities through the use standard interface definitions that allow new system elements to be added.

Good communication is a vital part of ensuring the interface management of a system. Many projects incorporate the use of an Interface Control Working Group (ICWG), which include members responsible for each of the interfacing elements. The ICWG can be focused on internal interfaces within a single system or the external interfaces between interoperating or enabling systems. The use of ICWGs formalizes and enhances collaboration within project teams or between project teams/organizations. The use of ICWGs is an effective approach that helps to ensure adequate consideration of all aspects of the interfaces.

One of the objectives of the interface management activities is to facilitate agreements with other stakeholders. This includes roles and responsibilities, the timing for providing interface information, and the identification of critical interfaces early in the project through a structured process. This is done through the project planning process (Section 5.1). Through specific interface focus as a part of the risk management process (Section 5.4), early identification of issues, risks, and opportunities can be managed, avoiding potential impacts, especially during integration. Interface management also enhances relationships between the different organizations, giving an open communication system of issues and cooperation, where problems can be resolved more effectively.

Finally, after establishing baselines for requirements, architecture, and design artifacts, the configuration management process (Section 5.5) provides the ongoing management and control of the interface requirements and definitions, as well as any associated artifacts (such as interface control documents, interface requirement specifications, and interface description/definition documents).

Interface management is intended to provide a simple but effective method to formally document and track the exchange of information as the life cycle processes are performed.

9.6.1 Interface Analysis Methods

There are several analysis methods and tools that aid the interface definition. These methods help to identify and understand the interfaces in the context of the system, the system elements, and/or the interfacing systems. Generally, the system analysis process (Section 4.6) is invoked by the system requirements definition, architecture definition, or design definition processes to perform the interface analysis.

N^2 diagrams (see Fig. 9.8) are a systematic approach to analyze interfaces. These apply to system interfaces,

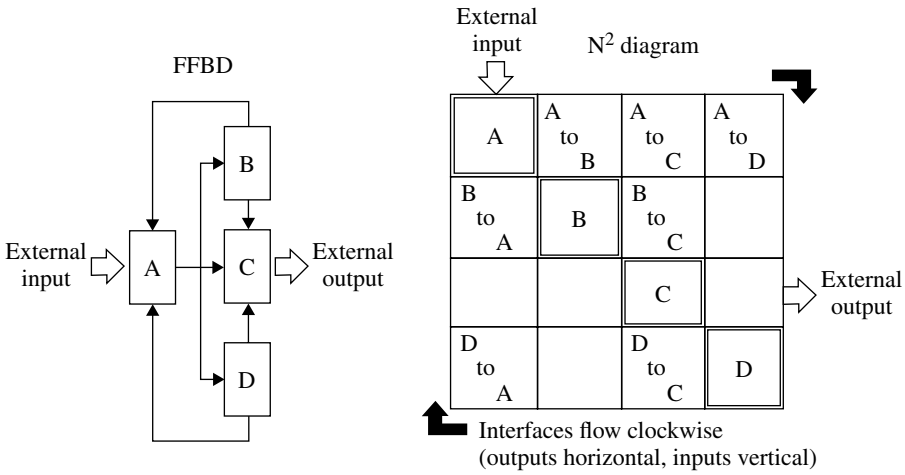


FIGURE 9.8 Sample FFBD and N^2 diagram. INCOSE SEH original figure created by Krueger and Forsberg. Usage per the INCOSE Notices page. All other rights reserved.

equipment (e.g., hardware) interfaces, or software interfaces. N^2 diagrams can also be used at later stages of the development process to analyze and document physical interfaces between system elements. For effective application, an N^2 diagram, which is a visual matrix, requires the systems engineer to generate complete definitions of all the system interfaces in a rigid bidirectional, fixed framework.

The system functions or physical elements are placed on the chart diagonal. The rest of the squares in the $N \times N$ matrix represent the interface inputs and outputs. Interfaces between functions flow in a clockwise direction. The entity being passed from function A to function B, for example, can be defined in the appropriate square. When a blank appears, there is no interface between the respective functions. When all functions have been compared to all other functions, then the chart is complete. If lower-level functions are identified in the process with corresponding lower-level interfaces, then they can be successively described in expanded or lower-level diagrams. Sometimes, characteristics of the entity passing between functions may be included in the box where the entity is identified. One of the main functions of the chart, besides interface identification, is to pinpoint areas where conflicts may arise between functions so that systems integration later in the development cycle can proceed efficiently (Becker et al., 2000; DSMC, 1983; Lano, 1977).

Alternatively, or in addition, FFBDs and DFDs can be used to characterize the flow of information among functions and between functions and the outside world. As the system architecture is decomposed to lower and lower levels, it is important to make sure that the interface definitions keep pace and that interfaces are not defined that ignore lower-level decompositions.

Other analysis methods that may be useful for interface definition include Design Structure Matrix (DSM) and the ibd (SysML).

- "Design Structure Matrix (DSM) is a straightforward and flexible modeling technique that can be used for designing, developing, and managing complex systems. DSM offers network modeling tools that represent the elements of a system and their interactions, thereby highlighting the system's architecture (or designed structure)." The DSM is very similar in appearance and usage to the N^2 diagram, but a different input and output convention is typically used (inputs on the horizontal rows and outputs on the vertical columns) (Eppinger and Browning, 2012).
- The ibd specifies the interconnection of parts of the system in SysML (see Section 9.1.9). They are used to describe the internal structure of a system in terms of its parts, ports, and connectors. The ibd provides the white box, or internal view, of a system block to represent the final assembly of all blocks within the main system block (Friedenthal et al., 2012).

9.7 INTEGRATED PRODUCT AND PROCESS DEVELOPMENT

Integrated Product Development (IPD) recognizes the need to consider all elements of the product life cycle, from conception through disposal, starting at the beginning of the life cycle. Important items to consider include quality, cost, schedule, user requirements, manufacturing, and support. IPD also implies the continuous integration of the entire product team, including engineering, manufacturing, verification, and support, throughout the product life cycle (DoD, 1998).

Risks inherent in concurrent product development are reduced by moving away from traditional hierarchical management structure and organized into Integrated Product Teams (IPTs). Productivity gains come through decentralization of processes, avoidance of previous problems, and better integration between engineering and manufacturing. Traditional development with serial activities may be so lengthy such that the product becomes obsolete before it is completed. With good interface definition and control, IPD, involving the entire team, can speed up the development process.

IPPD further recognizes the importance of *process*. The following definitions apply to IPPD:

- *Integrated Product Development Team (IPDT)*—A multidisciplinary group of people who are collectively responsible for delivering a defined product or process.
- *IPPD*—The process of using IPDTs to simultaneously develop the design for a product or system and the methods for manufacturing the product or system. The process verification may consist of review of a process description by an IPDT. It may also include a demonstration to an IPDT of a process.

- *Concurrent engineering*—Is a management/operational approach that aims to improve product design, production, operation, and maintenance by developing environments in which personnel from all disciplines (e.g., design, marketing, production engineering, process planning, and support) work together and share data throughout all stages of the product life cycle.

Integrated development has the potential to introduce *more* risk into a development program because downstream activities are initiated on the *assumption* that upstream activities will meet their design and interface requirements. However, the introduction of a hierarchy of cross-functional IPDTs, each developing and delivering a product, can reduce risks and provide better products faster.

IPPD also improves team communications through IPDTs, implements a proactive risk process, makes decisions based on timely input from the IPDT, and improves customer involvement.

9.7.1 IPDT Overview

An IPDT is a process-oriented, integrated set of cross-functional teams (i.e., an overall team comprising many smaller teams) given the appropriate resources and charged with the responsibility and authority to define, develop, produce, and support a product or process (and/or service). Each team is staffed with the skills necessary to complete their assigned processes, which may include all or some of the development and production steps.

The general approach is to form cross-functional IPDTs for all products and services. The typical types of IPDTs are a Systems Engineering and Integration Team (SEIT), a Product Integration Team (PIT), and a Product Development Team (PDT). These teams each mimic a small, independent project focusing on individual elements and/or their integration into more complex system elements. The SEIT balances requirements between product teams, helps integrate the other IPDTs, focuses on the integrated system and system processes, and addresses systems issues, which, by their nature, the other IPDTs would most likely relegate to a lower priority. Although the teams are organized on a process basis, the organizational structure of the team of teams may approach a hierarchical structure for the product, depending upon the way the product is assembled and integrated.

The focus areas for these IPDT team types and their general responsibilities are summarized in Table 9.1. This arrangement often applies to large, multielement, multiple subsystem programs but must be adapted to the specific project. For example, on smaller programs, the number of PIT teams can be reduced or eliminated. In service-oriented projects, the system hierarchy, focus, and responsibilities of the teams must be adapted to the appropriate services.

Team members' participation will vary throughout the product cycle, and different members may have

TABLE 9.1 Types of IPDTs and their focus and responsibilities

System hierarchy	Team type + focus responsibilities
External interface and system	Systems engineering and integration team (SEIT) • Integrated system and processes • External and program issues • System issues and integrity • Integration and audits of teams
Upper-level elements	Product Integration Teams (PITs) • Integrated H/W and S/W • Deliverable item issues and integrity • Support to other teams (SEIT and PDTs)
Lower-level elements	Product Development Teams (PDTs) • Hardware and software • Product issues and integrity • Primary participants (design and Mfg.) • Support to other teams (SEIT and PITs)

primary, secondary, or minor support roles as the effort transitions from requirements development through the different stages of the life cycle. For example, the manufacturing and verification representatives may have minor, part-time advisory roles during the early product definition stage but will assume primary roles later, during manufacture and verification. Team members participate to the degree necessary from the outset to ensure their needs and requirements are reflected in overall project requirements and planning to avoid costly changes later. It is also good for some of the team to remain throughout the product cycle to retain the team's "project memory."

IPDTs must be empowered with full life cycle responsibility for their products and systems and with the authority to get the job done. They should *not* be looking to higher management for key decisions. They should, however, be required to justify their actions and decisions to others, including interfacing teams, the systems integration team, and project management.

9.7.2 IPDT Process

The basic principle of IPDT is to get all disciplines involved at the beginning of product development to ensure that needs and requirements are completely understood for the full life cycle of the product. Requirements are developed initially at the system level, then successively at lower levels as the requirements are flowed down. Teams, led by systems engineers, perform the up-front SE activities at each level.

IPDTs do their own internal integration in an IPPD environment. A SEIT representative belongs to each product team (or several) with internal and external team responsibilities. There is extensive iteration between the product teams and the SEIT to converge on requirements and design concepts, although this effort should slow down appreciably after the preliminary design review and as the design firms up.

Systems engineers participate heavily in the SEIT and PIT and to a much lesser extent in the PDT. Regardless, the iterative SE processes described in this handbook are just as applicable to all teams in the IPPD environment. It is even easier to apply the processes throughout the program because of the day-to-day presence of systems engineers on all teams.

IPDTs have many roles, and their integration roles overlap based on the type of team and the integration level. Figure 9.9 gives examples of program processes and system activities.

The three bars on the left show the roles of the types of product teams at different levels of the system. For example, the SEIT leads and audits in external integration and in systems integration activities, as indicated by the shaded bar. For program processes involving lower-level elements (e.g., parts, components, or subassemblies), the appropriate PDTs are the active lead and audit participants, supported by the SEIT and the PIT.

Basic system activities include system requirements derivation, system functional analysis, requirements allocation and flowdown, system trade-off analysis, system synthesis, systems integration, TPM definition, and system verification. The bars for system functions 1, 2, and 3 in the chart show that the SEIT leads and audits activities on different system activities while the element teams participate. The lower-level system element teams provide additional support, if requested.

The column at the right side of Figure 9.9 shows other integration areas where all teams have some involvement. The roles of the various teams must also be coordinated for these activities but should be similar to the example.

9.7.2.1 Organizing and Running a High-Performance IPDT The basic steps and key activities to organize and run an IPDT are as follows:

1. *Define the IPDT teams for the project*—Develop IPDT teams that cover all project areas.
2. *Delegate responsibility and authority to IPDT leaders*—Select experienced team leaders early in the development process and avoid frequent budget changes throughout the life cycle.
3. *Staff the IPDT*—Candidates must work well in a team environment, communicate well, and meet their commitments:
   - Balance the competency, availability, and full-time commitment of the core team.
   - Plan when competencies are needed and not needed.
   - Identify issues where specialists are needed.
4. *Understand the team's operating environment*—Recognize how the team directly or indirectly influences other teams and the project as a whole.

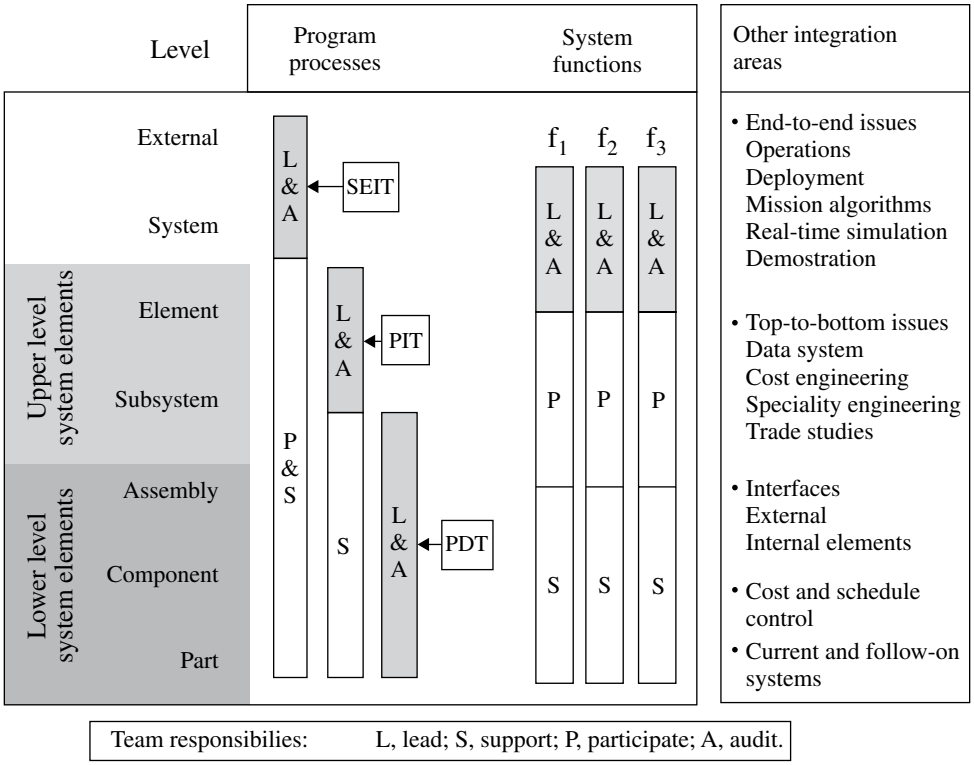


FIGURE 9.9 Examples of complementary integration activities of IPDTs. Adapted from Bob Lewis as INCOSE SEH v1 Figure 6.3. Usage per the INCOSE Notices page. All other rights reserved.

5. *Plan and conduct the "kick-off meeting"*—Recommend two kick-off meetings, one for the project as a whole and one for the individual IPDTs. Well-planned kick-off meetings will set the project off on the right foot.
6. *Train the team*—Training for the project is a critical element. The following recommended topics should be covered:
   - Tailored SE process for the project
   - Project description, stakeholders, purpose, mission, organization, schedule, and budget
   - Terminology and nomenclature
   - Access to project products
   - Communications skills
   - Project IPDT procedures, measures, and reporting

   Additional training sessions should be held and self-learning guides should be developed to help new team members come up to speed on the project when staff turnover occurs.
7. *Define the team vision and objectives*—Use collaborative brainstorming in the initial IPDT meetings to develop the team's vision and objectives such that each member has an ownership. It most likely will be necessary to bring in other IPDT members, management, and customers to flesh out the vision and objectives of the team.
8. *Have each team expand the definition of its job*—Once the higher-level project plan has been reviewed, each team must identify the tasks, roles, responsibilities, and milestones of the team and each of the members. Members need to understand how their individual tasks fit into the higher-level project–program tasks.
9. *Establish an expectation of routine process assessment and continuous improvement*—Each team must document the process they are using and the key measures to be monitored. The teams must have the mindset of continuous improvement, monitor their own activities, and continually make course corrections along the way.

10. *Monitor team progress via measures and reports*—Each team will have a set of measures and reports to monitor its own progress. These reports and measures must be reviewed by the SEIT that coordinates among the other IPDTs. These measures may include an earned value report and technical measures, such as a defect rate report. The selected measures are dependent on the team's role on the project.
11. *Sustain and evolve the team throughout the project*—Personnel assignments to a team will vary as each team grows, shrinks, and changes skill mix over the project life cycle. As issues arise, technical specialists may need to join the team to help address these specific issues. Services such as marketing, program controls, procurement, finance, legal, and human resources generally support the team at a steady, low level of effort, or as required.
12. *Document team products*—The team's products should be well defined. Because of the IPDT structure, the overhead of cross-organizational communication varies and should be reduced. When multiple documents are required, different team members with identified backups should be assigned as the responsible author with contributions from others. The IPDT should maintain a log of activities in addition to the mission, vision, objectives, deliverables, meeting minutes, decisions, tailored processes, agreements, team project information, and contact information.
13. *Close the project and conduct follow-up activities*—In conjunction with step 12, the IPDT should maintain records as though the project may be reengineered at some future time and all closeout products must be accessible. All IPDT logs should be organized the same way, when possible, such that they can be easily integrated into an overall project report. The closeout should include lessons learned, recommended changes, and a summary of measures for the team.

Project managers should review team staffing plans to ensure proper composition and strive for continuity of assignments. The advantages of a full-time contributor outweigh the work of many part-time team members. Similarly, the loss of a knowledgeable key team member can leave the team floundering. It is important to have people who can work well together and communicate, but team results may suffer without outstanding technical specialists and professionals who can make a difference. Recommended techniques for achieving high performance in an IPDT are as follows:

- *Carefully select the staff*—Excellent people do excellent work.
- *Establish and maintain positive team interaction dynamics*—All should know what is expected of the team and each individual and strive to meet commitments. Anticipate and surface potential problems quickly (internally and externally). Interactions should be informal but efficient and a "no blame" environment where problems are fixed and the team moves on. Acknowledge and reward good work.
- *Generate team commitment and buy-in*—Team alignment to the vision, objectives, tasks, and schedules. Maintain a team leader's notebook.
- *Breakdown the job into manageable activities*—Those that can be accurately scheduled, assigned, and followed up on weekly.
- *Delegate and spread out routine administrative tasks among the team*—Free up the leader to participate in technical activities. Give every team member some administrative/managerial experience.
- *Schedule frequent team meetings with mandatory attendance for quick information exchanges*—Ensure everyone is current. Assign action items with assignee and due date.

9.7.3 Potential IPDT Pitfalls

There are ample opportunities to go astray before team members and leaders go through several project cycles in the IPDT framework and gain the experience of working together. Table 9.2 describes some pitfalls common to the IPDT environment that teams should watch out for.

9.8 LEAN SYSTEMS ENGINEERING

SE is regarded as an established, sound practice, but not always delivered effectively. Recent US Government Accountability Office (GAO, 2008) and NASA (2007a) studies of space systems document major budget and

TABLE 9.2 Pitfalls of using IPDT

IPDT pitfalls	What to do
Spending too much time defining the vision and objectives	Converge and move on
Insufficient authority—IPDT members must frequently check with management for approval	Give team leader adequate responsibility, or put the manager on the team
IPDT members are insensitive to management issues and overcommit or overspend	Team leader must remain aware of overall project objectives and communicate to team members
Teams are functionally oriented rather than cross-functionally process oriented	Review the steps in organizing and running an IPDT (see preceding text)
Insufficient continuity of team members throughout the project	Management should review staffing requirements
Transition to the next stage team specialists occurs too early or too late in the schedule	Review staffing requirements
Overlapping assignments for support personnel compromises their effectiveness	Reduce the number of teams
Inadequate project infrastructure	Management involvement to resolve

schedule overruns. Similarly, recent studies by the MIT-based Lean Advancement Initiative (LAI) have identified a significant amount of waste in government programs, averaging 88% of charged time (LAI MIT, 2013; McManus, 2005; Oppenheim, 2004; Slack, 1998). Most programs are burdened with some form of waste: poor coordination, unstable requirements, quality problems, and management frustration. This waste represents a vast productivity reserve in programs and major opportunities to improve program efficiency.

Lean development and the broader methodology of lean thinking have their roots in the Toyota "just-in-time" philosophy, which aims at "producing quality products efficiently through the complete elimination of waste, inconsistencies, and unreasonable requirements on the production line" (Toyota, 2009). Lean SE is the application of lean thinking to SE and related aspects of organization and project management. SE is focused on the discipline that enables flawless development of complex technical systems. Lean thinking is a holistic paradigm that focuses on delivering maximum value to the customer and minimizing wasteful practices. A popular description of lean is "Doing the right job right the first time" and "working smarter, not harder." Lean thinking has been successfully applied in manufacturing, aircraft depots, administration, supply chain management, healthcare, and product development, including engineering.

Lean SE is the area of synergy between lean thinking and SE, with the goal to deliver the best life cycle value for technically complex systems with minimal waste. The early use of the term lean SE is sometimes met with concern that this might be a "repackaged faster, better, cheaper" initiative, leading to cuts in SE at a time when the profession is struggling to increase the level and quality of SE effort in programs. Lean SE does not take away anything from SE and it does not mean *less* SE. It means more and better SE with higher responsibility, authority, and accountability, leading to better, waste-free workflow with increased mission assurance. Under the lean SE philosophy, mission assurance is nonnegotiable, and any task that is legitimately required for success must be included, but it should be well planned and executed with minimal waste.

> Lean thinking: "Lean thinking is the dynamic, knowledge-driven, and customer-focused process through which all people in a defined enterprise continuously eliminate waste with the goal of creating value" (Murman, 2002).
>
> *Lean SE*: The application of lean principles, practices, and tools to SE to enhance the delivery of value to the system's stakeholders.

Three concepts are fundamental to the understanding of lean thinking: value, waste, and the process of creating value without waste (also known as lean principles).

9.8.1 Value

The value proposition in engineering programs is often a multiyear, complex, and expensive acquisition process involving numerous stakeholders and resulting in

hundreds or even thousands of requirements, which, notoriously, are rarely stable. In lean SE, "value" is defined simply as mission assurance (i.e., the delivery of a flawless complex system, with flawless technical performance, during the product or mission development life cycle) and satisfying the customer and all other stakeholders, which implies completion with minimal waste, minimal cost, and the shortest possible schedule.

> "*Value* is a measure of worth (e.g., benefit divided by cost) of a specific product or service by a customer, and potentially other stakeholders and is a function of (1) the product's usefulness in satisfying a customer need, (2) the relative importance of the need being satisfied, (3) the availability of the product relative to when it is needed, and (4) the cost of ownership to the customer" (McManus, 2004).

9.8.2 Waste in Product Development

The LAI classifies waste into seven categories: overprocessing, waiting, unnecessary movement, overproduction, transportation, inventory, and defects (McManus, 2005). Lately, the eighth category is increasingly added: the waste of human potential.

> *Waste*: "The work element that adds no value to the product or service in the eyes of the customer. Waste only adds cost and time" (Womack and Jones, 1996).

When applying lean thinking to SE and project planning, consider each waste category and identify areas of wasteful practice. The following illustrates some waste considerations for SE practice in each of the LAI waste classifications:

- *Overprocessing*—Processing more than necessary to produce the desired output. Consider how projects "overdo it" and expend more time and energy than needed:
  - Too many hands on the "stuff" (material or information)
  - Unnecessary serial production
  - Excessive/custom formatting or reformatting
  - Excessive refinement, beyond what is needed for value
- *Waiting*—Waiting for material or information, or information or material waiting to be processed. Consider "things" that projects might be waiting for to complete a task:
  - Late delivery of material or information
  - Delivery too early—leading to eventual rework
- *Unnecessary movement*—Moving people (or people moving) to access or process material or information. Consider any unnecessary motion in the conduct of the task:
  - Lack of direct access—time spent finding what you need
  - Manual intervention
- *Overproduction*—Creating too much material or information. Consider how more "stuff" (e.g., material or information) is created than needed:
  - Performing a task that nobody needs or using a useless metric
  - Creating unnecessary data and information
  - Information overdissemination and pushing data
- *Transportation*—Moving material or information. Consider how projects move "stuff" from place to place:
  - Unnecessary hand-offs between people
  - Shipping "stuff" (pushing) when not needed
  - Incompatible communication—lost transportation through communication failures
- *Inventory*—Maintaining more material or information than is needed. Consider how projects stockpile information or materials:
  - Too much "stuff" buildup
  - Complicated retrieval of needed "stuff"
  - Outdated, obsolete information
- *Defects*—Errors or mistakes causing the effort to be redone to correct the problem. Consider how projects go back and do it again:
  - Lack of adequate review, verification, or validation
  - Wrong or poor information
- *Waste of human potential*—Not utilizing or even suppressing human enthusiasm, energy, creativity, and ability to solve problems and general willingness to perform excellent work.

9.8.3 Lean Principles

Womack and Jones (1996) captured the process of creating value without waste into six lean principles. The principles (see Fig. 9.10) are abbreviated as value, value stream, flow, pull, perfection, and respect for people and are defined in detail in the following.

When applying lean thinking to SE, evaluate project plans, preparations of people, processes and tools, and organization behaviors using the lean principles. Consider how the customer defines *value* in the products and processes, then describe the *value stream* for creating products and processes, optimize *flow* through that value stream and eliminate waste, encourage *pull* from each node in that value stream, and strive to *perfect* the value stream to maximize value to the customer. These activities should all be conducted within a foundation of *respect* for customers, stakeholders, and project team members.

In 2009, the INCOSE Lean SE Working Group released a new online product entitled *Lean Enablers for Systems Engineering* (LEfSE), Version 1.0. It is a collection of practices and recommendations formulated as "do's" and "don'ts" of SE based on lean thinking. The practices cover a large spectrum of SE and other relevant enterprise management practices, with a general focus on improving program value and stakeholder satisfaction and reducing waste, delays, cost overruns, and frustrations. LEfSE are currently listed as 147 practices (referred to as subenablers) organized under 47 nonactionable topical headings called enablers and grouped into the six lean principles described below (Oppenheim, 2011):

1. Under the *value principle*, subenablers promote a robust process of establishing the value of the end product or system to the customer with crystal clarity early in the program. The process should be customer focused, involving the customer frequently and aligning employees accordingly.
2. The subenablers under the *value stream principle* emphasize detailed program planning and waste-preventing measures, solid preparation of the

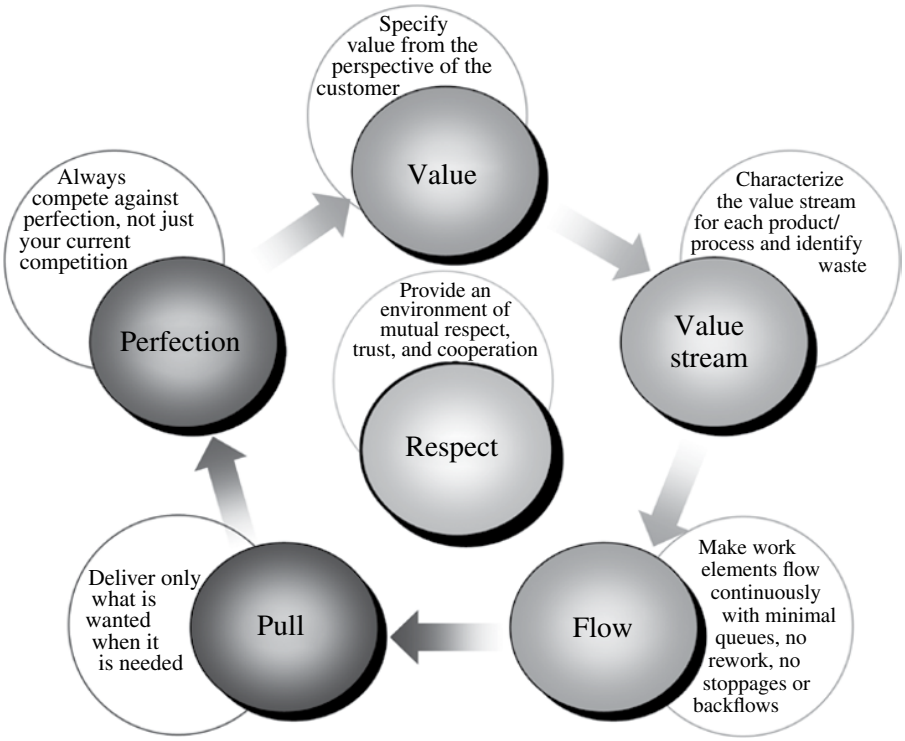


FIGURE 9.10 Lean development principles. Reprinted with permission from Bohdan Oppenheim. All other rights reserved.

personnel and processes for subsequent efficient workflow, and healthy relationships between stakeholders (e.g., customer, contractor, suppliers, and employees); program frontloading; and use of leading indicators and quality measures. Systems engineers should prepare for and plan all end-to-end linked actions and processes necessary to realize streamlined value, after eliminating waste.

3. The *flow principle* lists subenablers that promote the uninterrupted flow of robust quality work and first-time right products and processes, steady competence instead of hero behavior in crises, excellent communication and coordination, concurrency, frequent clarification of the requirements, and making program progress visible to all.
4. The subenablers listed under the *pull principle* are a powerful guard against the waste of rework and overproduction. They promote pulling tasks and outputs based on internal and external customer needs (including rejecting others as waste) and better coordination between the pairs of employees handling any transaction before their work begins so that the result can be first-time right.
5. The *perfection principle* promotes excellence in the SE and organization processes, the use of the wealth of lessons learned from previous programs in the current program, the development of perfect collaboration policy across people and processes, and driving out waste through standardization and continuous improvement. Imperfections should be made visible in real time, and continuous improvement tools (root cause analysis and permanent fix) should be applied. A category of these subenablers calls for a more important role of systems engineers, with responsibility, accountability, and authority for the overall technical success of the program.
6. Finally, the *respect-for-people principle* contains subenablers that promote the enterprise culture of trust, openness, honesty, respect, empowerment, cooperation, teamwork, synergy, and good communication and coordination and enable people for excellence.

In 2011, a follow-on major project undertaken jointly by the Project Management Institute (PMI), INCOSE, and the LAI at Massachusetts Institute of Technology in the leading role developed *Lean Enablers for Managing Engineering Programs (LEfMEP)* (Oehmen, 2012), incorporating all LEfSE, adding lean enablers for project and program management, and holistically integrating lean program management with lean SE. A major section of the book is devoted to a rigorous analysis of challenges in managing engineering programs. They are presented under the following 10 top challenge themes:

1. Firefighting—reactive program execution
2. Unstable, unclear, and incomplete requirements
3. Insufficient alignment and coordination of the extended enterprise
4. Processes that are locally optimized and not integrated for the entire enterprise
5. Unclear roles, responsibilities, and accountability
6. Mismanagement of program culture, team competency, and knowledge
7. Insufficient program planning
8. Improper metrics, metric systems, and key performance indicators
9. Lack of proactive program risk management
10. Poor program acquisition and contracting practices

The 326 lean enablers in Oehmen (2012) are listed in several convenient ways: under the six lean principles, under the 10 major challenge themes, under the SE processes used in this volume, and under the management performance domains defined in (PMI, 2013).

The LEfSE and LEfMEP are not intended to become mandatory practices. Instead, they should be used as a checklist of excellent holistic practices validated by the community of practice. Awareness of the enablers should improve the thinking at work and significantly improve program quality. Early feedback from the organizations practicing lean enablers indicates significant benefits (Oppenheim, 2011).

The *INCOSE Lean SE Working Group* public website (2011) contains a rich menu of publications and case studies related to both LEfSE and LEfMEP.

9.9 AGILE SYSTEMS ENGINEERING

Historically, agile software engineering processes came into awareness in 2001 with the declaration of the Agile Manifesto (Beck et al., 2001), which spawned interest in

a number of methodologies, with names such as Scrum and Extreme Programming. But adoption of those methodologies and consideration of how they might inform nonsoftware engineering (Carson, 2013) has tended to focus on software-related specific practices rather than fundamental frameworks. In contrast, a cross-industry study in 1991 (Nagel, 1992) observed that technology and the environment in which it is deployed were coevolving at an increasing rate, outpacing the adaptation capabilities of most organized human endeavors. Agility, as a systemic characteristic, was thus identified, and subsequently studied to identify domain-independent metrics, architecture, and design principles (Dove, 2001).

Agility is a capability exhibited by systems and processes that enables them to sustain effective operation under conditions of unpredictability, uncertainty, and change. The value proposition of an agile SE process is risk management, appropriate when development speed and customer satisfaction are likely to be affected by requirements understandings that evolve during system development.

Common causes for requirements evolution include insufficient initial understanding, new understandings revealed during development, and evolving knowledge of the deployment environment. If ignored, requirements evolution reduces or eliminates customer satisfaction. If unmitigated, requirements evolution causes rework and scrapped work, a principal source of time and cost overruns.

An agile system architecture incurs expense in infrastructure and modularity design, which should be weighed against probabilistic costs for requirements evolution. This is risk management. The purpose of an agile SE process is to reduce the technical, cost, and schedule risks associated with accommodating beneficial requirements evolution.

Agility is the ability to respond effectively to surprises—good or bad. Practices and techniques should be chosen for compatibility and synergy with the nature of the project (Carson, 2013; Sillitto, 2013) and the cultural environment in which they will be employed.

9.9.1 Agile SE Framework

Agile SE (Forsberg et al., 2005) is summarized as follows:

- Leverages an agile architecture for SE (process), enabling reconfiguration of goals, requirements, plans, and assets, predictably.
- Leverages an architecture for agile SE (product), enabling changes to the product (system) during development and fabrication, predictably.
- Leverages an empowered intimately involved "product owner" (chief systems engineer, customer, or equivalent responsible authority on product vision), enabling broad-level systems thinking to inform real-time decision making as requirements understanding evolve.
- Leverages human productivity factors that affect engineering, fabrication, and customer satisfaction in an unpredictable and uncertain environment.

9.9.2 Agile Metric Framework

Agility measures are enabled and constrained principally by architecture—in both the process and the product of development:

- Time to respond, measured in both the time to understand a response is necessary and the time to accomplish the response
- Cost to respond, measured in both the cost of accomplishing the response and the cost incurred elsewhere as a result of the response
- Predictability of response capability, measured before the fact in architectural preparedness for response and confirmed after the fact in repeatable accuracy of response time and cost estimates
- Scope of response capability, measured before the fact in architectural preparedness for comprehensive response capability within mission and confirmed after the fact in repeatable evidence of broad response accommodation

9.9.3 Agile Architectural Framework

Agile SE and agile-systems engineering are two different things (Haberfellner and de Weck, 2005) with a shared common architecture that enables the agility in each (Dove, 2012). The architecture will be recognized in a simple sense as a drag-and-drop plug-and-play loosely coupled modularity, with some critical aspects not often called to mind with the general thoughts of a modular architecture.

There are three critical elements in the architecture: a roster of drag-and-drop *encapsulated* modules, a passive

infrastructure of minimal but sufficient rules and standards that enable and constrain plug-and-play operation, and an active infrastructure that designates specific responsibilities that sustain agile operational capability:

- *Modules*—Modules are self-contained encapsulated units complete with well-defined interfaces that conform to the plug-and-play passive infrastructure. They can be dragged and dropped into a system of response capability with relationship to other modules determined by the passive infrastructure. Modules are encapsulated so that their interfaces conform to the passive infrastructure, but their methods of functionality are not dependent on the functional methods of other modules except as the passive infrastructure dictates.
- *Passive infrastructure*—The passive infrastructure provides drag-and-drop connectivity between modules. Its value is in isolating the encapsulated modules so that unexpected side effects are minimized and new operational functionality is rapid. Selecting passive infrastructure elements is a critical balance between requisite variety and parsimony—just enough in standards and rules to facilitate module connectivity but not so much to overly constrain innovative system configurations.
- *Active infrastructure*—An agile system is not something designed and deployed in a fixed event and then left alone. Agility is most active as new system configurations are assembled in response to new requirements—something which may happen very frequently, even daily in some cases. In order for new configurations to be enabled when needed, four responsibilities are required: the collection of available modules must evolve to be always what is needed, the modules that are available must always be in deployable condition, the assembly of new configurations must be accomplished, and both the passive infrastructure and active infrastructure must have evolved when new configurations require new standards and rules. Responsibilities for these four activities must be designated and embedded within the system to ensure that effective response capability is possible at unpredictable times:
  - Module mix—Who (or what process) is responsible for ensuring that new modules are added to the roster and existing modules are upgraded in time to satisfy response needs?
  - Module readiness—Who (or what process) is responsible for ensuring that sufficient modules are ready for deployment at unpredictable times?
  - System assembly—Who (or what process) assembles new system configurations when new situations require something different in capability?
  - Infrastructure evolution—Who (or what process) is responsible for evolving the passive and active infrastructures as new rules and standards are anticipated and become appropriate?

9.9.4 Agile Architectural Design Principles

Ten reusable, reconfigurable, scalable design principles are briefly itemized in this section:

Reusable principles are as follows:

- *Encapsulated modules*—Modules are distinct, separable, loosely coupled, independent units cooperating toward a shared common purpose.
- *Facilitated interfacing (plug compatibility)*—Modules share well-defined interaction and interface standards and are easily inserted or removed in system configurations.
- *Facilitated reuse*—Modules are reusable and replicable, with supporting facilitation for finding and employing appropriate modules.

Reconfigurable principles are as follows:

- *Peer–peer interaction*—Modules communicate directly on a peer-to-peer relationship; and parallel rather than sequential relationships are favored.
- *Distributed control and information*—Modules are directed by objective rather than method; decisions are made at point of maximum knowledge, and information is associated locally and accessible globally.
- *Deferred commitment*—Requirements can change rapidly and continue to evolve. Work activity, response assembly, and response deployment that are deferred to the last responsible moment avoid costly wasted effort that may also preclude a subsequent effective response.
- *Self-organization*—Module relationships are self-determined where possible, and module interaction is self-adjusting or self-negotiated.

Scalable principles are as follows:

- *Evolving standards*—Passive infrastructure standardizes intermodule communication and interaction, defines module compatibility, and is evolved by designated responsibility for maintaining current and emerging relevance.
- *Redundancy and diversity*—Duplicate modules provide capacity right-sizing options and fail-soft tolerance, and diversity among similar modules employing different methods is exploitable.
- *Elastic capacity*—Modules may be combined in responsive assemblies to increase or decrease functional capacity within the current architecture.

10

SPECIALTY ENGINEERING ACTIVITIES

The objective of this chapter is to give enough information to systems engineers to appreciate the significance of various engineering specialty areas, even if they are not an expert in the subject. It is recommended that subject matter experts are consulted and assigned as appropriate to conduct specialty engineering analysis. The topics in this chapter are covered in alphabetical order by topic title to avoid giving more weight to one topic over another. More information about each specialty area can be found in references to external sources.

With a few exceptions, the forms of analysis presented herein are similar to those associated with SE. Most analysis methods are based on the construction and exploration of models that address specialized engineering areas, such as electromagnetic compatibility (EMC), reliability, safety, and security. Not every kind of analysis and associated model will be applicable to every application domain.

10.1 AFFORDABILITY/COST-EFFECTIVENESS/ LIFE CYCLE COST ANALYSIS

As stated in Blanchard and Fabrycky (2011),

> Many systems are planned, designed, produced, and operated with little *initial* concern for affordability and the total cost of the system over its intended lifecycle… The technical [*aspects* are] usually considered first, with the economic [aspects] deferred until later.

This section addresses economic and cost factors under the general topics of affordability and cost-effectiveness. The concept of life cycle cost (LCC) is also discussed.

10.1.1 Affordability Concepts

Improving design methods for affordability (Bobinis et al., 2013; Tuttle and Bobinis, 2013) is critical for all application domains. The INCOSE and the National Defense Industrial Association (NDIA) (the Military Operations Research Society (MORS) has also adapted these definitions) have addressed "affordability" through ongoing affordability working groups started in late 2009 and have defined system affordability through these ongoing working groups. Both organizations have defined system affordability as follows:

- INCOSE Affordability Working Group definitions (June 2011):

  Affordability is the balance of system performance, cost and schedule constraints over the system life while satisfying mission needs in concert with strategic investment and organizational needs.

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.

> Design for affordability is the systems engineering practice of balancing system performance and risk with cost and schedule constraints over the system life satisfying system operational needs in concert with strategic investment and evolving stakeholder value.

- NDIA Affordability Working Group definition (June 2011):

  > Affordability is the practice of ensuring program success through the balancing of system performance (KPPs), total ownership cost, and schedule constraints while satisfying mission needs in concert with long-range investment, and force structure plans of the DOD.

The concept of affordability can seem straightforward. The difficulty arises when an attempt is made to specify and quantify the affordability of a system. This is significant when writing a specification or when comparing two affordable solutions to conduct an affordability trade study. Even though affordability has been defined by the INCOSE, NDIA, and MORS, in discussions at an MORS Special Meeting on *Affordability Analysis: How Do We Do It?*, it was noted that all industry groups have discovered that affordability analysis is contextually sensitive, often leading to a misunderstanding and incompatible perspectives on what an "affordable system *is*." The various industry working groups have recommended developing and formalizing affordability analysis processes, including recognizing the difference between cost and affordability analyses. As a result of these high-level discussions, the key affordability takeaways include:

- Affordability context, system(s), and portfolios (of systems capabilities) need to be consistently defined and included in any understanding of what an affordable system is.
- An affordability process/framework needs to be established and documented.
- Accountability (system governance) for affordability needs to be assigned across the life cycle, which includes stakeholders from the various contextual domains.

10.1.1.1 "Cost-Effective Capability" Is a Contextual Attribute As defined in "Better Buying Power: Mandate for Restoring Affordability and Productivity in Defense Spending" (Carter, 2011), "affordability means conducting a program at a cost constrained by the maximum resources the Department can allocate for that capability." Affordability includes acquisition cost and average annual operating and support cost. It is expanded to encompass additional elements required for the LCC of a system, as an outcome of various hierarchal contexts in which any system is embedded. Therefore, in the SE domain, affordability as an attribute must be determined both inside the boundaries of the system of interest (SOI) and outside (see Fig. 10.1). This defines, in practical

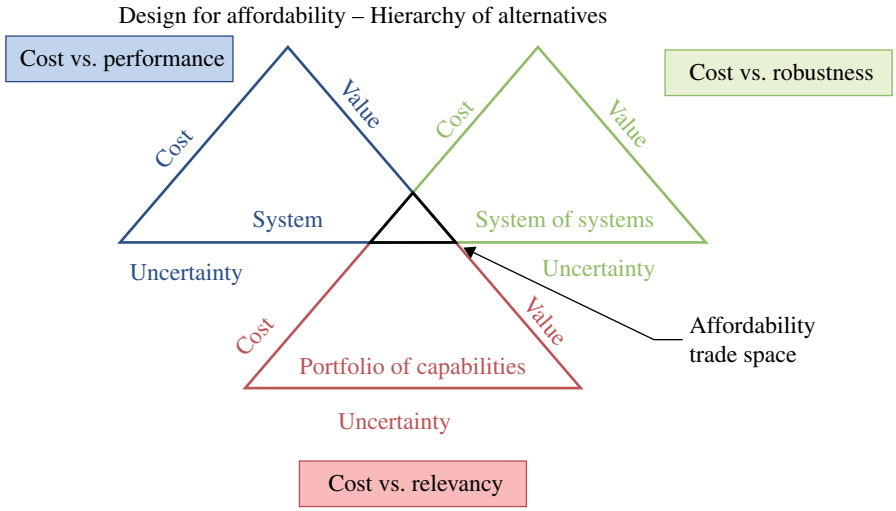


FIGURE 10.1 Contextual nature of the affordability trade space. Derived from Bobinis et al. (2013) Figure 1. Reprinted with permission from Joseph Bobinis. All other rights reserved.

terms, the link between system capability, cost, and what we call "affordability." Thus, the concept of affordability must encompass everything from a portfolio (e.g., family of automobiles) to an individual program (specific car model). Affordability as a design attribute of a system versus a program versus a domain remains contextually dependent on the stakeholder's context and the life cycle of the SOI under examination.

10.1.1.2 Design Model for Affordability As previously stated, an affordability design model must be able to provide the ability to effectively manage and evolve systems over long life cycles. The derived requirements we will focus on in this section are as follows:

- Design perspective that assumes the system will change based on environmental influences, new uses, and disabling system causing performance deterioration
- Causes of system and system element life cycle differences (technology management)
- Feedback functions and measurement of system behaviors with processes to address emergence (life cycle control systems)
- A method to inductively translate system behaviors into actionable engineering processes (adaptive engineering)

One of the major assumptions for measuring the affordability of competing systems is that given two systems, which produce similar output capabilities, it will be the *nonfunctional* attributes of those systems that differentiate system value to its stakeholders. The affordability model is concerned with operational attributes of systems that determine their value and effectiveness over time, typically expressed as the system's "ilities" or specialty engineering as they are called in this handbook.

These attributes are properties of the system as a whole and as such represent the salient features of the system and are measures of the ability of the system to deliver the capabilities it was designed for over time. "System integration, and its derivatives across the life cycle, requires additional discipline and a long term perspective during the SE and design phase. This approach includes explicit consideration of issues such as system reliability, maintainability and supportability to address activities pertaining to system operation, maintenance, and logistics. There is also a need to address real-world realities pertaining to changing requirements and customer expectations, changing technologies, and evolving standards and regulations" (Gallios and Verma, n.d.) (see Fig. 10.2).

10.1.1.3 Impact to Affordability Managing a system within an affordability trade space means that we are concerned with the actual performance of the fielded system, defined in one or more appropriate metrics, bounded by cost over time. ("System performance" can be expressed in whatever way makes sense for the system under study.) The time dimension extends a specific "point analysis" (static) to a continuous life cycle perspective (dynamic). Quantifying a relationship between cost, performance, and time defines a functional space

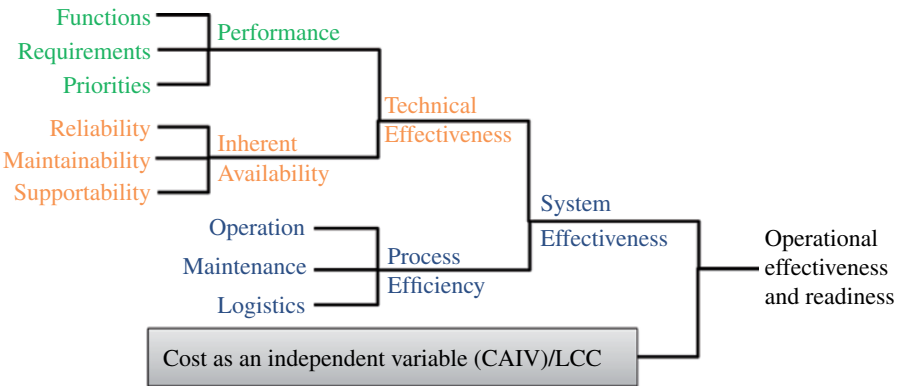


FIGURE 10.2 System operational effectiveness. Derived from Bobinis et al. (2013) Figure 4. Reprinted with permission from Joseph Bobinis. All other rights reserved.

that can be graphed and analyzed mathematically. Then it becomes possible to examine how the output (performance, availability, capability, etc.) changes due to changes in the input (cost constraints or budget availability). Hence, utilizing this functional relationship between cost and outcome defines an affordability trade space. Done correctly, it is possible to analyze specifically the relationship between money spent and system performance and possibly determine the point of diminishing returns.

However, it is frequently necessary to estimate the value of one variable when the values of the others are known or specified. These kinds of point solutions (which in this sense are "coordinates" within the space) are useful in examining specific relationships, including making predictions of "average" behavior. Answering questions regarding "expected value" of some parameter in terms of others, often for specific points in time, frequently falls into this category. The overall trade space can then be thought of as the totality of all such point solutions.

10.1.1.4 Affordability Trade Space throughout the System Life Cycle The affordability trade space must reflect the SE focus on meeting operational and performance characteristics and developing highly reliable systems. Supportability analysis, which should be done in conjunction with design, is concerned with developing highly maintainable systems. These disciplines share a common goal: fielding robust, capable systems that are available for use by the end user when needed. Operational availability (A_o) is then simply another performance parameter that can be examined as a function of cost within this space. It is reasonable to conclude that improvements to the design, in this case driven by a stringent A_o requirement, can lower operation and support costs in the fielded system. In fact, A_o is an implicit performance measure used to calculate expected system effectiveness. That has to be applied, along with use/market size and operational requirements, to determine the overall cost-effective capability of the system under study.

Supporting a system throughout its life cycle requires systems engineers to account for changes in the system design as circumstances change over time, such as changes in the threat environment, diminishing material shortage (DMS) issues, and improvements in technology, SoS relationships, and impact of variables outside the system.

Given that designers do not control the variables in the environment, the optimization problem for affordability is different or could be different at any point in the life cycle. This optimization problem can be managed by functional criticality, surge, and adaptive requirements or as a set of predetermined technology refresh cycles. It can be optimized for cost or performance whichever is of most *value*. The intention is to be able to measure both as a function of operational performance efficiency. The affordability model must enhance value engineering (VE) analysis through the ability to measure all functional contributions to operational performance. The designer must be provided with the ability to choose the range of functions to adjust for optimal impact and cost.

In all cases, the affordability trade space must be able to accommodate changes but always driven by the same key concepts: what does it cost to implement such a change, and what do we get for it. Consequently, identifying "cost-effective capability" is still the key to analysis within the affordability trade space across the system life cycle.

For instance, consider a system that is unique in the inventory but is suffering an unacceptably low A_o. Then the question arises, can we upgrade this system to improve field performance and do so in a cost-effective way? Is it possible to improve the existing reliability and maintainability characteristics of this system and increase A_o, given that the system itself has many years of service life remaining? So the trade space analysis must show that the upgrades will pay for themselves over time through lower operation and support costs, increased capability, or both.

In general, the factors that must be considered within the trade space across the system life cycle include:

- Cost versus benefits of different design solutions
- Cost versus benefits of different support strategies
- Methods and rationale used to develop these comparisons
- Ability to identify and obtain data required to analyze changes

From a programmatic perspective, analysis of these alternatives must also be done in conjunction with identification, classification, and analysis of associated risk and development of a costed plan for implementation.

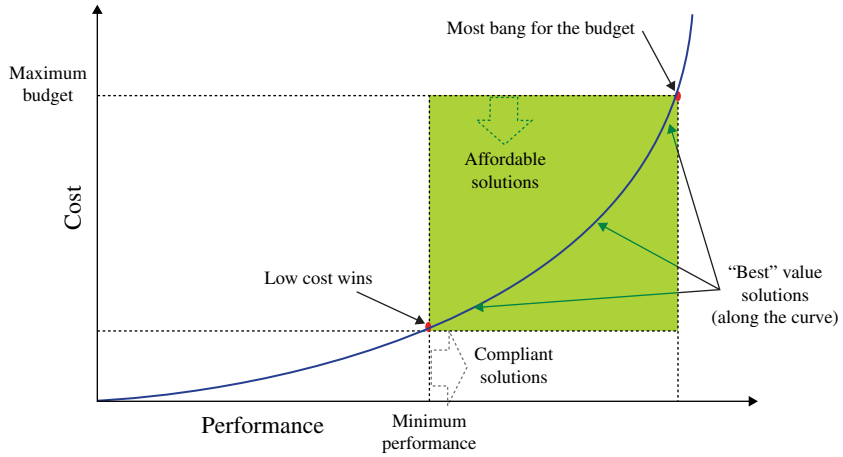


FIGURE 10.3 Cost versus performance. Reprinted with permission from Joseph Bobinis. All other rights reserved.

10.1.1.5 Affordability Implementation When one considers the entire SOI, both the primary and enabling systems should be treated as an SoS. In the example in Figure 10.3, the mission-effectiveness affordability trade space brings together primary and enabling systems into an SoS. Note that the SoS is treated as a closed-loop system where requirements are modified as the mission needs evolve. These iterations allow for technology insertion as the design is updated. Design-to-cost (DTC) targets are set for the primary and enabling systems, which ensure that affordability throughout the system life cycle is considered, even as the system evolves. Affordability measurements that feed back into the system assessment are KPPs and operational availability, which ensures that the mission can be accomplished over time (e.g., those KPPs are met across the system life cycle; see Fig. 10.4).

To define affordability for a particular program or system (see Fig. 10.3 as an example of an affordability range), we must define selected affordability components. As such, the following could be specified:

1. Required capabilities
   (a) Identify the required capabilities and the time phasing for inclusion of the capabilities.
2. Required capabilities performance
   (a) Identify and specify the required MOEs for each of the capabilities.
   (b) Define time phasing for achieving the MOEs.
3. Budget
   (a) Identify the budget elements to include in the affordability evaluation.
   (b) Time-phased budget, either
      (i) for each of the budget elements or
      (ii) as the total budget.

At least one of the affordability elements needs to be designated as the decision criteria that will be used in either a trade study or as the basis for a contract award. The affordability elements that are not designated as the decision criteria become constraints, along with the constraints being specified. This is illustrated by an example depicted in Figure 10.3. Here, the capabilities and schedule have been fixed leaving either the cost or the performance to be the evaluation criteria, while the other becomes the constraint. This results in a relatively simple relationship between performance and cost. The maximum budget and the minimum performance are identified. Below the maximum budget line lie solutions that meet the definition of "…conducting a program at a cost constrained by the maximum resources…." The solutions to the right of the minimum performance line satisfy the threshold requirement. Thus, in the shaded rectangle lie the solutions to be considered since they meet the minimum performance and are less than the maximum budget. On the curve lay the solutions that are the "best value," in the sense that for a given cost the corresponding point on the curve is the maximum

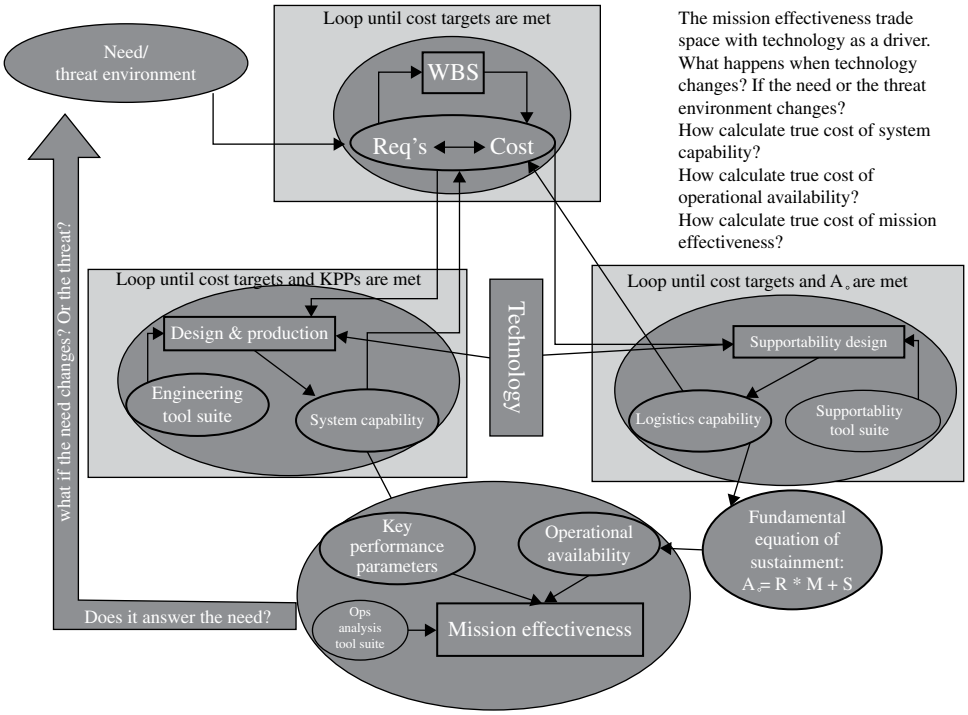


FIGURE 10.4 Affordability cost analysis framework. Derived from Bobinis et al. (2010). Reprinted with permission from Joseph Bobinis. All other rights reserved.

performance that can be achieved. (*Note: In the real world, the curve is rarely smooth or continuous.*) Similarly, for a given performance, the corresponding point on the curve is the minimum cost for which that performance can be achieved. Selecting the decision criterion as cost will result in achieving the threshold performance. Similarly, if the decision criterion is performance, all of the budget would be expended. Consequently, to specify affordability for a system or program requires determining which affordability element is the basis for the decision criteria and which elements are being specified as constraints.

Affordability is the result of a disciplined decision-making process—requiring systematic methodologies that support selection of the most affordable technologies and systems.

10.1.2 Cost-Effectiveness Analysis

As mentioned in the preceding section, a systems engineer can no longer afford the luxury of ignoring cost as an SE area of responsibility or as a major architectural driver. In essence, a systems engineer must be conversant in business and economics as well as engineering.

Cost-effectiveness analysis (CEA) is a form of business analysis that compares the relative costs and performance characteristics of two or more courses of action. At the system level, CEA helps derive critical system performance and design requirements and supports data-based decision making.

CEA begins with clear goals and a set of alternatives for reaching those goals. Comparisons should only be made for alternatives that have similar goals. A straightforward CEA cannot compare options with different goals and objectives.

Experimental or quasiexperimental designs can be used to determine effectiveness and should be of a quality capable of justifying reasonably valid conclusions. If not, there is nothing in the CEA method that will rescue the results. What CEA adds is the ability to consider the results of different alternatives relative to the costs of achieving those results. It does not change the criteria for what is a good effectiveness study. Alternatives being assessed should address a common specific goal

where attainment of that goal can be measured such as miles per gallon, kill radius, or people served.

CEA is distinct from cost–benefit analysis (CBA), which assigns a monetary value to the measure of effect. The approach to measuring costs is similar for both techniques, but in contrast to CEA where the results are measured in performance terms, CBA uses monetary measures of outcomes. This approach has the advantage of being able to compare the costs and benefits in monetary values for each alternative to see if the benefits exceed the costs. It also enables a comparison among projects with very different goals as long as both costs and benefits can be placed in monetary terms.

Other closely related, but slightly different, formal techniques include cost–utility analysis, economic impact analysis, fiscal impact analysis, and social return on investment (SROI) analysis.

In both CEA and CBA, the cost of risk and the risk of cost need to be included into the study. Risk is usually handled using probability theory. This can be factored into the discount rate (to have uncertainty increasing over time), but is usually considered separately. Particular consideration is often given to risk aversion—the irrational preference to avoid loss over achieving gain. Uncertainty in parameters (as opposed to risk of project failure) can be evaluated using sensitivity analysis, which shows how results and cost respond to parameter changes. Alternatively, a more formal risk analysis can be undertaken using Monte Carlo simulations. Subject matter experts in risk and cost should be consulted.

The concept of cost-effectiveness is applied to the planning and management of many types of organized activity. It is widely used in many aspects of life. Some examples are:

1. Studies of the desirable performance characteristics of commercial aircraft to increase an airline's market share at lowest overall cost over its route structure (e.g., more passengers, better fuel consumption)
2. Urban studies of the most cost-effective improvements to a city's transportation infrastructure (e.g., buses, trains, motorways, and mass transit routes and departure schedules)
3. In health services, where it may be inappropriate to monetize health effect (e.g., years of life, premature births averted, sight years gained)
4. In the acquisition of military hardware when competing designs are compared not only for purchase price but also for such factors as their operating radius, top speed, rate of fire, armor protection, and caliber and armor penetration of their guns

10.1.3 LCC Analysis

LCC refers to the total cost incurred by a system, or product, throughout its life. This "total" cost varies by circumstances, the stakeholders' points of view, and the product. For example, when you purchase an automobile, the major cost factors are the cost of acquisition, operation, maintenance, and disposal (or trade-in value). A more expensive car (acquisition cost) may have lower LCC because of lower operation and maintenance costs and greater trade-in value. However, if you are the manufacturer, other costs like development and production costs, including setting up the production line, need to be considered. The systems engineer needs to look at costs from several aspects and be aware of the stakeholders' perspectives. In some literature, LCC is equated to total cost of ownership (TCO) or total ownership cost (TOC), but many times, these measures only include costs once the systems is purchased or acquired.

Sometimes, it is argued that the LCC estimates are only to support internal program trade-off decisions and, therefore, must only be accurate enough to support the trade-offs (relative accuracy) and not necessarily realistic. By itself, this is usually a bad practice and, if done, is a risk element that should be tracked for some resolution of veracity. The analyst should always attempt to prepare as accurate cost estimates as possible and assign risk as required. These estimates are often reviewed by upper management and potential stakeholders. The credibility of results is significantly enhanced if reviewers sense the costs are "about right," based on their past experience. Future costs, while unknown, can be predicted based on assumptions and risk assigned. All assumptions when doing LCC analysis should be documented.

LCC analysis can be used in affordability and system cost-effectiveness assessments. The LCC is *not* the definitive cost proposal for a program since LCC "estimates" (based on future assumptions) are often prepared early in a program's life cycle when there is insufficient detailed design information. Later, LCC estimates should be updated with actual costs from early program stages

and will be more definitive and accurate due to hands-on experience with the system. A major purpose of LCC studies is to help identify cost drivers and areas in which emphasis can be placed during the subsequent substages to obtain the maximum cost reduction. Accuracy in the estimates will improve as the system evolves and the data used in the calculation is less uncertain.

LCC analysis helps the project team understand the total cost impact of a decision, compare between program alternatives, and support trade studies for decisions made *throughout* the system life cycle. LCC normally includes the following costs, represented in Figure 10.5:

- *Concept costs*—Costs for the initial concept development efforts. Can usually be estimated based on average manpower and schedule spans and include overhead, general and administrative (G&A) costs, and fees, as necessary.
- *Development costs*—Costs for the system development efforts. Similar to concept costs, can usually be estimated based on average manpower and schedule spans and include overhead, G&A costs, and fees, as necessary.
- *Production costs*—Usually driven by tooling and material costs for large-volume systems. Labor cost estimates are prepared by estimating the cost of the first production unit and then applying learning curve formula to determine the reduced costs of subsequent production units. For an item produced with a 90% learning curve, each time the production lot size doubles (2, 4, 8, 16, 32, … etc.) the average cost of units in the lot is 90% of the average costs of units in the previous lot. A production cost specialist is usually required to estimate the appropriate learning curve factor(s).
- *Utilization and support costs*—Typically based on future assumptions for ongoing operation and maintenance of the system, for example, fuel costs, manning levels, and spare parts.
- *Retirement costs*—The costs for removing the system from operation and includes an estimate of trade-in or salvage costs. Could be positive or negative and should be mindful of environmental impact to dispose.

Common methods/techniques for conducting LCC analysis are as follows:

- *Expert judgment*—Consultation with one or more experts. Good for sanity check, but may not be sufficient.
- *Analogy*—Reasoning by comparing the proposed project with one or more completed projects that are judged to be similar, with corrections added for known differences. May be acceptable for early estimations.

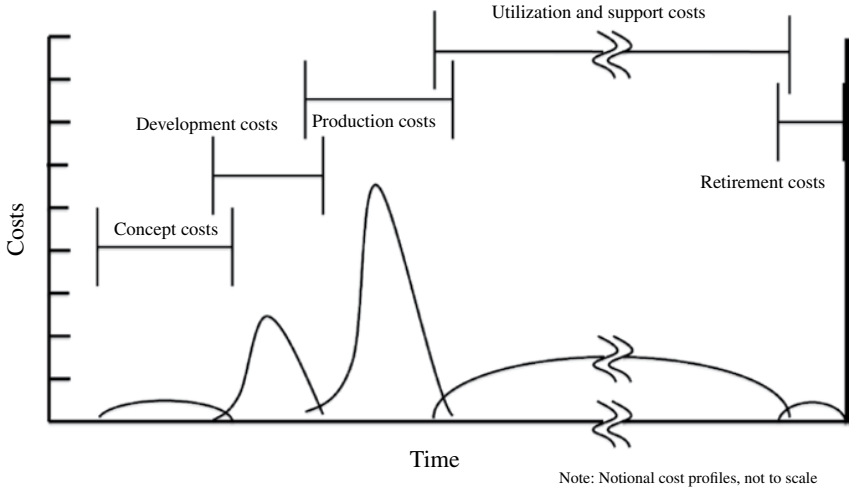


FIGURE 10.5 Life cycle cost elements (not to scale). Derived from INCOSE SEH v1 Figure 9.3. Usage per the INCOSE Notices page. All other rights reserved.

- *Parkinson technique*—Defines work to fit the available resources.
- *Price to win*—Focuses on providing an estimate, and associated solution, at or below the price judged necessary to win the contract.
- *Top focus*—Based on developing costs from the overall characteristics of the project from the top level of the architecture.
- *Bottoms up*—Identifies and estimates costs for each element separately and sums the contributions.
- *Algorithmic (parametric)*—Uses mathematical algorithms to produce cost estimates as a function of cost-driver variables, based on historical data. This technique is supported by commercial tools and models.
- *DTC*—Works on a design solution that meet a predetermined production cost.
- *Delphi techniques*—Builds estimates from multiple technical and domain experts. Estimates are only as good as the experts.
- *Taxonomy method*—Hierarchical structure or classification scheme for the architecture.

10.2 ELECTROMAGNETIC COMPATIBILITY

EMC is the engineering discipline concerned with the behavior of a system in an electromagnetic (EM) environment. A system is considered to be electromagnetically compatible when it can operate without malfunction in an EM environment together with other systems or system elements and when it does not add to that environment as to cause malfunction to other systems or system elements. When a system causes interference, the term electromagnetic interference (EMI) is often used. In EMC, the EM environment not only includes all phenomena and effects that are classically attributed to electromagnetics (such as radiation) but also electrical effects (conduction).

Successfully achieving EMC during system development requires a typical SE process, as shown in Figure 10.6.

10.2.1.1 Electric and Electromagnetic Environmental Effects Analysis An electric and electromagnetic environmental effects (E^4) analysis describes all the threats (natural and man-made) that a system may encounter during its life cycle. MIL-STD-464C (DoD, 2010) can be used to guide this analysis, which should contain all the information needed to determine EMC requirements of the system.

10.2.1.2 EMC Requirements (Standards and Specifications) EMC standards and specifications are used to regulate the EM environment in which a system is operating. It usually governs both the system's ability to function within its intended EM environment (sensitivity or susceptibility) and its contribution to that environment (emissions).

Standards and specifications are available for conducted emissions, conducted susceptibility, radiated emissions, and radiated susceptibility.

It is rare for a system to have custom-developed EMC requirements that do not follow existing standards and specifications. However, existing standards and

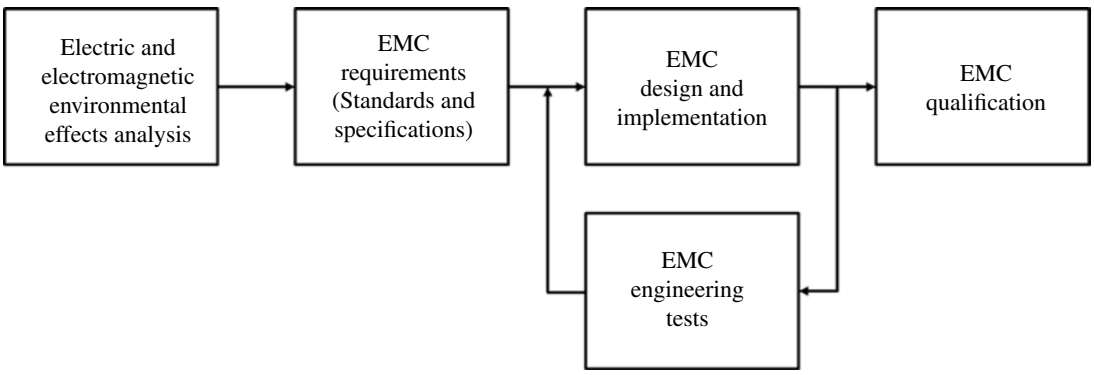


FIGURE 10.6 Process for achieving EMC. Reprinted with permission from Arnold de Beer. All other rights reserved.

specifications (whether commercial, military, avionic, automotive, or medical) classify a requirement into a class or category depending on the severity of the potential malfunction. It is a SE function to determine the correct EMC requirements with class or category according to the outcome of the E^4 analysis.

10.2.3.1 EMC Design and Implementation The EMC requirements are inputs to the concept and development stages. It is important that the EMC requirements are fixed at the beginning of physical design as the EMC design includes both mechanical and electrical/electronic hardware implementations that are not part of the other functional requirements.

For any EMC design, it is best to follow a process of zoning, where a system is divided into zones that can easily be managed for EMC. Typically, major system elements are grouped together in zones that are low in emissions or of similar emissions. Very sensitive circuits (typically analog/measurement circuits) are grouped together and protected. The interfaces between zones are controlled. Where connections interface between zones, filters are used. Where there is a change in radiated interference, screening and/or physical separation is employed.

A structured approach to the control of interference during the design stage of a system is to develop an EMI control plan. The EMI control plan typically includes all EMC requirements, zoning strategy, filtering and shielding, cabling, and detail mechanical and electrical design pertaining to EMC.

10.2.1.4 EMC Engineering Tests Prequalification tests may be required during the development stage. This is typically done on a system element level and even as low as single printed circuit board assembly level. Since EMC results are difficult to predict, the best way of ensuring design success is to test at lower system levels for compliance in order to maximize the probability of system compliance.

10.2.1.5 EMC Qualification EMC qualification tests are performed to verify the EMC design of a system against its requirements. The first part of this activity is to compile an EMC test plan, which maps each requirement to a test and test setup.

The EMC test setup is an integral part of EMC SE as test results can vary according to the setup. This setup can be challenging as the system or system element under test must be in operational mode during emission testing and it must be possible to detect malfunctions during susceptibility testing. Interfacing with the system or system element must be done while not compromising the EM zone in which it is tested. This may require special system-related EMC test equipment. When it is impractical to test a large system (such as a ship, aircraft, or complete industrial plant), the qualification tests of the system elements are used to qualify the larger system.

10.3 ENVIRONMENTAL ENGINEERING/ IMPACT ANALYSIS

The European Union, the United States, and many other governments recognize and enforce regulations that control and restrict the environmental impact that a system may inflict on the biosphere. Such impacts include emissions to air, water, and land and have been attributed to cause problems such as eutrophication, acidification, soil erosion and nutrient depletion, loss of biodiversity, and damage to ecosystems (UNEP, 2012). The focus of environmental impact analysis is on potential harmful effects of a proposed system's development, production, utilization, support, and retirement stages. All governments that have legally expressed their concern for the environment restrict the use of hazardous materials (e.g., mercury, lead, cadmium, chromium 6, and radioactive materials) with a potential to cause human disease or to threaten endangered species through loss of habitat or impaired reproduction. Concern extends over the full life cycle of the system, from the materials used and scrap waste from the production process, operations of the system replacement parts, and consumables and their containers to final disposal of the system. These concerns are made evident by the European Union's 2006 resolution to adopt a legal restriction that system developers and their suppliers retain lifetime liability for decommissioning systems that they build and sell.

The ISO 14000 series of environmental management standards (ISO, 2004) are an excellent resource for organizations of methods to analyze and assess their operations and their impacts on the environment. Failure to comply with environmental protection laws carries penalties and should be addressed in the earliest phases of requirements analysis (Keoleian and Menerey, 1993). The Øresund Bridge (see Section 3.6.2) is an example of

how early analysis of potential environmental impacts ensures that measures are taken in the design and construction to protect the environment with positive results. Two key elements of the success of this initiative were the continual monitoring of the environmental status and the integration of environmental concerns into the requirements from the owner.

Disposal analysis is a significant analysis area within environmental impact analysis. Traditional landfills for nonhazardous solid wastes have become less available within large city areas, and disposal often involves transporting the refuse to distant landfills at considerable expense. The use of incineration for disposal is often vigorously opposed by local communities and citizen committees and poses the problem of ash disposal since the ash from incinerators is sometimes classified as hazardous waste. Local communities and governments around the world have been formulating significant new policies to deal with the disposal of nonhazardous and hazardous wastes.

One goal of the architecture design is to maximize the economic value of the residue system elements and minimize the generation of waste materials destined for disposal. Because of the potential liability that accompanies the disposal of hazardous and radioactive materials, the use of these materials is carefully reviewed and alternatives used wherever and whenever possible. The basic tenet for dealing with hazardous waste is the "womb-to-tomb" control and responsibility for preventing unauthorized release of the material to the environment. This may include designing for reuse, recycling, or transformation (e.g., composing, biodegradation).

In accordance with the US and European Union laws, system developers and supporting manufacturers must analyze the potential impacts of the systems that they construct and must submit the results of that analysis to government authorities for review and approval to build the system. Failure to conduct and submit the environmental impact analysis can result in severe penalties for the system developer and may result in an inability to build or deploy the system. It is best when performing environmental impact analysis to employ subject matter experts who are experienced in conducting such assessments and submitting them for government review. Methods associated with life cycle assessment (LCA) and life cycle management (LCM) are increasingly sophisticated and supported by software (Magerholm et al., 2010). Government acquisitions are subject to legislation for Green Public Procurement (GPP) (Martin, 2010). Consumers of commercial products are offered assistance in their purchasing decisions through Environmental Product Declarations and labeling, such as the Nordic Swan, and Blue Angel (Salzman, 1997). Another effort in the ISO community is the development of a standard for product carbon footprints as an indicator of the global environmental impact of a product expressed in carbon emission equivalents (Draucker et al., 2011).

10.4 INTEROPERABILITY ANALYSIS

Interoperability depends on the compatibility of elements of a large and complex system (which may be an SoS or a family of systems (FoS)) to work as a single entity. This feature is increasingly important as the size and complexity of systems continue to grow. Pushed by an inexorable trend toward electronic digital systems and pulled by the accelerating pace of digital technology invention, commercial firms and national organizations span the world in increasing numbers. As their spans increase, these commercial and national organizations want to ensure that their sunken investment in legacy elements of the envisioned new system is protected and that new elements added over time will work seamlessly with the legacy elements to form a unified system.

Standards have also grown in number and complexity over time, yet compliance with standards remains one of the keys to interoperability. The standards that correspond to the layers of the *ISO-OSI Reference Model* for peer-to-peer communication systems once fit on a single wall chart of modest size. Today, it is no longer feasible to identify the number of standards that apply to the global communications network on a wall chart of any size. Interoperability will increase in importance as the world grows smaller due to expanding communications networks and as nations continue to perceive the need to communicate seamlessly across international coalitions of commercial organizations or national defense forces.

The Øresund Bridge (see Section 3.6.2) demonstrates the interoperability challenges faced when just two nations collaborate on a project, for example, the meshing of regulations on health and safety and the resolution of two power supply systems for the railway.

10.5 LOGISTICS ENGINEERING

Logistics engineering (Blanchard and Fabrycky, 2011), which may also be referred to as product support engineering, is the engineering discipline concerned with the identification, acquisition, procurement, and provisioning of all support resources required to sustain operation and maintenance of a system. Logistics should be addressed from a life cycle perspective and be considered in all stages of a program and especially as an inherent part of system concept definition and development. The emphasis on addressing logistics in these stages is based on the fact that (through past experience) a significant portion of a system's LCC can be attributed directly to the operation and support of the system in the field and that much of this cost is based on design and management decisions made during early stages of system development. Furthermore, logistics should be approached from a system perspective to include all activities associated with design for supportability, the acquisition and procurement of the elements of support, the supply and distribution of required support material, and the maintenance and support of systems throughout their planned period of utilization.

The scope of logistics engineering is thus (i) to determine logistics support requirements, (ii) to design the system for supportability, (iii) to acquire or procure the support, and (iv) to provide cost-effective logistics support for a system during the utilization and support stages (i.e., operations and maintenance). Logistics engineering has evolved into a number of related elements such as supply chain management (SCM) in the commercial sector and integrated logistics support (ILS) in the defense sector. Further logistics engineering developments include acquisition logistics and performance-based logistics. Logistics engineering is also closely related to reliability, availability, and maintainability (RAM) (refer to Section 10.8), since these attributes play an important role in the supportability of a system.

10.5.1 Support Elements

Support of a system during the utilization and support stages requires personnel, spares and repair parts, transportation, test and support equipment, facilities, data and documentation, computer resources, etc. Support planning starts with the definition of the support and maintenance concept (in the concept stage) and continues through supportability analysis (in the development stage) to the ultimate development of a maintenance plan. Planning, organization, and management activities are necessary to ensure that the logistics requirements for any given program are properly coordinated and implemented and that the following elements of support are fully integrated with the system:

- *Product support integration and management*—Plan and manage cost and performance across the product support value chain, from the concept to retirement stages.
- *Design interface*—Participate in the SE process to impact the design from inception throughout the life cycle. Facilitate supportability to maximize availability, effectiveness, and capability at the lowest LCC. Design interface evaluates all facets of the product from design to fielding, including the product's operational concept for support impacts and the adequacy of the support infrastructure. Prior to the establishment of logistics requirements, logistics personnel accomplish planning, trade-offs, and analyses to provide a basis for establishing support requirements and subsequent resources. These include support system effectiveness inputs to the system specifications and goals and integration of reliability and maintainability program requirements. Consideration of support alternatives and design into conceptual programs must be initiated at this early stage for effective problem identification and resolution. Logistics personnel conduct analyses to assist in identifying potential postproduction support problems and to contribute to possible LCC and support solutions.
- *Sustaining engineering*—This effort spans those technical tasks (engineering and logistics investigations and analyses) to ensure continued operation and maintenance of a system with managed (i.e., known) risk. Technical surveillance of critical safety items, approved sources for these items, and the oversight of the design configuration baselines (basic design engineering responsibility for the overall configuration including design packages, maintenance procedures, and usage profiles) for the fielded system to ensure continued certification compliance are also part of the sustaining engineering effort. Periodic technical review of the

in-service system performance against baseline requirements, analysis of trends, and development of management options and resource requirements for resolution of operational issues should be part of the sustaining effort.

- *Maintenance planning*—Identify, plan, fund, and implement maintenance concepts and requirements to ensure the best possible system capability is available for operations, when needed, at the lowest possible LCC. The support concept describes the support environment in which the system will operate. It also includes information related to the system maintenance concept: the support infrastructure, expected durations of support, reliability and maintainability rates, and support locations. The support concept is the foundation that drives the maintenance planning process. Establish general overall repair policies such as "repair or replace" criteria.
- *Operation and maintenance personnel*—Identify, plan, fund, and acquire personnel, with the training, experience, and skills required to operate, maintain, and support the system.
- *Training and training support*—Plan, fund, and implement a strategy to train operators and maintainers across the system life cycle. As part of the strategy, plan, fund, and implement actions to identify, develop, and acquire Training Aids, Devices, Simulators, and Simulations (TADSS) to maximize the effectiveness of the personnel to operate and sustain the system equipment at the lowest LCC.
- *Supply support*—Consists of all actions, procedures, and techniques necessary to determine requirements to acquire, catalog, receive, store, transfer, issue, and dispose of spares, repair parts, and supplies. This means having the right spares, repair parts, and all classes of supplies available, in the right quantities, at the right place, at the right time, at the right price. The process includes provisioning for initial support, as well as acquiring, distributing, and replenishing inventories.
- *Computer resources (hardware and software)*—Computers, associated software, networks, and interfaces necessary to support all logistics functions. Includes resources and technologies necessary to support long-term data management and storage.
- *Technical data*, *reports*, *and documentation*—Represents recorded information of scientific or technical nature, regardless of form or character (such as equipment technical manuals and engineering drawings), engineering data, specifications, and standards. Procedures, guidelines, data, and checklists needed for proper operations and maintenance of the system, including:
  - System installation procedures
  - Operating and maintenance instructions
  - Inspection and calibration procedures
  - Engineering design data
  - Logistics provisioning and procurement data
  - Supplier data
  - System operational and maintenance data
  - Supporting databases
- *Facilities and infrastructure*—Facilities (e.g., buildings, warehouses, hangars, waterways, etc.) and infrastructure (e.g., IT services, fuel, water, electrical service, machine shops, dry docks, test ranges, etc.) required to support operation and maintenance.
- *Packaging*, *handling*, *storage*, *and transportation* (*PHS&T*)—The combination of resources, processes, procedures, design, considerations, and methods to ensure that all system, equipment, and support items are preserved, packaged, handled, and transported properly, including environmental considerations, equipment preservation for the short and long storage, and transportability. Some items require special environmentally controlled, shock-isolated containers for transport to and from repair and storage facilities via all modes of transportation (land, rail, sea, air, and space).
- *Support equipment*—Support equipment consists of all equipment (mobile or fixed) required to sustain the operation and maintenance of a system. This includes, but is not limited to, ground handling and maintenance equipment, trucks, air conditioners, generators, tools, metrology and calibration equipment, and manual and automatic test equipment.

10.5.2 Supportability Analysis

Supportability analysis is an iterative analytical process by which the logistics support requirements for a system are identified and evaluated. It uses quantitative methods to aid in (i) the initial determination and establishment of

supportability requirements as an input to design; (ii) the evaluation of various design options; (iii) the identification, acquisition, procurement, and provisioning of the various elements of maintenance and support; and (iv) the final assessment of the system support infrastructure throughout the utilization and support stages.

Supportability analysis constitutes a design analysis process that is part of the overall SE effort. Functional analysis is used to define all system functions early in the concept stage and to appropriate levels in the system hierarchy. The resulting functional breakdown structure, together with the system requirements and the support and maintenance concept, provides the starting point of a supportability analysis as shown in Figure 10.7.

Supportability analysis may include analyses such as FMECA, fault tree analysis (FTA), reliability block diagram (RBD) analysis, Maintenance Task Analysis (MTA), RCM, and LORA. Products and activities identified in Figure 10.7 are described as follows:

- *Functional failure analysis*—A functional breakdown structure is used as reference to perform functional FMECA and/or FTA and RBD analysis. These analyses can be used to identify functional failure modes and to classify them according to criticality (i.e., severity of failure effects and probability of occurrence). The functional failure analysis can also provide valuable system design input (e.g., redundancy requirements).
- *Physical definition*—During system design, the physical breakdown structure of a system should be developed to assist in identifying the actual location of items in the system. This breakdown structure is used as baseline at various stages throughout the life cycle of the system, and it should be continuously updated and refined to the required level of detail.
- *Physical failure analysis*—The physical breakdown structure is used as reference to perform hardware FMECA and/or FTA and RBD analysis with the objective of identifying all maintenance tasks for potential failure modes. The criticalities of failure modes are used to prioritize corrective and preventive maintenance task requirements.
- *Task identification and optimization*—Corrective maintenance tasks are primarily identified using FMECA, while preventive maintenance tasks are identified using RCM. Trade-off studies may be required to achieve an optimized maintenance strategy.
- *Detail task analysis*—Detail procedures for corrective and preventive maintenance tasks should be developed, and support resources identified and allocated to each task. A LORA may be used to determine the most appropriate location for executing these tasks.
- *Support element specifications*—Support element specifications should be developed for all support

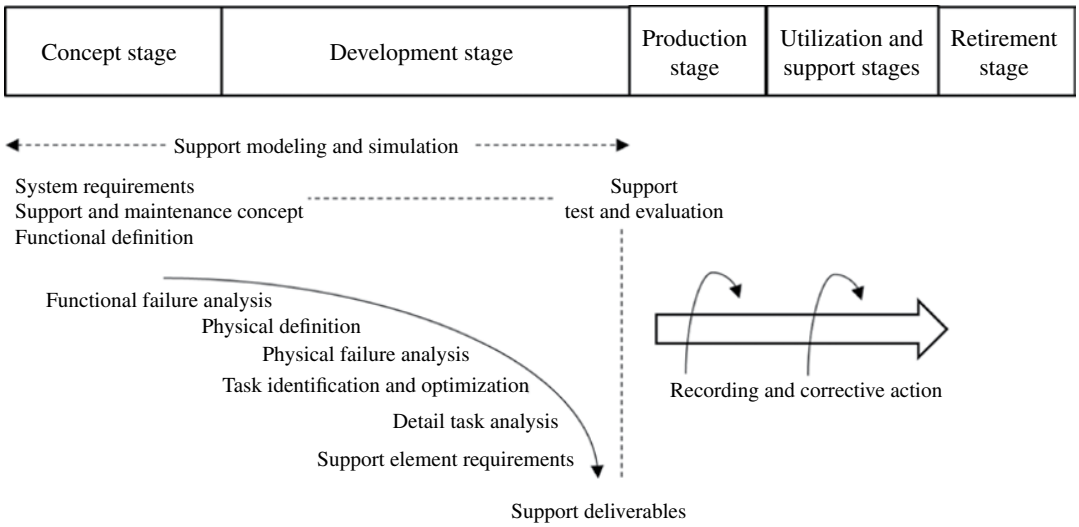


FIGURE 10.7 Supportability analysis. Reprinted with permission from Corrie Taljaard. All other rights reserved.

deliverables. Depending on the system, specifications may be required for training aids, support equipment, publications, and packaging material.

- *Support deliverables*—All support deliverables should be acquired or procured based on the individual specifications. Support element plans describing the management procedures for the support elements should also be developed.
- *Support modeling and simulation*—RAM and LCC modeling and simulation are integral parts of supportability analysis that should be initiated during the early stages to develop an optimized system design, maintenance strategy, and support system.
- *Support test and evaluation*—The support deliverables should be tested and evaluated against both support element specifications and the overall system requirements.
- *Recording and corrective action*—Failure recording and corrective action during the utilization and support stages form the basis for continuous improvement. System availability metrics should be used to continuously monitor the system in order to improve support where deficiencies are identified.

10.6 MANUFACTURING AND PRODUCIBILITY ANALYSIS

The capability to manufacture or produce a system element is as essential as the ability to properly define and design it. A designed product that cannot be manufactured causes design rework and program delays with associated cost overruns. For this reason, producibility analysis and trade studies for each design alternative form an integral part of the architectural design process. One objective is to determine if existing proven processes are satisfactory since this could be the lowest risk and most cost-effective approach. The Maglev train contractor (see Section 3.6.3) experienced a steep learning curve to produce an unprecedented system from scientific theory.

Producibility analysis is a key task in developing low-cost, quality products. Multidisciplinary teams work to simplify the design and stabilize the manufacturing process to reduce risk, manufacturing cost, lead time, and cycle time and to minimize strategic or critical material use. Critical producibility requirements are identified during system analysis and design and included in the program risk analysis, if necessary. Similarly, long-lead-time items, material limitations, special processes, and manufacturing constraints are evaluated. Design simplification also considers ready assembly and disassembly for ease of maintenance and preservation of material for recycling. When production engineering requirements create a constraint on the design, they are communicated and documented. The selection of manufacturing methods and processes is included in early decisions.

Manufacturing analyses draw upon the production concept and support concept. Manufacturing test considerations are shared with the engineering team and are taken into account in built-in test and automated test equipment.

IKEA® is often used as an example of supply chain excellence. IKEA® has orchestrated a value creating chain that begins with motivating customers to perform the final stages of furniture assembly in exchange for lower prices and a fun shopping experience. They achieve this through designs that support low-cost production and transportability (e.g., the bookcase that comes in a flat package and goes home on the roof of a car).

10.7 MASS PROPERTIES ENGINEERING

Mass Properties Engineering (MPE) ensures that the system or system element has the appropriate mass properties to meet the requirements (SAWE). Mass properties include weight, the location of center of gravity, inertia about the center of gravity, and product of the inertia about an axis.

Typically, the initial sizing of the physical system is derived from other requirements, such as minimum payload, maximum operating weight, or human factor restrictions. Mass properties estimates are made at all stages of the system life cycle based on the information that is available at the time. This information may range from parametric equations to a three-dimensional product model to actual inventories of the product in service. A risk assessment is conducted, using techniques such as uncertainty analysis or Monte Carlo simulations, to verify that the predicted mass properties of the system will meet the requirements and that the system will operate within its design limits. MPE is conducted at the end of the production stage to assure all parties that the

delivered system meets the requirements and then several times during the utilization stage to ensure the safety of the system, system element, or human operator. For a large project, such as oil platform or warship, the MPE level of effort is significant.

One trap in MPE is believing that three-dimensional modeling tools can be exclusively used to estimate the mass properties of the system or system element. This is problematic because (i) not all parts are modeled on the same schedule and (ii) most parts are modeled neat, that is, without such items as manufacturing tolerances, paint, insulation, fittings, etc., which can add from 10 to 100% to the system weight. For example, the liquid in piping and tanks can weigh more than the structural tank or metallic piping that contain it.

MPE usually includes a reasonableness check of all estimates using an alternative method. The simplest method is to justify the change between the current estimate and any prior estimates for the same system or the same system element on another project. Another approach is to use a simpler estimating method to repeat the estimate and then justify any difference.

10.8 RELIABILITY, AVAILABILITY, AND MAINTAINABILITY

To be reliable, a system must be robust—it must avoid failure modes even in the presence of a broad range of conditions including harsh environments, changing operational demands, and internal deterioration (Clausing and Frey, 2005). Reliability can thus be seen as the proper functioning of a system during its expected life under the full range of conditions experienced in the field.

Reliability engineering refers to the specialized engineering discipline that addresses the reliability of a system during its total life cycle. It includes related aspects such as availability and maintainability of a system. Therefore, reliability engineering is often used as collective term for the engineering discipline concerned with the RAM of a system.

RAM are important attributes or characteristics of a given system. However, RAM should not actually be viewed as characteristics, but rather as nonfunctional requirements. It is therefore essential that SE processes should include RAM activities, selected, planned, and executed in an integrated manner with other technical processes.

Reliability engineering activities support other SE processes in two ways. Firstly, reliability engineering activities should be used to influence system design (e.g., the system architecture depends on reliability requirements). Secondly, reliability engineering activities should be used as part of system verification (e.g., system analysis or system test).

10.8.1 Reliability

The objectives of reliability engineering, in the order of priority, are (O'Connor and Kleyner, 2012):

- To apply engineering knowledge and specialist techniques to prevent or to reduce the likelihood or frequency of failures
- To identify and correct the causes of failures that do occur, despite the efforts to prevent them
- To determine ways of coping with failures that do occur, if their causes have not been corrected
- To apply methods for estimating the likely reliability of new designs and for analyzing reliability data

The priority emphasis is important, since proactive prevention of failure is always more cost-effective than reactive correction of failure. Timely execution of appropriate reliability engineering activities is of utmost importance in achieving the required reliability during operations.

Traditionally, reliability has been defined as *the probability that an item will perform a required function without failure under stated conditions for a stated period of time* (O'Connor and Kleyner, 2012). The emphasis on probability in the definition of reliability (to quantify reliability) resulted in a number of potentially misleading or even incorrect practices (e.g., reliability prediction and reliability demonstration of electronic systems).

Modern approaches to reliability place more emphasis on the engineering processes required to prevent failure during the expected life of a system (i.e., failure-free operation). The concept of "design for reliability" has recently shifted the focus from a reactive "test–analyze–fix" approach to a proactive approach of designing reliability into the system. "Failure mode avoidance" approaches are aligned with other SE processes and attempt to improve reliability of a system early in the

development stages (Clausing and Frey, 2005). It is performed by evaluating system functions, technology maturity, system architecture, redundancy, design options, etc. in terms of potential failure modes. The most significant improvements in system reliability can be achieved by avoiding physical failure modes in the first place and not by minor improvements after the system has been conceived, designed, and produced.

"Design for reliability" implies that reliability should be specified as a requirement in order to receive adequate attention during requirements analysis. Reliability requirements may be specified either in qualitative or quantitative terms, depending on the specific industry. Care should be taken with quantitative requirements, since verification of reliability is often not practical (especially for high reliability requirements). Also, the misuse of reliability metrics (e.g., mean time between failure (MTBF)) frequently results in "playing the numbers game" during system development, instead of focusing on the engineering effort necessary to achieve reliability (Barnard, 2008). For example, MTBF is often used as an indicator of "average life" of an item, which may be completely incorrect. It is therefore recommended that other reliability metrics be used for quantitative requirements (e.g., reliability (as success probability) at a specific time).

10.8.1.1 Development of a Reliability Program Plan Reliability engineering activities are often neglected during system development, resulting in a substantial increase in risk of project failure or customer dissatisfaction. It is therefore recommended that reliability engineering activities be formally integrated with other SE technical processes. A practical way to achieve integration is to develop a Reliability Program Plan at the start of the project.

Appropriate reliability engineering activities should be selected and tailored according to the objectives of the specific project. These activities should be captured in the Reliability Program Plan. The plan should indicate which activities will be performed, the planned timing of the activities, the level of detail required for the activities, and the persons responsible for execution of the activities.

ANSI/GEIA-STD-0009-2008, *Reliability program standard for systems design, development, and manufacturing*, can be referenced for this purpose. This standard addresses not only hardware and software failures but also other common failure causes such as manufacturing, operator error, operator maintenance, training, quality, etc. "At the heart of the standard is a systematic 'design-reliability-in' process, which includes three elements:

- Progressive understanding of system-level operational and environmental loads and the resulting loads and stresses that occur throughout the structure of the system.
- Progressive identification of the resulting failure modes and mechanisms.
- Aggressive mitigation of surfaced failure modes."

ANSI/GEIA-STD-0009-2008, which supports a system life cycle approach to reliability engineering, consists of the following objectives:

- Understand customer/user requirements and constraints.
- Design and redesign for reliability.
- Produce reliable systems/products.
- Monitor and assess user reliability.

The Reliability Program Plan thus provides a forward-looking view on how to achieve reliability objectives. Complementary to the Reliability Program Plan is the Reliability Case that provides a retrospective (and documented) view on achieved objectives during the system life cycle.

Figure 10.8 indicates a few relevant questions that may be used to develop a Reliability Program Plan for a specific project.

10.8.1.2 Reliability Engineering Activities Reliability engineering activities can be divided into two groups, namely, engineering analyses and tests and failure analyses. These activities are supported by various reliability management activities (e.g., design procedures, design checklists, design reviews, electronic part derating guidelines, preferred parts lists, preferred supplier lists, etc.).

Engineering analyses and tests refer to traditional design analysis and test methods to perform, for example, load–strength analysis during design. Included in this group are finite element analysis, vibration and shock analysis, thermal analysis and measurement, electrical stress analysis, wear-out life prediction, highly accelerated life testing (HALT), etc.

Failure analyses refer to traditional RAM analyses to improve understanding of cause-and-effect relationships

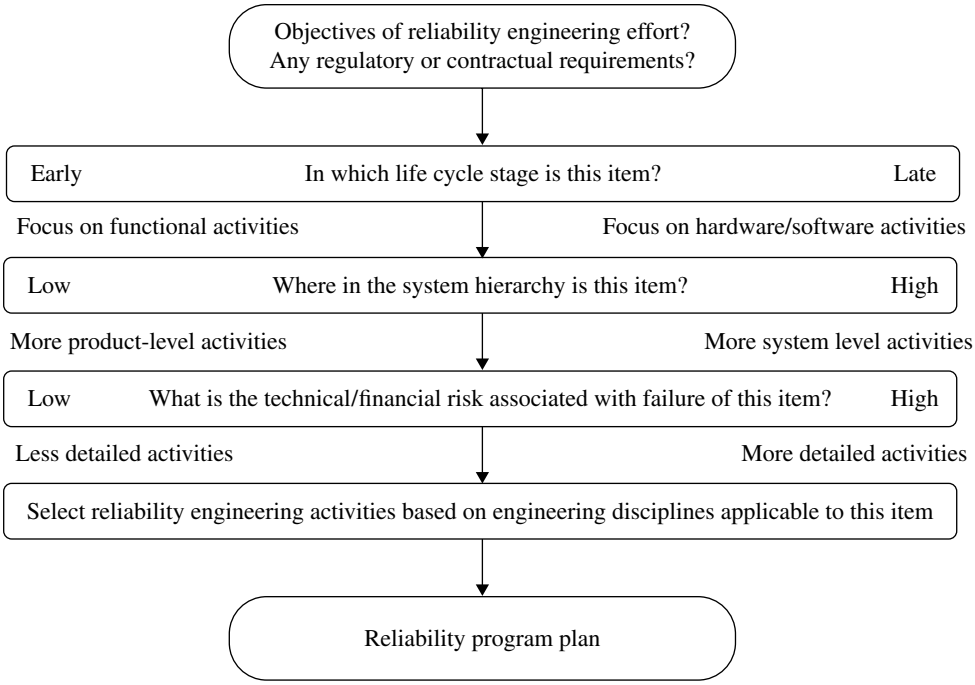


FIGURE 10.8 Reliability Program Plan development. Reprinted with permission from Albertyn Barnard. All other rights reserved.

during design and operations. Included in this group are Failure Mode and Effects Analysis (FMEA), FTA, RBD analysis, systems modeling and simulation, root cause failure analysis, etc.

10.8.2 Availability

Availability is defined as the probability that a system, when used under stated conditions, will operate satisfactorily at any point in time as required. Availability is therefore dependent on the reliability and maintainability of the system, as well as the support environment during the utilization and support stages. It may be expressed and defined as inherent, achieved, or operational availability (Blanchard and Fabrycky, 2011):

- *Inherent availability* (A_i) is based only on the inherent reliability and maintainability of the system. It assumes an ideal support environment (e.g., readily available tools, spares, maintenance personnel) and excludes preventive maintenance, logistics delay time, and administrative delay time.
- *Achieved availability* (A_a) is similar to inherent availability, except that preventive (i.e., scheduled) maintenance is included. It excludes logistics delay time and administrative delay time.
- *Operational availability* (A_o) assumes an actual operational environment and therefore includes logistics delay time and administrative delay time.

10.8.3 Maintainability

An objective in systems engineering is to design and develop a system that can be maintained effectively, safely, in the least amount of time, at the least cost, and with a minimum expenditure of support resources without adversely affecting the mission of that system. Maintainability is the *ability* of a system to be maintained, whereas maintenance constitutes a series of actions to be taken to restore or retain a system in an effective operational state. Maintainability must be inherent or "built into" the design, while maintenance is the result of design.

Maintainability can be expressed in terms of maintenance times, maintenance frequency factors, maintenance

labor hours, and maintenance cost. Maintenance can be broken down into corrective maintenance (i.e., unscheduled maintenance accomplished, as a result of failure, to *restore* a system to a specified level of performance) and preventive maintenance (i.e., scheduled maintenance accomplished to *retain* a system at a specified level of performance by providing systematic inspection and servicing or preventing impending failures through periodic item replacements) (Blanchard and Fabrycky, 2011).

10.8.4 Relationship with Other Engineering Disciplines

Reliability engineering is closely related to other engineering disciplines, such as safety engineering and logistics engineering. The primary objective of reliability engineering is prevention of failure. The primary objective of safety engineering is prevention and mitigation of harm under both normal and abnormal conditions (see Section 10.10). The primary objective of logistics engineering is development of efficient logistics support (e.g., preventive and corrective maintenance; see Section 10.5).

These three disciplines not only have "failure" as common theme, but they may also use similar activities, albeit from different viewpoints. For example, an FMEA may be applicable to reliability, safety, and logistics engineering. However, a design FMEA will be different to a safety or logistics FMEA, due to the different objectives. Common to all disciplines is the necessity of early implementation during the system life cycle.

While reliability is concerned with failures (or rather the absence of failures), maintainability refers to the ability of a system to be maintained (or the ease of maintenance). Availability is a function of both reliability and maintainability and may include logistics aspects (as in the case of operational availability). The LCC of a system is highly dependent on reliability and maintainability, which are considered major drivers in support resources and related in-service costs.

10.9 RESILIENCE ENGINEERING

10.9.1 Introduction

The general definition of resilience is "…the act of rebounding or springing back" (Little et al., 1973). For engineered systems, as defined in this handbook, resilience has taken on the following meaning (Haimes, 2012):

> Resilience is the ability to prepare and plan for, absorb or mitigate, recover from, or more successfully adapt to actual or potential adverse events.

Although this definition can apply to the resilience of any engineered system including both physical assets and humans, early work (Hollnagel et al., 2006) focused on the resilience of organizational systems. Although this definition is widely used, some domains, for example, the military (Richards, 2009), define resilience to include only the recovery phase of a disruption.

Resilience has taken on a particular importance at a governmental level (NRC, 2012; The White House, 2010) with the resilience of infrastructure systems being of the highest priority. Infrastructure systems include fire protection, law enforcement, power, water, healthcare, transportation, telecommunication, and other systems. The principles and practices outlined here can apply to any engineered system. Infrastructure systems are generally SoS as defined in Section 2.4 and pose a particular challenge to achieving resilience arising from the distinctive features of SoS. The SOIs are not limited to safety-critical systems. The resilience in question may apply to the restoration of a service, such as water, power, healthcare, and so forth. Water, power, and healthcare may most likely be safety-critical systems, contributing to safety functions, such as sprinkling systems (water), provision of life (health), and power (supporting safety-critical systems, such as grid, and critical infrastructure).

10.9.2 Description

Resilience pertains to the anticipation, survival, and recovery from a variety of disruptions caused by both human-made and natural threats. External human-originated threats include terrorist attacks. Internal human-originated threats include operator and design error. Natural threats include extreme weather, geological events, wildfires, and so forth. Threats may be single or multiple. Threats confronted after the first in a multiple-threat scenario may result from attempts to correct for the initial threat. Multiple threats may also result from cascading failures, which are common in infrastructure systems.

Resilience is an emergent and nondeterministic property of a system (Haimes, 2012). It is emergent because it cannot be determined by the examination of individual

elements of the system. The entire system and the interaction among the elements must be examined. It is nondeterministic because the wide variety of possible system states at the time of the disruption cannot be characterized either deterministically or probabilistically. Statistical data analysis (extreme amounts) may allow for probabilistic assessment. For example, in reference to Fukushima, data exist on earthquakes and tsunamis to make a quantitative prediction. Moreover, data are available on cooling system configuration and probability of failure under earthquake and tsunami conditions, making it possible to evaluate these events on a probabilistic basis. Because of these emergent and nondeterministic properties, resilience cannot be measured, nor outcomes of particular threats be accurately predicted except by iterative analytical trials of threats and system configurations.

The purpose of engineering a resilient system is to determine the architecture and/or other system characteristics that will anticipate, survive, and recover from a disruption or multiple disruptions. Figure 10.9 is a model of a disruption.

Figure 10.9 shows a disruption occurring in three states: the pre-event initial state, the intermediate states due to the event, and the post-event final state. The figure also shows a feedback loop that represents the multiple-threat scenario. The ability of a system to accomplish these desirable outcomes depends on the application of one or more principles (Jackson and Ferris, 2013). These principles are abstract, allowing a system developer to design specific implementations that, in turn, will result in specific resilience characteristics. Principles can be either scientifically validated rules or heuristics. The characterization of these principles at the abstract level allows them to apply to any domain. The principles must be invoked during one or more of the phases in the diagram in Figure 10.9. The system developer can only determine which principles are preferred in a particular situation by proposing design solutions and modeling their effect. In addition, it has been determined (Jackson and Ferris, 2013) that resilience is achieved when the principles must be implemented in appropriate combinations. Hence, the following principle when implemented in the appropriate combination can be considered an integrated model of resilience. The system developer can develop concrete design proposals by following the reasoning that "an abstraction is a simplified replica of the concrete" (Lonergan, 1992). The top-level abstract principles and their associated dominant characteristics are described in the following text. Subprinciples to these principles can be found in the primary source (Jackson and Ferris, 2013).

The engineering of a resilient system is not a separate discipline. Its principles, listed in the following text, are recognized in other disciplines, for example, architecture design, reliability, and safety. Reliability is a key consideration in safety. They have one thing in common: the ability to enhance the resilience of an engineered system. The goal of each principle is to support a particular attribute or feature of the system that will enhance resilience. The following principles are listed according to the attribute they support:

- *Attribute: Capacity*—the ability to withstand a threat
  - Absorption: System capable of absorbing design threat level.

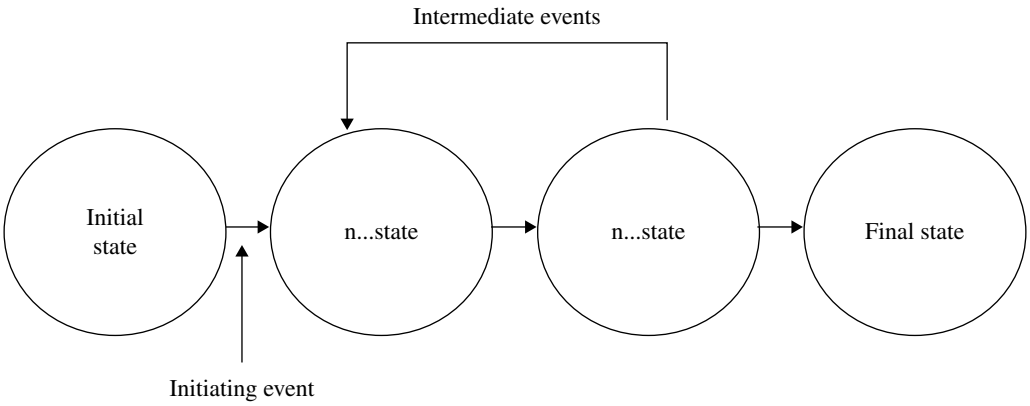


FIGURE 10.9 Resilience event model. Reprinted with permission from Scott Jackson. All other rights reserved.

  - Physical redundancy: System consists of two or more identical and independent branches.
  - Functional redundancy: Also called layered diversity, system consists of two or more different and independent branches and is not vulnerable to common cause failure.
- *Attribute: Buffering*—the ability to maintain a distance from the boundary of unsafe operation or collapse
  - Layered defense: System does not have a single point of failure.
  - Reduce complexity: System capable of reducing the number of elements, interfaces, and/or variability among its elements.
  - Reduce hidden interactions: System capable of detecting undesirable interactions among its elements.
- *Attribute: Flexibility*—the ability to bend or restructure
  - Reorganization: System capable of restructuring itself in the face of a threat
  - Repairability: System capable of repairing itself following a disruption
- *Attribute: Adaptability*—the ability to prevent the system from drifting into unsafe behaviors
  - Drift correction: System capable of detecting approaching threat and performing corrective action.
  - Neutral state: System capable of entering neutral state to allow decisions to be made.
  - Human in the loop: System has human elements where needed.
  - Loose coupling: System resistant to cascading failure by slack and delays at the nodes.
- *Attribute: Tolerance*—the ability to degrade gracefully
  - Localized capacity: Individual elements of a system are capable of independent operation following failure of other elements.
- *Attribute: Cohesion*—the ability of the elements of a system to operate together as a system
  - Internode interaction: System has connections among all its nodes.

Key inputs for resiliency engineering are as follows:

- Threats: number, type, characteristics
- Objectives and priorities
- SOI: type and purpose
- Candidate principles: potentially appropriate for the SOI
- Solution proposals

Key outputs for resiliency engineering are as follows:

- Preferred system characteristics
- System-predicted response to selected threats
- Loss and recovery of function, service, and financial impact
- Recovery time

Key activities of the resiliency engineering process are as follows:

- Create models, including system characteristics and threats.
- Select candidate resilience principles and combinations of principles appropriate to the relevant scenarios.
- Select a measure, or measures, of effectiveness.
- Propose candidate solutions for each principle including inputs and outputs for each system element.
- Model threats for a selected range of types and magnitudes relevant to the scenarios:
  - Identify potential impacts of unanticipated threats.
- Execute the model for the range of threats and relevant system states.
- Conduct an impact analysis to determine the loss and recovery of function, service, or financial impact of the evaluated system.

10.10 SYSTEM SAFETY ENGINEERING

System safety engineering is an applied derivative of SE that builds upon the fundamentals of good systems thinking and applies them analytically through each of the system's life cycle phases. At the core of system safety engineering is the analysis of each of requirement, each system element, and each macro-to-micro behavior within the context of the system being developed, operated, or sustained to identify and eliminate or control safety risk potential. Safety risk potential is defined as any condition that would produce undesired conditions of the system resulting in damage to the system, harm to

the humans involved in the operations and support of the system, or damage to the environment.

The primary objective of system safety engineering is to influence the design with safety-related requirements for the development, production, utilization, support, and retirement stages of a safe system. The benefits of a safe system are numerous and include, but not limited to, the reduction of risk associated with cost, schedule, operational effectiveness, system availability, and legal liability.

10.10.1 The Role of SE in System Safety

As today's systems increase in size and complexity, the SE approaches used become more critical. System-level properties such as safety must be designed into these systems. They cannot be added on afterward and expected to be safe. Systems are designed to achieve specific goals to satisfy the requirements and constraints. SE must develop a means to organize the engineering design process to ensure the goals of system safety are included.

SE must embed the system safety engineering effort into its engineering processes from the beginning such that safety can be designed into the system as engineering design decisions are made. With respect to system safety engineering, SE determines the goals of the system and participates in the identification and documentation of potential hazards to be avoided. From this information, a set of system functional requirements, safety requirements, and constraints can be identified and documented. These requirements will lay the foundation for design and operation of the system to ensure safety is designed into the system. SE must establish system safety engineering from the early concept stages and continue this process throughout the life cycle of the system. SE should ensure design decisions are guided by safety considerations while taking system requirements and constraints into account.

10.10.2 Identify and Integrate System Safety Requirements

System safety engineers review and identify applicable "best practice" system safety design requirements and guidelines from federal, military, national, and industry regulations, codes, standards, and other documents for the system to be designed and developed (e.g., Federal Motor Vehicle Safety Standards (FMVSS), Military Standards (MIL-STDs), National Electrical Code (NEC), and Registration, Evaluation, Authorization, and Restriction of Chemicals (REACH)). These initial requirements are used to derive additional system safety design requirements that are provided to design engineers to eliminate or reduce hazard risks to acceptable levels. These requirements are integrated into the high-level system requirements and design documents. The system safety requirements are then related to the hazards identified in the system.

System safety engineering, along with SE, ensures that the system safety design requirements and guidelines are developed, refined, completely and correctly specified, and properly translated into system element requirements to ensure they are implemented in the design and development of the system hardware, software, and user interface. In addition, applicable safety requirements are identified for incorporation into procedures, processes, warnings, and cautions for use in the operator, user, and diagnostic manuals.

10.10.3 Identify, Analyze, and Categorize Hazards

Within the discipline of system safety engineering, there are numerous analytical methods, techniques, and products that are considered best practice and acceptable within the industry. For example, SAE International possesses specific methods defined in SAE ARP 4754 and 4761 that are commonly used within the aviation industry. The US Department of Defense MIL-STD-882 has also defined specific analysis techniques that are useful in the defense domain (DoD, 2010b). Regardless of the standard or guidance used, the following nonexhaustive list of analysis techniques and safety engineering artifacts reflects SE best practice:

- Preliminary hazard analysis (PHA)
- Functional hazard analysis (FHA)
- System element hazard analysis (SEHA)
- System hazard analysis (SHA)
- Operations and support hazard analysis (O&SHA)
- Health hazard analysis (HHA)
- FTA
- Probabilistic risk assessment (PRA)
- Event tree analysis (ETA)

System safety engineers begin the identification of hazards at the beginning of system concept definition. A

hazard analysis is initiated to identify potential hazards and their mishap potential that may be inherent in the concepts under consideration. The system safety engineer draws from safety experience on similar systems, including mishap/incident hazard tracking logs, safety lessons learned, and design guidelines to develop this list. As the system matures through the development cycle, the hazard analysis is updated to identify and analyze new hazards resulting from design changes.

An in-depth causal analysis is conducted for each identified hazard. This analysis identifies all contributions from hardware, software, and any/all control entities including the human operator, which could cause the hazard to occur. The identification of each causal factor allows the system safety engineer, along with the systems engineer, to identify specific system safety requirements/constraints necessary to mitigate the hazard and reduce the risk to an acceptable level.

Each hazard is analyzed to determine its severity and probability of occurrence. The hazard's severity and probability determine its risk categorization. Table A.I of MIL-STD-882E shows an example of mishap severity categories (DoD, 2010b), and Table A.II shows mishap probability levels. Mishap risk classification is performed by using a mishap risk assessment matrix. This matrix is used to rank the mishap risk potential of the hazards. Table A.III of MIL-STD-882E shows an example of the mishap risk assessment matrix (DoD, 2010b). This matrix is used to prioritize engineering efforts to mitigate the system hazards.

Hazards associated with software cannot rely solely on the hazard probability as identified in Table A.II of MIL-STD-882E (DoD, 2010b). Categorization of a software failure is generally determined by its hazard severity and the degree of command, control, and autonomy the software functionality exercises over the hardware. The software control categories include the following: (i) autonomous control, (ii) software exercises control or displays information allowing time for intervention by either an independent safety system or an operator, (iii) software issues commands or generates information requiring operator action to complete the control, and (iv) software does not control safety-critical hardware or provide safety-critical information. A software criticality index is then established using the severity categories and the control categories. This matrix can then be used to prioritize the level of rigor (LOR) assigned to the design, code, and test of the software. Safety-significant software is developed to a specific LOR to bring confidence that the software performs as expected functionally and does not perform unintended functions.

Hazards are prioritized so that corrective action efforts can be focused on the most serious hazards first. The goal of the system safety effort is to work with engineering to design systems that contain no hazards. Since it is impossible or impractical to design a system completely free from hazards, the effort is focused on developing a system design where there are no hazards with an unacceptable level of mishap risk. Each hazard identified is analyzed to determine the requirements to be incorporated into the design to reduce the risk associated with the hazard to an acceptable level. The system safety order of precedence is used to define the order followed for implementing system safety requirements to reduce the mishap risk. The order of precedence is as follows:

1. Eliminate hazards through design selection.
2. Reduce risk through design alteration.
3. Incorporate safety devices.
4. Provide warning devices.
5. Develop procedures and training.

Best practices dictate a function whose failure to operate or whose incorrect operation will directly result in a mishap of either catastrophic (death, permanent total disability, irreversible significant environmental impact, or monetary loss equal to or exceeding $10M) or critical (hospitalization of at least three personnel, reversible significant environmental impact, or monetary loss equal to or exceeding $1M but less than $10M). Severity cannot be mitigated solely through procedural mitigations.

The results of the hazard analysis activities are captured in a hazard tracking system (HTS). The HTS is used to track each hazard by documenting the implementation of the safety requirements, verification results, and residual mishap risk. The HTS is updated throughout the life cycle of the system.

10.10.4 Verify and Validate System Safety Requirements

System safety engineers, along with the systems engineers, provide input to all tests, demonstrations, models, and inspections to verify compliance of the system with

the identified system safety requirements. This is done to ensure the safety of the design is adequately demonstrated for all hazards not eliminated by design. System safety engineers typically witness verification/validation activities for those hazards that were categorized as catastrophic and critical. Results from all test activities performed to validate/verify system safety requirements are reviewed by system safety engineers and captured in the HTS as well as the test and evaluation reports on hardware or software.

10.10.5 Assess Safety Risk

System safety engineers perform and document a comprehensive evaluation of the mishap risk being assumed prior to each key milestone of the program (preliminary design review, critical design review, program completion, etc.) as well as key test or operation activities. The safety assessment identifies all safety features of the hardware, software, and system design. It also identifies procedural, hardware, and software related hazards that may be present in the system at each milestone, test, or operation activity. Specific procedural controls and precautions are identified for those hazards still present in the system. The safety assessment also identifies and documents hazardous materials used in the design, operation, or maintenance of the system as well as those that will be generated by the system. An assessment as to why non- or less hazardous materials could not be used is developed and documented.

10.10.6 Summary

As an integral part of SE, system safety engineering focuses heavily on ensuring that the design engineers are provided a complete set of safety-related requirements to minimize the safety risk potential of the system during the development, production, utilization, support, and retirement stages. The end objective is to deploy, operate, and maintain a system that possesses an acceptable safety risk. The objective is ideally to be accident-free.

10.11 SYSTEM SECURITY ENGINEERING

System security engineering is focused on ensuring a system can function under disruptive conditions associated with misuse and malicious behavior. System security engineering involves a disciplined application of SE principles in analyzing threats and vulnerabilities to systems and assessing and mitigating risk to the information assets of the system during its life cycle. It applies a blend of technology, management principles and practices, and operational rules to ensure sufficient protections are available to the system at all times.

System security engineering considers and accounts for the system and the associated environment. Sources of potential disruptive conditions (threats) are many and varied. They may be natural (e.g., weather) or man-made. They may emanate from external sources (e.g., political or power interruptions) or may be caused by internal forces (e.g., user or supporting systems). A disruption may be unintentional or intentional (malicious) in nature. The security capabilities, whether implemented through design, policy, or practice, must be usable from the user's perspective.

To be effective, system security engineering is applied throughout the life cycle of the system. System security engineering activities can be applied to each life cycle stage:

- *Concept*—System security engineering explores technology trends and advancements to identify potential technologies and promising security strategies to address current and future threats and support architectural and operational concepts that provide security to support and protect operational needs.
- *Development*—System security engineering ensures security concepts are translated into functional with verifiable requirements and defines security-level effectiveness during this stage.
- *Production*—System security engineering provides support during fabrication, build, and assembly to ensure that security settings are properly initialized and delivered with the final system and establishes security-level effectiveness during this stage.
- *Utilization*—System security engineering maintains security effectiveness during use by considering changes in operational, user, and threat environments.
- *Support*—System security engineering ensures that security features are updated and remain effective after maintenance and monitors security events to maintain security effectiveness.

- *Retirement*—System security engineering ensures that effective security practices are employed during retirement of the system and associated information.

10.11.1 Systems Engineer and System Security Engineer Roles and Responsibilities

System security engineering brings security-focused disciplines, technology, and concerns into the SE process to ensure that protection considerations are given the proper weight in decisions concerning the system (Dove et al., 2013). Systems engineers must employ security subject matter experts and security engineers on a timely basis to perform system security analysis and provide effective security recommendations.

System security engineering consists of subspecialties such as antitamper, supply chain risk management, hardware assurance, information assurance, software assurance, system assurance, and others that the system security engineer needs to trade off to provide a balanced system security engineering view. An example of the relationship of SE and system security engineering to the security expert domain is illustrated by the joint contribution to defining and deploying an engineering solution to meet user needs. Some of these joint work products include:

- System security plan
- Vulnerability assessment plan
- Security risk management plan
- System security architecture views
- Security test plan
- Deployment plan
- Disaster recovery and continuity plan

10.11.2 System Security Engineering Activities for Requirements

Stakeholder security interests include intellectual property, information assurance, security laws, supply chain compliance, and security standards. Examples of standards include ISO/IEC 27002, Information security standard (2013); Chapter 13 of the Defense Acquisition Guide (DAU, 2010); and the Engineering for Systems Assurance Guide (NDIA, 2008). Systems engineers need to consider stakeholder security interests during the stakeholder needs and requirements definition process and should be captured in the stakeholder requirements.

During system requirements definition, the critical functions and data are identified that are most in need of protection. A risk-based analysis pattern of criticality analysis, threat assessment, vulnerability assessment, and identification of potential protection controls and mitigations are used as inputs to a CBA. The CBA takes into account impacts to system performance, affordability, and usage compatibility. Security scenarios are developed for both normal security processing and misuse/abuse situations to assist with system requirements definition. These scenarios are also important for use during verification and validation.

System requirements definition needs to consider system security protections. System security protections can be grouped into three categories: prevention, detection, and response. Prevention includes access control and critical function isolation and separation. Detection includes functions that monitor and log security-related behavior. Response includes mitigations that switch to a degraded mode when the primary function or data has been compromised.

The results of these activities are a set of system security requirements (including process requirements), security OpsCon and support concept, and a set of scenarios to be used for verification and validation.

10.11.3 System Security Engineering for Architecture Definition and Design Definition

Engagement of system security engineering in the architecture definition and design definition processes is important because system architecture enables or impedes system security. Adversaries learn system protective measures and change their methods rapidly. Architecture should enable rapid change of protective measures, facilitating operational-time engineering intervention. This is achieved by modifying architectural structure and security functional detail and is enabled by an agile architectural strategy articulated in the system OpsCon. Evolving threats can be countered with a security architecture composed of loosely coupled encapsulated security functional system elements that can be replaced, augmented with additional functionality, and reconfigured for different interconnections (Dove et al., 2013).

Long system life expectancies are especially vulnerable to unadaptable, rigid security architectures.

Architecture focuses on the high-level allocation of responsibilities between different system elements of the SOI and defines the interactions and connectivity between those system elements. Responsibility for security requirements established during the requirements processes is allocated to functional system elements and security-specialty system elements as appropriate during the architectural design process.

Resilience engineering works closely with system security engineering. System resilience permits a system to operate while under, and recover after, attack, perhaps with degraded performance but with continued delivery of critical functionality. Resilience is an architectural feature that is difficult to provide later in the development life cycle and is very costly after deployment.

Long life systems will have functional upgrades and system element replacements throughout their life. Insider threat and supply chain threat may manifest as system elements developed with embedded malicious capabilities, which may lie dormant until activated on demand. This suggests self-protective system elements that distrust communications and behaviors of interconnected system elements, rather than relying on system perimeter protection or trusted environment expectations.

10.11.4 System Security Engineering Activities for Verification and Validation

Verification and validation processes develop strategies for verifying and validating the SOI. System security engineering's involvement is needed for verifying security-related requirements and security impacts to enabling systems. Verification methods are identified for each security requirement with objective pass/fail criteria for inspection, analysis, demonstration, and test. Milestone reviews are conducted to confirm that measures have been planned, threat and vulnerability assessments are current, and system security requirements under test map to a comprehensive criticality analysis. Validation is performed on systems using assessment scenarios that focus on evolving threats in operational environments; risk evaluations and criticality assessments are updated based on current threats and vulnerabilities; and end-to-end scenarios are updated to address new vulnerabilities and threats.

10.11.5 System Security Engineering Activities for Maintenance and Disposal

System security engineering responsibilities do not end when the system is delivered. As part of maintenance, work instructions need to indicate authorized maintenance activities to prevent vulnerability insertion. As part of sustainment operations, threats and security features need to be reevaluated to determine if there are new vulnerabilities in the existing systems. As part of any capability upgrade or technology refresh, criticality analysis must be repeated in the context of new threats, end-to-end scenarios reevaluated to confirm effective protection, and supply chain vulnerabilities evaluated for updated equipment. Prior to system disposal, hardware and software must be brought to a state that cannot be reverse engineered.

10.11.6 System Security Engineering Activities for Risk Management

A mission criticality analysis, a threat assessment, and a vulnerability assessment can be used as security inputs to the risk management process to improve the objectivity of the risk identification and risk-level determination. Balancing security risk reduction within the context of the overall system performance, cost, and schedule requires an objective CBA. The earlier the security risk identification and analysis is done, the more effectively security can be built into the designs, development process, and supply chain. Because of the dynamic innovative nature of threats and the continuous discovery of vulnerabilities, the risk identification and analysis needs to be repeated often throughout the system life cycle.

10.11.7 System Security Engineering Activities for Configuration and Information Management

Configuration and information management ensures that the state of the system is known in total only to those authorized to understand its configuration and capability. Configuration and information management should allow members of the design team to interact with the portions of the system they are authorized to see in order to maintain a consistent and accurate view of the system as it evolves from concept to delivery. Changes to the configuration must be controlled to restrict access for changing the system and documented to provide insight

for forensic analysis in the event of attack or discovery of vulnerabilities.

10.11.8 System Security Engineering Activities for Acquisition and Supply

System security considerations go beyond the SOI. The enabling systems and supply chain must be protected, and all aspects of assembling a solution must be known. System security analysis, evaluation, protection implementation, and updates should be part of the request for proposal to ensure that the acquired system or system element is delivered with acceptable risk.

Vulnerability assessments must be made and updated throughout the life cycle. Hardware and software products that are COTS, while often considered for affordability, may have limited assurance that malicious insertion has not occurred. COTS products may also be used outside the SOI, which allows vulnerability discovery that could be exploited against the SOI.

The system security engineer evaluates the consequence and likelihood of losing business/mission capability in order to select system elements that have secure supply chains. In some cases, the best-performing COTS system element may not be selected if the supply chain risk cannot be reduced through countermeasures to an acceptable level.

10.12 TRAINING NEEDS ANALYSIS

Training needs analyses support the development of products and processes for training the users, maintainers, and support personnel of a system. Training analysis includes the development of personnel capabilities and proficiencies to accomplish tasks at any point in the system life cycle to the level they are tasked. These analyses address initial and follow-on training necessary to execute required tasks associated with system use and maintenance. An effective training analysis begins with a thorough understanding of the concept documents and the requirements for the SOI. A specific list of functions or tasks can be identified from these sources and represented as learning objectives for operators, maintainers, administrators, and other system users. The learning objectives then determine the design and development of the training modules and their means of delivery.

Important considerations in the design of training include who, what, under what conditions, how well each user must be trained, and what training will meet the objectives. Each of the required skills identified must be transformed into a positive learning experience and mapped onto an appropriate delivery mechanism. The formal classroom environment is rapidly being replaced with or augmented by simulators, computer-based training, Internet-based distance delivery, and in-systems electronic support, to name a few. Updates to training content use feedback from trainees after they have some experience to improve training effectiveness.

10.13 USABILITY ANALYSIS/HUMAN SYSTEMS INTEGRATION

Human systems integration (HSI) is the interdisciplinary technical and management process for integrating human considerations within and across all system elements. HSI focuses on the human, an integral element of every system, over the system life cycle. It is an essential enabler to SE practice as it promotes a "total system" approach that includes humans, technology (e.g. hardware, software), the operational context, and the necessary interfaces between and among the elements to make them all work in harmony (Bias and Mayhew, 1994; Blanchard and Fabrycky, 2011; Booher, 2003; Chapanis, 1996; ISO 13407, 1999; Rouse, 1991). The "human" in HSI includes all personnel who interact with the system in any capacity, such as:

- System owners
- Operators
- Maintainers
- Trainers
- Users/customers
- Decision makers
- Support personnel
- Peripheral personnel

Humans are an element of most systems, so many systems benefit from HSI application. HSI establishes human-centered disciplines and concerns into the SE process to improve the overall system design and performance. The primary objective of HSI is to ensure that human capabilities and limitations are treated as critical system elements, regardless of whether humans in the system operate as individuals, crews, teams, units, or organizations. The technology elements of the system have inherent capabilities; similarly, humans possess particular knowledge, skills, and abilities (KSAs), expertise, and

cultural experiences. Deliberate design effort is essential to ensure development of quality interfaces between technology elements and the system's intended users, operators, maintainers, and support personnel in operational environments. It is also important to acknowledge that humans outside the system may be affected by its operation.

While many systems and design engineers intuitively understand that the human operator and maintainer are part of the system under development, they often lack the expertise or information needed to fully specify and incorporate human capabilities. HSI brings this technical expertise into the SE process and serves as the focal point for human considerations in the system concept, development, production, utilization, support, and retirement stages. The comprehensive application of HSI to system development, design, and acquisition is intended to optimize total system performance (e.g., humans, hardware, and software) while accommodating the characteristics of the population that will use, operate, maintain, and support the system and also support efforts to reduce LCC.

A key method of HSI is trade studies and analyses. HSI analyses, especially requirements analyses that include human issues and implications, often result in insights not otherwise realized. Trade studies that include human-related issues are critical to determining the design that is the most effective, efficient, suitable (including useful and understandable), usable, safe, and affordable.

HSI helps systems engineers focus on long-term costs since much of that cost is directly related to human element areas. One example (unfortunately, of many) is the Three Mile Island power plant nuclear incident in the United States:

> The accident was caused by a combination of personnel error, design deficiencies, and component failures. The problems identified from careful analysis of the events during those days have led to permanent and sweeping changes in how NRC regulates its licensees—which, in turn, has reduced the risk to public health and safety. (NRC, 2005)

Failure to include HSI within a comprehensive SE framework resulted in loss of confidence in nuclear power in the United States and delayed progress in the field for almost 30 years. The cleanup costs, legal liability, and significant resources associated with responding to this near catastrophe trace directly to lack of attention to the human element of a highly complex system, resulting in flawed operations technology and work methods. It also emphasized that while human performance includes raw efficiency in terms of tasks accomplished per unit time and accuracy, human performance directly impacts the overall system performance.

10.13.1 HSI Is Integral to the SE Process

The foundation for program success is rooted in requirements development. Human performance requirements are derived from and bounded by other performance requirements within the system. Front-end analyses (FEA) generate system requirements and incorporate HSI-related requirements. Effective FEA start with a thorough understanding of the mission of the new system and the work to be performed, successes or problems with any predecessor systems, and the KSAs and training associated with the people who are likely to interact with the proposed system technology. HSI modeling views the system as a collection of interrelating elements that behave according to a shared organizing principle. This perspective underpins models and simulations with mathematical rigor approaching that of other engineering disciplines (SEBoK, 2014). It also highlights the essential HSI truth—there are no unpopulated systems. Simulation early in the development process, particularly before system hardware and software elements are developed, ensures all the interfaces are captured for requirements definition, trade space analysis, and iterative design activities. HSI analyses allocate human-centered functions within the system and identify potential human (or system) capability gaps. For example, humans excel at solving induction problems, and machines excel at deduction (Fitts, 1954). The requirement for inductive or deductive decision making is inherent in the structure of the system design.

It is critical to include HSI early in system development (during stakeholder requirements generation) and continuously through the development process to realize the greatest benefit to the final system solution and substantial LCC savings. Fully utilizing HSI in IPPD helps ensure that systems will not require expensive "train-arounds" or late-stage fixes to correct ineffective usability and operational inefficiencies driven by bad, unspecified, or undefined human interfaces. The systems

engineer plays a critical role in engaging HSI expertise to support the IPDTs. A knowledgeable, interdisciplinary HSI team is generally required to address the full spectrum of human considerations, and the systems engineer is key to ensuring that HSI is included throughout the system's life cycle. Program managers, chief engineers, and systems engineers should ensure that HSI practitioners are actively participating in design reviews, working groups, and IPDTs. Consistent involvement and communications with customers, users, developers, scientists, testers, logisticians, engineers, and designers (human, hardware, and software) are essential.

10.13.2 Technical and Management HSI Processes

HSI must be addressed by all program-level IPDTs and at program, technical, design, and decision reviews throughout the life of the system. HSI influences the design and acquisition of all systems and system modifications and makes explicit the role that the human plays in system performance and cost and how these factors are shaped by design decisions. In addition, HSI is one of the essential components of engineering practice for system development, contributing technical and management support to the development process itself.

10.13.2.1 HSI Domains HSI processes facilitate trade-offs among interdependent, human-centered domains without replacing individual domain activities, responsibilities, or reporting channels. The human-centered domains typically cited by various organizations may differ in what they are called or in number, but the human considerations addressed are quite similar. The following human-centered domains with recognized application to HSI serve as a good foundation of human considerations that need to be addressed in system design and development, but clearly are not all inclusive:

- *Manpower*—Addresses the number and type of personnel and the various occupational specialties required and potentially available to train, operate, maintain, and support the deployed system.
- *Personnel*—Considers the type of KSAs, experience levels, and aptitudes (e.g., cognitive, physical, and sensory capabilities) required to operate, maintain, and support a system and the means to provide (e.g., recruit and retain) such people.
- *Training*—Encompasses the instruction and resources required to provide personnel with requisite KSAs to properly operate, maintain, and support systems. The training community develops and delivers individual and collective qualification training programs, placing emphasis on options that:
  - Enhance user capabilities to include operator, maintainer, and support personnel
  - Maintain skill proficiencies through continuation training and retraining
  - Expedite skill and knowledge attainment
  - Optimize the use of training resources

  Training systems, such as simulators and trainers, should be developed in conjunction with the emerging system technology.
- *Human factors engineering* (*HFE*)—Involves an understanding of human capabilities (e.g., cognitive, physical, sensory, and team dynamic) and comprehensive integration of those capabilities into system design beginning with conceptualization and continuing through system disposal. A key objective for HFE is to clearly characterize the actual work to be performed and then use this information to create effective, efficient, and safe human/hardware/software interfaces to achieve optimal total system performance. This "optimal performance" is the achievement of the following:
  - Conducting task analyses and design trade-off studies to optimize human activities, creating workflow
  - Making the human goals and performance the design driver to assure an intuitive system for humans who will use, operate, maintain, and support it
  - Providing deliberately designed primary, secondary, backup, and emergency tasks and functions
  - Meeting or exceeding performance goals and objectives established for the system
  - Conducting analyses to eliminate/minimize the performance and safety risks leading to task errors and system mishaps across all expected operational, support, and maintenance environments

  HFE uses task and function analyses (including cognitive task analysis) supported by increasingly

sophisticated tools and methods to define system functions and interfaces. These efforts should recognize the increasing complexity of technology and the associated demands on people, giving careful consideration to the capabilities and limitations of humans. The design should not demand unavailable or unachievable skills. HFE seeks to maximize usability for the targeted range of users/customers and to minimize design characteristics that induce frequent or critical errors. HSI/HFE tools provide data that can be directly incorporated into system design elements such as the information architecture.

HFE works with the IPDTs to ensure that representative personnel are tested in situations to determine whether the human can operate, maintain, and support the system in adverse environments while working under the full range of anticipated mission stress and endurance conditions.

- *Environment*—In the context of HSI, this domain involves environmental considerations that can affect operations and requirements, particularly human performance.
- *Safety*—Promotes system design characteristics and procedures to minimize the risk of accidents or mishaps that cause death or injury to operators, maintainers, and support personnel; threaten the operation of the system; or cause cascading failures in other systems. Prevalent issues include the following:
  - Factors that threaten the safety of personnel and their operation of the system
  - Walking/working surfaces, emergency egress pathways, and personnel protection devices
  - Pressure and temperature extremes
  - Prevention/control of hazardous energy releases (e.g., mechanical, electrical, fluids under pressure, ionizing or nonionizing radiation, fire, and explosions)
- *Occupational health*—Promotes system design features and procedures that serve to minimize the risk of injury, acute or chronic illness, and disability and to enhance job performance of personnel who operate, maintain, or support the system. Prevalent issues include the following:
  - Noise and hearing protection
  - Chemical exposures and skin protection
  - Atmospheric hazards (e.g., confined space entry and oxygen deficiency)
  - Vibration, shock, acceleration, and motion protection
  - Ionizing/nonionizing radiation and personnel protection
  - Human factors considerations that can result in chronic disease or discomfort (e.g., repetitive motion injuries or other ergonomic-related problems)
- *Habitability*—Involves characteristics of system living and working conditions, such as the following:
  - Lighting and ventilation
  - Adequate space
  - Availability of medical care, food, and/or drink services
  - Suitable sleeping quarters, sanitation and personal hygiene facilities, and fitness/recreation facilities
- *Survivability*—Addresses human-related characteristics of a system (e.g., life support, body armor, helmets, plating, egress/ejection equipment, airbags, seat belts, electronic shielding, alarms, etc.) that reduce susceptibility of the total system to mission degradation or termination, injury or loss of life, and partial or complete loss of the system or any of its elements.

The domains outlined above must be considered simultaneously since decisions made in one domain can have significant impacts on other domains. Each individual domain decision generates the need to concurrently assess HSI issues across all the domains and against mission performance, prior to making formal programmatic decisions. This approach mitigates the potential for unintended, adverse consequences, including increased technical risk and cost.

10.13.2.2 Key HSI Activities and Tenets HSI programs have distilled the following HSI activities and associated key actionable tenets:

- *Initiate HSI early and effectively*—HSI should begin in early system concept development with FEA and requirements definition.

  HSI-related requirements include not just those of the individual domains but also those that arise from interactions among the HSI domains.

HSI requirements must be developed in consonance with other system requirements and consider any constraints or capability gaps and must be reconsidered, refined, and revised as program documents and system requirements are updated. The human considerations identified in the requirements must address the capabilities and limitations of users, operators, maintainers, and other personnel as they interact with and within the system. Doing this early, continuously, and comprehensively as part of the SE process provides the opportunity to identify risks and costs associated with program decisions.

HSI should be conducted by professionals who are trained in the domains outlined earlier, who are resourced with appropriate tools, and who have access to data from other concurrent IPDT activities.

- *Identify issues and plan analysis*—Projects/programs must identify HSI-related issues that require analysis to ensure thorough consideration.

  The systems engineer must have a comprehensive plan for HSI early in the acquisition process and summarize HSI planning in the acquisition strategy. This plan can be stand-alone or integrated into the SEMP and include details associated with the analyses in SE language to support HSI trade-offs. Systems engineers should address HSI throughout the entire acquisition cycle as part of the SE process.

  Efficient, timely, and effective planning/replanning and FEA are the cornerstones of HSI efforts, ensuring human considerations are effectively integrated into capability requirements, system concept development, and acquisition. HSI FEA establishes and assesses criteria for success and helps determine when a system design change is required.
- *Document HSI requirements*—Systems engineers derive HSI requirements, as needed at each level of the system hierarchy, using HSI plans, analyses, and reports as sources (or rationale) for the derived requirements.HSI requirements should be cross-referenced with other documents, plans, and reports and captured in the requirements traceability documents and maintained in system requirements databases in the same manner as all other system requirements.
- *Make HSI a factor in source selection for contracted development efforts*—HSI requirements must be explicit in source selection planning and implementation with adequate priority in the SOW.
- *Execute integrated technical processes*—HSI domain integration begins in early system concept development with FEA and requirements definition and continues through development, operations, sustainment, modification, and all the way to eventual system disposal.

  HSI activities and considerations must be included in each key project planning document (e.g., acquisition strategy, SEMP, test plan, verification, etc.) and in the system architectural framework.

  Systems engineers and HSI personnel must be prepared to exchange technical data and to present accurate, integrated cost data whenever possible to demonstrate reduced LCC, thereby justifying trade-off decisions that may increase design and acquisition costs.
- *Conduct proactive trade-offs*—In conducting trade-off analyses both within HSI domains and across the system, the primary goal is to ensure the system meets or exceeds the performance requirements without compromising survivability, environment, safety, occupational health, and habitability.
- *Conduct HSI assessments*—The purpose of the HSI assessment process is to evaluate the application of HSI principles throughout the system life cycle. The process enables cross-discipline HSI collaboration in acquisition program evaluations and provides an integral avenue for HSI issue identification and resolution.

  HSI assessment should be initiated early in the system life cycle and addressed throughout the system development process, particularly in SE technical reviews and during working group meetings, design reviews, logistical assessments, verification, and validation.

  HSI assessments should be based on sound data collection and analyses. Deficiencies should be captured and include a detailed description of the deficiency, its operational impact, recommended corrective action, and current status.

10.14 VALUE ENGINEERING

VE, value management (VM), and value analysis (VA) are all terms that pertain to the application of the VE process (Bolton et al., 2008; Salvendy, 1982; SAVE, 2009).

This chapter discusses how VE supports, compliments, and "adds value" to the SE discipline.

VE originated in studies of product changes resulting from material shortages during World War II. During that period, materials identified in designs were substituted without sacrificing quality and performance. Lawrence D. Miles noticed this success and developed a formal methodology to utilize teams to examine the function of products manufactured by General Electric.

VE uses a systematic process (e.g., a formal job plan), VE-certified facilitators/team leads, and a multidisciplinary team approach to identify and evaluate solutions to complex problems in the life cycle of a project, process, or system. The VE process utilizes several industry standard problem-solving/decision-making techniques in an organized effort directed at independently analyzing the functions of programs, projects, organizations, processes, systems, equipment, facilities, services, and supplies. The objective is to achieve the essential functions at the lowest LCC consistent with required performance, reliability, availability, quality, and safety. VE is not a cost reduction activity but a function-oriented method to improve the value of a product. There is no limit to the field in which VE may be applied.

The objective of performing VE is to improve the economical value of a project, product, or process by reviewing its elements to accomplish the following:

- Achieve the essential functions and requirements
- Lower total LCC (resources)
- Attain the required performance, safety, reliability, quality, etc.
- Meet schedule objectives

Value is defined as a fair return or the equivalent in goods, services, or money for something exchanged. In other words, value is based on "what you get" relative to "what it cost." It is represented by the relationship:

$$\text{Value} = \text{function}/\text{cost}$$

Function is measured by the requirements from the stakeholder. Cost is calculated in the materials, labor, price, time, etc., required to accomplish that function.

According to the Society of American Value Engineers (SAVE) International, to qualify as a value study, the following conditions must be satisfied:

- The value study team follows an organized, six-phase VE job plan (see following text) and performs function analysis.
- The value study team is a multidisciplinary group, chosen based on their expertise.
- The value study team leader (i.e., facilitator) is trained in the value methodology.

10.14.1 Systems Engineering Applicability

VE supports the various aspects and techniques of SE as well as being a comprehensive approach to SE project initiation. VE is implemented through a systematic, rational process consisting of a series of team-based techniques, including:

- Mission definition, strategic planning, or problem-solving techniques to determine current and future states. High-level requirements definition can be initiated in these processes.
- Functional analysis to define what a system or system element does or its reason for existence. This technique serves as the basis or structure for technical, functional, and/or operational requirements.
- Innovative, creative, or speculative techniques to generate new alternatives. Trade studies are often defined in VE workshops from the more "viable" alternatives. Time can be given to conduct in-depth analyses, such as feasibility studies, cost estimates, etc. The VE workshop can then be reconvened (e.g., 1–6 months later) to determine the preferred alternative.
- Evaluation techniques to select preferred alternatives. These range in complexity from completely subjective to quantitative, depending on the LOR and justification required.

VE can be used to effectively and efficiently initiate an SE project. The team approach brings the customer and other stakeholders together along with engineering, planning, finance, marketing, etc. so that the strategic elements of the project can be discussed or developed. Stakeholder needs can be identified and established as requirements. Functions are brainstormed and alternatives developed. In general, the boundaries of the scope of work are identified and clarified with all affected

groups. The SE effort can then proceed with initial planning already established and stakeholder input integrated. Some of the uses and benefits of VE include:

- Clarify objectives
- Solve problems (obtain "buy-in" to solution)
- Improve quality and performance
- Reduce costs and schedule
- Assure compliance
- Identify and evaluate additional concepts and options
- Streamline and validate processes and activities
- Strengthen teamwork
- Reduce risks
- Understand customer requirements

10.14.2 VE Job Plan

The VE process uses a formal job plan, which is a scientific method of problem solving that has been optimized over the last 50 years. The premise of VE is to use a beginning-to-end process that analyzes functions in such a way that creates change or value-added improvement in the end. A typical six-phase VE job plan is as follows:

- *Phase 0: Preparation/planning*—Plan the scope of the VE effort.
- *Phase 1: Information gathering*—Clearly identify the problem(s) to be solved or the objective of the work scope. This includes gathering background information pertinent to meeting the objective and identifying and understanding customer/stakeholder requirements and needs.
- *Phase 2: Functional analysis*—Define the project functions using an active verb and measurable noun and then review and analyze the functions to determine which are critical, need improvements, or are unwanted. Several techniques can be used to enhance functional analysis. Functions can be organized in a WBS, in a Function Analysis System Technique (FAST) diagram (discussed in the following text), or in a flow diagram.

  Functions can be assigned a cost. Depending on the objective of the VE study, determining the cost/function relationships is one method to identify where unnecessary costs exist within the scope of the study. Other criteria to identify those areas needing improvement are personnel, environmental safety, quality, reliability, construction time, etc. Cost/function relationships provide direction for the team related to areas that provide the greatest opportunity for cost improvement and the greatest benefits to the project. This can be captured in a cost/function worksheet.
- *Phase 3: Creativity*—Brainstorm different ways to accomplish the function(s), especially those that are high cost or low value.
- *Phase 4: Evaluation*—Identify the most promising functions or concepts. The functions can also be used in this phase to generate full alternatives that accomplish the overall objective. Then the alternatives can be evaluated using the appropriate structured evaluation technique.
- *Phase 5: Development*—Develop the viable ideas into alternatives that are presented to decision makers to determine the path forward. Alternatively, the full alternatives that were evaluated in Phase 4 can be further developed into proposals, which may include rough-order-of-magnitude cost, estimated schedule, resources, etc.
- *Phase 6: Presentation/implementation*—It may be desirable to present the alternatives/proposals to management or additional stakeholder for final direction. This is where the implementation plan concepts are identified and a point of contact assigned to manage the actions.

10.14.3 FAST Diagram

The FAST diagramming technique was developed in 1964 by Charles W. Bytheway to help identify the dependencies and relationships between functions. A FAST diagram (see Fig. 10.10) is not time oriented like a Program Evaluation Review Technique (PERT) chart or flowchart. It is a function-oriented model that applies intuitive logic to verify the functions that make up the critical path.

There are several different types of FAST diagrams as well as varying levels of complexity. The purpose is to develop a statement of the function (verb and noun) for each part or element of the item or process under study. Functions are classified as basic and secondary. The critical path is made up of the main functions within the scope of the activity or project. The basic function is

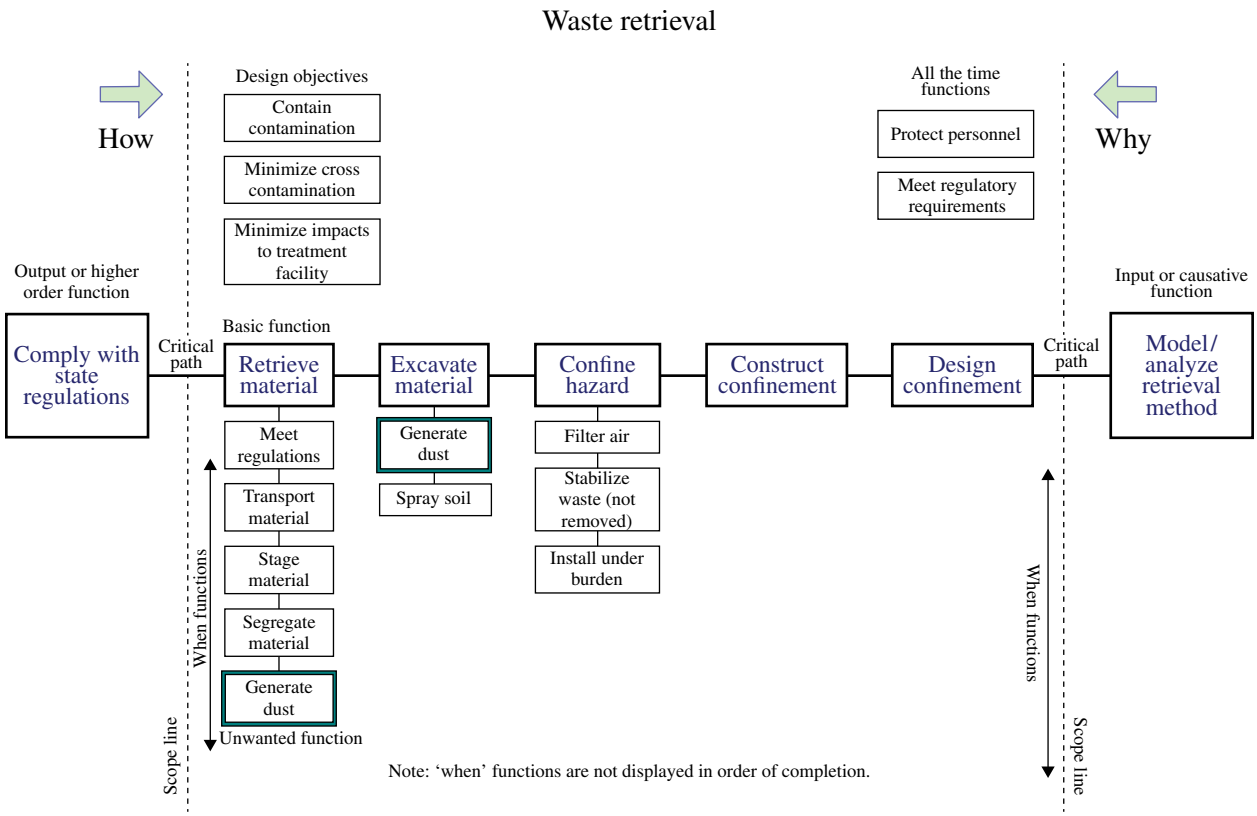


FIGURE 10.10 Sample Function Analysis System Technique (FAST) diagram. Reprinted with permission from Doug Hamelin. All other rights reserved.

typically the deliverable or end state of the effort. The higher-order function is the ultimate goal and is the answer to "why" the basic function is performed. Functions that "happen at the same time" or "are caused by" some function on the critical path are known as "when" functions and are placed below the critical path functions. Functions that happen "all the time," such as safety, aesthetic functions, etc., are placed above the critical path functions.

Unwanted functions are noted by a double lined box. They are not essential to the performance of the scope, but are a consequence of the selected design solution. Limiting unwanted functions and minimizing the cost of basic/critical path functions result in an item of "best value" that is consistent with all performance, reliability, quality, maintainability, logistics support, and safety requirements.

There is no "correct" FAST model; team discussion and consensus on the final diagram is the goal. Using a team to develop the FAST diagram is beneficial for several reasons:

- Applies intuitive logic to verify functions
- Displays functions in a diagram or model
- Identifies dependence between functions
- Creates a common language for team
- Tests validity of functions

10.14.4 VE Certification

A Certified Value Professional should be used to facilitate VE workshops. They have the training and experience to manage a team, implement the methodology, and maximize the benefits to the customer.

The Certification Program is composed of two major elements: individual professional certification and

educational program approval. The highest level is the *Certified Value Specialist* (CVS), which is recognition of the individual who has met all certification requirements, both technical and experience, and whose principal career is VE.

The *Associate Value Specialist* (AVS) program recognizes those individuals who have decided to become professional value engineers but who have not yet acquired all the experience or technical skills expected of a CVS. The *Value Methodology Practitioner* (VMP) program was established to recognize those individuals who acquired the basic skills of VE/VA but their principal career is not VE.

The CVS and VMP must recertify every 4 years. Although considered an entry-level certification, the AVS may be maintained indefinitely as long as all certification maintenance fees are paid. Membership in the SAVE is not a requirement for individual certification or for educational program approval.

10.14.5 Conclusion

VE is a best business practice. Projects that use VE in the early life cycle stages have shown high rates of success. A VE study is more rigorous than the typical project review. Each VE study brings together an impartial and independent team of technical experts with a common purpose of improving and optimizing the project's value. VE is seen as a systematic and creative approach for increasing the "return on investment" in systems, facilities, and other products. For more information on VE, consult the website of the SAVE International (2009).

APPENDIX A: REFERENCES

Achenbach, J. (2009). Mars Mission Has Some Seeing Red. *The Washington Post*, 11 February 2009.

Adams, J. L. (1990). *Conceptual Blockbusting*, 3rd Ed. San Francisco, CA: San Francisco Book Company, Inc.

Alavi, M., & Leidner, D. E. (1999). Knowledge Management Systems: Issues, Challenges, and Benefits. *Communications of the AIS*, 1(2).

Albright, D., Brannan, P., & Walrond, C. (2010). *Did Stuxnet Take Out 1,000 Centrifuges at the Natanz Enrichment Plant?* Washington, DC: Institute for Science and International Security. Retrieved from http://isis-online.org/isis-reports/detail/did-stuxnet-take-out-1000-centrifuges-at-the-natanz-enrichment-plant/ (accessed January 26, 2015).

Albright, D., Brannan, P., & Walrond, C. (2011). Stuxnet Malware and Natanz: Update of ISIS December 22, 2010 Report. Institute for Science and International Security. Retrieved from http://isis-online.org/uploads/isis-reports/documents/stuxnet_update_15Feb2011.pdf (accessed October 24, 2014).

ANSI/AIAA G-043A. (2012). *Guide to the Preparation of Operational Concept Documents*. Reston, VA: American National Standards Institute/American Institute of Aeronautics and Astronautics.

ANSI/EIA 632. (2003). *Processes for Engineering a System*. Arlington, VA: American National Standards Institute/Electronic Industries Association.

ANSI/EIA 649B. (2011). Configuration Management Standard. TechAmerica.

ANSI/GEIA-STD-0009. (2008). Reliability Program Standard for Systems Design, Development, and Manufacturing. American National Standards Institute/Government Electronic Industries Association.

Arnold, S., & Lawson, H. (2003). Viewing Systems from a Business Management Perspective: The ISO/IEC 15288 Standard. *Systems Engineering*, 7(3), 229–42.

ARP 4754A. (2010). *Guidelines for Development of Civil Aircraft and Systems*. Warrendale, PA: SAE International.

Ashby, W. R. (1956). *Introduction to Cybernetics*. London, UK: Methuen.

ASQ. (2007). Quality Progress. In *Quality Glossary*. Milwaukee, WI: American Society for Quality Control.

AT&T. (1993). *AT&T Engineering Guides for Managing Risk*. TX: McGraw-Hill.

Barnard, R. W. A. (2008). What Is Wrong with Reliability Engineering? *Proceedings of the 18th Annual INCOSE International Symposium*. Utrecht, the Netherlands: International Council on Systems Engineering.

Barton, T. L., Shenkir, G., & Walker, P. L. (2002). *Making Enterprise Risk Management Pay Off: How Leading Companies Implement Risk Management*. Upper Saddle

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.

River, NJ: Financial Times/Prentice Hall PTR/Pearson Education Company.

BBC. (2002). China's Supertrain Takes to Tracks. Retrieved from BBC News World Edition: Asia-Pacific: http://news.bbc.co.uk/2/hi/asia-pacific/2182975.stm (accessed May 29, 2003).

Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., … Thomas, D. (2001). Manifesto for Agile Software Development. Retrieved from http://agilemanifesto.org (accessed October 24, 2014).

Becker, O., Ben-Ashe, J., & Ackerman, I. (2000). A Method for Systems Interface Reduction Using N2 Charts. *Systems Engineering, Systems Engineering*, 3(1), 27–37.

Beer, S. (1959). *Cybernetics and Management*. New York, NY: John Wiley & Sons, Inc.

Bellinger, G. (2013). Systems Thinking World. Retrieved from http://www.systemswiki.org/ (accessed October 24, 2014).

von Bertalanffy, L. (1950). The Theory of Open Systems in Physics and Biology. *Science*, 111(2872), 23–9.

von Bertalanffy, L. (1968). *General System Theory: Foundations, Development, Applications*. New York, NY: Braziller.

Bias, R. G., & Mayhew, D. J. (1994). *Cost Justifying Usability*. Boston, MA: Academic Press.

Blanchard, B., & Fabrycky, W. (2011). *Systems Engineering and Analysis*, 5th Ed. Boston, MA: Prentice Hall.

Boardman, J., & Sauser, B. (2008). *Systems Thinking—Coping with 21st Century Problems*. Boca Raton, FL: CRC Press.

Bobinis, J., Dean, E., Mitchell, T., & Tuttle, P. (2010). INCOSE Affordability Working Group Mission. *2010 ISPA/SCEA Joint Annual Conference & Training Workshop Proceedings*. Society for Cost Estimating and Analysis.

Bobinis, J., Haimowitz, J., Tuttle, P., Garrison, C., Mitchell, T., & Klingberg, J. (2013). Affordability Considerations: Cost Effective Capability. *Proceedings of the 23rd Annual INCOSE International Symposium*. Philadelphia, PA: International Council on Systems Engineering.

Boehm, B. (1986), A Spiral Model of Software Development and Enhancement. *ACM SIGSOFT Software Engineering Notes, ACM*, 11(4), 14–24.

Boehm, B. (1996). Anchoring the Software Process. *Software*, 13(4), 73–82.

Boehm, B., & Lane, J. (2007). Using the Incremental Commitment Model to Integrate System Acquisition, Systems Engineering, and Software Engineering. *CrossTalk*, 20, 4–9.

Boehm, B., & Turner, R. (2004). *Balancing Agility and Discipline*. Boston, MA: Addison-Wesley.

Boehm, B., Lane, J., Koolmanojwong, S., & Turner, R. (2014). *The Incremental Commitment Spiral Model: Principles and Practices for Successful Systems and Software*. Boston, MA: Addison-Wesley Professional.

Bogdanich, W. (2010, January 23). Radiation Offers New Cures, and Ways to Do Harm. *The New York Times*.

Bolton, J. D., Gerhardt, D. J., Holt, M. P., Kirk, S. J., Lenaer, B. L., Lewis, M. A., … Vicers, J. R. (2008). *Value Methodology: A Pocket Guide to Reduce Cost and Improve Value through Function Analysis*. Salem, NH: GOAL/QPC.

Booher, H. R. (Ed.). (2003). *Handbook of Human Systems Integration*. New York, NY: John Wiley & Sons, Inc.

Brenner, M. J. (n.d.). TQM, ISO 9000, Six Sigma: Do Process Management Programs Discourage Innovation? Retrieved from Knowledge@Wharton, University of Pennsylvania: http://knowledge.wharton.upenn.edu/article/tqm-iso-9000-six-sigma-do-process-management-programs-discourage-innovation/ (accessed January 26, 2015).

Briedenthal, J., & Forsberg, K. (2007). Organization of Systems Engineering Plans According to Core and Off-Core Processes. Pasadena, CA: California Institute of Technology, Jet Propulsion Laboratory.

Brykczynski, D. B., & Small, B. (2003). Securing Your Organization's Information Assets. *CrossTalk*, 16(5), 12–6.

Buede, D. M. (2009). *The Engineering Design of Systems: Models and Methods* (2nd Ed). Hoboken, NJ: John Wiley & Sons, Inc.

Carpenter, S., Delugach, H., Etzkorn, L., Fortune, J., Utley, D., & Virani, S. (2010). The Effect of Shared Mental Models on Team Performance. *Industrial Engineering Research Conference*. Cancun, Mexico: Institute of Industrial Engineers.

Carson, R. (2013). Can Systems Engineering be Agile? Development Lifecycles for Systems, Hardware, and Software. *Proceedings of the 23rd Annual INCOSE International Symposium*. Philadelphia, PA: International Council on Systems Engineering.

Carter, A. (2011). Better Buying Power: Mandate for Restoring Affordability and Productivity in Defense Spending. *Memorandum for Defense Acquisition and Logistics Professionals*. Under Secretary of Defense for Acquisition, Technology, and Logistics.

Chang, C. M. (2010). *Service Systems Management and Engineering: Creating Strategic Differentiation and Operational Excellence*. Hoboken, NJ: John Wiley & Sons, Inc.

Chapanis, A. (1996). *Human Factors in Systems Engineering*. New York, NY: John Wiley & Sons, Inc.

Chase, W. P. (1974). *Management of System Engineering*. New York, NY: John Wiley & Sons, Inc.

Checkland, P. (1975). The Origins and Nature of Hard Systems Thinking. *Journal of Applied Systems Analysis*, 5(2), 99–110.

Checkland, P. (1998). *Systems Thinking, Systems Practice*. Chichester, UK: John Wiley & Sons, Ltd.

Christensen, C. M. (2000). *The Innovator's Dilemma*. New York, NY: HarperCollins Publishers.

Churchman, C. W., Ackoff, R. L., & Arnoff, E. L. (1950). *Introduction to Operations Research*. New York, NY: John Wiley & Sons, Inc.

CJCS. (2012). *CJCSI 3150.25E, Joint Lessons Learned Program*. Washington, DC: Office of the Chairman of the Joint Chiefs of Staff.

Clausing, D., & Frey, D. D. (2005). Improving System Reliability by Failure Mode Avoidance including Four Concept Design Strategies. *Systems Engineering*, 8(3), 245–61.

Cloutier, R., DiMario, M., & Pozer, H. (2009). Net Centricity and Systems of Systems. In M. Jamshidi, (Ed.), *Systems of Systems Engineering*. Wiley Series in Systems Engineering. Boca Raton, FL: CRC Press/Taylor & Francis Group.

CMMI Product Team. (2010). Capability Maturity Model Integration, Version 1.3 (CMU/SEI-2010-TR-033). Software Engineering Institute, Carnegie Mellon University. Retrieved from CMMI Institute: http://cmmiinstitute.com (accessed October 24, 2014).

Cockburn, A. (2000). Selecting a Project Methodology. *IEEE Software*, 7(4), 64–71.

Conrow, E. H. (2003). *Effective Risk Management*, (2 Ed). Reston, VA: American Institute of Aeronautics and Astronautics, Inc.

Conway, M. E. (1968). How Do Committees Invent? *Datamation*, 14(5), 28–31. Retrieved from http:www.melconway.com/research/committees.html (accessed October 24, 2014).

Cook, M. (2004). Understanding the Potential Opportunities provided by Service-Oriented Concepts to Improve Resource Productivity. In T. Bhamra, & B. Hon (Eds.), *Design and Manufacture for Sustainable Development* (pp. 123–34). Suffolk, UK: Professional Engineering Publishing Limited.

Crosby, P. B. (1979). *Quality Is Free*. New York, NY: New American Library.

Dahmann, J. (2014). System of Systems Pain Points. *24th Annual INCOSE International Symposium*. Las Vegas, NV: International Council on Systems Engineering.

Daskin, M. S. (2010). *Service Science*. New York, NY: John Wiley & Sons, Inc.

DAU. (1993). *Committed Life Cycle Cost against Time. 3.1*. Fort Belvoir, VA: Defense Acquisition University.

DAU. (2010). *Defense Acquisition Guidebook*. Fort Belvoir, VA: Defense Acquisition University. Retrieved from https://acc.dau.mil/CommunityBrowser.aspx?id=22907&lan=en-US (accessed October 24, 2014).

Deming, W. E. (1986). *Out of the Crisis*. Cambridge, MA: MIT Center for Advanced Engineering Study.

DeRosa, J. K. (2005). Enterprise Systems Engineering. *Air Force Association, Industry Day*. Danvers, MA.

DeRosa, J. K. (2006). An Enterprise Systems Engineering Model. *16th Annual INCOSE International Symposium*. Orlando, FL: International Council on Systems Engineering.

DoD. (1998). Integrated Product and Process Development Handbook. August. Retrieved from http://www.acq.osd.mil/se/docs/DoD-IPPD-Handbook-Aug98.pdf (accessed October 24, 2014).

DoD 5000.59. (2007). *Directive:DoD Modeling and Simulation (M&S) Management*. Washington, DC: U.S. Department of Defense.

DoD. (2010a). *MIL-STD-464C. Electromagnetic Environmental Effects, Requirements for Systems*. Washington, DC: U.S. Department of Defense.

DoD. (2010b). *MIL-STD-882. System Safety Program Requirements*. Washington, DC: US Department of Defense.

DoDAF. (2010). DoD Architecture Framework, Version 2.02. Retrieved from http://dodcio.defense.gov/dodaf20.aspx (accessed October 24, 2014).

DoD and US Army. (2003). PSM Guide V4.0c, Practical Software and Systems Measurement: A Foundation for Objective Project Management. Picatinny Arsenal, NJ.

Domingue, J., Fensel, D., Davies, J., Gonzalez-Cabero, R., & Pedrinaci, C. (2009). The Service Web: A Web of Billions of Services. In G. Tselentis, J. Domingue, A. Galis, A. Gavras, D. Hausheer, S. Krco, … T. Zeheriadis (Eds.), *Toward the Future Internet—A European Research Perspective*. Amsterdam, the Netherlands: IOS Press.

Dove, R. (2001). *Response Ability: The Language, Structure, and Culture of the Agile Enterprise*. New York, NY: John Wiley & Sons, Inc.

Dove, R. (2012). Agile Systems and Processes: Necessary and Sufficient Fundamental Architecture (Agile 101). *INCOSE Webinar*. Retrieved from http://www.parshift.com/s/Agile Systems-101.pdf (accessed September 19).

Dove, R., Popick, P., & Wilson, E. (2013). The Buck Stops Here: Systems Engineering is Responsible for System Security. *Insight*, 16(2), 6–10.

Draucker, L., Kaufman, S., Kuile, R. T., & Meinrenken, C. (2011). Moving Forward on Product Carbon Footprint Standards. *Journal of Industrial Ecology*, 15(2), 169–71.

DSMC. (1983). *Systems Engineering Management Guide*. Fort Belvoir, VA: Defense Systems Management College.

Edwards, W., Miles Jr., R. F., & Von Winterfeldt, D. (2007). *Advances in Decision Analysis: From Foundations to Applications*. New York, NY: Cambridge University Press.

Eisner, H. (2008). *Essentials of Project and Systems Engineering Management*. Hoboken, NJ: John Wiley & Sons, Inc.

Elm, J., & Goldenson, D. (2012). The Business Case for Systems Engineering Study: Results of the Systems Engineering Effectiveness Study. Software Engineering Institute, Carnegie Mellon University. Retrieved from http://resources.sei.cmu.edu/library/asset-view.cfm?assetID=34061 (accessed January 26, 2015).

Engel, A. (2010). *Verification, Validation, and Testing of Engineered Systems*. Hoboken, NJ: John Wiley & Sons, Inc. Retrieved from INCOSE Systems Engineering Center of Excellence: http://www.incose.org/secoe/0105.htm (accessed September 23).

Eppinger, S., & Browning, T. (2012). *Design Structure Matrix Methods and Applications*. Cambridge, MA: MIT Press.

Estefan, J. (2008). Survey of Model-Based Systems Engineering (MBSE) Methodologies, Rev. B, Section 3.2. NASA Jet Propulsion Laboratory.

FAA. (2006). Systems Engineering Manual, Version 3.1. Federal Aviation Administration.

Failliere, N. L. (2011). W32.Stuxnet Dossier Version 1.4. Wired. Retrieved from http://www.wired.com/images_blogs/threatlevel/2011/02/Symantec-Stuxnet-Update-Feb-2011.pdf (accessed October 24, 2014).

Fairley, R. E. (2009). *Managing and Leading Software Projects*. Los Alamitos, CA: IEEE Computer Society; John Wiley & Sons, Inc.

Fitts, P. M. (1954). The Information Capacity of the Human Motor System in Controlling the Amplitude of Movement. *Journal of Experimental Psychology*, 47(6), 381–91.

Flood, R. L. (1999). *Rethinking the Fifth Discipline: Learning with the Unknowable*. London, UK: Routledge.

Flood, R. L., & Carson, E. R. (1993). *Dealing with Complexity: An Introduction to the Theory and Application of Systems Science* (2 Ed). New York, NY: Plenum Press.

Forrester, J. W. (1961). *Industrial Dynamics*. Waltham, MA: Pegasus Communications.

Forsberg, K. (1995). If I Could Do That, Then I Could …: Systems Engineering in a Research and Development Environment. *Proceedings of the Fifth Annual INCOSE Symposium*. St. Louis, MO: International Council on Systems Engineering.

Forsberg, K., & Mooz, H. (1991), The Relationship of System Engineering to the Project Cycle. *Proceedings of the National Council for Systems Engineering (NCOSE) Conference*, Chattanooga, TN, pp. 57–65. October.

Forsberg, K., Mooz, H., & Cotterman, H. (2005). *Visualizing Project Management*, (3 Ed). Hoboken, NJ: John Wiley & Sons, Inc.

Fossnes, T. (2005). Lessons from Mt. Everest Applicable to Project Leadership. *Proceedings of the Fifteenth Annual INCOSE Symposium*. Rochester, NY: International Council on Systems Engineering.

Friedenthal, S. (1998). Object Oriented Systems Engineering. *Process Integration for 2000 and Beyond: Systems Engineering and Software Symposium*. New Orleans, LA: Lockheed Martin Corporation.

Friedenthal, S., Moore, A., & Steiner, R. (2012). *A Practical Guide to SysML: The Systems Modeling Language*, (2 Ed). New York, NY: Morgan Kaufmann Publishers, Inc.

Gallios, B., & Verma, D. (n.d.). System Design and Operational Effectiveness (SDOE): Blending Systems and Supportability Engineering Education. *Partnership in RMS Standards*, 5(1), 2.

GAO. (2008). Best Practices—Increased Focus on Requirements and Oversight Needed to Improve DOD's Acquisition Environment and Weapon System Quality. Washington, DC: U.S. Government Accountability Office. Retrieved from http://www.gao.gov/new.items/d08294.pdf (accessed October 24, 2014).

Giachetti, R. E. (2010). *Designing of Enterprise Systems: Theory, Architecture, and Methods*. Boca Raton, FL: CRC Press.

Gilb, T. (2005). *Competitive Engineering*. Philadelphia, PA: Elsevier.

Gilb, T., & Graham, D. (1993). *Software Inspection*. Reading, MA: Addison-Wesley-Longman.

Grady, J. O. (1994). *System Integration*. Boca Raton, FL: CRC Press.

Gupta, J., & Sharma, S. (2004). *Creating Knowledge Based Organizations*. Boston, MA: Ida Group Publishing.

Haberfellner, R., & de Weck, O. (2005). Agile SYSTEMS ENGINEERING versus AGILE SYSTEMS Engineering. *Proceedings of the 15th Annual INCOSE International Symposium*. Rochester, NY: International Council on Systems Engineering.

Haimes, Y. (2012). Modelling Complex Systems of Systems with Phantom System Models. *Systems Engineering*, 15(3), 333–46.

Hall, A. (1962). *A Methodology for Systems Engineering*. Princeton, NJ: Van Nostrand.

Heijden, K., Bradfield, R., Burt, G., Cairns, G., & Wright, G. (2002). *The Sixth Sense: Accelerating Organizational Learning with Scenarios*. Chichester, UK: John Wiley & Sons, Ltd.

Herald, T., Verma, D., Lubert, C., & Cloutier, R. (2009). An Obsolescence Management Framework for System Baseline Evolution—Perspectives through the System Life Cycle. *Systems Engineering*, 12, 1–20.

Hipel, K., Jamshidi, M., Tien, J., & White, C. (2007). The Future of Systems, Man, and Cybernetics: Application Domains and Research Methods. *IEEE Transactions on Systems, Man and Cybernetics—Part C: Applications and Reviews*, 13(5), 726–43.

Hitchens, D. K. (2003). *Advanced Systems Thinking, Engineering, and Management*. Boston, MA: Artech House.

Hollnagel, E., Woods, D. D., & Leveson, N. (Eds.). (2006). *Resilience Engineering: Concepts and Precepts*. Aldershot, UK: Ashgate Publishing Limited.

Honour, E. (2013). Systems Engineering Return on Investment. Ph.D. Thesis, Defense and Systems Institute, University of South Wales. Retrieved from http://www.hcode.com/seroi/index.html (accessed January 26, 2015).

Hughes, T. (1998). In *Rescuing Prometheus* (pp. 141–95). New York, NY: Pantheon Books.

Hybertson, D. (2009). *Model-Oriented Systems Engineering Science: A Unifying Framework for Traditional and Complex Systems*. Boca Raton, FL: Auerback/CRC Press.

IEEE Std 828. (2012). *IEEE Standard for Configuration Management in Systems and Software Engineering*. New York, NY: Institute of Electrical and Electronics Engineers.

IEEE 1012. (2012). *IEEE Standard for System and Software Verification and Validation*. New York, NY: Institute of Electrical and Electronics Engineers.

IIBA. (2009). *A Guide to the Business Analysis Body of Knowledge (BABOK Guide)*. Toronto, ON: International Institute of Business Analysis.

INCOSE. (2004). What Is Systems Engineering? Retrieved from http://www.incose.org/practice/whatissystemseng.aspx (accessed June 14, 2004).

INCOSE. (2006). INCOSE Code of Ethics. International Council on Systems Engineering. Retrieved from http://www.incose.org/about/ethics.aspx (accessed February 15).

INCOSE. (2007). Systems Engineering Vision 2020. International Council on Systems Engineering, Technical Operations. Retrieved from http://www.incose.org/ProductsPubs/pdf/SEVision2020_20071003_v2_03.pdf (accessed October 24, 2014).

INCOSE. (2010a). Model-Based Systems Engineering (MBSE) Wiki, hosted on OMG server. Retrieved from http://www.omgwiki.org/MBSE/doku.php (accessed October 24, 2014).

INCOSE. (2010b). Systems Engineering Measurement Primer, TP-2010-005, Version 2.0. Measurement Working Group, San Diego, CA: International Council on Systems Engineering. Retrieved from http://www.incose.org/ProductsPubs/pdf/INCOSE_SysEngMeasurementPrimer_2010-1205.pdf (accessed October 24, 2014).

INCOSE. (2011). Lean Enablers for Systems Engineering. Lean Systems Engineering Working Group. Retrieved from INCOSE Connect https://connect.incowse.org/tb/leansw/ (accessed February 1).

INCOSE. (2012). Guide for the Application of Systems Engineering in Large Infrastructure Projects. TP-2010-007-01. INCOSE Infrastructure Working Group. San Diego, CA. June.

INCOSE RWG. (2012). *Guide for Writing Requirements*. San Diego, CA: International Council on Systems Engineering, Requirements Working Group.

INCOSE & PSM. (2005). Technical Measurement Guide, Version 1.0, December 2005, http://www.incose.org and http://www.psmsc.com (accessed October 24, 2014).

INCOSE UK. (2010). Systems Engineering Competencies Framework. INCOSE United Kingdom, Ltd. Retrieved from https://connect.incose.org/products/SAWG%20Shared%20Documents/Other%20Technical%20Products/FRAMEWORK_WEB_Jan2010_Issue3.pdf (accessed October 24, 2014).

ISM. (n.d.). Institute for Supply Management. Retrieved from http://www.ism.ws/ (accessed October 24, 2014).

ISO 9001. (2008). *Quality management systems—Requirements*. Geneva, Switzerland: International Organization for Standardization.

ISO 10007. (2003). *Quality Management Systems—Guidelines for Configuration Management*. Geneva, Switzerland: International Organization for Standardization.

ISO 10303-233. (2012). *Industrial Automation Systems and Integration—Product Data Representation and Exchange—Part 233: Application Protocol: Systems Engineering*. Geneva, Switzerland: International Organization for Standardization.

ISO 13407. (1999). *Human-Centered Design Processes for Interactive Systems*. Geneva, Switzerland: International Organization for Standardization.

ISO 14001. (2004). *Environmental Management Systems—Requirements with Guidance for Use*. Geneva, Switzerland: International Organization for Standardization. Retrieved from http://14000store.com (accessed October 24, 2014).

ISO 14971. (2007) *Medical Devices—Application of Risk Management to Medical Devices*. Geneva, Switzerland: International Organization for Standardization.

ISO 17799. (2005) *Information Technology—Security Techniques: Code of Practice for Information Security Management*. Geneva, Switzerland: ISO/IEC.

ISO 26262. (2011). *Road Vehicles—Functional Safety*. Geneva, Switzerland: International Organization for Standardization.

ISO 31000. (2009). *Risk Management—Principles and Guidelines*. Geneva, Switzerland: International Organization for Standardization.

ISO 31010. (2009). *Risk Management—Risk Assessment Techniques*. Geneva, Switzerland: International Organization for Standardization.

IOS Guide 73. (2002). Risk Management—Vocabulary. International Organization for Standardization.

ISO Guide 73. (2009). *Risk Management—Vocabulary*. Geneva, Switzerland: International Organization for Standardization.

ISO/IEC 16085. (2006). *Systems and Software Engineering—Life Cycle Processes: Risk Management*. Geneva, Switzerland: International Organization for Standardization.

ISO/IEC 27002. (2013). *Information Technology—Security Techniques: Code of Practice for Information Security Controls*. Geneva, Switzerland: International Organization for Standardization.

ISO/IEC Guide 51. (1999). *Safety Aspects—Guidelines for the Inclusion in Standards*. Geneva, Switzerland: International Organization for Standardization.

ISO/IEC TR 19760. (2003) *Systems Engineering—A Guide for the Application of ISO/IEC 15288*. Geneva, Switzerland: International Organization for Standardization.

ISO/IEC TR 24748-1. (2010). *System and Software Engineering—Life Cycle Management—Part 1: Guide for Life Cycle Management*. Geneva, Switzerland: International Organization for Standardization. Retrieved from http://standards.iso.org/ittf/PubliclyAvailableStandards/index.html (accessed October 24, 2014).

ISO/IEC TR 24748-2. (2010). *Systems and Software Engineering—Life Cycle Management—Part 2: Guide to the Application of ISO/IEC 15288*. Geneva, Switzerland: International Organization for Standardization.

ISO/IEC/IEEE 15288. (2015). *Systems and Software Engineering—System Life Cycle Processes*. Geneva, Switzerland: International Organization for Standardization.

ISO/IEC/IEEE 15939, 2007. *Systems and Software Engineering—Measurement Process, ISO/IEC 2007*. Geneva, Switzerland: International Organization for Standardization.

ISO/IEC/IEEE 16326. (2009). *Systems and Software Engineering—Life Cycle Processes: Project Management*. Geneva, Switzerland: International Organization for Standardization.

ISO/IEC/IEEE 24748-4. (2014). *Systems and Software Engineering—Life Cycle Management—Part 4: Systems Engineering Planning*. Geneva, Switzerland: International Organization for Standardization.

ISO/IEC/IEEE 24765. (2010). *Systems and Software Engineering—Vocabulary*. Geneva, Switzerland: International Organization for Standardization. Retrieved from http://pascal.computer.org/sev_display/index.action (accessed October 24, 2014).

ISO/IEC/IEEE 29110 Series (2014) *Systems and Software Engineering—Lifecycle Profiles for Very Small Entities (VSEs)*. International Organization for Standardization.

ISO/IEC/IEEE 29119. (2013). *Software Testing Standard*. Geneva, Switzerland: International Organization for Standardization.

ISO/IEC/IEEE 29148. (2011). *Systems and Software Engineering—Life Cycle Processes: Requirements Engineering*. Geneva, Switzerland: International Organization for Standardization.

ISO/IEC/IEEE 42010. (2011). *Systems and Software Engineering—Recommended Practice for Architectural Descriptions of Software-Intensive Systems*. Geneva, Switzerland: International Organization for Standardization.

Jackson, M. (1989). Which Systems Methodology When? Initial Results from a Research Program. In R. Flood, M. Jackson, & P. Keys (Eds.), *Systems Prospects: The Next Ten Years of Systems Research*. New York, NY: Plenum Press.

Jackson, S., & Ferris, T. (2013). Resilience Principles for Engineered Systems. *Systems Engineering* 19(2), 152–164.

Jacky, J. (1989). Programmed for Disaster. *The Sciences*, 29, 22–7.

Jensen, J. (2014). The Øresund Bridge—Linking Two Nations. Retrieved from http://www.cowi.dk (accessed October 24, 2014).

JHUAPL. (2011). Tutorial Material—Model-Based Systems Engineering Using the Object-Oriented Systems Engineering Method (OOSEM). The Johns Hopkins University Applied Physics Laboratory. Retrieved from APL Technology Transfer http://www.jhuapl.edu/ott/technologies/copyright/sysml.asp (accessed October 24, 2014).

Johnson, S. (2010). *Where Good Ideas Come From: The Natural History of Innovation*. New York, NY: Riverhead Books.

Juran, J. M. (Ed.). (1974). *Quality Control Handbook*, 3rd Ed. New York, NY: McGraw-Hill.

Kaposi, A., & Myers, M. (2001). *Systems for All*. London, UK: Imperial College Press.

Katzan, H. (2008). *Service Science*. Bloomington, IN: iUniverse Books.

KBS. (2010). IDEF Family of Methods. Knowledge Based Systems, Inc. Retrieved from IDEF: Integrated DEFinition Methods http://www.idef.com/ (accessed October 24, 2014).

Keeney, R. L. (2002). Common Mistakes in Making Value Trade-Offs. *Operations Research*, 50(6), 935–45.

Keeney, R., & Gregory, R. (2005). Selecting Attributes to Measure the Achievement of Objectives. *Operations Research*, 15(1), 1–11.

Keoleian, G. A., & Menerey, D. (1993). Life Cycle Design Guidance Manual: Environmental Requirements and the Product System, EPA/600/R-92/226. Environmental Protection Agency, Risk Reduction Engineering Laboratory. Retrieved from http://css.snre.umich.edu/css_doc/CSS93-02.pdf (accessed October 24, 2014).

Klir, G. (1991). *Facets of Systems Science*. New York, NY: Plenum Press.

Kotter, J. P. (2001). *What Leaders Really Do*. Boston, MA: Harvard Business Review: Best of HBR.

Kruchten, P. (1999). The Software Architect and the Software Architecture Team. In P. Donohue (Ed.), *Software*

Architecture (pp. 565–583). Boston, MA: Kluwer Academic Publishers.

LAI MIT. (2013). Lean Enterprise Value Phase 1. Retrieved from MIT Lean Advancement Initiative http://lean.mit.edu/about/history/phase-i?highlight=WyJwaGFzZSIsMV0= (accessed October 24, 2014).

LAI, INCOSE, PSM, and SEARI. (2010). Systems Engineering Leading Indicators Guide, Version 2.0, January 29. Retrieved from http://www.incose.org and http://www.psmsc.com (accessed October 24, 2014).

Langley, M., Robitaille, S., & Thomas, J. (2011). Toward a New Mindset: Bridging the Gap Between Program Management and Systems Engineering. *PM Network*, 25(9), 24–6. Retrieved from http://www.pmi.org (accessed October 24, 2014).

Langner, R. (2012). Stuxnet Deep Dive. Miami Beach, FL: SCADA Security Scientific Symposium (S4). Retrieved from http://vimeopro.com/user10193115/s4-2012#/video/35806770 (accessed October 24, 2014).

Lano, R. (1977). *The N2 Chart*. Euclid, OH: TRW, Inc.

Larman, C., & Basili, V. (2003, June). Iterative and Incremental Development: A Brief History. *IEEE Software* 36(6): 47–56.

Lawson, H. (2010). *A Journey Through the Systems Landscape*. Kings College, UK: College Publications.

Lefever, B. (2005). ScSCE Methodology. Retrieved from SeCSE Service Centric Systems Engineering: http://www.secse-project.eu/ (accessed January 26, 2015).

Leveson, N., & Turner, C. S. (1993). An Investigation of the Therac-25 Accidents. *IEEE Computer*, 26(7), 18–41.

Lewin, K. (1958). *Group Decision and Social Change*. New York, NY: Holt, Rinehart and Winston.

Lin, F., & Hsieh, P. (2011). A SAT View on New Service Development. *Service Science*, 3(2), 141–57.

Lindvall, M., & Rus, I. (2000). Process Diversity in Software Development. *IEEE Software*. 17(4), 14–8.

Little, W., Fowler, H. W., Coulson, J., & Onions, C. T. (1973). *The Shorter Oxford English Dictionary on Historical Principles*, 3rd Ed. Oxford, UK: Oxford University Press.

LMCO. (2008). Object-Oriented Systems Engineering Method (OOSEM) Tutorial, Version 3.11. Bethesda, MD: Lockheed Martin Corporation and San Diego, CA: INCOSE OOSEM Working Group.

Lombardo, M. M., & Eichinger, R. W. (1996). *The Career Architect Development Planner*, 1st Ed. Minneapolis, MN: Lominger.

Lonergan, B. (1992). Insight: A Study of Human Understanding. In F. E. Crowe, & R. M. Doran (Eds.), *Collected Works of Bernard Lonergan* (5 Ed, Vol. 3). Toronto, ON: University of Toronto Press.

Long, D. (2013). The Holistic Perspective—Systems Engineering as Leaders. *INCOSE Great Lakes Regional Conference Key Note Address*. West Lafayette, IN.

Luzeaux, D., & Ruault, J. R. (Eds.). (2010). *Systems of Systems*. New York, NY: John Wiley & Sons, Inc.

Lykins, H., Friedenthal, S., & Meilich, A. (2000). Adapting UML for an Object-Oriented Systems Engineering Method (OOSEM). *Proceedings of the 20th Annual INCOSE International Symposium*. Chicago, IL: INCOSE.

M&SCO. (2013). Verification, Validation, & Accreditation (VV&A) Recommended Practices Guide (RPG). Retrieved from U.S. DoD Modeling & Simulation Coordination Office: http://www.msco.mil/VVA_RPG.html (accessed October 24, 2014).

Magerholm, F. A., Shau, E. M., & Haskins, C. (2010). A Framework for Environmental Analyses of Fish Food Production Systems Based on Systems Engineering Principles. *Systems Engineering*, 13, 109–18.

Maglio, P., & Spohrer, J. (2008). Fundamentals of Service Science. *Journal of the Academy of Marketing Science*, 36(1), 18–20.

Maier, M. W. (1998). Architecting Principles for Systems of Systems. *Systems Engineering*, 1(4), 267–84.

Maier, M. W., & Rechtin, E. (2009). *The Art of Systems Architecting*, 3rd Ed. Boca Raton, FL: CRC Press.

Martin, B. (2010). New Initiatives in the Army Green Procurement Program. Presentation, U.S. Army Public Health Command. Retrieved from http://www.dtic.mil/dtic/tr/fulltext/u2/a566679.pdf (accessed October 24, 2014).

Martin, J. N. (1996). *Systems Engineering Guidebook: A Process for Developing Systems and Products*. Boca Raton, FL: CRC Press.

Martin, J. N. (2011). Transforming the Enterprise Using a Systems Approach. *Proceedings of the 21st Annual INCOSE International Symposium*. Denver, CO: International Council on Systems Engineering.

McAfee, A. (2009). *Enterprise 2.0: New Collaborative Tools for Your Organizations Toughest Challenges*. Boston, MA: Harvard Business School Press.

McConnel, S. (1998, May). The power of process. *IEEE Computer* 31(5), 100–102.

McDonough, W. (2013). McDonough Innovations: Design for the Ecological Century. Retrieved from http://www.mcdonough.com/ (accessed October 24, 2014).

McGarry, J., Card, D., Jones, C., Layman, B., Clark, E., Dean, J., & Hall, F. (2001). *Practical Software Measurement: Objective Information for Decision Makers*, Boston, MA: Addison-Wesley.

McGrath, D. (2003). China Awaits High Speed Maglev. Wired News. Retrieved from http://archive.wired.com/science/discoveries/news/2003/01/57163 (accessed January 26, 2015).

McManus, H. L. (2004). *Product Development Value Stream Mapping Manual, LAI Release Beta*. Boston, MA: MIT Lean Advancement Initiative.

McManus, H. L. (2005). *Product Development Transition to Lean (PDTTL) Roadmap, LAI Release Beta*. Boston, MA: MIT Lean Advancement Initiative.

Michel, R. M., & Galai, D. (2001). *Risk Management*. New York, NY: McGraw-Hill.

Miller, G. A. (1956). The Magical Number Seven, Plus or Minus Two: Some Limits on our Capacity for Processing Information. *Psychological Review*, 63(2), 81.

MoDAF. (n.d.). The Website for MODAF Users and Implementers. UK Secretary of State for Defence (run by Model Futures, Ltd.). Retrieved from http://www.modaf.org.uk/ (accessed October 24, 2014).

Murman, E. M. (2002). *Lean Enterprise Value*. New York, NY: Palgrave.

Nagel, R. N. (1992). 21st Century Manufacturing Enterprise Strategy Report. Prepared for the Office of Naval Research. Retrieved from http://www.dtic.mil/cgi-bin/GetTRDoc?AD=ADA257032 (accessed October 24, 2014).

NASA. (2007a). NASA Pilot Benchmarking Initiative: Exploring Design Excellence Leading to Improved Safety and Reliability. Final Report. National Aeronautic and Space Administration.

NASA. (2007b). NASA Systems Engineering Handbook. National Aeronautic and Space Administration. Retrieved from http://www.es.ele.tue.nl/education/7nab0/2013/doc/NASA-SP-2007-6105-Rev-1-Final-31Dec2007.pdf (accessed October 24, 2014).

NDIA. (2008). Engineering for Systems Assurance, Version 1.0. National Defense Industrial Association, Systems Assurance Committee. Retrieved from http://www.acq.osd.mil/se/docs/SA-Guidebook-v1-Oct2008.pdf (accessed October 24, 2014).

NDIA. (2011). System Development Performance Measurement Report. Arlington, VA: National Defense Industrial Association, December.

Nissen, J. (2006). The Øresund Link. *The Arup Journal*, 31(2), 37–41.

NIST. (2012). National Vulnerability Database, Version 2.2. National Institute of Standards and Technology, Computer Science Division. Retrieved from http://nvd.nist.gov/ (accessed December 19)

Norman, D. (1990). *The Design of Everyday Things*. New York, NY: Doubleday.

NRC (Nuclear Regulatory Commission). (2005). Fact Sheets, Three Mile Island Accident. U.S. Nuclear Regulatory Commission. Report NUREG/BR-0292. Retrieved from http://pbadupws.nrc.gov/docs/ML0825/ML082560250.pdf (accessed October 24, 2014).

NRC (National Research Council). (2008). Pre-Milestone A and Early-Phase Systems Engineering. National Research Council of the National Academies. Washington, DC: The National Academies Press. Retrieved from http://www.nap.edu/ (accessed October 24, 2014).

NRC (National Resilience Coalition). (2012). Definition of Resilience. National Resilience Coalition. Washington, DC: The Infrastructure Security Partnership.

O'Connor, P. D., & Kleyner, A. (2012). *Practical Reliability Engineering*, 5th Ed. Hoboken, NJ: John Wiley & Sons, Inc.

Oehmen, J. (Ed.). (2012). The Guide to Lean Enablers for Managing Engineering Programs, Version 1.0. Cambridge, MA: Joint MIT-PMI-INCOSE Community of Practice on Lean in Program Management. Retrieved from http://hdl.handle.net/1721.1/70495/ (accessed October 24, 2014).

Office of Government Commerce. (2009). *ITIL Lifecycle Publication Suite Books*. London, UK: The Stationery Office.

OMG. (2013a). Documents Associated with Unified Profile for DoDAF and MODAF (UPDM), Version 2.1. Object Management Group, Inc. Retrieved from http://www.omg.org/spec/UPDM/Current (accessed October 24, 2014).

OMG. (2013b). OMG Systems Modeling Language (SysML). Object Management Group, Inc. Retrieved from http://www.omgsysml.org (accessed October 24, 2014).

Oppenheim, B. W. (2004). Lean Product Development Flow. *Systems Engineering*, 7(4), 352–76.

Oppenheim, B. W. (2011). *Lean for Systems Engineering with Lean Enablers for Systems Engineering*. Hoboken, NJ: John Wiley & Sons, Inc.

Parnell, G. S., Bresnick, T., Tani, S., & Johnson, E. (2013). *Handbook of Decision Analysis*. Hoboken, NJ: John Wiley & Sons, Inc.

Patanakul, P., & Shenhar, A. (2010). Exploring the Concept of Value Creation in Program Planning and Systems Engineering Processes. *Systems Engineering*, 13, 340–52.

Pineda, R. (2010). Understanding Complex Systems of Systems Engineering. *Fourth General Assembly Cartagena Network of Engineering*, Metz, France.

Pineda, Martin, & Spoherer, (2014), in SEBoK, Part 3, Value of Service Systems Engineering.

PLCS. (2013). Product Life Cycle Support (PLCS). Retrieved from http://www.plcs-resources.org/ (accessed October 24, 2014)..

PMI. (2000). *A Guide to the PMBOK*. Newton Square, PA: Project Management Institute.

PMI. (2009). *Practice Standard for Project Risk Management*. Newton Square, PA: Project Management Institute.

PMI. (2013). *The Standards for Program Management*, 3rd Ed. Newton Square, PA: Project Management Institute.

Porrello, A. M. (n.d.). Death and Denial: The Failure of the THERAC-25, A Medical Linear Accelerator. California Polytechnic State University, San Luis Obispo, CA. Retrieved from http://users.csc.calpoly.edu/~jdalbey/SWE/Papers/THERAC25.html (accessed October 24, 2014).

Qiu, R. (2009). Computational Thinking of Service Systems: Dynamics and Adaptiveness Modeling. *Service Science*, 1(1), 42–55.

Rebovich, G. (2006). Systems thinking for the enterprise: new and emerging perspectives. *Proceedings 2006 IEEE /SMC International Conference on System of Systems Engineering*. Los Angeles, CA: Institute of Electrical and Electronics Engineers.

Rebovich, G., & White, B. E. (Eds.). (2011). *Enterprise Systems Engineering: Advances in the Theory and Practice*. Boca Raton, FL: CRC Press.

Richards, M. G. (2009). *Multi-Attribute Tradespace Exploration for Survivability*. Boston: Massachusetts Institute of Technology.

Roedler, G. (2010). Knowledge Management Position. *Proceedings of the 20th Annual INCOSE International Symposium*. Chicago, IL: International Council on Systems Engineering.

Roedler, G. J., & Jones, C. (2006). Technical Measurement: A Collaborative Project of PSM, INCOSE, and Industry. INCOSE Measurement Working Group. INCOSE TP-2003-020-01.

Roedler, G., Rhodes, D. H., Schimmoler, H., & Jones, C. (Eds.). (2010). Systems Engineering Leading Indicators Guide, v2.0. Boston: MIT; INCOSE; PSM.

Ross, A. M., Rhodes, D. H., & Hastings, D. E. (2008). Defining Changeability: Reconciling Flexibility, Adaptability, Scalability, Modifiability, and Robustness for Maintaining System Lifecycle Value. *Systems Engineering*, 11, 246–62.

Rouse, W. B. (1991). *Design for Success: A Human-Centered Approach to Designing Successful Products and Systems*. New York, NY: John Wiley & Sons, Inc.

Rouse, W. B. (2005). Enterprise as Systems: Essential Challenges and Enterprise Transformation. *Systems Engineering*, 8(2), 138–50.

Rouse, W. B. (2009). Engineering the Enterprise as a System. In A. P. Sage, & W. B. Rouse (Eds.), *Handbook of Systems Engineering and Management*, 2nd Ed. New York, NY: John Wiley & Sons, Inc.

Royce, W. W. (1970). Managing the Development of Large Software Systems, *Proceedings, IEEE WESCON*, pp. 1–9. August.

Ryan, A. (2008). What Is a Systems Approach? *Journal of Non-Linear Science*, arXiv, 0809.1698.

Ryan, M. (2013). An Improved Taxonomy for Major Needs and Requirements Artifacts. *Proceedings of the 23rd Annual INCOSE International Symposium*. Philadelphia, PA: International Council on Systems Engineering.

SAE Aerospace Quality Standard AS9100:C. (2009). *Quality Management Systems—Requirements for Aviation, Space, and Defense Organizations*. Warrendale, PA: Society of Automotive Engineers.

SAE ARP 4754. (2010). *Guidelines for Development of Civil Aircraft and Systems*. Warrendale, PA: Society of Automotive Engineers.

SAE JA 1011 (2009). Evaluation criteria for Reliability-Centered Maintenance (RCM) processes. Warrendale, PA: Society of Automotive Engineers.

Salter, K. (2003). Presentation Given at the Jet Propulsion Laboratory. Pasadena, CA.

Salvendy, G. (Ed.). (1982). *Handbook of Industrial Engineering*. New York, NY: John Wiley & Sons, Inc.

Salzman, J. (1997). Informing the Green Consumer: The Debate Over the Use and Abuse of Environmental Labels. *Journal of Industrial Ecology*, 1(2), 11–21.

SAVE. (2009). Welcome to SAVE International. SAVE International. Retrieved from http://www.value-eng.org/ (accessed October 24, 2014).

SAWE. (n.d.). Standards and Practices. Retrieved from Society of Allied Weight Engineers, Inc.: http://www.sawe.org/rp (accessed October 24, 2014).

Scholtes, P. R. (1988). *The Team Handbook: How the Use Teams to Improve Quality*. Madison, WI: Joiner Associates, Inc.

SE VOCAB. (2013). Software and Systems Engineering Vocabulary. Retrieved from http://pascal.computer.org/sev_display/index.action (accessed October 24, 2014).

SEBoK. (2014). BKCASE Editorial Board. The Guide to the Systems Engineering Body of Knowledge (SEBoK), version 1.3. R.D. Adcock (EIC). Hoboken, NJ: The Trustees of the Stevens Institute of Technology. www.sebokwiki.org. BKCASE is managed and maintained by the Stevens Institute of Technology Systems Engineering Research Center, the International Council on Systems Engineering, and the Institute of Electrical and Electronics Engineers Computer Society.

Software Engineering Institute. (2010). CMMI® (Measurement and Quantitative Management Process Areas), Version 1.3, November. Retrieved from http://www.sei.cmu.edu (accessed October 24, 2014).

Senge, P. (1990). *The Fifth Discipline: The Art & Practice of the Learning Organization*. New York, NY: Crown Business.

Shaw, T. E., & Lake, J. G. (1993). Systems Engineering: The Critical Product Development Enabler. *36th APICS International Conference Proceedings*. American Production and Inventory Control Society.

Sheard, S. (1996). Twelve Systems Engineering Roles. *Proceedings of the 6th Annual INCOSE International Symposium*. Boston, MA: International Council on Systems Engineering.

Shewhart, W. A. (1939). *Statistical Method from the Viewpoint of Quality Control*. New York, NY: Dover.

Sillitto, H. G. (2012). Integrating Systems Science, Systems Thinking, and Systems Engineering: Understanding the Differences and Exploiting the Synergies. *Proceedings of the 22nd Annual INCOSE International Symposium*. Rome, Italy: International Council on Systems Engineering.

Sillitto, H. G. (2013). Composable Capability—Principles, Strategies, and Methods for Capability Systems Engineering. *Proceedings of the 23rd Annual INCOSE International Symposium*. Philadelphia, PA: International Council on Systems Engineering.

Skanska. (2013). Øresund Bridge: Improving Daily Life for Commuters, Travelers, and Frogs. Retrieved from Skanska: Øresund Consortium: http://www.group.skanska.com/Campaigns/125/Oresund-Bridge/ (accessed October 24, 2014).

Skyttner, L. (2006). *General Systems Theory: Perspectives, Problems, Practice*, 2nd Ed. Singapore: World Scientific Publishing Company.

Slack, R. A. (1998). Application of Lean Principles to the Military Aerospace Product Development Process. Master of Science—Engineering and Management Thesis, Massachusetts Institute of Technology.

SMTDC. (2005). Shanghai Maglev Project Background. Retrieved from Shanghai Maglev Train: http://www.smtdc.com/en/ (accessed January 26, 2015).

Sols, A., Romero, J., & Cloutier, R. (2012). Performance-based Logistics and Technology Refreshment Programs: Bridging the Operational-Life Performance Capability Gap in the Spanish F-100 Frigates. *System Engineering*, 15, 422–32.

Spath, D., & Fahnrich, K. P. (Eds.). (2007). *Advances in Services Innovations*. Berlin/Heidelberg, Germany: Springer-Verlag.

Spohrer, J. C. (2011). Service Science: Progress & Direction. International Joint Conference on Service Science. Taipei, Taiwan.

Spohrer, J. C., & Maglio, P. P. (2010). Services Science: Toward a Smarter Planet. In G. Slavendy, & W. Karwowski (Eds.), *Introduction to Service Engineering*. Hoboken, NJ: John Wiley & Sons, Inc.

Srinivansan, J. (2010). Towards a Theory Sensitive Approach to Planning Enterprise Transformation. 5th EIASM Workshop on Organizational Change and Development. Vienna, Austria: European Institute for Advanced Studies in Management.

Stoewer, H. (2005). Modern Systems Engineering: A Driving Force for Industrial Competitiveness. Presentation to Members of the Japan INCOSE Chapter. Tokyo, Japan.

The White House. (2010). *National Security Strategy*. Washington, DC: The White House.

Theilmann, W., & Baresi, L. (2009). Multi-level SLAs for Harmonized Management in the Future Internet. In G. Tselentix, J. Dominque, A. Galis, A. Gavras, D. Hausheer, S. Krco, . . . T. Zehariadis (Eds.), *Towards the Future Internet—A European Research Perspective*. Amsterdam, the Netherlands: IOS Press.

Tien, J., & Berg, D. (2003). A Case for Service Systems Engineering. *Journal Systems Science and Systems Engineering*, 12(1), 13–38.

Toyota. (2009). Toyota Production System: Just-in-Time—Productivity Improvement. Retrieved from http://www.toyota-global.com/company/vision_philosophy/toyota_production_system/ (accessed October 24, 2014).

Transrapid. (2003, May 23). Transrapid International. Retrieved from http://www.transrapid.de/cgi/en/basics.prg?session=86140b5952535725_590396 (accessed October 24, 2014).

Tuttle, P., & Bobinis, J. (2013). Specifying Affordability. *Proceedings of the 23rd Annual INCOSE International Symposium*. Philadelphia, PA: International Council on Systems Engineering.

UNEP. (2012). GEO-5: Global Environmental Outlook. Retrieved from United National Environment Programme: http://www.unep.org/geo (accessed October 24, 2014).

Urwick, L. E. (1956). The Manager's Span of Control. Harvard Business Review. May/June.

Vargo, S. L., & Akaka, M. A. (2009). Service-Dominant Logic as a Foundation for Service Science: Clarifications. *Service Science*, 1(1), 32–41.

Walden, D. (2007). YADSES: Yet Another Darn Systems Engineering Standard. *Proceedings of the 17th Annual INCOSE International Symposium*. San Diego, CA: International Council on Systems Engineering.

Warfield, J. (2006). *An Introduction to Systems Science*. Hackensack, NJ: World Scientific Publishing Company.

Waters Foundation. (2013). Systems Thinking. Retrieved from Systems Thinking in Schools: http://watersfoundation.org/systems-thinking/overview/ (accessed October 24, 2014).

Wideman, R. M. (2002). Comparative Glossary of Project Management Terms, Version 3.1. Retrieved from http://maxwideman.com/pmglossary/ (accessed October 24, 2014).

Wideman, R. M. (Ed.). (2004). *Project and Program Risk Management: A Guide to Managing Project Risks and*

Opportunities. Newtown Square, PA: Project Management Institute.

Wiener, N. (1948). *Cybernetics or Control and Communication in the Animal and the Machine*. (Hermann, & Cie, Eds.) New York, NY: John Wiley & Sons, Inc.

Wild, J. P., Jupp, J., Kerley, W., Eckert, W., & Clarkson, P. J. (2007). Towards a Framework for Profiling of Products and Services. 5th International Conference on Manufacturing Research (ICMR). Leicester, UK.

Womack, J. P., & Jones, D. T. (1996). *Lean Thinking*. New York, NY: Simon & Schuster.

Wymore, A.W. (1967). *A Mathematical Theory of Systems Engineering: The Elements*. Malabar, FL: Krieger Publication Co.

Wymore, A. W. (1993). *Model-Based Systems Engineering*. Boca Raton, FL: CRC Press.

Yourdon, E. (1989). *Modern Structured Analysis*. Upper Saddle River, NJ: Yourdon Press.

Zachman, J. A. (1987). A Framework for Information Systems Architecture. *IBM Systems Journal*, 26(3), 276–92. Retrieved from http://www.zifa.com (accessed October 24, 2014).

APPENDIX B: ACRONYMS

A_a	Achieved availability
A_i	Inherent availability
A_o	Operational availability
act	Activity diagrams [SysML™]
AECL	Atomic Energy Commission Limited [Canada]
AIAA	American Institute of Aeronautics and Astronautics [United States]
ANSI	American National Standards Institute [United States]
API	Application programming interface
ARP	Aerospace Recommended Practice
AS	Aerospace Standard
ASQ	American Society for Quality
ASAM	Association for Standardization of Automation and Measuring Systems
ASEP	Associate Systems Engineering Professional [INCOSE]
AUTOSAR	AUTomotive Open System ARchitecture
AVS	Associate Value Specialist [SAVE]
bdd	Block definition diagram [SysML™]
BRS	Business Requirements Specification
CAIV	Cost as an Independent Variable
CBA	Cost–benefit analysis
CBM	Condition-based maintenance
CCB	Configuration Control Board
CE	Conformité Européenne [EU]
CEA	Cost-effectiveness analysis
CFR	Code of Federal Regulations [United States]
CI	Configuration item
CMMI®	Capability Maturity Model® Integration [CMMI Institute]
CMP	Configuration management plan
ConOps	Concept of operations
COTS	Commercial off-the-shelf
CSEP	Certified Systems Engineering Professional [INCOSE]
CVS	Certified Value Specialist [SAVE]
DAU	Defense Acquisition University [United States]
DFD	Data flow diagrams
DMS	Diminishing material shortages
DoD	Department of Defense [United States]
DoDAF	Department of Defense Architecture Framework [United States]
DSM	Design Structure Matrix
DTC	Design to cost
ECP	Engineering Change Proposal
ECR	Engineering change request

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.

EIA	Electronic Industries Alliance
EM	Electromagnetic
EMC	Electromagnetic compatibility
EMI	Electromagnetic interference
EN	Engineering notice
ER	Entity relationship diagram
ESEP	Expert Systems Engineering Professional [INCOSE]
ETA	Event tree analysis
FAST	Function Analysis System Technique
FBSE	Functions-based systems engineering
FEA	Front-end analyses
FEP	Fuel enrichment plant
FFBD	Functional flow block diagram
FHA	Functional hazard analysis
FMEA	Failure Mode and Effects Analysis
FMECA	Failure modes, effects, and criticality analysis
FMVSS	Federal Motor Vehicle Safety Standards [United States]
FoS	Family of systems
FTA	Fault tree analysis
G&A	General and administrative
GAO	Government Accountability Office [United States]
GEIA	Government Electronic Industries Alliance
GENIVI	Geneva In-Vehicle Infotainment Alliance
GNP	Gross national product
GPP	Green Public Procurement
HALT	Highly accelerated life testing
HFE	Human factors engineering
HHA	Health hazard analysis
HSI	Human systems integration
HTS	Hazard tracking system
ibd	Internal block diagram [SysML™]
IBM	International Business Machines
ICD	Interface control document
ICS	Industrial control system
ICSM	Incremental Commitment Spiral Model
ICWG	Interface Control Working Group
IDEF	Integrated definition for functional modeling
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
IFWG	Interface Working Group
IID	Incremental and iterative development
ILS	Integrated logistics support
INCOSE	International Council on Systems Engineering
IPAL	INCOSE Product Asset Library [INCOSE]
IPD	Integrated Product Development
IPDT	Integrated Product Development Team
IPO	Input–process–output
IPPD	Integrated Product and Process Development
IPT	Integrated Product Team
ISO	International Organization for Standardization
IT	Information technology
IV&V	Integration, verification, and validation
JSAE	Japan Society of Automotive Engineers [Japan]
JPL	Jet Propulsion Laboratory [United States]
KDR	Key driving requirement
KM	Knowledge management
KPP	Key Performance Parameter
KSA	Knowledge, skills, and abilities
LAI	Lean Advancement Initiative
LCA	Life cycle assessment
LCC	Life cycle cost
LCM	Life cycle management
LEfMEP	Lean Enablers for Managing Engineering Programs
LEfSE	Lean Enablers for Systems Engineering
LINAC	Linear accelerator
LOR	Level of rigor
LORA	Level of Repair Analysis
MBSE	Model-based systems engineering
MIT	Massachusetts Institute of Technology
MoC	Models of computation
MODA	Multiple objective decision analysis
MoDAF	Ministry of Defense Architecture Framework [United Kingdom]
MOE	Measure of effectiveness
MOP	Measure of performance
MORS	Military Operations Research Society
MOS	Measure of suitability
MPE	Mass Properties Engineering
MTA	Maintenance Task Analysis
MTBF	Mean time between failure
MTBR	Mean time between repair
MTTR	Mean time to repair
N^2	N-squared diagram
NASA	National Aeronautics and Space Administration [United States]

NEC	National Electrical Code [United States]
NCOSE	National Council on Systems Engineering (pre-1995)
NCS	Network-Centric Systems
NDI	Nondevelopmental item
NDIA	National Defense Industrial Association [United States]
O&SHA	Operations and support hazard analysis
OAM&P	Operations, administration, maintenance, and provisioning
OEM	Original Equipment Manufacturer
OMG	Object Management Group
OOSEM	Object-Oriented Systems Engineering Method
OpEMCSS	Operational Evaluation Modeling for Context-Sensitive Systems
OPM	Object-Process Methodology
OpsCon	Operational concept
OSI	Open System Interconnect
par	Parametric diagram [SysML™]
PBL	Performance-based logistics
PBS	Product breakdown structure
PDT	Product Development Team
PERT	Program Evaluation Review Technique
PHA	Preliminary hazard analysis
PHS&T	Packaging, handling, storage, and transportation
PIT	Product Integration Team
pkg	Package diagram [SysML™]
PLC	Programmable logic controller
PLCS	Product Life Cycle Support
PLM	Product line management
PMI	Project Management Institute
PRA	Probabilistic risk assessment
PSM	Practical Software and Systems Measurement
QA	Quality assurance
QM	Quality management
R&D	Research and development
RAM	Reliability, availability, and maintainability
RBD	Reliability block diagram
RCM	Reliability-centered maintenance
REACH	Registration, Evaluation, Authorization, and Restriction of Chemical Substances
req	Requirement diagram [SysML™]
RFC	Request for change
RFP	Request for proposal
RFQ	Request for quote
RFV	Request for variance
RMP	Risk management plan
ROI	Return on investment
RUP	Rational Unified Process [IBM]
RUP-SE	Rational Unified Process for Systems Engineering [IBM]
RVTM	Requirements Verification and Traceability Matrix
SA	State Analysis [JPL]
SAE	SAE International [formerly the Society of Automotive Engineers]
SAR	Safety Assessment Report
SAVE	Society of American Value Engineers
SCM	Supply chain management
SCN	Specification change notice
sd	Sequence diagram [SysML™]
SE	Systems engineering
SEARI	Systems Engineering Advancement Research Institute
SEBoK	Guide to the Systems Engineering Body of Knowledge
SEH	Systems Engineering Handbook [INCOSE]
SEHA	System element hazard analysis
SEIT	Systems Engineering and Integration Team
SEMP	Systems engineering management plan
SEMS	Systems Engineering Master Schedule
SEP	Systems engineering plan
SHA	System hazard analysis
SLA	Service-level agreement
SOI	System of interest
SoS	System of systems
SOW	Statement of work
SROI	Social return on investment
SRR	System Requirements Review
SSDP	Service system design process
STEP	Standard for the Exchange of Product Model Data
stm	State machine diagram [SysML™]
StRS	Stakeholder Requirements Specification
SWOT	Strength–weakness–opportunity–threat
SysML™	Systems Modeling Language [OMG]
SyRS	System Requirements Specification
SYSPG	Systems Engineering Process Group
TADSS	Training Aids, Devices, Simulators, and Simulations
TCO	Total cost of ownership
TOC	Total ownership cost

TOGAF	The Open Group Architecture Framework
TPM	Technical performance measure
TRL	Technology readiness level
TRP	Technology refreshment program
TQM	Total quality management
TR	Technical report
uc	Use case diagram [SysML™]
UIC	International Union of Railways
UK	United Kingdom
UL	Underwriters Laboratory [United States and Canada]
UML™	Unified Modeling Language™ [OMG]
US	United States
USB	Universal Serial Bus
USD	US dollars [United States]
V&V	Verification and validation
VA	Value analysis
VE	Value engineering
VM	Value management
VMP	Value Methodology Practitioner [SAVE]
VSE	Very small entities
VSME	Very small and micro enterprises
VV&A	Verification, validation, and accreditation
WBS	Work breakdown structure
WG	Working group

APPENDIX C: TERMS AND DEFINITIONS

Words not included in this glossary carry meanings consistent with general dictionary definitions. Other related terms can be found in SE VOCAB (2013).

Acquirer	The stakeholder that acquires or procures a product or service from a supplier
"*-ilities*"	The developmental, operational, and support requirements a program must address (named because they typically end in "ility"—availability, maintainability, vulnerability, reliability, supportability, etc.)
Acquisition logistics	Technical and management activities conducted to ensure supportability implications are considered early and throughout the acquisition process to minimize support costs and to provide the user with the resources to sustain the system in the field
Activity	A set of cohesive tasks of a process
Agile	Project execution methods can be described on a continuum from "adaptive" to "predictive." Agile methods exist on the "adaptive" side of this continuum, which is not the same as saying that agile methods are "unplanned" or "undisciplined"
Agreement	The mutual acknowledgment of terms and conditions under which a working relationship is conducted
Architecture	(System) fundamental concepts or properties of a system in its environment embodied in its elements, relationships, and in the principles of its design and evolution (see ISO 42010)

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.

Baseline
The gate-controlled step-by-step elaboration of business, budget, functional, performance, and physical characteristics, mutually agreed to by buyer and seller, and under formal change control. Baselines can be modified between formal decision gates by mutual consent through the change control process.
An agreed-to description of the attributes of a product at a point in time, which serves as a basis for defining change (ANSI/EIA-649-1998)

Black box/ white box
Black box represents an external view of the system (attributes). White box represents an internal view of the system (attributes and structure of the elements)

Capability
An expression of a system, product, function, or process ability to achieve a specific objective under stated conditions

Commercial off-the-shelf (COTS)
Commercial items that require no unique acquirer modifications or maintenance over the life cycle of the product to meet the needs of the procuring agency

Commonality
(Of a product line) refers to functional and nonfunctional characteristics that can be shared with all member products within a product line (ISO 26550 2nd CD)

Configuration
A characteristic of a system element, or project artifact, describing their maturity or performance

Configuration item (CI)
A hardware, software, or composite item at any level in the system hierarchy designated for configuration management. (The system and each of its elements are individual CIs.) CIs have four common characteristics:
1. Defined functionality
2. Replaceable as an entity
3. Unique specification
4. Formal control of form, fit, and function

Decision gate
A decision gate is an approval event (often associated with a review meeting). Entry and exit criteria are established for each decision gate; continuation beyond the decision gate is contingent on the agreement of decision makers

Derived requirements
Detailed characteristics of the system of interest (SOI) that typically are identified during elicitation of stakeholder requirements, requirements analysis, trade studies, or validation

Design constraints
The boundary conditions, externally or internally imposed, for the SOI within which the organization must remain when executing the processes during the concept and development stages

Domain asset
Is the output of a subprocess of domain engineering that is reused for producing two or more products in a product line. A domain asset may be a variability model, an architectural design, a software component, a domain model, a requirements statement or specification, a plan, a test case, a process description, or any other element useful for producing products and services. Syn: domain artifact (ISO 26550 2nd CD)

Note: In systems engineering, domain assets may be subsystems or components to be reused in further system designs. Domain assets are considered through their original requirements and technical characteristics. Domain assets include, but are not limited to, use cases, logical principles, environmental behavioral data, and risks or opportunities learned from the previous projects

Domain assets are not physical products available off-the-shelf and ready for commissioning. Physical products (e.g., mechanical parts, electronic components, harnesses, optic lenses) are stored and managed according to the best practices of their respective disciplines

Note: In software engineering, domain assets can include source or object code to be reused during the implementation

Note: Domain assets have their own life cycles. ISO/IEC/IEEE 15288 may be used to manage a life cycle

Domain scoping — Identifies and bounds the functional domains that are important to an envisioned product line and provide sufficient reuse potential to justify the product line creation. Domain scoping builds on the definitions of the product scoping (ISO 26550 2nd CD)

Element — See system element

Enabling system — A system that supports a SOI during its life cycle stages but does not necessarily contribute directly to its function during operation

Enterprise — A purposeful combination of interdependent resources that interact with each other to achieve buisness and operational goals (Rebovich and White, 2011)

Environment — The surroundings (natural or man-made) in which the SOI is utilized and supported or in which the system is being developed, produced, and retired

Facility — The physical means or equipment for facilitating the performance of an action, for example, buildings, instruments, and tools

Failure — The event in which any part of an item does not perform as required by its specification. The failure may occur at a value in excess of the minimum required in the specification, that is, past design limits or beyond the margin of safety

Functional configuration audit — An evaluation to ensure that the product meets baseline functional and performance capabilities (adapted from ISO/IEC/IEEE 15288)

Human factors — The systematic application of relevant information about human abilities, characteristics, behavior, motivation, and performance. It includes principles and applications in the areas of human-related engineering, anthropometrics, ergonomics, job performance skills and aids, and human performance evaluation

Human systems integration — The interdisciplinary technical and management processes for integrating human considerations within and across all system elements; an essential enabler to SE practice

Interface — A shared boundary between two functional units, defined by functional characteristics, common physical interconnection characteristics, signal characteristics, or other characteristics, as appropriate (ISO 2382-1)

Integration definition for functional modeling (IDEF) — A family of modeling languages in the fields of systems and software engineering that provide a multiple-page (view) model of a system that depicts functions and information or product flow. Boxes illustrate functions and arrows illustrate information and product flow (KBS, 2010). Alphanumeric coding is used to denote the view:

- IDEF0—functional modeling method
- IDEF1—information modeling method
- IDEF1X—data modeling method
- IDEF3—process description capture method
- IDEF4—object-oriented design method
- IDEF5—ontology description capture method

IPO diagram — Figures in this handbook that provide a high-level view of the process of interest. The diagram summarizes the process activities and their inputs and outputs from/to external actors; some inputs are categorized as controls and enablers. A control governs the accomplishments of the process; an enabler is the means by which the process is performed

Life cycle cost (LCC) — The total cost of acquisition and ownership of a system over its entire life. It includes all costs associated with the system and its use in the concept, development, production, utilization, support, and retirement stages

Life cycle model — A framework of processes and activities concerned with the life cycle, which also acts as a common reference for communication and understanding

Measures of effectiveness — Measures that define the information needs of the decision makers with respect to system effectiveness to meet operational expectations

Measures of performance — Measures that define the key performance characteristics the system should have when fielded and operated in its intended operating environment

N^2 diagrams — Graphical representation used to define the internal operational relationships or external interfaces of the SOI

Operator — An individual who, or an organization that, contributes to the functionality of a system and draws on knowledge, skills, and procedures to contribute the function

Organization — Person or a group of people and facilities with an arrangement of responsibilities, authorities, and relationships (adapted from ISO 9001:2008)

Performance — A quantitative measure characterizing a physical or functional attribute relating to the execution of a process, function, activity, or task; performance attributes include quantity (how many or how much), quality (how well), timeliness (how responsive, how frequent), and readiness (when, under which circumstances)

Physical configuration audit — An evaluation to ensure that the operational system or product conforms to the operational and configuration documentation (adapted from ISO/IEC/IEEE 15288)

Process — A set of interrelated or interacting activities that transforms inputs into outputs (adapted from ISO 9001:2008)

Product line —
1. Group of products or services sharing a common, managed set of features that satisfy specific needs of a selected market or mission. ISO/IEC/IEEE 24765 (2010), Systems and software engineering vocabulary
2. A collection of systems that are potentially derivable from a single-domain architecture. IEEE 1517-1999 (R2004) IEEE standard for information technology—Software life cycle processes—Reuse processes (3.14) (ISO/IEC FCD 24765.5)

Product line scoping — Defines the products that will constitute the product line and the major (externally visible) common and variable features among the products, analyzes the products from an economic point of view, and controls and schedules the development, production, and marketing of the product line and its products. Product management is primarily responsible for this process (ISO 26550 2nd CD)

Project — An endeavor with defined start and finish criteria undertaken to create a product or service in accordance with specified resources and requirements

Proof of concept — A naïve realization of an idea or technology to demonstrate its feasibility

Prototype — A production-ready demonstration model developed under engineering supervision that is specification compliant and represents what manufacturing should replicate

Qualification limit — Proving that the design will survive in its intended environment with margin. The process includes testing and analyzing hardware and software configuration items to prove that the design will survive the anticipated accumulation of acceptance test environments, plus its expected handling, storage, and operational environments, plus a specified qualification margin

Requirement — A statement that identifies a system, product, or process characteristic or constraint, which is unambiguous, clear, unique, consistent, stand-alone (not grouped), and verifiable, and is deemed necessary for stakeholder acceptability

Resource — An asset that is utilized or consumed during the execution of a process

Return on investment — Ratio of revenue from output (product or service) to development and production costs, which determines whether an organization benefits from performing an action to produce something (ISO/IEC 24765.5 FCD; ISO/IEC/IEEE 24765, 2010)

Reuse	1. The use of an asset in the solution of different problems. [IEEE 1517-1999 (R2004)] 2. Building a software system at least partly from existing pieces to perform a new application. [ISO/IEC/IEEE 24765 (2010)]
Specialty engineering	Analysis of specific features of a system that requires special skills to identify requirements and assess their impact on the system life cycle
Stage	A period within the life cycle of an entity that relates to the state of its description or realization
Stakeholder	A party having a right, share, or claim in a system or in its possession of characteristics that meet that party's needs and expectations
Supplier	An organization or an individual that enters into an agreement with the acquirer for the supply of a product or service
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) A combination of interacting elements organized to achieve one or more stated purposes (ISO/IEC/IEEE 15288)
System element	Member of a set of elements that constitutes a system
System life cycle	The evolution with time of a SOI from conception to retirement
System of interest	The system whose life cycle is under consideration
System of systems	A SOI whose system elements are themselves systems; typically, these entail large-scale interdisciplinary problems with multiple, heterogeneous, distributed systems
Systems engineering	Systems engineering (SE) 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. SE 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)
Systems engineering effort	Systems engineering effort integrates multiple disciplines and specialty groups into a set of activities that proceed from concept to production and to operation. SE considers both the business and the technical needs of all stakeholders with the goal of providing a quality system that meets their needs
Systems engineering management plan (SEMP)	Structured information describing how the systems engineering effort, in the form of tailored processes and activities, for one or more life cycle stages, will be managed and conducted in the organization for the actual project
Tailoring	The manner in which any selected issue is addressed in a particular project. Tailoring may be applied to various aspects of the project, including project documentation, processes and activities performed in each life cycle stage, the time and scope of reviews, analysis, and decision making consistent with all applicable statutory requirements
Technical performance measures	Measures that define attributes of a system element to determine how well a system or system element is satisfying or expected to satisfy a technical requirement or goal

Trade-off — Decision-making actions that select from various requirements and alternative solutions on the basis of net benefit to the stakeholders

User — Individual who or group that benefits from a system during its utilization

Validation — Confirmation, through the provision of objective evidence, that the requirements for a specific intended use or application have been fulfilled (ISO/IEC/IEEE 15288)

Note: Validation is the set of activities ensuring and gaining confidence that a system is able to accomplish its intended use, goals, and objectives (i.e., meet stakeholder requirements) in the intended operational environment

Value — A measure of worth (e.g., benefit divided by cost) of a specific product or service by a customer, and potentially other stakeholders and is a function of (i) the product's usefulness in satisfying a customer need, (ii) the relative importance of the need being satisfied, (iii) the availability of the product relative to when it is needed, and (iv) the cost of ownership to the customer (McManus, 2004)

Variability — Of a product line refers to characteristics that may differ among members of the product line (ISO 26550 2nd CD)

Variability constraints — Denotes constraint relationships between a variant and a variation point, between two variants, and between two variation points (ISO 26550 2nd CD)

Verification — Confirmation, through the provision of objective evidence, that specified requirements have been fulfilled (ISO/IEC/IEEE 15288)

Note: Verification is a set of activities that compares a system or system element against the required characteristics. This may include, but is not limited to, specified requirements, design description, and the system itself

Waste — Work that adds no value to the product or service in the eyes of the customer (Womack and Jones, 1996)

APPENDIX D: N^2 DIAGRAM OF SYSTEMS ENGINEERING PROCESSES

Figure D.1 illustrates the input/output relationships between the various SE processes presented in the handbook and shows the interactions depicted on the IPO diagrams throughout this handbook. The primary flows represent a typical system development program.

The individual processes are placed on the diagonal by abbreviation to the process names, as follows:

EXT	External inputs and outputs
BMA	Business or mission analysis
SNRD	Stakeholder needs and requirements definition
SRD	System requirements definition
AD	Architecture definition
DD	Design definition
SA	System analysis
IMPL	Implementation
INT	Integration
VER	Verification
TRAN	Transition
VAL	Validation
OPER	Operation
MAINT	Maintenance
DISP	Disposal
PP	Project planning
PAC	Project assessment and control
DM	Decision management
RM	Risk management
CM	Configuration management
INFOM	Information management
MEAS	Measurement
QA	Quality assurance
ACQ	Acquisition
SUP	Supply
LCMM	Life cycle model management
INFRAM	Infrastructure management
PM	Portfolio management
HRM	Human resource management
QM	Quality management
KM	Knowledge management
TLR	Tailoring

The off-diagonal squares represent the inputs/outputs interface shared by the processes that intersect at a given square. Outputs flow horizontally; inputs flow vertically and can be read in a clockwise fashion.

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.

Note 1: The absence of an *x* in an intersection does not preclude tailoring to create a relationship between any two processes.

Note 2: This is the result of one possible instance of the life cycle processes. Other instances of the process relationships are possible.

EXT	X	X													X								X	X	X	X	X	X	X	X	X
	BMA	X				X									X		X	X	X	X	X										X	
	X	SNRD	X	X	X	X	X	X	X	X	X	X	X	X	X		X	X	X	X	X										X	
			SRD	X	X	X			X						X		X	X	X	X	X										X	
				AD	X	X	X								X		X	X	X	X	X										X	
					DD	X	X	X	X						X		X	X	X	X	X										X	
						SA									X	X	X	X	X	X											X	
	X	X	X	X	X	X	IMPL	X		X		X	X		X	X	X	X	X	X			X								X	
	X	X	X	X	X	X		INT	X						X	X	X	X	X	X			X								X	
	X	X	X	X	X	X			VER	X	X				X	X	X	X	X	X			X								X	
	X	X	X	X	X	X				TRAN	X	X	X		X	X	X	X	X	X			X								X	
	X	X	X	X	X	X					VAL	X	X	X	X	X	X	X	X	X			X	X							X	
	X	X	X	X	X	X						OPER	X	X	X	X	X	X	X	X			X								X	
	X	X	X	X	X	X						X	MAINT	X	X	X	X	X	X	X			X								X	
	X	X	X	X	X	X								DISP	X	X	X	X	X	X			X	X							X	
	X	X													PP	X	X	X	X	X			X			X		X			X	
															X	PAC	X	X	X	X	X						X				X	
															X	X	DM	X	X	X											X	
															X	X	X	RM	X	X											X	
															X	X	X	X	CM	X											X	
															X	X	X	X	X	INFOM											X	
															X	X	X	X	X	X	MEAS										X	
															X	X	X	X	X	X		QA								X	X	
X								X							X	X	X	X	X	X			ACQ								X	
X															X	X	X	X	X	X				SUP			X				X	
X															X						X				LCMM					X	X
X																										INFRAM				X	
X															X											X	PM	X		X	
X															X													HRM		X	
X															X	X						X			X				QM	X	
X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	KM	
															X										X						TLR

FIGURE D.1 Input/output relationships between the various SE processes. INCOSE SEH original figure created by David Walden. Usage per the INCOSE Notices page. All other rights reserved.

APPENDIX E: INPUT/OUTPUT DESCRIPTIONS

Combined list of all inputs and outputs defined in the processes described in Chapters 4–8.

Accepted system or system element	System element or system is transferred from supplier to acquirer and the product or service is available to the project
Acquired system	The system or system element (product or service) is delivered to the acquirer from a supplier consistent with the delivery conditions of the acquisition agreement
Acquisition agreement	An understanding of the relationship and commitments between the project organization and the supplier. The agreement can vary from formal contracts to less formal interorganizational work orders. Formal agreements typically include terms and conditions
Acquisition need	The identification of a need that cannot be met within the organization encountering the need or a need that can be met in a more economical way by a supplier
Acquisition payment	Payments or other compensations for the acquired system. Includes remitting and acknowledgement
Acquisition record	Permanent, readable form of data, information, or knowledge related to acquisition
Acquisition reply	The responses of one or more candidate suppliers in response to a request for supply
Acquisition report	An account prepared for interested parties in order to communicate the status, results, and outcomes of the acquisition activities

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.

Acquisition strategy	Approaches, schedules, resources, and specific considerations required to acquire system elements. May also include inputs to determine acquisition constraints
Agreements	Agreements from all applicable life cycle processes, including acquisition agreements and supply agreements
Alternative solution classes	Identifies and describes the classes of solutions that may address the problem or opportunity
Analysis situations	The context information for the analysis including life cycle stage, evaluation drivers, cost drivers, size drivers, team characteristics, project priorities, or other characterization information and parameters that are needed to understand analysis and represent the element being analyzed. Relevant information from the process that invokes the analysis. Any existing models related to the element being analyzed. Any data related to the element being analyzed, including historical, current, and projected data. Can originate from any life cycle process
Applicable laws and regulations	International, national, or local laws or regulations
Architecture definition record	Permanent, readable form of data, information, or knowledge related to architecture definition
Architecture definition strategy	Approaches, schedules, resources, and specific considerations required to define the selected system architecture that satisfies the requirements
Architecture traceability	Bidirectional traceability of the architecture characteristics
Business or mission analysis record	Permanent, readable form of data, information, or knowledge related to business or mission analysis
Business or mission analysis strategy	Approaches, schedules, resources, and specific considerations required to conduct business or mission analysis and ensure business needs are elaborated and formalized into business requirements
Business requirements	Definition of the business framework within which stakeholders will define their requirements. Business requirements govern the project, including agreement constraints, quality standards, and cost and schedule constraints. Business requirements may be captured in a Business Requirements Specification (BRS), which is approved by the business leadership *Note: Business requirements may not always be formally captured in the system life cycle*
Business requirements traceability	Bidirectional traceability of the business requirements
Candidate configuration items (CIs)	Items for configuration control. Can originate from any life cycle process
Candidate information items	Items for information control. Can originate from any life cycle process
Candidate risks and opportunities	Risks and opportunities that arise from any stakeholder. In many cases, risk situations are identified during the project assessment and control process. Can originate from any life cycle process
Concept of operations (ConOps)	The ConOps is a verbal and/or graphic statement prepared for the organization's leadership that describes the assumptions or intent regarding the overall operation or series of operations of the enterprise, to include any new capability (ANSI/AIAA, 2012; ISO/IEC/IEEE 29148, 2011)
Configuration baselines	Items placed under formal change control. The required configuration baseline documentation is developed and approved in a timely manner to support required systems engineering (SE) technical reviews, the system's acquisition and support strategies, and production
Configuration management record	Permanent, readable form of data, information, or knowledge related to configuration management

Configuration management report	An account prepared for interested parties in order to communicate the status, results, and outcomes of the configuration management activities. Documents the impact to any process, organization, decision (including any required change notification), products, and services affected by a given change request
Configuration management strategy	Approaches, schedules, resources, and specific considerations required to perform configuration management for a project. Describes and documents how to make authorized changes to established baselines in a uniform and controlled manner
Customer satisfaction inputs	Responses to customer satisfaction surveys or other instruments
Decision management strategy	Approaches, schedules, resources, and specific considerations required to perform decision management for a project
Decision record	Permanent, readable form of data, information, or knowledge related to decision management
Decision report	An account prepared for interested parties in order to communicate the status, results, and outcomes of the decision management activities. Should include a recommended course of action, an associated implementation plan, and key findings through effective trade space visualizations underpinned by defendable rationale grounded in analysis results that are repeatable and traceable. As decision makers seek to understand root causes of top-level observations and build their own understanding of the trade-offs, the ability to rapidly drill down from top-level trade space visualizations into lower-level analyses supporting the synthesized view is often beneficial
Decision situation	Decisions related to decision gates are taken on a prearranged schedule; other requests for a decision may arise from any stakeholder, and initial information can be little more than broad statements of the situation. Can originate from any life cycle process
Design definition record	Permanent, readable form of data, information, or knowledge related to design definition
Design definition strategy	Approaches, schedules, resources, and specific considerations required to define the system design that is consistent with the selected system architecture and satisfies the requirements
Design traceability	Bidirectional traceability of the design characteristics, the design enablers, and the system element requirements
Disposal constraints	Any constraints on the system arising from the disposal strategy including cost, schedule, and technical constraints
Disposal enabling system requirements	Requirements for any systems needed to enable disposal of the system of interest
Disposal procedure	A disposal procedure that includes a set of disposal actions, using specific disposal techniques, performed with specific disposal enablers
Disposal record	Permanent, readable form of data, information, or knowledge related to disposal
Disposal report	An account prepared for interested parties in order to communicate the status, results, and outcomes of the disposal activities. May include an inventory of system elements for reuse/storage and any documentation or reporting required by regulation or organization standards
Disposal strategy	Approaches, schedules, resources, and specific considerations required to ensure the system or system elements are deactivated, disassembled, and removed from operations

Disposed system	Disposed system that has been deactivated, disassembled, and removed from operations
Documentation tree	Defines the hierarchical representation of the set of system definition products for the system under development. Based on the evolving system architecture
Enabling system requirements	Enabling system requirements from all applicable life cycle processes, including implementation enabling system requirements, integration enabling system requirements, verification enabling system requirements, transition enabling system requirements, validation enabling system requirements, operation enabling system requirements, maintenance enabling system requirements, and disposal enabling system requirements
Final Requirements Verification and Traceability Matrix (RVTM)	Final list of requirements, their verification attributes, and their traces. Includes any proposed changes to the system requirements due to the verification actions
Human resource management plan	Approaches, schedules, resources, and specific considerations required to identify the skill needs of the organization and projects. Includes the organizational training plan needed to develop internal personnel and the acquisition of external personnel
Human resource management record	Permanent, readable form of data, information, or knowledge related to human resource management
Human resource management report	An account prepared for interested parties in order to communicate the status, results, and outcomes of the human resource management activities
Implementation constraints	Any constraints on the system arising from the implementation strategy including cost, schedule, and technical constraints
Implementation enabling system requirements	Requirements for any systems needed to enable implementation of the system of interest
Implementation record	Permanent, readable form of data, information, or knowledge related to implementation
Implementation report	An account prepared for interested parties in order to communicate the status, results, and outcomes of the implementation activities
Implementation strategy	Approaches, schedules, resources, and specific considerations required to realize system elements to satisfy system requirements, architecture, and design
Implementation traceability	Bidirectional traceability of the system elements
Information management record	Permanent, readable form of data, information, or knowledge related to information management
Information management report	An account prepared for interested parties in order to communicate the status, results, and outcomes of the information management activities
Information management strategy	Approaches, schedules, resources, and specific considerations required to perform information management for a project
Information repository	A repository that supports the availability for use and communication of all relevant project information artifacts in a timely, complete, valid, and, if required, restricted manner
Infrastructure management plan	Approaches, schedules, resources, and specific considerations required to define and sustain the organizational and project infrastructures
Infrastructure management record	Permanent, readable form of data, information, or knowledge related to infrastructure management

Infrastructure management report	An account prepared for interested parties in order to communicate the status, results, and outcomes of the infrastructure management activities. Includes cost, usage, downtime/response measures, etc. These can be used to support capacity planning for upcoming projects
Initial RVTM	A preliminary list of requirements, their verification attributes, and their traces
Installation procedure	An installation procedure that includes a set of installation actions, using specific installation techniques, performed with specific transition enablers
Installed system	Installed system ready for validation
Integrated system or system element	Integrated system element or system ready for verification. The resulting aggregation of assembled system elements
Integration constraints	Any constraint on the system arising from the integration strategy including cost, schedule, and technical constraints
Integration enabling system requirements	Requirements for any systems needed to enable integration of the system of interest
Integration procedure	An assembly procedure that groups a set of elementary assembly actions to build an aggregate of implemented system elements, using specific integration techniques, performed with specific integration enablers
Integration record	Permanent, readable form of data, information, or knowledge related to integration
Integration report	An account prepared for interested parties in order to communicate the status, results, and outcomes of the integration activities. Includes documentation of the integration testing and analysis results, areas of nonconformance, and validated internal interfaces
Integration strategy	Approaches, schedules, resources, and specific considerations required to integrate the system elements
Interface definition	The logical and physical aspects of internal interfaces (between the system elements composing the system) and external interfaces (between the system elements and the elements outside the system of interest)
Interface definition update identification	Identification of updates to interface requirements and definitions, if any
Knowledge management plan	Establishes how the organization and projects within the organization will interact to ensure the right level of knowledge is captured to provide useful knowledge assets. Includes a list of applicable domains; plans for obtaining and maintaining knowledge assets for their useful life; characterization of the types of assets to be collected and maintained along with a scheme to classify them for the convenience of users; criteria for accepting, qualifying, and retiring knowledge assets; procedures for controlling changes to the knowledge assets; and definition of a mechanism for knowledge asset storage and retrieval
Knowledge management report	An account prepared for interested parties in order to communicate the status, results, and outcomes of the knowledge management activities
Knowledge management system	Maintained knowledge management system. Project suitability assessment results for application of existing knowledge. Lessons learned from execution of the organizational SE processes on projects. Should include mechanisms to easily identify and access the assets and to determine the level of applicability for the project considering its use. Can be used by any life cycle process

Life cycle concepts	Articulation and refinement of the various life cycle concepts consistent with the business needs in the form of life cycle concept documents on which the system of interest is based, assessed, and selected. The architecture is based on these concepts, and they are essential in providing context for proper interpretation of the system requirements. Typical concepts include: • Acquisition concept • Deployment concept • Operational concept (OpsCon) • Support concept • Retirement concept
Life cycle constraints	Constraints from all applicable life cycle processes, including implementation constraints, integration constraints, verification constraints, transition constraints, validation constraints, operation constraints, maintenance constraints, and disposal constraints
Life cycle model management plan	Approaches, schedules, resources, and specific considerations required to define a set of organizational life cycle models. Includes identification of new needs and the evaluation of competitiveness from the perspective of the organization strategy. Includes criteria for assessments and approvals/disapprovals
Life cycle model management record	Permanent, readable form of data, information, or knowledge related to life cycle model management
Life cycle model management report	An account prepared for interested parties in order to communicate the status, results, and outcomes of the life cycle model management activities
Life cycle models	Life cycle model or models appropriate for the project. Includes definition of the business and other decision-making criteria regarding entering and exiting each life cycle stage. The information and artifacts are collected and made available to be used and reused
Maintenance constraints	Any constraints on the system arising from the maintenance strategy including cost, schedule, and technical constraints
Maintenance enabling system requirements	Requirements for any systems needed to enable operation of the system of interest
Maintenance procedure	A maintenance procedure that includes a set of maintenance actions, using specific maintenance techniques, performed with specific maintenance enablers
Maintenance record	Permanent, readable form of data, information, or knowledge related to maintenance
Maintenance report	An account prepared for interested parties in order to communicate the status, results, and outcomes of the maintenance activities
Maintenance strategy	Approaches, schedules, resources, and specific considerations required to perform corrective and preventive maintenance in conformance with operational availability requirements
Major stakeholder identification	List of legitimate external and internal stakeholders with an interest in the solution. Major stakeholders are also derived from analysis of the ConOps
Measurement data	Measurement data from all applicable life cycle processes, including measure of effectiveness (MOE) data, measure of performance (MOP) data, technical performance measures (TPM) data, project performance measures data, and organizational process performance measures data
Measurement needs	Measurement needs from all applicable life cycle processes, including MOE needs, MOP needs, TPM needs, project performance measures needs, and organizational process performance measures needs
Measurement record	Permanent, readable form of data, information, or knowledge related to measurement

Measurement report	An account prepared for interested parties in order to communicate the status, results, and outcomes of the measurement activities. Includes documentation of the measurement activity results, the measurement data that was collected and analyzed and results that were communicated, and any improvements or corrective actions driven by the measures with their supporting data
Measurement repository	A repository that supports the availability for use and communication of all relevant measures in a timely, complete, valid, and, if required, confidential manner
Measurement strategy	Approaches, schedules, resources, and specific considerations required to perform measurement for a project. Addresses the strategy for performing measurement: describing measurement goals, identifying information needs and applicable measures, and defining performance and evaluation methodologies
MOE data	Data provided for the identified measurement needs
MOE needs	Identification of the MOEs (Roedler and Jones, 2006), which define the information needs of the decision makers with respect to system effectiveness to meet operational expectations
MOP data	Data provided for the identified measurement needs
MOP needs	Identification of the MOPs (Roedler and Jones, 2006), which define the key performance characteristics the system should have when fielded and operated in its intended operating environment
Operation constraints	Any constraints on the system arising from the operational strategy including cost, schedule, and technical constraints
Operation enabling system requirements	Requirements for any systems needed to enable operation of the system of interest
Operation record	Permanent, readable form of data, information, or knowledge related to operation
Operation report	An account prepared for interested parties in order to communicate the status, results, and outcomes of the operation activities
Operation strategy	Approaches, schedules, resources, and specific considerations required to perform system operations
Operator/ maintainer training materials	Training capabilities and documentation
Organization infrastructure	Resources and services that support the organization. Organizational-level facilities, personnel, and resources for hardware fabrication, software development, system implementation and integration, verification, validation, etc.
Organization infrastructure needs	Specific requests for infrastructure products or services from the organization, including commitments to external stakeholders
Organization lessons learned	Organizational-related lessons learned. Results from an evaluation or observation of an implemented corrective action that contributed to improved performance or increased capability. A lesson learned also results from an evaluation or observation of a positive finding that did not necessarily require corrective action other than sustainment
Organization portfolio direction and constraints	Organization business objectives, funding outlay and constraints, ongoing research and development (R&D), market tendencies, etc., including cost, schedule, and solution constraints

Organization strategic plan	The overall organization strategy, including the business mission or vision and strategic goals and objectives
Organization tailoring strategy	Approaches, schedules, resources, and specific considerations required to incorporate new or updated external standards into the organization's set of standard life cycle processes
Organizational policies, procedures, and assets	Items related to the organization's standard set of life cycle processes, including guidelines and reporting mechanisms. Organization process guidelines in the form of organization policies, procedures, and assets for applying the system life cycle processes and adapting them to meet the needs of individual projects (e.g., templates, checklists, forms). Includes defining responsibilities, accountability, and authority for all SE processes within the organization
Organizational process performance measures data	Data provided for the identified measurement needs
Organizational process performance measures needs	Identification of the organizational process performance measures, which measure how well the organization is satisfying it objectives
Portfolio management plan	Approaches, schedules, resources, and specific considerations required to define a project portfolio
Portfolio management record	Permanent, readable form of data, information, or knowledge related to portfolio management
Portfolio management report	An account prepared for interested parties in order to communicate the status, results, and outcomes of the portfolio management activities
Preliminary interface definition	The preliminary logical and physical aspects of internal interfaces (between the system elements composing the system) and external interfaces (between the system elements of the system and the elements outside the system of interest)
Preliminary life cycle concepts	Preliminary articulation of the various life cycle concepts consistent with the business needs in the form of life cycle concept documents on which the system of interest is based, assessed, and selected. The architecture is based on these concepts, and they are essential in providing context for proper interpretation of the system requirements. Typical concepts include: • Acquisition concept • Deployment concept • OpsCon • Support concept • Retirement concept
Preliminary MOE data	Preliminary data provided for the identified measurement needs
Preliminary MOE needs	Preliminary identification of the MOEs (Roedler and Jones, 2006), which define the information needs of the decision makers with respect to system effectiveness to meet operational expectations
Preliminary TPM data	Preliminary data provided for the identified measurement needs
Preliminary TPM needs	Preliminary identification of the TPM (Roedler and Jones, 2006), which measure attributes of a system element to determine how well a system or system element is satisfying or expected to satisfy a technical requirement or goal
Preliminary validation criteria	The preliminary validation criteria (the measures to be assessed), who will perform validation activities, and the validation environments of the system of interest
Problem or opportunity statement	Description of the problem or opportunity. Should be derived from the organization strategy and provide enough detail to understand the gap or new capability that is being considered
Procedures	Procedures from all applicable life cycle processes, including integration procedure, verification procedure, installation procedure, validation procedure, maintenance procedure, and disposal procedure

Project assessment and control record	Permanent, readable form of data, information, or knowledge related to project assessment and control
Project assessment and control strategy	Approaches, schedules, resources, and specific considerations required to perform assessment and control for a project
Project budget	A prediction of the costs associated with a particular project. Includes labor, infrastructure, acquisition, and enabling system costs along with reserves for risk management
Project change requests	Requests to update any formal baselines that have been established. In many cases, the need for change requests is identified during the project assessment and control process. Can originate from any life cycle process
Project constraints	Any constraints on the system arising from the technical management strategy including cost, schedule, and technical constraints
Project control requests	Internal project directives based on action required due to deviations from the project plan. New directions are communicated to both project team and customer, when appropriate. If assessments are associated with a decision gate, a decision to proceed, or not to proceed, is taken
Project direction	Organizational direction to the project. Includes sustainment of projects meeting assessment criteria and redirection or termination of projects not meeting assessment criteria
Project human resource needs	Specific requests for human resources needed by the project, including commitments to external stakeholders
Project infrastructure	Resources and services that support a project. Project-level facilities, personnel, and resources for hardware fabrication, software development, system implementation and integration, verification, validation, etc.
Project infrastructure needs	Specific requests for infrastructure products or services needed by the project, including commitments to external stakeholders
Project lessons learned	Project-related lessons learned. Results from an evaluation or observation of an implemented corrective action that contributed to improved performance or increased capability. A lesson learned also results from an evaluation or observation of a positive finding that did not necessarily require corrective action other than sustainment (CJCS, 2012)
Project performance measures data	Data provided for the identified measurement needs
Project performance measures needs	Identification of the project performance measures, which measure how well the project is satisfying it objectives
Project planning record	Permanent, readable form of data, information, or knowledge related to project planning
Project portfolio	The necessary information for all of the organizations' projects. The initiation of new projects or the setting up of a product line management approach. Includes the project goals, resources, budgets identified and allocated to the projects, and clearly defined project management accountability and authorities
Project schedule	A linked list of a project's milestones, activities, and deliverables with intended start and finish dates. May include a top-level milestone schedule and multiple levels (also called tiers) of schedules of increasing detail and task descriptions with completion criteria and work authorizations
Project status report	An account prepared for interested parties in order to communicate the status, results, and outcomes of the overall project activities. Includes status on meeting the objectives set out for the project, information on the health and maturity of the project work effort, status on project tailoring and execution, and status on personnel availability and effectiveness for the project

Project tailoring strategy	Approaches, schedules, resources, and specific considerations required to incorporate and tailor the organization's set of standard life cycle processes for a given project
Quality management (QM) corrective actions	Actions taken when quality goals are not achieved. Resulting from project-related and process-related reviews and audits
Qualified personnel	The right people with the right skills are assigned at the right time to projects per their skill needs and timing
Quality assurance evaluation report	An account prepared for interested parties in order to communicate evidence of whether the project's quality assurance activities are effective. Includes the assessment of all the project-related process and any suggested improvements or necessary corrective actions. Provides constructive input for improvements to an organization's life cycle model implementation
Quality assurance plan	The set of project quality assurance activities, tailored to the project, designed to monitor development and SE processes. Describes the quality assurance organization and applicable audit, evaluation, and monitoring activities. This includes the set of policies and procedures, including specific methods and techniques that apply to quality assurance practices within the organization and within individual projects. It also includes quality objectives for processes and systems that are measurable, along with linkages to the assigned accountability and authority for QM within the organization. The plan also references activities performed by other organizations or functions that are monitored or audited by the quality assurance organization
Quality assurance record	Permanent, readable form of data, information, or knowledge related to quality assurance
Quality assurance report	An account prepared for interested parties in order to communicate the status, results, and outcomes of the quality assurance activities. Includes information on deviations from nominal conditions during the product life cycle and actions to be taken when quality assurance goals and objectives are not achieved
Quality management evaluation report	An account prepared for interested parties in order to communicate evidence of whether the organization's QM activities are effective. Includes the assessment of all the organizational-related process and any suggested improvements or necessary corrective actions. Provides constructive input for improvements to an organization's life cycle model implementation
Quality management guidelines	Guidelines for quality practices within the organization, within individual projects, and as part of the execution of system life cycle processes
Quality management plan	The overarching guidance that explains the organization's quality philosophy and quality organization. Describes the QM organization and applicable audit, evaluation, and monitoring activities. This includes the set of policies and procedures, including specific methods and techniques that apply to QM practices within the organization. It also includes quality objectives for processes and systems that are measurable, along with the assigned accountability and authority for QM within the organization. The set of project QM activities form the basis of the project quality assurance
Quality management record	Permanent, readable form of data, information, or knowledge related to QM
Quality management report	An account prepared for interested parties in order to communicate the status, results, and outcomes of the QM activities. Includes the results of any customer satisfaction surveys and any issues that need to be addressed

Records	Records from all applicable life cycle processes, including business or mission analysis record, stakeholder needs and requirements definition record, system requirements definition record, architecture definition record, design definition record, system analysis record, implementation record, integration record, verification record, transition record, validation record, operation record, maintenance record, disposal record, project planning record, project assessment and control record, decision record, risk record, configuration management record, information management record, measurement record, quality assurance record, acquisition record, supply record, life cycle model management record, infrastructure management record, portfolio management record, human resource management record, and QM record
Reports	Project reports from all applicable life cycle processes, including system analysis report, implementation report, integration report, verification report, transition report, validation report, operation report, maintenance report, disposal report, decision report, risk report, configuration management report, information management report, measurement report, quality assurance report, acquisition report, and supply report (other reports go to other process areas and are not aggregated here)
Request for supply	A request to an external supplying organization to propose a solution to meet a need for a system element or system (product or service). The organization can identify candidate suppliers that could meet this need. Inputs are received from the project personnel in the organization with the need
Risk management strategy	Approaches, schedules, resources, and specific considerations required to perform risk management for a project
Risk record	Permanent, readable form of data, information, or knowledge related to risk management
Risk report	An account prepared for interested parties in order to communicate the status, results, and outcomes of the risk management activities. The risks are documented and communicated along with rationale, assumptions, treatment plans, and current status. For selected risks, an action plan is produced to direct the project team to update the project plan and properly respond to the risks. If appropriate, change requests are generated to mitigate technical risk. Risk profiles and/or risk matrices summarize the risks and contain the findings of the risk management process
SEMP	*Systems engineering management plan.* The top-level plan for managing the SE effort. It defines how the project will be organized, structured, and conducted and how the total engineering process will be controlled to provide a product that satisfies stakeholder requirements. Includes identification of required technical reviews and their completion criteria, methods for controlling changes, risk and opportunity assessment and methodology, and identification of other technical plans and documentation to be produced for the project
Source documents	External documents relevant to the particular stage of procurement activity for the system of interest. Includes the written directives embodied in the source documents relevant to organizational strategies and policies
Stakeholder needs	Needs determined from communication with external and internal stakeholders in understanding their expectations, needs, requirements, values, problems, issues, and perceived risks and opportunities

Stakeholder needs and requirements definition record	Permanent, readable form of data, information, or knowledge related to stakeholder needs and requirements definition
Stakeholder needs and requirements definition strategy	Approaches, schedules, resources, and specific considerations required to reflect consensus among the stakeholder classes to establish a common set of acceptable requirements. Includes the approach to capture the stakeholder needs, transform them into stakeholder requirements, and manage them through the life cycle
Stakeholder requirements	Requirements from various stakeholders that will govern the project, including required system capabilities, functions, and/or services; quality standards; system constraints; and cost and schedule constraints. Stakeholder requirements may be captured in the Stakeholder Requirements Specification (StRS)
Stakeholder requirements traceability	Bidirectional traceability of the stakeholder requirements
Standards	This handbook and relevant industry, country, military, acquirer, and other specifications and standards. Includes new knowledge from industry-sponsored knowledge networks
Strategy documents	Strategies for all applicable life processes, including business or mission analysis strategy, stakeholder needs and requirements definition strategy, system requirements definition strategy, architecture definition strategy, design definition strategy, system analysis strategy, implementation strategy, integration strategy, verification strategy, transition strategy, validation strategy, operation strategy, maintenance strategy, disposal strategy, project assessment and control strategy, decision management strategy, risk management strategy, configuration management strategy, information management strategy, measurement strategy, acquisition strategy, and supply strategy
Supplied system	The system or system element (product or service) is delivered from the supplier to the acquirer consistent with the delivery conditions of the supply agreement
Supply agreement	An understanding of the relationship and commitments between the project organization and the acquirer. The agreement can vary from formal contracts to less formal interorganizational work orders. Formal agreements typically include terms and conditions
Supply payment	Payments or other compensations for the supplied system. Includes receipt and acknowledgement
Supply record	Permanent, readable form of data, information, or knowledge related to supply
Supply report	An account prepared for interested parties in order to communicate the status, results, and outcomes of the supply activities
Supply response	The organization response to the request for supply
Supply strategy	Approaches, schedules, resources, and specific considerations required to identify candidate projects for management consideration. May also include inputs to determine supply constraints. Should also include the identification of potential acquirers
System analysis record	Permanent, readable form of data, information, or knowledge related to system analysis
System analysis report	An account prepared for interested parties in order to communicate the status, results, and outcomes of the system analysis activities. Includes the results of costs analysis, risks analysis, effectiveness analysis, and other critical characteristics analysis. Also includes all models or simulations that are developed for the analysis
System analysis strategy	Approaches, schedules, resources, and specific considerations required to accomplish the various analyses to be carried out, including methods, procedures, evaluation criteria, or parameters

System architecture description	Description of the selected system architecture, typically presented in a set of architectural views (e.g., views from architecture frameworks), models (e.g., logical and physical models, although there are other kinds of models that might be useful), and architectural characteristics (e.g., physical dimensions, environment resistance, execution efficiency, operability, reliability, maintainability, modularity, robustness, safeguard, understandability, etc.) (ISO/IEC/IEEE 42010, 2010). Architecturally significant system elements are identified and defined to some degree in this artifact. (Other system elements might need to be added during the design definition process as the design is fleshed out)
System architecture rationale	Rationale for architecture selection, technological/technical system element selection, and allocation between system requirements and architectural entities (e.g., functions, input/output flows, system elements, physical interfaces, architectural characteristics, information/data elements, containers, nodes, links, communication resources)
System design description	Description of the selected system design. System elements are identified and defined
System design rationale	Rationale for design selection, system element selection, and allocation between system requirements and system element. Includes rationale of major selected implementation options and enablers
System element descriptions	Design characteristics description of the system elements contained in the system; the description depends on the implementation technology (e.g., data sheets, databases, documents, exportable data files)
System element documentation	Detailed drawings, codes, and material specifications. Updated design documentation, as required by corrective action or adaptations caused by acquisition or conformance to regulations
System elements	System elements implemented or supplied according to the acquisition agreement
System function definition	Definition of the functional boundaries of the system and the functions the system must perform
System function identification	Identification of the system functions
System functional interface identification	Identification and documentation of the functional interfaces with systems external to the boundaries and the corresponding information exchange requirements
System requirements	What the system needs to do, how well, and under what conditions, as required to meet project and design constraints. Includes types of requirements such as functional, performance, interface, behavior (e.g., states and modes, stimulus responses, fault and failure handling), operational conditions (e.g., safety, dependability, human factors, environmental conditions), transportation, storage, physical constraints, realization, integration, verification, validation, production, maintenance, disposal constraints, and regulation. System requirements may be captured in a document called the System Requirements Specification (SyRS) or just System Specification. This includes the requirements at any level in the system hierarchy
System requirements definition record	Permanent, readable form of data, information, or knowledge related to system requirements definition

System requirements definition strategy	Approaches, techniques, resources, and specific considerations required to be used to identify and define the system requirements and manage the requirements through the life cycle
System requirements traceability	Bidirectional traceability of the system requirements
TPM data	Data provided for the identified measurement needs
TPM needs	Identification of the TPM, which measure attributes of a system element to determine how well a system or system element is satisfying or expected to satisfy a technical requirement or goal
Trained operators and maintainers	Trained humans that will operate and maintain the system
Transition constraints	Any constraints on the system arising from the transition strategy including cost, schedule, and technical constraints
Transition enabling system requirements	Requirements for any systems needed to enable transition of the system of interest
Transition record	Permanent, readable form of data, information, or knowledge related to transition
Transition report	An account prepared for interested parties in order to communicate the status, results, and outcomes of the transition activities. Includes documentation of the transition results and a record of any recommended corrective actions, such as limitations, concessions, and ongoing issues. Should also include plans to rectify any problems that arise during transition
Transition strategy	Approaches, schedules, resources, and specific considerations required to transition the systems into its operation environment
Updated RVTM	An updated list of requirements, their verification attributes, and their traces
Validated requirements	Confirmation that the various requirements will satisfy the business and stakeholder requirements
Validated system	Validated system ready for supply and operation. Also informs maintenance and disposal
Validation constraints	Any constraint on the system arising from the validation strategy including cost, schedule, and technical constraints
Validation criteria	The validation criteria (the measures to be assessed), who will perform validation activities, and the validation environments of the system of interest
Validation enabling system requirements	Requirements for any systems needed to enable validation of the system of interest
Validation procedure	A validation procedure that includes a set of validation actions, using specific validation techniques, performed with specific validation enablers
Validation record	Permanent, readable form of data, information, or knowledge related to validation
Validation report	An account prepared for interested parties in order to communicate the status, results, and outcomes of the validation activities. Includes validation results and the objective evidence confirming that the system satisfies its stakeholder requirements and business requirements or not. Should also communicate an assessment of the confidence level of the findings or results
Validation strategy	Approaches, schedules, resources, and specific considerations required to accomplish the selected validation actions that minimize costs and risks while maximizing operational coverage of system behaviors
Verification constraints	Any constraint on the system arising from the verification strategy including cost, schedule, and technical constraints
Verification criteria	The verification criteria (the measures to be assessed), who will perform verification activities, and the verification environments of the system of interest

Verification enabling system requirements	Requirements for any systems needed to enable verification of the system of interest
Verification procedure	A verification procedure that includes a set of verification actions, using a specific verification method/technique, performed with specific verification enablers
Verification record	Permanent, readable form of data, information, or knowledge related to verification
Verification report	An account prepared for interested parties in order to communicate the status, results, and outcomes of the verification activities. Includes verification results and the objective evidence confirming that the system fulfills its requirements, architectural characteristics, and design properties or not. Should also communicate an assessment of the confidence level of the findings or results
Verification strategy	Approaches, schedules, resources, and specific considerations required to accomplish the selected verification actions that minimize costs and risks while maximizing operational coverage of system behavior
Verified system	Verified system (or system element) ready for transition
WBS	The *work breakdown structure* is the decomposition of a project into smaller components and provides the necessary framework for detailed cost estimating and control. Includes a data dictionary. The costs for and description of the physical end products (hardware and software) may be captured in a product breakdown structure (PBS). The PBS supports bottoms up and algorithmic (parametric) cost estimating (see 10.1.3). The PBS is a key ingredient of commercial cost estimating tools

APPENDIX F: ACKNOWLEDGMENTS

SEH V4 CONTRIBUTIONS

The *INCOSE Systems Engineering Handbook* version 4 editorial team owes a debt of gratitude to all the contributors to prior editions (versions 1, 2, 2A, and 3). Tim Robertson led the effort to create version 1 of the handbook. Version 2 was led by James Whalen (ESEP) and Richard Wray (ESEP). Version 3 was led at various times by Kevin Forsberg (ESEP), Terje Fossnes (ESEP), Douglas Hamelin, Cecilia Haskins (ESEP), Michael Krueger (ESEP), and David Walden (ESEP). The framework they provided gave a solid basis for moving ahead with this version. This revision reflects changes to the previous version based on three primary objectives: first, to reflect the updated ISO/IEC/IEEE 15288:2015 standard; second, to reflect the state of the practice based on inputs from the relevant INCOSE Working Groups (WGs); and third, to be consistent with the Systems Engineering Body of Knowledge (SEBoK) wherever possible. Version 4 also corrected several minor issues identified by the INCOSE community.

A great deal of effort and enthusiasm was provided by the section leads and key authors, most of whom also serve as INCOSE WG Chairs or SEBoK authors. We acknowledge them in alphabetical order: Erik Aslaksen (CSEP), Albertyn Barnard, Joe Bobinis, Barry Boehm, Ed Casey, Dan Cernoch, Hugo Chale Gongora, Matthew Cilli, John Clark (CSEP), Bjorn Cole, Judith Dahmann, Arnold de Beer, Charles Dickerson, Rick Dove, Joe Elm (ESEP), Tom Fairlie, Alain Faisandier, Gauthier Fanmuy, Paul Frenz (CSEP), Sandy Friedenthal, Katri Hakola, Alan Harding, Cecilia Haskins (ESEP), Mimi Heisey, Eric Honour (CSEP), Scott Jackson, Ken Kepchar (ESEP), Alain Kouassi, Gary Langford, Claude Laporte, Alain LePut, Howard Lykins, Ray Madachy, James Martin, Jen Narkevicius, Warren Naylor, Bohdan Oppenheim, Ricardo Pineda, Paul Popick, Derek Price, Melinda Reed, Kevin Robinson, Jean-Claude Roussel (ESEP), Mike Ryan, Frank Salvatore (CSEP), Hillary Sillitto (ESEP), Jack Stein, Richard Swanson (ASEP), Corrie Taljaard, Chris Unger, Beth Wilson (ESEP), and Mark Wilson (ESEP).

The INCOSE Technical Operations review team led by Quoc Do generated excellent comments that significantly improved the handbook. Other individual reviewers also generated useful review comments. We acknowledge them in alphabetical order: Aaron Chia, Stephen Cook, Judith Dahmann, Bruce Douglass, Nick

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.

Dutton (ASEP), Jeff Grady (ESEP), Robert Halligan, Alan Harding, Cecilia Haskins (ESEP), Ray Hentzschel, Charles Homes, Rainer Ignetik, Vernon Ireland, Julian Johnson (ASEP), Mario Kossmann (CSEP), Michael Krueger (ESEP), Paul Logan, Bill Miller, Kevin Patrick, Jack Ring, Steve Saunders, Paul Schreinemakers, David Schultz, Zane Scott, Despina Tramoundanis, Joyce van den Hoek Ostende, Charles Wilson (CSEP), and Kenneth Zemrowski (ESEP). We would also like to thank the certification beta exam participants and the specific and anonymous reviewers who provided comments on v3.2, v3.2.1, and v3.2.2. Their inputs were much appreciated.

The editors wish to acknowledge the Idaho National Laboratory Systems Analysis Department for their significant support of this update and thank them for their contribution of technical editor Douglas Hamelin. We also wish to acknowledge Lockheed Martin Corporation for their significant support of this update and thank them for their contribution of Thomas Shortell (CSEP), who was the lead for the handbook figures and the electronic versions. The editors wish to thank Vitech Corporation for the use of their CORE tool, which was used to create an underlying process model that helped ensure consistency in the handbook IPO diagrams.

Any errors introduced as part of the editorial process rest with the editors, not the contributors.

We apologize if we unintentionally omitted anyone from these lists.

Gratefully, David Walden (ESEP), Garry Roedler (ESEP), and Kevin Forsberg (ESEP).

APPENDIX G: COMMENT FORM

Reviewed document:	INCOSE SE Handbook v4.0
Name of submitter:	Given FAMILY (given name and family name)
Date submitted:	DD-MMM-YYYY
Contact info:	john.doe@anywhere.com (email address)
Type of submission:	Group (individual/group)
Group name and number of contributors:	INCOSE XYZ WG (if applicable)
Comments:	Detailed comments with reference to document section, paragraph, etc. Please include detailed recommendations, as shown in the table below

Send comments to info@incose.org

Comment ID	**Category** (TH, TL, E, G)	**Section number** (e.g., 3.4.2.1, no alpha)	**Specific reference** (e.g., paragraph, line, figure, table)	**Issue, comment, and rationale** (rationale must make comment clearly evident and supportable)	**Proposed change/ new text—mandatory entry** (must be substantial to increase the odds of acceptance)

E, editorial; G, general; TH, technical high; TL, technical low.

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.

INDEX

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.

WILEY END USER LICENSE AGREEMENT

Go to www.wiley.com/go/eula to access Wiley's ebook EULA.