Chapter 7

Systems Engineering in Professional Practice

ROGER C. BURK, Ph.D.

I have been involved with many milestone reviews as the principal for the Defense Acquisition Board. I see examples of well-run and poorly run projects. The difference becomes clear in the first few minutes of the review. What is also clear is the important role that systems engineering plays in making a project run smoothly, effectively, and efficiently, as well as the contrary, where the lack of systems engineering and the discipline that comes with proper implementation can cause tremendous problems [1].

—Michael W. Wynne, Secretary of the Air Force

7.1 THE SYSTEMS ENGINEER IN THE ENGINEERING ORGANIZATION

The typical job for a professional systems engineer is technical integrator supporting a Program Manager who is developing a complex system.

Systems Engineering as a discipline was introduced in Section 1.7, with Chapter 6 describing it in more detail. This chapter is focused on the practitioner of the discipline: his or her place in the organization, responsibilities, and specific activities and tasks, so as to convey what it is like to be a systems engineer. I also want to give some references for important aspects of the systems

Decision Making in Systems Engineering and Management, Second Edition
Edited by Gregory S. Parnell, Patrick J. Driscoll, Dale L. Henderson

engineer's job that are beyond the scope of this book. This section discusses the job in general, including the title and the organizational placement. Section 7.2 describes the activities during each part of the system life cycle. Section 7.3 describes relationships with other practitioners involved in system development. Section 7.4 goes into team formation and building, since systems engineers often lead interdisciplinary teams, either formally or informally. Section 7.5 gives more detail on typical responsibilities of a systems engineer—for instance, which documents he or she would be expected to prepare. The next two sections reflect on the nature of the systems engineering job: 7.6 describes the various roles an SE can play, and 7.7 gives the personality characteristics conducive to being a good systems engineer. Section 7.8 summarizes the chapter.

Figure 7.1 is a map of the concepts presented in this chapter. Typically, a professional systems engineer works for a Chief Systems Engineer, who in turn works for a Program Manager. The systems engineer is the technical interface with clients, users, and consumers and is often the one responsible for building a team. This team will adopt an attitude concerning the challenge (good or bad) and usually lasts throughout the team's life cycle. The systems engineer has roles and responsibilities, and ideally it has certain personal characteristics that contribute significantly to success. The specific activities that a systems engineer performs are distributed (nonuniformly) over the system life cycle. Throughout this chapter, the acronym SE will be used to describe both a systems engineer and the discipline of systems engineering. The distinction will be clear from the context within which the acronym is used.

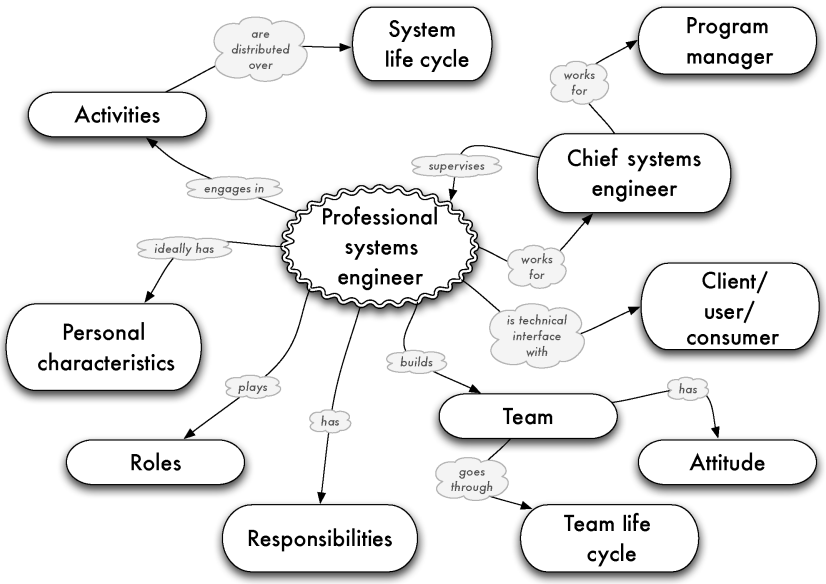

Figure 7.1 Concept map for Chapter 7.

The Systems Engineering Job

Anyone in any walk of life can use systems thinking to find good solutions to complex problems in a technological setting. To some extent, any professional engineer will use systems considerations to (a) determine the requirements for a system he or she is designing, (b) define its interfaces with other systems, and (c) evaluate how well it is performing. However, in professional practice some people are given specific big-picture engineering responsibility for a system or group of systems. These people have "systems engineer" in their job title or job description, and they are responsible for making sure that the technical efforts of everyone involved work together to produce an operational system that meets all requirements. For instance, a team developing a new helicopter will include mechanical engineers to make sure all the moving parts work together, aeronautical engineers to make sure the vehicle flies, electrical engineers to make sure the control systems work, software engineers to make sure the onboard computer systems operate properly, and many other discipline or specialty engineers: reliability engineers, test engineers, human factors engineers, and so on. It is the systems engineer who takes the overall view and is responsible for ensuring that all these engineers work together at the technical level to produce a system that meets the customer's needs. This chapter is about what these professional systems engineers actually do.

The International Council on Systems Engineering (INCOSE) has produced a Systems Engineering Handbook [2] that covers much of the material of this chapter in more detail, but in condensed handbook form for practitioners.

Three Systems Engineering Perspectives

There are three perspectives on the practice of systems engineering: the organization's, the system's, and the individual's. The organization is interested in (a) the process and what people do and (b) meeting customer needs and expectations. However, we also have to remember the point of view of the system: the product and its development throughout its life cycle. This is the long-term point of view, concerned with the design, manufacture, test, deployment, and operation of the system, regardless of the developing organization. Finally, the individual SE has a different perspective, concerned with the here and now, tasks and responsibilities, and getting the job done. This chapter stresses the organizational perspective and also describes the system and individual perspectives as they relate to the engineering organization.

Organizational Placement of Systems Engineers

A civil or mechanical engineer sometimes works on his or her own, doing small design projects for individual customers. Such an engineer can also work for a small engineering company that takes on contracts to build bridges, levees, industrial facilities, and so on. In contrast, most systems engineers work for larger companies, because those are the companies that take on complex projects that cannot be

done without a systems approach. The systems engineer will be a part of a team developing a complex system such as an aircraft or a telecommunications network, or perhaps he or she will be in a systems engineering staff overseeing a number of smaller projects. If a systems engineer works for a small company, it is generally a consulting firm whose clients are large companies or government agencies seeking advice on how to develop or acquire major complex systems.

The organizational placement of systems engineers can vary widely based on the scale and complexity of the system involved and on the technical practices of the organization. For a major system such as an enterprise software system, an aircraft, or a spacecraft, where the development cost is hundreds of millions of dollars or more, there will often be a dedicated chief systems engineer who reports to the program manager. This chief SE may have a staff of systems engineers working for him or her, especially early in system development when key decisions about system architecture and system requirements are made. In this case, "Systems Engineering" (SE) can refer to a functional organization. The chief SE may also be given responsibility for technical specialists, such as configuration management personnel. For a smaller system, there may be a chief systems engineer working with a smaller staff, or even working alone as an advisor to the program manager on overall technical integration of the program. Sometimes the program manager him- or herself is also designated as the program systems engineer. If the organization is responsible for a family of relatively small systems, such as a set of communications or electronic devices, there may be a single SE with technical oversight for all of them.

7.2 SYSTEMS ENGINEERING ACTIVITIES

An SE coordinates technical efforts over the lifetime of the system, but he or she is usually busiest toward the beginning, when he or she defines the system requirements, develops the system concept, and coordinates and integrates the efforts of the other design engineers.

Whether a system is large or small, its development will go through a series of system life cycle stages (described in Chapter 3). The execution of the fundamental SE tasks (described in Chapter 6) will be distributed in a sequence of activities during these stages. The distribution is not uniform. Task 1 ("Use an interdisciplinary approach . . . ") applies constantly throughout the system life cycle, but Task 2 ("Convert customer needs . . . ") is predominantly in the early stages, and Task 3 ("Lead the requirements analysis . . . ") has peaks of activity both early, when system requirements are established, and later, during systems test, when proper function at the system level is verified. The systems engineer's changing set of activities during these life cycle stages can be described as follows. Table 7.1 summarizes them.

TABLE 7.1 Summary of Systems Engineering Activities During a Systems Life Cycle

Life Cycle Stage	Major SE Activities
Establish system need	Stakeholder interaction (especially users)
	Requirements definition
	Functional analysis
Develop system concept	Stakeholder interaction (especially users)
	Team building
	Requirements analysis and management
	Architectural tradeoff studies
	Definition of system architecture, elements, boundaries, and interfaces
	Creation of systems engineering master plan
Design and develop the system	Stakeholder interaction (especially users)
	Interface control
	Overall design coordination
	Requirements analysis and management
	Configuration control
	Specialty engineering
	Coordinating major design reviews
	System development testing
Produce system	Coordination with production engineers
	System acceptance testing
Deploy system	Coordinating deployment with users and other stakeholders
Operate system	Gathering data on system performance and user feedback
	Coordinating problem resolutions and system upgrades
Retire system	Coordinating hand-off of mission to replacement system
	Coordinating system disposal

Establish System Need

Stakeholder interaction is a key role for SEs during this stage. The SE will talk to the stakeholders, paying special attention to consumers of the system's products and services, the system users, and the system owner, in order to understand and document the needs that are driving the system development. For instance, if the system is a passenger aircraft, the consumer is the airline passenger, the user is the aircrew, and the owner is the airline. The SE will use the needs to identify the system functions and define the specific technical requirements for the system. The SE often acts as an interface between the consumer or user and the designer. He or she translates between "consumer-speak" or "user-speak" and "engineer-speak" and makes sure they understand each other correctly. The result is usually a formal statement of the system's form, fit, and function, agreed to in writing by all parties. After this stage, the SE continues to manage the requirements, ensuring no change without sufficient cause and agreement by all parties.

Develop System Concept

At this point, the need is established and a project initialized, and the SEs play perhaps their most important role. In the organizational dimension, this is a time of team building, organization, and planning. This is when people are recruited, background research in the problem area is done, key tasks identified, and a preliminary schedule created. At the technical level, the SE continues to manage and refine the system requirements. The SE also examines different candidate system architectures and helps make the selection of the one to be pursued. This crucial process is described in more detail in Section 7.5; it includes identifying system functions, system boundaries, and major system elements and defining their interfaces. A Systems Engineering Management Plan may also be written during this stage, to plan SE activities for the rest of the system's life cycle. A Test and Evaluation Master Plan may be written to describe how testers will ensure that the finished system meets requirements.

Design and Develop the System

Detailed design is primarily the responsibility of discipline-specific engineers, but the SE also has a role to ensure system-level requirements are met. The SE maintains technical cognizance over the whole program to ensure that all elements will work together as they should. The SE often coordinates interfaces between system elements and between the system and its environment. Often these interfaces will be defined in a set of interface control documents (ICDs), which will be under configuration control, meaning that they can be changed only after review by the SE and approval by a Configuration Control Board (CCB). The ICDs will define the interfaces exactly, often in mind-numbing detail. For instance, the ICD for a data exchange will identify each data field, the type of data in it, the units of measurement, the number of characters, and so forth. An SE is also often involved in specialty engineering such as reliability, maintainability, and usability (see Section 7.5 and Chapter 8 for more on specialty engineering and its role in SE). When it is time to integrate the various elements and test them at the system level, the SE is usually in charge. He or she will be involved in planning, executing, and evaluating operational tests, in which test personnel not involved in system development use the system under field conditions. This is the second peak of activity for fundamental systems engineering Task 3 as described in Chapter 6: the time when the SE ensures system validation and successful system realization.

Produce System

The SE may be involved in the production plan and in monitoring performance to ensure that the system is built as designed. During this stage, the SE plays a key role in the analysis and approval of engineering change proposals and block upgrades.

Deploy System

The SE will help plan the deployment to ensure that all operability needs are met.

Operate System

The SE examines how well the system meets the consumers' needs, users' needs, and the client's expectations. This often involves gathering data and feedback from the consumers and users after the system is deployed. These provide input for deciding on modifications for future systems, problem fixes or upgrades for systems already in use, and improvements on systems engineering practices.

Retire System

The SE ensures that system retirement and disposal are properly planned for and executed.

7.3 WORKING WITH THE SYSTEMS DEVELOPMENT TEAM

Because of their wide responsibilities, SEs often find themselves working with a wide variety of other professionals, who may have widely varying engineering expertise. The following sections sketch out how an individual SE typically works with other professionals. Of course, people do not always fill these roles as a full-time job, though the larger the program, the more likely that they will. These relationships are summarized in Table 7.2.

The SE and the Program Manager

Broadly speaking, every project or program will have a manager who has overall responsibility for both technical and business success. In private organizations, the program manager (PM) is responsible for making sure that the program makes money for the company. In public organizations, the program manager is responsible for making sure the program provides the product or service for a given budget. This is the most responsible job in the program. The chief SE is the PM's chief advisor on overall technical aspects of the program, and the SE staff provides system integration for the PM. Table 7.3 compares the responsibilities of the PM and the SE.

The SE and the Client, the User, and the Consumer

The client pays to develop the system, the user operates it, and the consumer receives products or services from it. These may not be the same individuals, or even in the same organization. For instance, in Army acquisition the customer is a program manager in Army Material Command stationed at a place like Redstone Arsenal in Alabama, whereas the user and the consumer may be soldiers in the

TABLE 7.2 Summary of Program Roles in Relation to the Systems Engineer

Individual	Basic Responsibility	Provides to SE	Receives from SE
Program manager (PM)	Business and technical management of program	Program direction	Technical advice and leadership
Customer	Acquire the best system for his or her organization	System requirements	Technical information and recommendations
User	Operate the system	System requirements	Technical information
CTO or CIO	Coordinate a company's technology policy and/or information systems	Guidance and cooperation	Technical information and recommendations
Operations researcher or system analyst	Mathematical modeling to support decision making	Well-supported technical recommendation	Tasking for trade studies and analyses
Configuration manager	Ensure no changes occur without agreed level of review	Assurance	Cooperation
Life cycle cost estimator	Estimate system costs	Cost information	Technical information
Engineering manager	Appropriate engineering processes	Well-managed discipline engineering	Cooperation
Discipline engineer	Detailed design	Sound design and interface requirements	System architecture
Test engineer	Ensure materials, components, elements, or systems meet specifications and requirements	Test results	Requirements
Specialty engineer	Ensure system meets requirements in area of specialization	Specialized expertise	Requirements
Industrial engineer	Technical operation of industrial plant	Efficient plant	Coordination
Quality assurance	Ensure that manufacture and test is performed as intended	Assurance	Cooperation

TABLE 7.3 Comparison of the Program Manager and the Systems Engineer [5]

Domain	Program Manager	Systems Engineer
Risk	Manages risk	Develops risk management process
	Sets guidelines	Analyzes risk
Changes	Controls change	Analyzes changes
		Manages configuration
Outside interfaces	Primary customer interface	Primary user interface
Internal interfaces	Primary internal management interface	Primary internal technical interface
Resources	Provides and manages resources	Delineates needs
		Uses resources

field. For an information technology (IT) system, the client may be an IT organization within a company, whereas the consumers may be the clients of the company and the users may be distributed throughout the company's other divisions. Clients, users, and consumers are all critically important, but by definition the client has control over the acquisition process. The program manager is responsible for relations with the client and the SE is the PM's primary advisor on technical issues in that relationship. The SE is also responsible for coordinating relationships with consumers and users, especially developing system requirements that take into account consumer and user needs and desires. This process is much easier if the client has good relations with the consumers and users.

The SE and the CTO or CIO

A company oriented toward research, technology, or systems development may designate a high-level executive to have responsibility for technical issues such as research and development and strategic technical direction. This person is often called the chief technology officer (CTO). The CTO is responsible for using technology to achieve the organization's mission. The exact scope of responsibilities varies widely from company to company, and some high-tech companies do not use this title. The CTO's role is somewhat like that of a systems engineer for the entire company as a single enterprise. Other companies designate an executive as chief information officer (CIO), who will be responsible for the company's information strategy and for information technology systems to achieve the organization's mission.

The SE and the Operations Researcher or System Analyst

These specialties involve studying existing or proposed systems and environments and evaluating system performance. They are sometimes regarded as part of systems engineering, though their emphasis is on mathematical and computer-based

modeling and simulation (see Chapter 4) and not on such SE activities as managing user interfaces, requirements definition and allocation, and system performance verification. These individuals specialize in answering quantitative questions about complex systems, and they are often invaluable in making a sound and defensible decision when there is a lot at stake and the best course of action is not clear.

The SE and the Configuration Manager

Configuration management (CM) is the process of ensuring that things do not change without due review and approval and without all stakeholders being aware of the change and its implications. This includes key documents, such as those that describe system requirements, design, and interfaces, as well as the actual physical configuration of the system being built. Experience has shown that without strong CM discipline, people will have good ideas that cause them to introduce small changes into the system, and these changes will accumulate and cause chaos later when everyone has a slightly different version of the system. Typically, detailed configuration is under the control of the design engineer early in the design process. At some point the design is brought under configuration control; after that, any change requires paperwork, reviews, signatures, and approval by a configuration control board that is often chaired by the PM. The configuration manager administers this paperwork process. It is not a romantic job, but it is absolutely essential in developing a system of any significant complexity. In some organizations, CM is part of the SE shop; in others it reports independently to the PM.

The SE and the Life Cycle Cost Estimator

The program manager is deeply concerned with system costs in both the near and far terms, because costs help determine system viability and profitability. The total cost of a system over its entire life cycle is especially important. This includes costs to design, build, test, deploy, operate, service, repair, upgrade, and finally dispose of the system. Cost estimators have their own methods, models, and databases that they use to help estimate total life cycle cost (see Chapter 5). Some of these methods are empirical or economic in nature; others are technical, and the SE can expect to be involved in them. Since life cycle cost is a system-level criterion of great interest to the PM, the SE will want to to use life cycle cost as a key consideration in all systems decisions.

The SE and the Engineering Manager

An engineering manager (EM) is in charge of a group of engineers. He or she is concerned with ensuring that sound engineering methods are used, as well as performing the usual personnel management functions. To the extent that sound engineering always involves some element of a systems perspective, an EM will also be involved in promoting systems engineering. However, an EM's basic responsibility is sound discipline-specific engineering. When a functional or matrix

organization is used, the EM may be in charge of engineers working on many different programs and systems, so his or her system perspective may be weaker than the cognizant systems engineer. In contrast, the SE is primarily a technology leader, integrator, coordinator, and advisor.

The SE and the Discipline Engineer

By discipline engineer, we mean mechanical engineer, civil engineer, electrical engineer, aerospace engineer, software engineer, chemical engineer, environmental engineer, information security engineer, and so forth. These engineers design things, and they are responsible for every detail of what they design. In contrast, a systems engineer is responsible for the high-level structure of a system, and at some level of detail he turns it over to the appropriate discipline engineer for completion. Frequently an SE starts out professionally as a discipline engineer and moves into systems work as his or her career progresses. This provides a useful background that enables the SE to work with other engineers, particularly in his or her original discipline. However, the SE must have (or develop) the knack for dealing with experts in fields other than his or her own. He must convince them that he can grasp the essentials of a sound argument in the expert's field. Also, the SE often spends a great deal of time translating what the user or consumer says into precise language that the design engineer can use, as well as translating what the design engineer says into language that the user or consumer can understand.

The SE and the Test Engineer

Testing can occur at the material, component, element, system, or architecture level. System test is usually considered a systems engineering responsibility, and elements that are complex enough to be treated as systems in themselves may also be given to systems engineers to test. An engineer who specializes in system test is a specialized SE; other test engineers specialize in material or component testing, and they are considered specialty engineers.

The SE and the Specialty Engineer

Specialty engineers are those who concentrate on one aspect of design engineering, such as human factors, reliability, maintainability, or information assurance (see Section 7.5 below for a longer list). Sometimes these specialties are collectively referred to as *systems effectiveness* or "the ilities." These specialties require a systems outlook, though they are narrower in focus than general systems engineering. An SE should have a basic understanding and appreciation of these specialty engineering disciplines. In a large program, there may be one or more engineers specializing in each of these areas, and they may be organized separately or within a systems engineering office.

The SE and the Industrial Engineer

An industrial engineer (IE) can be regarded as a systems engineer for an industrial operation, such as a manufacturing plant. An IE's responsibilities might include facility layout, operation scheduling, and materials ordering policies. Other IEs design efficient processes for service industries; yet others deal with human factors in commercial operations. A program SE can expect to interact with a plant IE when working out producibility and related issues. Industrial Engineering as an academic discipline predates the emergence of SE, and many universities teach both in the same department because of the related history and substantial overlap in material.

The SE and Quality Assurance

Quality assurance (QA) means making sure an item is produced exactly as it was designed. In many organizations, there is a separate quality assurance organization that reports directly to a high-level manager. QA personnel are not concerned directly with engineering, but with process. They ensure that all checks are made, that all necessary reviews are completed, and that all required steps are executed. QA personnel provide an important independent check to ensure that the fabrication and test process was executed exactly as the engineers intended. A QA person may carry a personal stamp to apply to paperwork to verify that it has been reviewed and that all necessary signatures are on it. That QA stamp will be required before work can proceed to the next step.

7.4 BUILDING AN INTERDISCIPLINARY TEAM

Systems engineers often form and lead interdisciplinary teams to tackle particular problems.

Because of their role as both leader and integrator, systems engineers often find themselves in the organizational role of assembling teams of people from various backgrounds to work on a particular task or project. The systems engineer must identify early on the people with the best mix of skills to work the given problem. The team membership will vary by the nature of the problem. The team may include electrical, mechanical, and civil engineers; it can also include architects, computer or political scientists, lawyers, doctors, and economists. Technical skills are only a part of the mix.

Team Fundamentals

Figure 7.2 shows the key ingredients and products of a successful team. The vertices of the triangle show the products of a successful team. The sides and inner triangles describe what it takes to make the results happen. In *The Wisdom of Teams* [3],

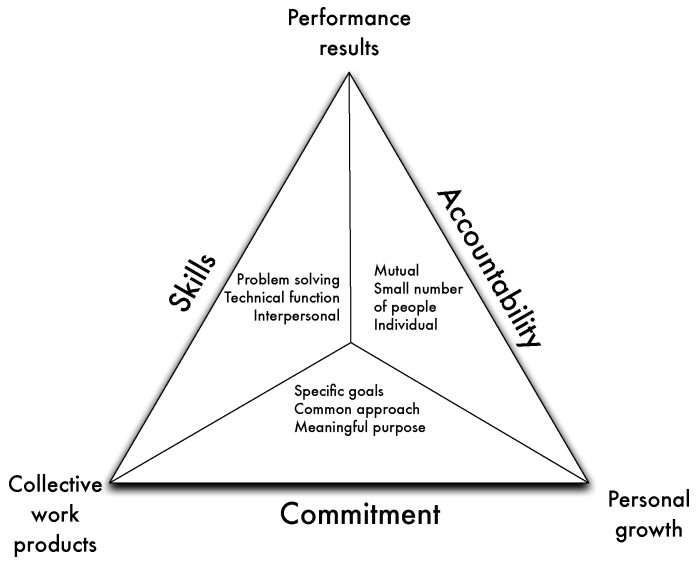

Figure 7.2 Fundamentals of a successful team [3].

Katzenbach and Smith stress that the performance ethic of the team, comprising accountability and commitment, is essential for team success. They build on this to create a definition that distinguishes a team from "a mere group of people with a common assignment."

> A team is a small number of people with complementary skills who are committed to a common purpose, performance goals, and approach for which they hold themselves mutually accountable.

Each member of the team must understand and be committed to the answers to three fundamental questions:

1. Why are we here?
2. What are we to accomplish?
3. What does success look like, and how will we know when we get there?

The answers to these questions will differ based on the type of team assembled.

Team Attitude

It is vital that team members have an attitude that "only the team can succeed or fail." This is difficult to foster in an ad hoc team drawn from many sources.

Members not fully assimilated into the team may have a loyalty to their home organization and seek what is best for their constituency. A key to building an effective SE team is to assemble the complete team early, rather than adding members over the life of the project. This encourages cooperation, buy-in, and a sense of ownership early by everyone. A common occurrence is the addition of expertise such as finance or marketing during later stages, when their contribution is more directly related. However, this often leads to a feeling of outsider-ship that can result in lackluster enthusiasm, if not outright sabotage.

Team Selection

Building and managing a successful team requires several up-front decisions by the lead systems engineer [4]:

- What is the right mix of skills and power level? Should members represent a spectrum of rank and authority to promote varied viewpoints, or should the power level be the same to avoid undue influence by superiors?
- What attitude toward collaboration and problem solving is required?
- How much time is available?
- How well-defined must the problem be to fully engage team members?

Team Life Cycle

The systems engineer must evaluate the impact of the team as it works through the problem. Katzenbach and Smith [3] designed the team performance curve shown in Figure 7.3 to illustrate how well various teams achieve their goals.

A working group relies on the sum of the individual "bests" for their performance. Members share information, best practices, or perspectives to make

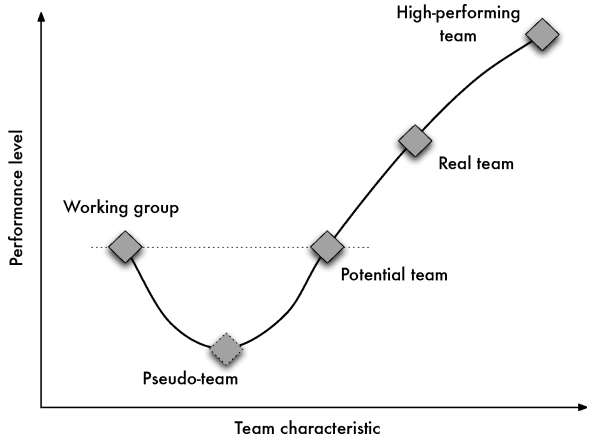

Figure 7.3 Team performance curve [3].

decisions to help each individual perform within his or her area of responsibility. Pseudo-teams are teams in name only and are not focused on the collective performance. Katzenbach and Smith rate their performance impact below that of work groups because their interactions detract from other individual performances without delivering any joint benefit. Potential teams have significant incremental performance and are trying to improve their impact. However, they have not yet established collective accountability. Their performance impact is about the same as a working group. Real teams are those that meet the definition of a small number of people with complementary skills who are equally committed to a common purpose, goals, and a working approach for which they hold themselves mutually accountable. Teams with the highest performance impact, high-performance teams, are those that are *real teams*, and their members are also deeply committed to one another's personal growth and success. That commitment usually transcends teams.

Cross-Cultural Teams

A word is in order about the special problems of teams with members from different ethnic, national, or religious cultures. Such teams are becoming more and more common as professionals from all countries become more mobile, as more international projects are undertaken, and as "virtual teams" that interact only electronically become more common. Even men and women from the same culture can sometimes have different behavioral expectations. If team members are not used to working with people from the other cultures on the team, misunderstandings can arise that interfere with team formation. Different cultures have different customs governing how business is done, including norms on very basic things, such as:

- How close to each other should two people stand when speaking together?
- What physical contact is appropriate between colleagues of the same or of the opposite sex?
- How should one dress in the workplace?
- How much privacy should one expect?
- How much organizational structure is needed?
- How many pleasantries must be exchanged before getting to business?
- How frank or tactful should one be when expressing dissatisfaction?
- How much deference should be shown to those in authority?
- How much individual ambition is appropriate to express?
- What kind of humor is acceptable?
- How diligent must one be when on the job?
- How scrupulously honest must one be in matters small and great?

Cultural norms like these are taken in with one's mother's milk. At first many people are hardly aware that they are only the customs of the people they grew up with and not the universal rules of decent behavior. They can be unaware of

how behavior that seems to them entirely normal and ordinary can be considered offensive according to other cultural norms. This is certainly true of some Americans, and it is equally true of some from Asia, Africa, Europe, Latin America, and every other place in the world.

Fortunately, if one is aware of the potential for misunderstanding and has a little bit of goodwill, these problems are not too hard to avoid, especially among the well-educated people likely to be on engineering teams. The important point here is to be aware of the potential problems due to cultural differences, to maintain a spirit of forbearance and understanding, and to be a good listener. Also, keep in mind that there is a lot of variation between individuals from the same culture, so not everyone will behave as one might expect based solely on their background. Harris and Moran [5] provide a text for those who want to better understand cultural issues and develop cross-cultural teamwork skills.

7.5 SYSTEMS ENGINEERING RESPONSIBILITIES

> Common specific assigned responsibilities of SEs include writing a systems engineering management plan, external technical interface, requirements analysis, requirements management, system architecting, interface control, writing a test and evaluation master plan, configuration management, specialty engineering, coordinating technical reviews, and system integration and test.

From the point of view of the individual, the following are the specific responsibilities that are often assigned to an SE. From the point of view of the organization, these are the tasks that the systems engineering office will accomplish.

Systems Engineering Management Plan (SEMP)

In a major project, a systems engineering management plan (SEMP)(often pronounced as one syllable: "semp") should be written when the system concept is defined but before design starts. This document describes how the ideal systems engineering process is going to be tailored for the problem at hand, and it is the basis for all technical planning. It relates technical management to program management, provides a technical management framework, sets coordination requirements and methods, and establishes control processes. It is a communication vehicle that lets the client, users, consumers, and everyone on the project know how systems engineering will be carried out. The SEMP is a living document that can be modified from time to time as circumstances change. A typical SEMP might contain the following:

- Description of the envisioned development process and system life cycle
- SE activities in each envisioned phase

- Participants and involved organizations
- Planned major system reviews, audits, and other control points, including success criteria
- Products and documentation to be produced by the SE
- Risk management plan
- System requirements, including method of testing
- Identification of key measures of technical progress
- Plan for managing internal and external interfaces, both physical and functional
- Description of any trade studies planned
- Integration plan for configuration management, quality assurance, system effectiveness engineering, and other specialties, as required

Technical Interface with Users and Consumers

Systems engineers will usually be charged with user and consumer relationships as described above. The SE will meet with the user and consumer, travel to their locations, conduct interviews, focus groups, or surveys, and do whatever else is required to ensure that the users' and consumers' needs and desires are captured. The SE will be responsible for ensuring that the written system requirements truly describe what the user and consumer need with sufficient precision to design the system, and with sufficient accuracy that a system that meets the requirements will also meet the needs. The SE also has to interpret engineering constraints for nontechnical stakeholders, so that they can understand when a particular requirement should perhaps be relaxed because of its disproportionate effect on system cost, reliability, or other criterion.

Analysis and Management of Systems Requirements

In any major system development, the SE should be in charge of the system requirements, and those requirements should be written down in a document that only the CCB can change. These requirements define what the system needs to do and be in order to succeed. They can determine whether or not a billion-dollar contract has been properly executed and how much (if anything) the contractor will be paid. There is an important tradeoff in determining requirements. Typically, if the performance requirements are set at the high level that users and customers would like, the resulting system will be too expensive, unreliable, or both. If the requirements are set too low, the system may not perform well enough to be worth building. Ultimately, it is the PM's responsibility to make a requirements tradeoff between cost and performance while remaining cognizant of schedule. It is the SE's responsibility to make sure the PM understands the consequences and risks of the decision. A common approach is to establish both *threshold* requirements, which must be met to have a worthwhile system, and *goal* (or *objective*) requirements, which represent a challenging but feasible level of higher performance.

Requirements development should start with *customer requirements* that describe what is expected from the system by the operator, the client, and other stakeholders. These requirements should cover where and how the system will be used, in what environments, what its minimum performance should be, how long the system life cycle should last, and so forth. Requirements can start with a high-level objective such as, "We're going to land a man on the moon and return him safely to Earth." Analysis of such high-level requirements will produce *functional requirements* for such things as rocket propulsion, temperature control, breathable atmosphere, and communications. *Nonfunctional requirements* may specify criteria that will be used to judge the system but are not related to specific functional behaviors—for instance, color, finish, or packaging. *Performance requirements* will specify how well or to what level functional requirements have to be performed—for instance, speed in knots, availability in percent, or reliability as a probability.

Further analysis of a moon mission requirement will produce requirements for a launch vehicle, a command module, a lunar lander, and so forth. At the lowest level there will be *design requirements* describing exactly what must be built, coded, or bought. High-level requirements should not vary with the implementation, but as detail increases, the requirements become more and more dependent on the particular technical design chosen. For instance, a lunar mission going directly to the Moon's surface without a rendezvous in lunar orbit would not have separate requirements for a command module and a lander. The value measures identified in the systems decision process (see Chapter 10) are natural bases for system-level requirements, if they are direct and natural measures suitable for formal testing.

Requirements that come from an analysis of other requirements, rather than directly from an analysis of what the system must do and be, are called *derived requirements*. One kind of derived requirement is an *allocated requirement*, which is laid upon a subsystem to partially fulfill a higher-level requirement. For instance, two subsystems may have allocated reliability requirements of 0.9 and 0.8 in order to meet a system reliability requirement of 0.72.

The SE has the job of documenting both high-level and derived system requirements, at least down to a certain level of detail. One common way to visualize the process is the systems engineering "V," an example of which is shown in Figure 7.4. The highest-level requirements should be written in language that the important stakeholders can readily understand, and they may be tested in an acceptance test. From these the SE may derive system requirements that are in engineering language and which will be tested at the system level. Further analysis may derive subsystem requirements, or may allocate system requirements directly to subsystems, and the subsystems will be tested in integration tests. At the lowest level are component requirements and testing. Thus a requirement to go to the Moon leads to a requirement for a booster with a certain performance, which leads to a requirement for a rocket motor of a certain power, which leads to a requirement for a rocket nozzle with certain characteristics. At the lower levels, the design and requirements allocation will be in the hands of discipline engineers, but the SE will have oversight of the integrity of requirements traceability.

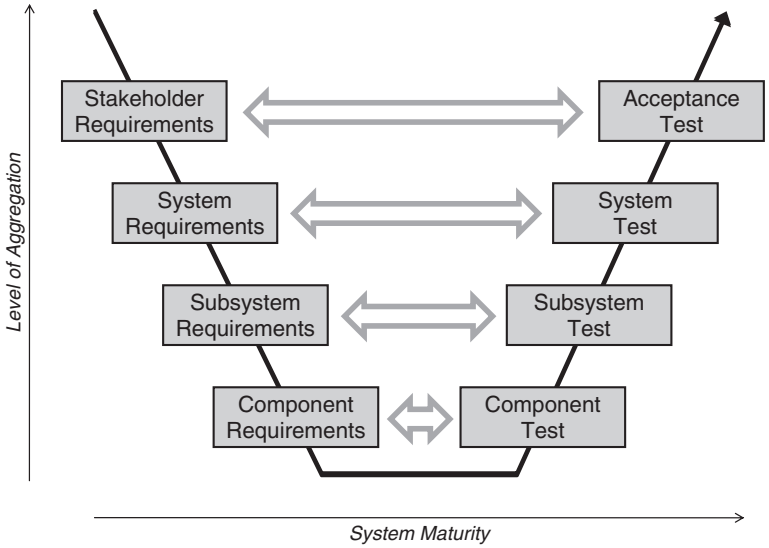

Figure 7.4 A systems engineering "V" [6].

Requirements can be of different types (sometimes overlapping):

- Customer requirements, which describe what the client expects
- Functional requirements, which define what the system has to do
- Nonfunctional requirements, which specify criteria not related to system behavior
- Performance requirements, which specify in engineering units how well the functions have to be performed
- Constraint requirements, which describe the constraints under which the performance needs to be demonstrated
- Design requirements, which specify in detail what is to be built, coded, or bought
- Derived requirements, which are developed from higher-level requirements
- Allocated requirements, which are derived from a higher-level requirement and assigned to a subsystem
- Physical requirements, which give the form, fit, and finish of the system

The SE should ensure that all requirements are:

- Unique, meaning that no two requirements overlap or duplicate each other
- Unambiguous, so that there can be no misunderstanding of exactly what is required
- Testable, so that there is a practical way of determining whether the delivered system actually meets the requirement
- Traceable, so that every high-level requirement is met by one or more low-level testable requirements, and every low-level requirement supports a documented high-level requirement

Once set, requirements should be changed only with extreme reluctance. Fluctuating requirements typically waste effort by requiring redesigns; they also promote integration problems when the full implications of the change are not grasped at first. It is also common for stakeholders to come late to the process with new requirements, leading to "requirements creep" that accumulates until the system becomes unaffordable or unworkable. The resulting schedule and cost impacts have led to the demise of many programs.

Later chapters in this book present techniques for requirements analysis and trade studies. These techniques include functional analysis (for functional requirements), screening criteria (for performance requirements), and value modeling (for goal requirements). These are described in Chapters 10 and 11 and their use is illustrated in Chapter 12. For the most complicated systems, requirements engineering can become a discipline of its own. See Hull, Jackson, and Dick [6] and Laplante [7] for in-depth treatments of requirements analysis.

System Architecting

This is the first stage in establishing the design for a complex system. For example, in the Apollo program, the decision to use one launch vehicle to put an entire mission into Earth orbit, rather than two launches with an Earth orbit rendezvous, was an architectural decision. So was the decision to have a separate lander to go from lunar orbit to the surface and back, rather than landing the entire spacecraft. These crucial decisions have to be made early in the design process. Making them wisely requires the participation of the best systems engineers, supported by the best discipline engineers and other specialists to provide expertise in their particular areas.

The first step in deciding on a system architecture is a functional analysis (see Chapter 10). A function is a task, action, or activity that the system must perform in order to accomplish its purpose. For a complex system, the SEs will be tasked to identify all system functions, analyze them to identify all required subfunctions, and develop a hierarchical functional architecture that documents them all. A function is identified using a verb and object. Adequately defining and describing

the requirements may require block diagrams, data flow diagrams, state transition diagrams, and so forth, depending on the nature and complexity of the system.

The second step is to define the major elements or subsystems of the system. This is the point where a particular solution starts to be defined. The decisions made here will have a fundamental effect on the outcome of the design, so they should be made very carefully. The SE defines the system boundary (see Chapter 2)—that is, what is to be regarded as part of the system being designed and what is part of the environment. For instance, in designing a cargo ship, the freight handling in ports could be defined as part of the system and subject to redesign, or as part of the environment that constrains the ship design. The SE defines the interface between the system and its environment, for instance, by defining the port facilities a ship must be able to use.

The SEs help conceptualize and analyze various system concepts to meet the need. Tools for conceptual systems design are described in Chapter 10. Systems engineers, along with relevant discipline engineers, can expect to put much effort into architectural design trade studies during this step. In these studies, they will develop models of different system architectures and evaluate how well they will be able to meet system requirements. A complex system may consist of many different physical elements in different locations, as a spacecraft system might include ground stations, communications relay stations, a mission control center, a satellite, and a launch vehicle. In other cases, a system may contain subsystems or elements that are primarily the domain of one engineering discipline, as a helicopter might have a power system, flight control system, electrical system, and data processing system. The SE is responsible for identifying such major elements and defining their relationships. To the greatest extent possible, the architecture should be selected such that each system requirement can be allocated to exactly one element. The final architecture is often documented in a hierarchical diagram called a work breakdown structure (WBS), which provides the framework for subsequent breakouts of tasks on the project.

The third and final step in system architecting is functional allocation. The system requirements are allocated to the architecture elements. Element interactions and interfaces are defined. External interfaces (between system elements and the environment) are also defined. System elements are turned over to discipline engineers for further design work, or to systems engineers for elements that themselves are unusually complex or require the integration of different engineering disciplines.

Architecting has become an subdiscipline of its own. There are several good texts that give extended advice on how to do it, including Maier and Rechtin [8].

Systems Engineering Tools and Formal Models

Some projects use standard description and documentation models, sometimes software-based. A systems engineer will usually be responsible for creating and maintaining such models. Some projects use a standard format, such as the Department of Defense Architecture Framework, or DoDAF [9]. Such a standard framework eases communication and defines several standard ways to view the system

description—for example, from the point of view of capabilities, operations, functions, or schedules. Many projects use a commercial software product to manage requirements and system models; examples are CORE® [10] from Vitech Corporation and Rational DOORS® [11] from IBM. These tools can be complicated to use and require a good deal of training and experience, but they automate much of the tedious record keeping and consistency checking that the SE would otherwise have to do by hand. They keep track of dependencies and interaction, making it easier to determine the effect of a change in one part of the system. They ensure that a requirements or configuration change made in one place is also reflected everywhere else. They can produce a draft of a System Requirements Document and other standard documents while ensuring that they are all consistent with each other.

Interface Control Documents (ICDs)

The interactions between system elements and between the system and its environment are recorded in ICDs, which are the responsibilities of systems engineers and configuration managers to write and to update as required, as described in Section 7.2. This is often one of the SE's major tasks during much of the system development period, after system architecting is complete and before system test starts, when much of the technical effort is in the hands of discipline engineers.

Test and Evaluation Master Plan (TEMP)

This document is sometimes assigned to the SE to write, and sometimes to a separate test organization. It describes all testing to be done on the project, from part and component test, to development testing of new element designs to evaluate how well they work, to systems testing of the entire system under field conditions to demonstrate that it meets user needs (see Figure 7.4). The more complex the system, the earlier it should be written.

Configuration Management (CM)

The role of CM has been described previously in Section 7.3. CM gains control of the major documents described above, the SEMP, the TEMP, and the ICDs, after the SE has written the initial versions and they are accepted by the configuration control board. The CCB will control many other documents, some of which will be written by systems engineers and some of which will only be reviewed by systems engineers.

Specialty Engineering

It is common for specialists in some or all of following areas to be part of the systems engineering organization, reporting to the chief SE. They are responsible for reviewing the entire design and development process for the impact on their areas of responsibility and for recommending areas for improvement.

- *Risk Management*. Systems that have a significant chance of total failure commonly have one or more engineers dedicated to risk management. Such systems include spacecraft, high-tech military systems, and complex systems using cutting-edge technology. Risk cannot be completely eliminated from systems like these. Risk management involves identifying and tracking the sources of risk (e.g., piece part failure or extreme environmental conditions), classifying them by likelihood of occurrence and severity of effect, and guiding risk-reduction efforts into areas with the highest expected payoff. Risk management is discussed in Chapter 3 and in Chapter 12.
- *Reliability, Maintainability, Availability* (RMA). Engineers who specialize in RMA are focused on producing a system that is working when the user needs it. Reliability refers to the likelihood of malfunction, maintainability to the ease with which the system can be serviced and repaired, and availability to the overall level of readiness for use (the result of reliability, maintainability, and spare parts availability). RMA engineers use models to calculate expected availability and recommend efforts to improve it. Chapter 8 discusses reliability models in some detail.
- *Producibility*. This is the attribute of being relatively easy and cheap to manufacture. Producibility engineering involves selecting the right design, materials, components, piece parts, and industrial processes for manufacture.
- *Quality*. The role of the quality assurance function was described earlier (Section 7.3). Quality engineers design engineering processes that produce high-quality output (i.e., items having few defects), and they design systems so that they tend to develop few defects.
- *Integrated Logistics Support* (ILS). This function encompasses the unified planning and execution of system operational support, including training, maintenance, repairs, field engineering, spares, supply, and transportation. ILS is particularly important in military systems, which often have dedicated ILS engineers from the very beginning of system development, when they are responsible for establishing logistics-related requirements.
- *Human Factors*. This specialty focuses on how the system interacts with people, particularly with users and consumers. It includes both cognitive and perceptual factors (man–machine interface, situational awareness) and ergonomics (fit, comfort, controls, etc.). Other areas of concern are workload, fatigue, human reliability, and the impact of stress. Human factors engineers specialize in the human elements of the system and their interfaces.
- *Safety*. Safety engineers are concerned with preventing not only injury and death from accidents during system manufacture and test, but also mishaps that damage high-value equipment or result in long schedule delays. Because of the importance of safety and the natural human tendency to take more and more risks when behind schedule, it is common to have a separate safety office reporting directly to the program manager or to another high-level officer.
- *Security and Information Assurance*. Engineers in these areas are responsible for ensuring that information about the program does not get to people the

customer does not want to have it, for either commercial or national security reasons. They also provide systems and procedures to protect privacy data and to protect computer systems and data from attack. Finally, they assist in designing a system so that it can operate without having information about it obtained or tampered with by others. This is a particularly important function for financial and for military systems.

- *Environmental Impact*. Major government projects often cannot be done without an environmental impact statement, and that statement is often a major hurdle. Environmental engineers will help write it and then ensure that the system is developed, built, and operated in accordance with it so that environmental impact can be kept as low as possible.
- *Independent Verification and Validation* (IV&V). Verification is ensuring that the system was built as designed; validation is ensuring that the system as designed and built meets the user's needs. An IV&V engineer is a disinterested authority (not involved in the original development) responsible for ensuring that the system is correctly designed and built. The function is especially important in software system testing. The military services also have IV&V organizations to ensure new systems meet requirements.

Major Program Technical Reviews

Chapter 3 described the various life cycle models that are used in system development. Regardless of the model, usual practice is to hold a formal review as a control point (or gate) when moving from one stage to the next. The purpose of these reviews is to allow inspection and assessment of the work, to gain concurrence and approval, and to educate the staff, the management, the customer, and the user. These reviews go by such names as system requirements review, preliminary design review, critical design review, design readiness review, and full rate production readiness review, depending on the life cycle model used and the stage of the project. Design engineers, specialty engineers, testers, quality assurance personnel, and others present the status of their work and any important open issues, as appropriate to the project stage.

A systems engineer will often be tasked with organizing and emceeing the review, as well as presenting such SE topics as requirements, risk management, and systems test. For a major system, these reviews can be lengthy affairs. An auditorium full of people will look at slide after slide of Microsoft PowerPoint, for several days. The major program decision makers (PM, chief SE, etc.) will sit through the whole thing; others may come and go based on their involvement in each topic. People often find these reviews to be of compelling interest when the topic is in one's own area of responsibility, and crushingly boring at other times.

The review will result in a list of action items that identify areas that need clarification or further work. These can range from minor points like small discrepancies between two presentations of the same data to "show-stoppers" that threaten the development. Successful completion of the review (as judged by the PM or customer) is required to enter the next stage.

System Integration and Test

When the element development work is done, it is time to put the system together and test it to make sure that all the elements work together as intended, that the system interacts with its environment as it should, and that it meets client, user, and consumer needs. These are the activities on the upper right-hand side of the systems engineering "V" (Figure 7.4). It is normally an SE's responsibility to coordinate these efforts. In acceptance testing, the customer or user should be closely involved, and sometimes runs the testing.

7.6 ROLES OF THE SYSTEMS ENGINEER

> Systems engineers can play a number of roles that may or may not align closely with their formally assigned responsibilities.
>
> —*adapted from Sheard* [12]

These are short statements of the roles often played by a designated SE, whether or not they are really part of systems engineering and whether or not they are formally assigned. This is a more subjective account of the different roles an SE as an individual may play.

Technical Client Interface. The PM often relies on the SE for dealing with the client on technical issues, when no business matters are at stake.

User and Consumer Interface. This is a primary SE job; it is part of translating possibly inchoate needs into engineering requirements.

Requirements Owner. The SE investigates the requirements, writes them down, analyzes them, and coordinates any required changes for the lifetime of the project.

System Analyst. The SE builds models and simulations (Chapter 4) and uses them to predict the performance of candidate system designs.

System Architect. The SE defines system boundaries, system interfaces, system elements, and their interactions, and assigns functions to them.

Glue among Elements. The SE is responsible for integrating the system, identifying risks, and seeking out issues that "fall through the cracks." He or she is the technical conscience of the program, a proactive troubleshooter looking out for problems and arranging to prevent them. Since many problems happen at interfaces, the SE carefully scrutinizes them to ensure that the elements do not interfere with each other.

Technical Leader. SEs frequently end up as the planners, schedulers, and trackers of technical work; sometimes the role is formally assigned by the PM.

Coordinator. SEs coordinate the efforts of the different discipline engineers, take charge of resolving system issues, chair *integrated product/process teams* (IPTs) assembled to provide cross-disciplinary oversight of particular areas, and head "tiger teams" assembled to resolve serious problems.

System Effectiveness Manager. SEs oversee reliability, availability, human factors, and the other specialty engineering areas that can make the difference between a usable and a worthless system.

Life Cycle Planner. SEs provide for such necessities as users' manuals, training, deployment, logistics, field support, operational evaluation, system upgrades, and eventual system disposal.

Test Engineer. SEs are usually in charge of overall test planning and evaluation, and of execution of system-level tests.

Information Manager. SEs often write key program documents, review all important ones, control document change, and manage system data and metrics.

7.7 CHARACTERISTICS OF THE IDEAL SYSTEMS ENGINEER

> A good SE has a systems outlook, user orientation, inquisitiveness, common sense, professional discipline, good communication skills, a desire to cooperate, and a willingness to stand up for what's technically right.
> —*adapted from SAIC* [13]

As a final look at systems engineering practice, we will describe the personality traits that make an individual a good SE. Like any other job, some people fit more naturally into it than others. While anyone with the necessary technical skills and discipline can become a good SE, people with the following characteristics will find themselves easily falling into the role, liking the job, and doing well.

Systems Outlook. A natural SE tends to take a holistic, systems-level view on problems. Other engineers may gravitate toward looking at the details and making sure that all the crucial little things are done right; many people tend to be most concerned with organizational relationships and personalities. A good SE looks at the system as a whole, considering both technical and human factors, and is comfortable leaving element details to other experts.

Client, User, and Consumer Orientation. The ideal SE has field experience relevant to the system being worked on, or at least can readily identify with the user's and customer's perspectives. The SE should feel or develop a strong affinity with the user and customer, since one of the SE's key jobs is facilitating the consumer–user–designer interfaces.

Inquisitiveness. An SE should have a natural curiosity, and should indulge it by inquiring into areas that "just don't look right." He or she wants to know as much about the system as can be absorbed by one person, and also about the design and development process. When the SE comes across something that does not seem to make sense, he or she presses the inquiry until the doubt is resolved. In this way, the SE gains a better systems-level understanding and also often uncovers problems that had escaped notice by others with more narrow responsibilities.

Intuition. A good SE has the ability to quickly grasp essentials of an unfamiliar field, and it has a good feel for what level of detail he or she should be able to understand. He or she has good judgment on when it is necessary to press a question and when it is safe to hold off and leave it to other experts.

Discipline. A good SE adheres to engineering processes, knowing that they are essential for imposing structure on the formless and that they enable both understanding of the state of progress and control of the development. This includes objectivity: The SE maintains an objective and systems-level view of the project, and it does not let him or herself become identified with any other group working on the project. A good SE will be accepted as an honest broker when there are internal disagreements.

Communication. This is essential to the SE's role as glue among the elements. It has three parts. The SE is ready to listen to everyone involved in the project, especially the users and others not in the same chain of command. The SE is also ready to talk to everyone, to make sure everyone has a common understanding of the big picture. Finally, the SE is ready to act on what he or she finds out, bringing problems to the attention of the appropriate people and getting them working together to find a solution.

Cooperation, but not Capitulation. A natural SE is cooperative and eager to get everybody working together toward a common goal. The SE works to get buy-in from all parties. However, he or she knows when not to give in to resistance. If the issue seems important enough, the SE will insist on an appropriate explanation or investigation and is willing to take the problem to the program manager (or perhaps to the Chief Technology Officer) if necessary. That is what the SE is paid for.

7.8 SUMMARY

This chapter has focused on the realities of those who have "systems engineer" in their job title. These SEs can have a great variety of jobs, but perhaps the most typical is as the technical leader and integrator supporting a program manager who is building a complex system like an aircraft or a telecommunications system. The SE will be responsible for coordinating technical efforts over the lifetime of the system, but he or she will probably be busiest toward the beginning, when he or she is in charge of defining the system requirements, developing the

system concept, and coordinating and integrating the efforts of the other design engineers.

The coordinating role of SEs means that they will work with a wide variety of other professionals and specialists, including discipline and specialty engineers, analysts, testers, inspectors, managers, executives, and so on. The SE will often have the task of forming and leading interdisciplinary teams to tackle particular problems. Other specific tasks assigned to SEs will vary with the organization and the project; they often include defining the top-level system architecture, performing risk analysis and other specialty engineering, coordinating major technical reviews, and analyzing and maintaining system requirements, system and element interfaces, and the system test plan.

Besides accomplishing these tasks, SEs may find themselves playing many different roles during system development, such as external and internal technical interface, requirements owner, system analyst and architect, system effectiveness manager, and overall technical coordinator and planner, again depending on the organization and what the program manager desires (or allows). The person most likely to enjoy and succeed at this kind of professional systems engineering is a person who naturally has a systems outlook, user orientation, inquisitiveness, common sense, professional discipline, good communication skills, a desire to cooperate, and a willingness to stand up for what is technically right. The SE is the one responsible to the program manager for making sure that all the elements work together in a system that meets the needs of the client, the user, and the consumer.

7.9 EXERCISES

7.1. Is the SE more important at the beginning of a project or at the end? Why?

7.2. Identify the client, the user, and the consumer in each of the following situations:

(a) You buy a car

(b) A taxi driver buys a taxicab

(c) A taxi company buys a fleet of taxicabs

7.3. Develop system requirements for a clock. Try to make them implementation-independent. Develop a list of functions for the clock.

7.4. Make a list of alternative system concepts for getting across a river.

7.5. Identify the functions and elements of an automobile. Draw a matrix that assigns each function to one or more elements.

7.6. Write an ICD for information exchange between a baseball pitcher and a catcher.

7.7. Explain why the performance of a newly formed task team often goes down at first, but then improves.

7.8. What factors allow a team to perform exceptionally well?

7.9. Which of the roles of a systems engineer seem to be most important to you? Justify your choice. Which is second? Third?

7.10. Research a major historical technical failure and write a three-page paper on how (or whether) better systems engineering could have averted or ameliorated it. Some examples follow; a little online research will provide a rich supply of engineering disasters.

(a) The San Francisco fire following the 1906 earthquake

(b) The sinking of the Titanic (1912)

(c) The crash of the airship Hindenburg (1937)

(d) The collapse of the Tacoma Narrows bridge (1940)

(e) The meltdown at the Three Mile Island nuclear power plant (1979)

(f) The loss of the Space Shuttle orbiter Challenger (1986) or the Columbia (2003)

(g) The flooding of New Orleans by Hurricane Katrina (2005)

7.11. Select a well-known individual from history or fiction and make a case why that person would or would not make a good systems engineer, based on his or her personality traits.

ACKNOWLEDGMENT

Dr. Paul West, United States Military Academy, provided the initial draft of the section on team building.

REFERENCES

1. Wynne, MW, Shaeffer, MD. Revitalization of systems engineering in DoD. *Defense AT&L*, 2005;XXXIV(2):14–17.
2. Haskins, C, editor. *Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities*, version 3, INCOSE-TP-2003-002-03. June 2006.
3. Katzenbach, JR, Smith, DK. *The Wisdom of Teams*. New York: HarperCollins, 1993.
4. Aranda, L, Conlon, K. Teams: *Structure, Process, Culture, and Politics*. Upper Saddle River, NJ: Prentice-Hall, 1998.
5. Harris, P, Moran, R. *Managing Cultural Differences*, 4th ed. Houston: Gulf Publishing, 1996.
6. Hull, E, Jackson, K, Dick, J. *Requirements Engineering*, 2nd ed. London: Springer, 2005.
7. Laplante, PA. *Requirements Engineering for Software and Systems*. Boca Raton, FL: CRC Press, 2009.
8. Maier, MW, Rechtin, E. *The Art of Systems Architecting*, 3rd ed. Boca Raton, FL: CRC Press, 2009.
9. DoDAF, http://cio-nii.defense.gov/sites/dodaf20/index.html. Accessed April 22, 2010.

10. CORE software, http://www.vitechcorp.com/products/Index.html. Accessed April 22, 2010.
11. DOORS software, http://www-01.ibm.com/software/awdtools/doors. Accessed April 22, 2010.
12. Sheard, SA. Systems engineering roles revisited. Software Productivity Consortium, NFP, 2000.
13. SAIC (Science Applications International Corporation). Systems Engineering Definitions & Scene Setting [briefing], 1995.