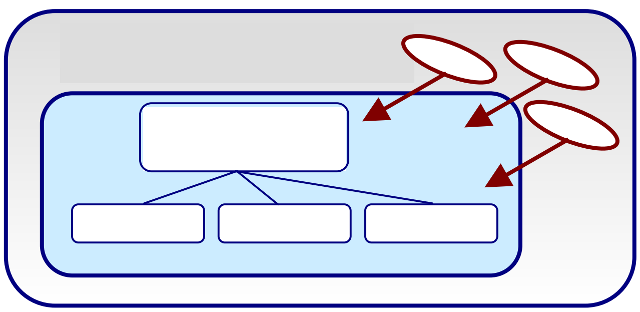
SMC

Systems Engineering Primer & Handbook

Concepts, Processes, and Techniques



Space & Missile Systems Center

U.S. Air Force

1st Edition

27 March 2003

SMC Systems Engineering

Concepts, Processes, and Techniques

Foreword

This booklet was prepared for the United States Air Force Space and Missile Systems Center. It is intended as a primer to systems engineering for your use. It is not all-inclusive and should be supplemented with Air Force and Department of Defense (DoD) directives, policies and procedures -- see http://deskbook.dau.mil/jsp/default.jsp

Preface

This Systems Engineering text is written to provide SMC personnel with fundamental systems engineering concepts and techniques as they apply to space and launch systems and the SMC environment. The intended audience is primarily the project officer, junior systems engineer, an engineer in another discipline that must perform Systems Engineering functions, or the experienced engineer who needs a suitable reference.

The authors recognize that systems engineering subject matter is very broad and approaches to performing systems engineering vary greatly. This interim release version is not intended to cover them all. This text addresses general concepts and common processes, tools, and techniques that are mostly familiar to SMC. It also provides information on recommended systems engineering practices and pitfalls to avoid. Many references are provided for the reader to consult for more in-depth knowledge.

This handbook describes systems engineering as it should be applied to the development of major space and launch systems. Systems engineering deals both with the system under development and the system that does the developing. Consequently, the handbook's scope properly includes systems engineering functions regardless of whether they are performed by the AFSPC operational User, SMC system program office (SPO), or a systems contractor.

This book is also prepared to accommodate the SMC systems engineering training program. It is written to accompany formal SMC systems engineering training courses. This book consists of 5 core chapters with each preceding chapter written to provide the foundation of knowledge to progress to the next: (1) an overview of key concepts and terms used in systems engineering, (2) an overview of systems engineering fundamentals and key definitions, (3) the systems engineering process and the end-to-end life cycle on a major space system, and (4) systems engineering tools. The basis for Chapter 5 on systems engineering management, Chapter 6 on specialty engineering integration, and Chapter 7 on validation and verification is established by the first three Chapters. The chapters are supplemented by appendices include templates and examples to illustrate topics.

Many different sources were used to prepare this book including the latest DoD Instruction and guidance on the subject, previously developed systems engineering handbooks developed for SMC, and a number of engineering publications that are cited throughout this book.

Finally, this text should be considered only a starting point. The SMC environment is undergoing rapid evolution. Over the next few years, the SMC Systems Engineering Revitalization (SER) initiatives will undoubtedly induce many changes to the conduct of engineering and acquisitions at the Center.

The release of this interim version will be followed by another version in October 2003. The October version will contain additional content as described in Appendix D. This version will also be updated to capture new Systems engineering related policies and initiatives as they pertain to SMC. AXE is soliciting your suggestions and content contributions to the October 2003 version. Therefore we provide a Customer Review & Feedback Form in Appendix E for your submissions to Mr. Dave Davis david.davis@losangeles.af.mil or Barry Portner bportner@tor.bdsys.com .

Acknowledgements

This work was conducted under the overall direction of Mr. Dave Davis, SMC Engineering

Many individuals contributed material reviewed various drafts, or otherwise provided valuable input to this handbook and it's predecessor – The GPS Systems Engineering Handbook. Our sincere apologies if the list below does not include others who provided content, edited, or shared their ideas and experience.

Principal Contributors:

Guy Hoshour

Barry Portner

Bill Crabtree

Brian Sullivan

Alan Adams

Rod Bennett

John Kern

Claude Kitchens

Dar Johnson

Bill Berrier

Jon Mathis

Kamran Shaik

Prepared by

CHAPTER 1

What is a System or a System of Systems?

System

A system is a set of interrelated elements which interact with one another in an organized fashion toward a common purpose. (Some naturally occurring organisms and phenomena – such as weather – can usefully be considered as systems.) The purpose of most military systems, often called weapon systems, is to provide a useful operational capability to the military forces. Some military systems may be used for operational support such as for training, testing, or characterizing the environment in which the forces and equipment must operate. Space systems include satellite systems (such as the Global Positioning System), launch systems (such as the Atlas V and Delta IV), and terrestrial control systems (such as the Air Force Satellite Control Network). The elements of a system may be quite diverse, consisting of hardware (equipment), software, people, data, and facilities.

The hardware or equipment and its installed software typically includes development and manufacturing tools and test sets, operational elements (to provide the needed capability), maintenance and support elements (to keep all elements working), and training elements (to train people in the use of the operational, maintenance, and other elements). The data include the procedures to manufacture, operate, and maintain the system and to responsibly dispose of expendables or equipment no longer needed after their use in development, manufacturing, operations, and maintenance. Facilities include control centers, launch pads, test and training facilities, and connecting roadways and power distribution.

A personal computer (PC) is an example of a system. The PC system which is partially shown in Figure 1 includes a processor, display, keyboard, and mouse. The processor in turn consists of the motherboard, connectors to attach peripheral elements, the case etc. The motherboard is further made up of parts and materials. Fasteners and electrical connectors join (or interface) the elements together. The soldering iron in Figure 1 is symbolic of the manufacturing and maintenance equipment that complete the system.

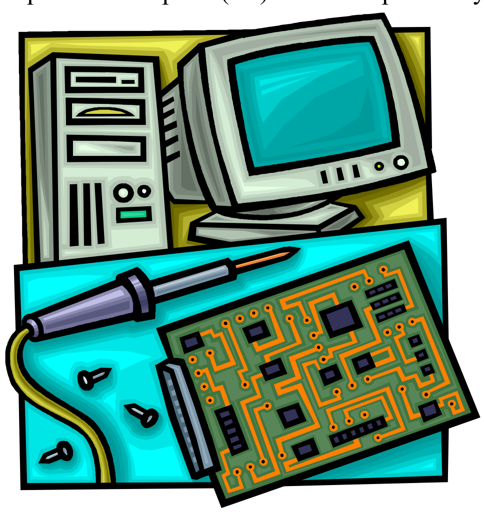


Figure 1. The Personal Computer System

Similarly, a space system might be made up of a satellite (or satellites), ground elements to control and maintain the satellites, and User elements that permit the operational military forces to take advantage of the capabilities that the space system has to offer. Each satellite is made up of its elements, typically the payload (that provides the basic mission capability such as communications, surveillance, navigation, etc.) and the spacecraft or bus (that typically supports the payload by providing electrical power, thermal control, attitude control, etc.). The payload and bus can, of course, be usefully subdivided into lower tier elements.

System of Systems

Most modern systems operate in the context of a broader system of interrelated systems. Thus each system must not only operate individually to provide the needed capability but must typically *interface* with a number of other systems. Such systems must be engineered and evaluated in the system-of-systems context. As a result, managers of a system of systems, on the advice of their systems engineers, establish capability objectives, set policies, determine constraints, and address the costs applicable to each individual system. In general, such policies, constraints, etc. define requirements that each system must meet. Accordingly, the system-of-systems managers may have oversight authority over the design and operational decisions for each system. An example is shown below.

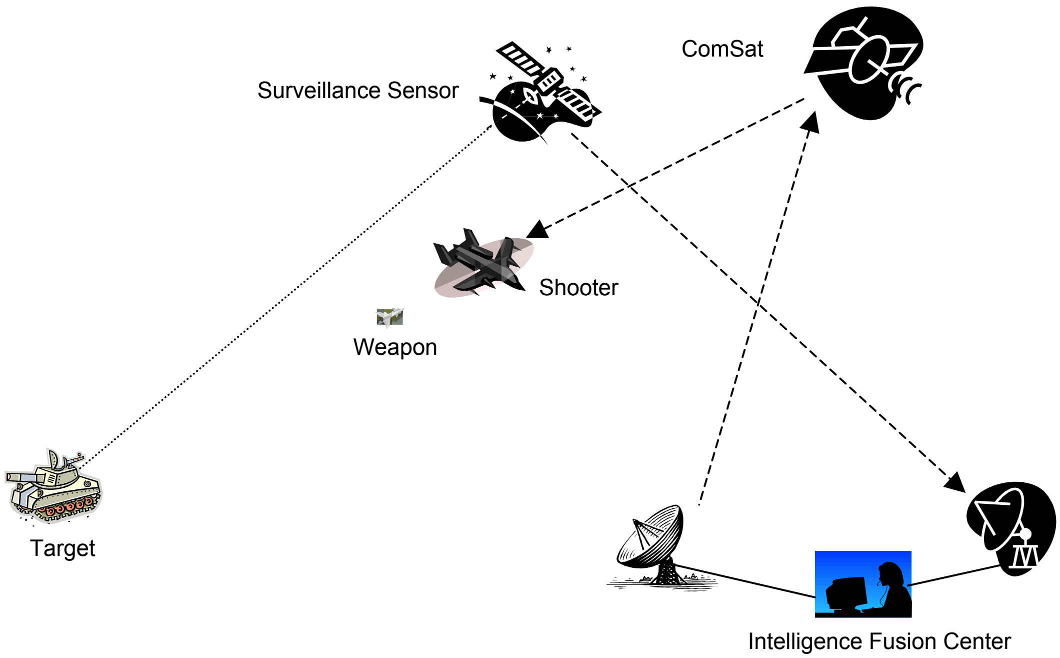


Figure 2. A System of Systems – Sensor to Intelligence Fusion Center to Shooter to Weapon

How does a military system come to be?

Modern military systems don't occur naturally or by happenstance. They result from extraordinarily complex processes involving a number of iterative steps, usually over many years. First, initial capabilities to be provided (or requirements to be satisfied) are defined. Then, multiple concepts to provide the capability (including maintenance and training) may be developed and evaluated to assess capability performance, affordability (cost), schedule, and risk. The evaluations may lead to refinements in the capabilities to be provided, further concept development, and, ultimately, the selection of a preferred concept. If the cost and risks are viewed as acceptable, an *acquisition program* may be initiated to complete development of the selected concept. The equipment that must be developed includes not only the operational elements to provide the capability but also the equipment to *train* the operational personnel and to *maintain* and *support* the operational equipment over its life cycle. Over the course of development, equipment design and software development is followed by verification that developmental items provide the needed capabilities. If successful, limited production of the equipment is typically followed by operational testing to verify that the operational and maintenance equipment with its installed software together with operations and maintenance instructions provides the desired capability in the intended operating environments. If the system proves acceptable, production continues and is followed by deployment of the operational and maintenance equipment to the operational military units along with support equipment and initial spare parts to a logistics center (depot).

In most cases, there is no known synthesis approach that can accomplish these steps based on first principles. Instead, some steps are often iterative and the entire process may be recursively applied before the system is ready for operations. Further, initial military capabilities often evolve through *incremental* or *spiral* acquisition processes to provide the full operational capability – as a result the capabilities to be provided are defined for time-phased increments or spirals. These processes are supported by other complex processes that provide the necessary budgets along with controls to prevent fraud, waste, and abuse of public funds.

The Department of Defense (DoD) has four interactive management systems that implement these processes. The Requirements Generation System[1] oversees the definition of the capabilities (or requirements) that are to be satisfied – it is directed by the Vice Chairman of the Joint Chiefs of Staff. The Defense Acquisition System[2] oversees the research, development, test and evaluation, production, and deployment activities that provide new capabilities – it is managed by the Under Secretary of Defense for Acquisition, Technology, and Logistics, USD(AT&L). The budget for each program is developed within the DoD through the biennial Planning, Programming and Budgeting System (PPBS)[3] which is managed by the Undersecretary of Defense. **All three of these systems are supported by systems engineering activities.** After approval by in the PPBS, the budgets are submitted by the President to the Congress for the annual Authorization and Appropriation of public funds. After public funds are appropriated by the Congress, they are managed by the DoD Financial Management System.[4]

What are System Capabilities or Requirements?

The new or upgraded capabilities that are to be provided (or requirements to be satisfied) can arise from a wide range of DoD activities. These activities generally fall in two broad categories. The first consists of opportunities created by new and evolving technologies from the science and technology (S&T) developed by OSD and the military services – the Air Force S&T program is carried out by the AF Research Laboratory (AFRL) – and by academic, industrial, commercial, and international sources. Such situations are sometimes called technology push or opportunities push. The second type of activity giving rise to new capabilities consists of operational problems or challenges which may be identified during training, exercises, operational testing, or military operations. Such capabilities are sometimes referred to as technology pull, operational pull, or operational challenges. Either category can give rise to formal statements of needed capabilities or requirements through a wide range of planning activities: strategic, operational, budget, or development planning (the latter is now called capability planning by the Air Force). As noted above, the operational capabilities to be provided are defined by the Requirements Generation System. Usually, the operational capabilities to be provided are translated into verifiable technical or engineering requirements by the System Program Office (SPO). AS a result, the capabilities or requirements are documented over time in a variety of ways or views – see the discussion of the Requirements View below.

---

1. Chairman of the Joint Chiefs of Staff Instruction 3170.01 Series, *Requirements Generation System.*
2. Interim DoD Instruction 5000.2, *Operation of the Defense Acquisition System.*
3. DoDD 7045.14; The Planning, Programming, and Budgeting System (PPBS); (Including Change 1); 28 July 1990
4. DoD Financial Management Regulation (FMR).

What is the System Environment?

All systems must operate in an environment defined by Mother Nature and can affect that environment. In addition, most military systems must be capable of functioning in a combat environment in which they are being directly attacked or in which the natural environment has been modified by intentional or unintentional enemy or friendly activity. Thus, a key aspect of the systems engineering process described in this handbook is to define the *constraints* imposed on the system by the operating environment. These include such factors as temperature extremes, humidity, salt water spray (for ocean-bound equipment), and the like for the natural environment. The effect of the system on the environment is typically constrained by public law and by governmental regulations that implement the law – these must also be identified by the systems engineering process. The definition of the constraints imposed by the combat environment usually start with a *system threat assessment* which may be conducted by the DoD intelligence community. But such constraints may also come from other sources. For example, public law requires that many systems be exposed to live-fire testing as part of the verification that they will provide the needed capability in the combat environment – such may give rise to the requirements both for features that facilitate the testing as well as very specific survivability features. In general, the requirements analysis and functional analysis and allocation activities (that are introduced later in Chapter 2 below and described in more detail in subsequent chapters) are systematic processes to define all of the constraints imposed by the operating environment.

What are Interfaces?

As noted under the discussion of System of Systems above, systems usually do not operate alone. The nature of the relationship between two systems as well as between subordinate elements of a system is called the interface. When the interface for a new system is to an existing system, the interface is a *constraint* on the design of the new system. Even when systems or system elements are designed in parallel but by separate design organizations, a point is reached in the development process where the interface eventually becomes a constraint on each design. As a result, interfaces are usually viewed as constraints having an effect on the design akin to that of the constraints imposed by the system environment as discussed just above.

Interfaces can be physical or functional. Physical interfaces include definitions of the means of attachment (bolt patterns, connectors, fasteners, etc.) and keep-out volumes. Functional interfaces include electrical, radio-frequency, and software.

As examples of interfaces, the Personal Computer discussed above usually must interface with a number of other systems including the power system to which it connects, other equipment such as a printer, and adaptor printed circuit cards such as those that provide for connection to the Internet or other networks. All of these involve both physical and functional interfaces.

As an example of formal documentation for an interface, the interface between two systems managed by different organizations – such as a satellite system and a launch system – may be captured in an interface specification or in an Interface Control Drawing or Document (ICD). Interfaces that are managed by a single organization may simply be captured in the design drawings.

Some interfaces have become *standards* used throughout an industry or even throughout much of the world. For example, the physical interface between the Personal Computer and the printer may be via a cable that meets the requirements of a standard parallel interface specified by a standards-issuing organization such as the Institute for Electrical and Electronics Engineers (IEEE) or one that has become a *de facto* standard such as those followed by certain cable manufacturers. Because of the problems that can result from informally defined interfaces, the interfaces for most military systems are defined by specifications, ICDs, or standards published by independent standards organizations.

As another example of an interface, a space launch system may use a liquid fuel and oxidizer. To achieve the planned performance, the liquids must meet certain requirements for purity, density, stability, etc. Such "interface" requirements are usually defined in specifications or standards to help ensure the needed launch performance.

What are the Views of a System?

Systems engineering must simultaneously consider a wide range of aspects of a system from the capabilities that the system is to provide and the environment in which it must operate to the system architecture to the design and physical implementation of the system. To facilitate doing this, a number of ways of looking at a system have been devised – this handbook will call these system views. Some of the views that have proven useful include Capability/Requirements, Functional, Physical, and Work.[5] As you read the following descriptions and subsequently apply various views in systems engineering work, an important point for you to keep in mind is that all of these views are abstract and therefore incomplete. Only the final physical implementation and its performance in thorough, realistic operational testing approach being complete definitions of the system and the capabilities it provides to help the military forces accomplish their mission.

Capability/Requirements View

The capability/requirements view is usually first shown via capabilities or requirements descriptions for which the preparation is led by the military operational community operating as part of the Requirements Generation System. As this handbook is being prepared, the formal process in the DoD is undergoing a change from a "requirements" oriented approach to a "capabilities" oriented approach. In the new approach, the process is to start with an analysis of doctrine, organization, training, materiel, leadership, personnel, and facilities (DOTMLPF) in an integrated, collaborative process to define desired capabilities to guide the development of a system or system of systems. Early in the development of a new capability, broad, time-phased operational goals and the requisite capabilities are to be described in the Initial Capabilities Document (ICD). Subsequently, the Requirements Generation System refines the capabilities to be provided by preparing the Capability Development Document (CDD) to support the initiation of a formal acquisition program.

For on-going programs based on the requirements oriented approach, the Requirements Generation System initially developed a Mission Need Statement (MNS) and then prepared an Operational Requirements Document (ORD) which is usually iteratively updated over the system life cycle.

The operational capabilities or requirements must usually be translated into verifiable technical or engineering requirements and completed by defining the additional requirements necessary to complete the system life cycle at reasonable cost and risk. The resulting technical requirements are usually

---

5. The Defense Acquisition System applies three related views, especially in joint programs for the acquisition of systems to be used by more than one military service or defense agency: operational, systems, and technical, as defined in the current Architectural Framework guidance. The Joint Staff leads development of the operational view to describe the joint capabilities that the user seeks and how to employ them. USD(AT&L) leads development of the systems view to characterize available technology and systems functionality. The systems view shall identify the kinds of systems and integration needed to achieve the desired operational capability. The Military Departments and Defense Agencies participate in the identification of the appropriate technical view consisting of standards that define and clarify the individual systems technical and integration requirements. The standards used to form the Technical Views of integrated architectures are selected from those contained in the current approved version of the Joint Technical Architecture.

formalized in a System Requirements Document (SRD), a Technical Requirements Document (TRD), or a system specification and associated ICDs or interface specifications.

Systems Engineering support to the Requirements Generation System in developing the ICD and CDD as well as the systems engineering necessary to complete the SRD, TRD, or system specifications is covered in Chapter 3 starting with Requirements Analysis.

Functional View: Functions and Functional Flow Block Diagram

To provide an operational capability, a number of functions must be carried out. A function can be thought of as a task to be performed to achieve a capability or satisfy an operational need or requirement. Functional views come in many forms. A common starting point to defining a functional view is the eight primary lifecycle functions from which all else can be derived:

- development, verification, production (and construction), training, deployment (or fielding), operations, support, and disposal.

An often used functional view is the functional flow block diagram (FFBD) supplemented by other devices such as timing diagrams that identify the timing relationship between functions. The FFBD shows order in which the functions must be carried out to provide the capability. The figure below shows the eight primary lifecycle functions organized into a simple, generic top-tier FFBD.

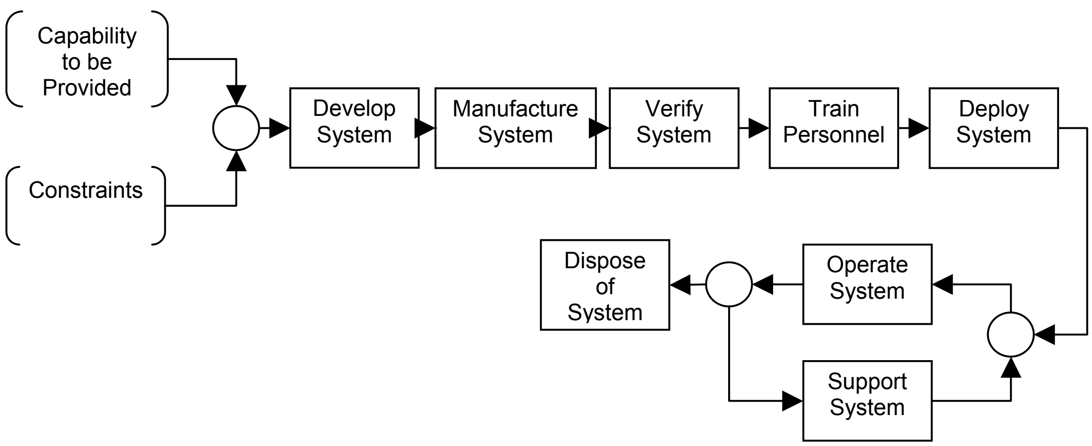


Figure 3. A top level functional flow block diagram showing a typical or generic relationship among the eight primary life cycle functions

When the functional view is used, the top tier flow shown above is refined to apply specifically to the system at hand and then each of the primary functions is further decomposed until the associated technical requirements can be directly associated with and allocated to the selected physical view of the system. This process is called functional analysis and allocation. The Functional Analysis section of Chapter 3 provides an overview of operational, system, and technical (interface) functional views and functional analysis approaches. See Appendix C.4, Techniques of Functional Analysis, for a more comprehensive treatment of functional analysis.

Physical hierarchical views of a system

Several physical views have been found useful: hierarchical or tree diagrams (that show the physical elements of the system to some level of detail), concept block diagrams (that typically show both key physical elements and a summary of the interfaces between those elements); and engineering drawings along with the procedures for manufacture, assembly, test, and deployment.

An often-used physical view is the product tree which shows the hierarchical relationship among the elements that make up the system. A simple example for a satellite system is shown in the figure below.

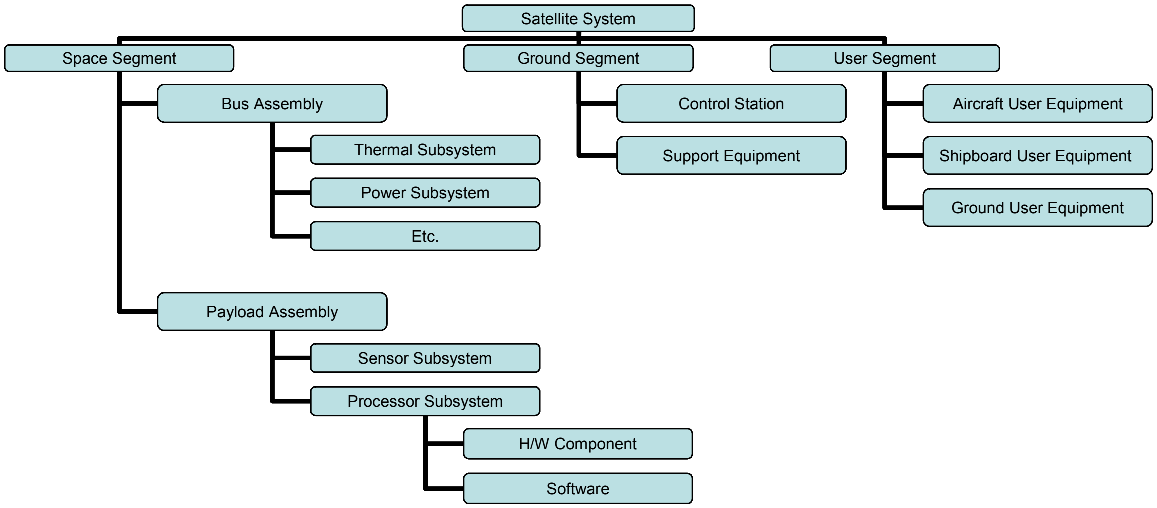


Figure 4. A simple satellite system product tree

The figure shows terms that are often used in describing and referring to various levels in a product tree. For example, the system is considered to be made up of segments. In the simple example, there is less nomenclature commonality at the next level as assembly, station, and equipment are all used. The assembly level is made up of subsystems while components are among the descriptions that might be used at the next level. Note that software is included in the example product tree wherever it is installed in a next higher level product. On the other hand, some programs have used trees that are a mix of products and functional specialties – in such trees, software may be collected and shown at a higher level. In programs that use the product trees to assign responsibility, authority, and accountability (RAA) for a complete element (such as the processor subsystem shown in the example), software should be placed as shown in the example so that the element can be fully verified to meet its requirements *in the system*. In addition, placing software where it logically falls in the system facilitates the allocation of requirements down the product tree to each element. A complete product tree is extended to the point that each element is identified that is either to be developed or produced by a different organizational element or that will be supported, inventoried, or procured as a separate element over the life cycle.

The product tree can be easily extended to provide other views. For example, by adding identifiers for the corresponding specifications for each system element, the tree in the figure becomes a specification tree. The documents defining interface constraints, whether contained in interface specifications or ICDs, can then be added to the specification tree to identify the interface constraints.

System product and specification trees evolve in detail and are refined as the system is developed using the system engineering process discussed further in Chapter 3, particularly under synthesis. Eventually, as synthesis progresses and matures, the specification tree and the associated specifications often become formally controlled – see system analysis and control in Chapter 3 and configuration management in Chapter 5.

Work view

A work view summarizes the work to be done to complete some activity in the life cycle of the system. One important work view is the product-oriented work breakdown structure (WBS) which uses the product tree as a point of departure and defines via a hierarchical outline and a set of definitions the work necessary to complete the next phase of activity in the system life cycle.[6] Using the product tree shown above as a point of departure, a simple, partially populated, WBS outline is shown in the table just below.

Table 1. A simple satellite system product-oriented Work Breakdown Structure

Level 1	**Level 2**	**Level 3**	**Level 5**	**Level 5**
00000 System	1000 Space Segment	1100 Bus Assembly	1110 Thermal Control	
			1120 Power System	
		1200 Payload Assembly	1210 Sensor	
			1220 Sensor Processor	1221 Sensor Processor Hdwr
				1222 Sensor Processor S/W
	2000 Ground Segment	2100 Ground Control Station		
		2200 Support Equipment		
	3000 User Segment	3100 Aircraft User Equipment		
		3200 Ship User Equipment		
		3300 Ground User Equipment		
	4000 Program Management/ Systems Engineering	4100 Program Management		
		4200 System Engineering	4210 Requirements Analysis	
			4220 Functional Analysis	
			4230 System Analysis and Control	
			4240 Synthesis	
			4250 Specialty Engineering	

6. You may also see another form of the WBS called the functional WBS which is a hierarchical outline of the functional specialties required to complete the activities. The product-oriented version is usually much more useful for systems engineering activities.

In the example WBS, required work not directly associated with a single product such as program management and systems engineering has been added to that for the products. Each entry has been given a number that can facilitate the creation of related views of the work such as a Statement of Work which defines the scope of the work for each entry in the WBS (down to a selected level) for a development contract. A more detailed product-oriented WBS and the associated definitions are shown in Appendix C.2

A related work view is the Integrated Master Plan (IMP) which identifies the significant accomplishments to be completed by each major event or milestone (such as a design review or major test) for each major entry in the WBS. Such IMPs can be sorted to show the work to be completed for each event and then by WBS or to show the work to be completed for each WBS. When the WBS entries can be assigned one-to-one to an organizational entity (such as an Integrated Product Team or IPT), IMPs sorted by WBS define the accomplishments to be completed by each organizational entity.

The IMP can also include a description of the processes that guide completing the significant accomplishments or such process descriptions can be in other documents such as the Systems Engineering Management Plan (SEMP). The IMP can, in turn, be extended to the Integrated Master Schedule (IMS) which shows the tasks needed to achieve each significant accomplishment on a calendar schedule. And finally, the tasks in the IMS can be extended to the work packages in an Earned Value Management System (EVMS) that define the resources planned to complete each task.

CHAPTER 2

What is Systems Engineering?

There are many definitions for Systems Engineering, but most have a common thread as an interdisciplinary process. According to the Defense Systems Management College (DSMC) text *Systems Engineering Fundamentals,* January 2001, Systems Engineering is "an interdisciplinary engineering management process that evolves and verifies an integrated, life-cycle balanced set of system solutions that satisfy customer needs." The systems engineering process is applied in a program or project that is established to complete the evolution and verification of the system solution. This is an intentionally broad definition that encompasses not only the work of people who may be called systems engineers but also the remaining total engineering effort associated with design, manufacture, coding, and testing of the system solution. In other words, systems engineering as used here means engineering of the system. For this reason, it is usually necessary to distinguish between the systems engineering process (in which almost all engineering personnel involved in a program participates) and the systems engineering organization (which may monitor the operation of the systems engineering process as a staff function for the program manager and carry out some of the steps in the process).

The systems engineering process is tightly coupled to the Life Cycle Phases of a Major System addressed in Chapter 3. As the systems engineering process progresses from concept through development, production, deployment, use, up-grading, support and finally, disposal, the systems engineers continuously re-evaluate and update designs, requirements, and operations in light of the latest feedback arising from the conduct of the process itself. The program constraints that shape the process are real-world factors such as cost, schedule, performance, risk, available technology, user capabilities, and governing legal requirements. Basically, systems engineering is a common sense approach to engineering.

The process has both engineering and management aspects. The engineering involves the analysis, synthesis and evaluation of candidate solutions. The technical management involves the orchestration of inputs from various sources and disciplines, and the tracking and documentation of the process.

An Orderly Process

Systems Engineering (Figure 5) is a methodical approach to achieving an optimum product to meet the customer's need, given the various constraints and influences that affect the product throughout its operational life. The five major activities of the process are:

Requirements Analysis — the evaluation of the need and translation to system requirements.

Functional Analysis and Allocation — the identification of the functionalities required to achieve

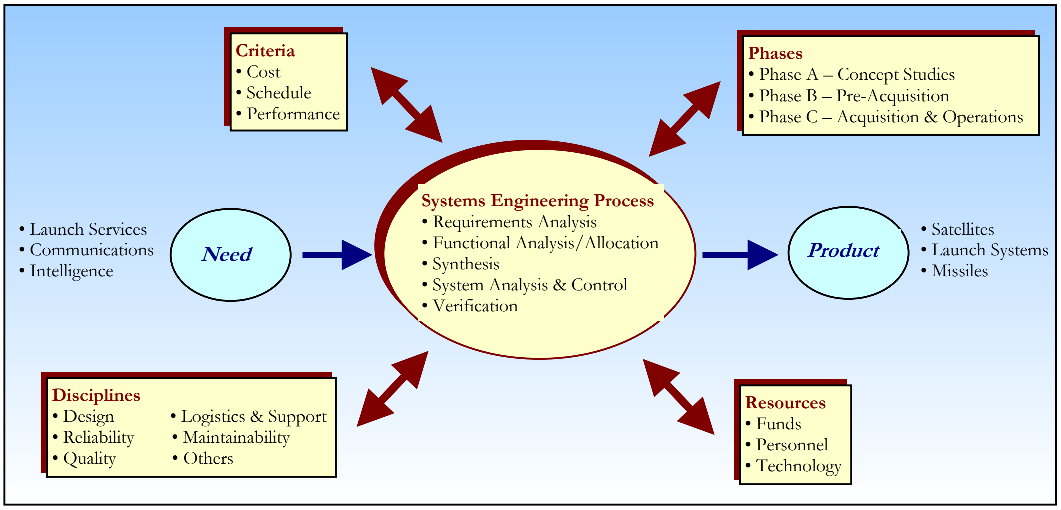


Figure 5. The System Engineering Process — Interacting to Provide Optimum Products

the system requirements and the allocation of requirements to these functions.

Synthesis — the development of candidate system and component solutions to meet the established requirements.

System Analysis and Control — the analysis and evaluation of the results and other activities to effect their coordination and refinement. Also, the management of technical efforts between disciplines and documentation of results.

Verification — the tasks, actions, and activities to be performed and the system elements required to evaluate progress and effectiveness of evolving system products and processes and to measure specification compliance.

In pursuing this process, Systems Engineering interacts with a variety of influences which mutually affect each other and modify the process. As shown in Figure 5, these influences are:

Programmatic Criteria — The programmatic criteria of cost, schedule, performance and risk place constraints on the process. In particular, technical constraints can place definite bounds on the system. Generally, a requirement that does not have to be derived and cannot be modified or dropped as a result of a trade-off study is considered a constraint. As the program progresses and the approaches mature, these criteria limit the options available. Interaction with program management to modify these criteria may be necessary should a particular approach be desirable.

Phases — The role of Systems Engineering evolves as the program moves through its phases. Early on, it is heavily involved in the analysis, allocation and synthesis activities. As designs mature, it engages in verification and documentation activities. Later in the program, its main activities are tracking configurations and sustaining operations. In all phases it has the responsibility to control and coordinate the other activities.

Disciplines — From the earliest stages of the process, Systems Engineering interacts with a variety of specialty disciplines to assist in the development of the optimum product. Design operations indicate possible alternate implementations of candidate systems. Research develops new technology or devices necessary to accomplish program goals. Manufacturing assures that the product is producible and develops the required processes for efficient production. Quality, Reliability and Maintainability analyze and suggest modifications to system and component designs to assure that necessary requirements are met. Integrated Logistics Support (ILS) identifies factors affecting the supportability of the product and develops logistics systems that will assure that the proper personnel, data and support hardware and software are available throughout the life cycle. Cost Estimating develops and exercises cost models to provide indications of the cost implications of various design options and tracks actual costs as the program progresses. Personnel Subsystems assures that any man-machine interfaces are ergonometrically sound. Systems Engineering orchestrates the inputs and outputs through these interfaces to assure that all disciplines are properly represented in the on-going design, that all are aware of any activities by others that may affect their areas, and that all design decisions reflect the best compromise between opposing requirements.

Resources — All projects are bounded by the funds, time, personnel and technology available. To some extent, these factors may be traded off between each other. For example, the development time can be extended to await new emerging technology, or additional or more experienced personnel may be obtained with a greater infusion of funds. Many of these factors are beyond the control of the Systems Engineer, but his inputs are valuable to the ultimate decision maker in that he can help define the possibilities.

The following paragraphs provide more detail on how the various factors affect Systems Engineering.

A Bounded Process

The initial impetus to the program is the customer's need. The need is couched in terms of what the customer wishes to accomplish (mission) and where he/she intends to accomplish it (environment) and when it is needed (schedule). At this point, questions such as "Is it feasible?" or "What will it cost?" are not raised by the customer; these are questions to be answered by Systems Engineering in the early stages of the program. As these questions are answered, the customer may be asked to modify his/her original requirements. It may be appropriate to determine if "a system that meets most of your needs but can be delivered sooner or at less cost is worthwhile."

The customer’s needs initiate the Systems Engineering process, but the criteria help to bound it. The programmatic factors of cost, schedule, performance and risk define the scope of the effort and are the basis for an evaluation of whether meeting the needs is a reasonable undertaking given the constraints. The customer cannot expect a Cadillac on a golf-cart budget or major technological breakthroughs on a shortened schedule.

As with many factors encountered in Systems Engineering, there are tradeoffs in the criteria. On one project the overriding requirement may be to deliver the product as quickly as possible. In this case higher funding, added risk, or even slightly diminished performance may be appropriate. On another project, performance is the driving requirement with cost, schedule, and risk being of lesser import. For each project, the balance of these criteria helps define the choices available to the Systems Engineers. Are we building the Taj Mahal or a lean-to?. . . today's need or tomorrow's expected requirement? . . .Indy 500 race car or family sedan? . . . black and white TV or high-definition color? Helping to establish the balance between the customer's needs and the program criteria of cost, schedule, performance and risk is one of the most important early tasks in the Systems Engineering process.

Table 2 lists the constraints that are commonly imposed at the upper and lower boundaries of the program criteria. Because there is interaction between the criteria, a single criterion does not completely define the program. For example, a low cost program does not always mean higher risk if the performance envelope is not being pushed and proven technology is being employed. Also there is a natural point for the criteria for any given program and going far below or beyond this natural point may have adverse effects. For instance, extending the program schedule of a program tends to reduce its cost since it allows more efficient use of resources and possible use of more effective evolving technology. However, at some point schedule stretch-out becomes costly as the expense of maintenance of facilities not fully used grows in respect to the overall program cost. Similarly, lower technology tends to go with lower cost, but not taking advantage of the latest proven techniques will drive up the cost. It is the responsibility of Systems Engineering to help the final decision makers understand the totality of the constraints being imposed by the criteria. However, once the criteria have been established, Systems Engineering is expected to work within those constraints. If it becomes apparent as the program develops that this is not possible, Systems Engineering must determine what is possible and then seek to renegotiate the criteria or the customer's original requirements into something which can be achieved.

Table 2. Characteristics of the Program – *Criteria Impose Bounds on the Systems Engineer*

Criterion	Characteristics of the Program at Criterion Limits	
Cost	**Low**	**High**
	• Lower Performance • Normal Schedule • Possibly Higher Risk • Older or Lower Technology	• Higher Performance • Possibly Shorter Schedule • Possibly Lower Risk • Newer or Higher Technology
Schedule	**Short**	**Extended**
	• Lower Performance • Possibly Higher Cost • Possibly Higher Risk • Older or Lower Technology	• Higher Performance • Possibly Lower Cost • Possibly Lower Risk • Newer or Higher Technology
Performance	**Low**	**High**
	• Possibly Shorter Schedule • Lower Cost • Lower Risk • Older or Lower Technology	• Possibly Longer Schedule • Higher Cost • Possibly Higher Risk • Newer or Higher Technology
Risk	**Low**	**High**
	• Possibly Higher Cost • Possibly Longer Schedule • Possibly Lower Performance • Older or Lower Technology	• Possibly Lower Cost • Possibly Shorter Schedule • Possibly Higher Performance • Newer or Higher Technology

An Evolutionary Process

As the program evolves, so does the role of the Systems Engineer. In the early phases, the emphasis is on analyzing and characterizing the customer needs, converting them into system requirements and developing system constructs that satisfy these requirements. Then candidate systems are evaluated against the established program criteria and the most promising selected for further analysis. Additional functional details are developed for the candidates and system requirements are flowed down to these functions. Possible hardware and software implementations of the functions are examined to determine fit with the requirements and criteria. The data from these investigations are reassembled at the system level for further candidate system screening. This iterative process continues throughout the life of the program. After one or two system architectures are selected, Systems Engineering is involved in the proof of concept and the direction of research in areas critical to the selected candidates. When the process settles on a final point design, Systems Engineering takes the leading role in developing and overseeing tests and analyses which verify that the selected system meets the customer needs and the program criteria. In the midst of this process, Systems Engineering ensures that the system configuration is documented and that subsequent changes are properly recorded in the data. As the system progresses through production, deployment and use, Systems Engineering is responsible for assuring that all production and field installation details are properly authorized and recorded and that they are consistent with the driving system requirements. The Systems Engineer must be sensitive to his changing role and to his responsibility for maintaining the overall integrity of the system design.

An Interdisciplinary Process

Systems Engineering does not operate in a vacuum. It must consult continually with a variety of disciplines, each interpreting the requirements and criteria from their particular viewpoint. Each holds Systems Engineering's feet to the fire for their area of expertise…Is the design feasible?…Is new technology required?…Is it producible?…Can quality be maintained?… Can it be supported in the field and are support costs reasonable? The specialty disciplines answer these questions and suggest alternate approaches where requirements are not being met. Systems Engineering ensures that the evolving design represents the optimum compromise among these often conflicting requirements.

The role of the Systems Engineer has been enhanced by the advent of Integrated Process and Product Development (IPPD) and the formation of Integrated Product Teams (IPTs). The use of IPTs emphasizes the importance of all disciplines in the development process and fosters closer cooperation between the Specialty engineers and the Systems Engineer and between each other. It promotes understanding of how the decisions of one participant affects the purposes of the others and makes for a consensus design that incorporates innovation in achieving the best design to meet the customer's need.

A Practical Process

In all things Systems Engineering must be pragmatic. The Systems Engineer is not a theorist. A Systems Engineer must propose real world solutions within the allotted funds and time, recognizing the capabilities of the personnel who will develop, operate, and maintain the system, and the technology available for use in the design and in the production of the resulting system. The Systems Engineer must first understand the problem before designing the solution. Too often engineers want to define the solution first and then find the problem the solution fits. According to James Martin, in his book, *Systems Engineering Guidebook*, "Systems Engineering is more properly concerned with the *engineering of systems* than with merely definition of the requirements and architecture for such a system. It is also not merely concerned with analysis of performance and effectiveness at the "system level" of a system hierarchy. Systems Engineering is really about common sense." As H. L. Mencken said, "There is always an easy solution to every human problem – neat, plausible, and wrong."

A Cost Effective Process

Systems engineering is performed in unison with program management. The system engineer's role includes providing assistance to the program managers to make the best decisions. The objective of systems engineering is to see to it that the system is designed, built, and operated so that it accomplishes its purpose in the most cost-effective way possible, considering performance, cost, schedule, and risk. A cost-effective system must provide a particular kind of balance between effectiveness and cost: the system must provide the most effectiveness for the resources expended or, equivalently, it must be the least expensive for the effectiveness it provides. The systems engineering process requires characterization alternative concepts/designs in terns of cost, performance, and risk of those concepts to support assessment of the alternatives. A cost-effective system provides a balance between value and cost. The system should provide the most value for the resources expended or, equivalently, it must be the least expensive for the effectiveness it provides. This condition is a weak one because there are usually many designs that meet the condition.

Cost

The cost of a system is the foregone value of the resources needed to design, build, and operate it. Because resources come in many forms: personnel and contractors, materials, energy, and the use of facilities and equipment such as wind tunnels, factories, offices, and computers—it is of en convenient to express these values in common terms by using monetary units (such as dollars).

Effectiveness

The effectiveness of a system is a quantitative measure of the degree to which the system's purpose is achieved. Effectiveness measures are usually very dependent upon system performance. For example, launch vehicle effectiveness depends on the probability of successfully injecting a payload onto a usable trajectory. The associated system performance attributes include the mass that can be put into a specified nominal orbit, the trade between injected mass and launch velocity, and launch availability.

Cost-Effectiveness

The cost-effectiveness of a system combines both the cost and the effectiveness of the system in the context of its objectives. While it may be necessary to measure either or both of those in terms of several numbers, it is sometimes possible to combine the components into a meaningful, single-valued objective function for use in design optimization. Even without knowing how to trade effectiveness for cost, designs that have lower cost and higher effectiveness are always preferred.

ElA/IS-632 defines Systems Engineering as "an interdisciplinary approach encompassing the entire technical effort to evolve and verify an integrated and life-cycle balanced set of system people, product, and process solutions that satisfy customer needs. Systems engineering encompasses (a) the technical efforts related to the development, manufacturing, verification, deployment, operations, support) disposal of, and user training for, system products and processes; (b) the definition and management of the system configuration; (c) the translation of the system definition into work breakdown structures; and (d) development of information for management decision making."

CHAPTER 3

How Does the Systems Engineering Process Work?

The Systems Engineering process is a series of repetitive operations whereby a universe of possible solutions to a stated need are narrowed to a single system that optimally satisfies the need. It is a continual excursion between the general and the specific… top down, bottom up… to propose solutions, check their possible implementation, and then propose new or modified solutions to be checked again. Even the most talented Systems Engineer cannot initially identify the optimum solution with certainty. "What worked before" is the obvious starting point, but if existing systems met all the requirements, there would be no need for a new system. In fact, with the present emphasis on evolutionary design under DoD 5000.1, one of the most important questions the Systems Engineer should ask is, "Can these requirements be satisfied using existing or slightly modified systems?" If the answer is yes, the customer's needs can be met much sooner and at lower cost. There is no need to reinvent the wheel!

Iteration, Iteration, Iteration

The Systems Engineering process is shown schematically in Figure 6.

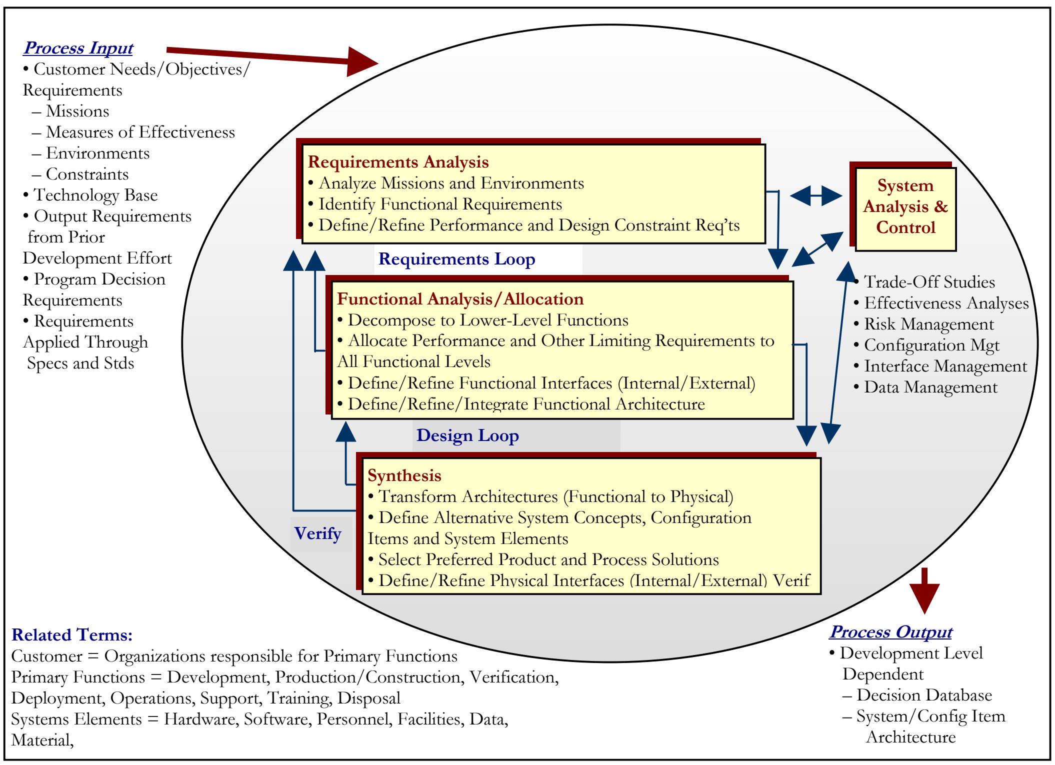


Figure 6. The System Engineering Process — Iterations to Determine the Optimum Solution

The customer's needs, objectives and requirements in terms of missions, measures of effectiveness, environments, and constraints initiate the process. Mission need statements indicate what is to be accomplished and how the system will be used. Measures of effectiveness quantify the results to be obtained and may be expressed as probabilities that the system will perform as required, e.g., the chance that a certain event will be recognized with a certain probability and that the probability of false alarm is below a certain percent. Environments refer to operating environments, i.e., space, airborne, ground, marine, or submarine. Internal environments, e.g., whether a particular system solution requires air conditioning or cryogenic cooling, are for the Systems Engineer to specify; it is of no consequence to the customer if the solution falls within the overall constraints and requirements. Customer-imposed constraints usually take the form of costs and schedules. However, some secondary constraints such as types of personnel to be utilized in operation and maintenance of the system, and fit with existing support systems may also be included.

The technology base and prior outputs are natural inputs to the process. Any good Systems Engineer builds on what has gone before. However, in analyzing existing technology for use on the current program, the System Engineer must identify critical areas where proof of the use of the technology in the given application is required. This may indicate the need for additional research.

Program decision requirements are another real-world check of the practicality of the program. Only mad scientists and billionaires can continue on their pet programs when all signals point to failure. The requirements may range from the formal multiple milestone procurements of the Government to informal program reviews, but somewhere along the line it is necessary to review cost projections and identify minimum progress required to warrant continued operations. By specifying these decision requirements the customer indicates what is necessary to maintain his interest and funding.

Specifications and standards imposed on the program should be tailored to the application. Of all the inputs to the Systems Engineering process, this is the one most likely to undergo repeated review and negotiation. Until designs begin to mature, detailed tailoring is difficult. Early on, care should be taken to ensure that potential cost savings, such as commercial off-the-shelf (COTS) solutions to the problem, are not eliminated. Every effort should be made to identify the major cost, schedule, performance and risk drivers contained in the imposed specifications and standards, and those which can, should be tailored or eliminated.

The major activities of the Systems Engineering Process are *Requirements Analysis, Functional Analysis and Allocation, Synthesis,* and *System Analysis and Control*. There is continual interaction and feedback among these activities and refinement of their outputs as the program progresses.

The initial interaction is through the Requirements Loop. The results of the mission and environments analysis and the identification of functional requirements are the input to the decomposition to lower level functions and the allocation of the requirements to the lower functions. As these analyses and allocations are accomplished, the results are fed back to the requirements analysis to verify their compliance or to determine whether modification of the requirements is compatible with achieving the mission.

The **Design Loop** operates in parallel with the **Requirements Loop**. Functional interfaces are established and functional architecture defined so that physical system configurations can be developed. As concepts are transformed to hardware and software designs, the design characteristics are analyzed against the allocated requirements. Functional architectures and allocations are re-examined and modified if necessary. Some results of the Design Loop may even reflect into the Requirements Analysis necessitating further re-evaluation.

The final feedback "loop" is the verification of the emerging detailed design against the originating requirements. This may be accomplished by analysis, simulation, demonstration, proof testing of critical components, or a combination of these. Note that verification can be interpreted as a loop or a process, and different authors have treated it different ways. For this Handbook, verification is considered to be a process, but there are certainly iterative aspects to the process that have the characteristics of a loop. What matters is that verification is accomplished thoroughly and correctly.

The System Analysis and Control activity functions as the planner, manager, judge, traffic cop and secretary of the process. This activity identifies the work to be performed and develops schedules and costs estimates for the effort. It coordinates the other activities and assures that all are operating from the same set of agreements and design iteration. It evaluates the outputs of the other activities and conducts independent studies to determine which of alternate approaches is best suited to the application. It determines when results of one activity require the action of another activity and directs the action to be performed. It documents the results of analyses and studies, maintains control of the evolving configuration, and measures and reports progress.

The output of the System Engineering Process is a decision database and a balanced system solution.

The database documents include:

- the design,
- all the decisions made to arrive at the design,
- defining specifications,
- verification requirements, and
- traceability of design features to imposed requirements, constraints, specifications and standards.

The balanced system solution is the best fit to all the final requirements and criteria imposed.

In the remainder of this chapter, we will look in more detail at the efforts involved in the four basic activities of the System Engineering Process. Sub-activities are identified to aid the discussion and to highlight specific efforts and outputs. However, they are not meant as isolated operations that toss their outputs "over the wall" for someone else to process further. Such activities are highly interactive and often performed by the same Systems Engineer. In fact, because they are usually so closely connected, at any given time the System Engineer may have difficulty determining on which he is working. This does not vitiate their value as discussion points and, on large programs they may, in fact, be conducted as separate operations.

Requirements Analysis

The Requirements Analysis is one of the first activities of the System Engineering Process and functions somewhat as an interface between the internal activities and the external sources providing inputs to the process. (The insert in the upper right of Figure shows the relationship of Requirements Analysis to the other Systems Engineering activities previously presented in Figure 2.) It examines, evaluates, and translates the external inputs into a set of functional and performance requirements that are the basis for the Functional Analysis and Allocation. It links with the Functional Analysis and Allocation to form the Requirements Loop of the System Engineering Process.

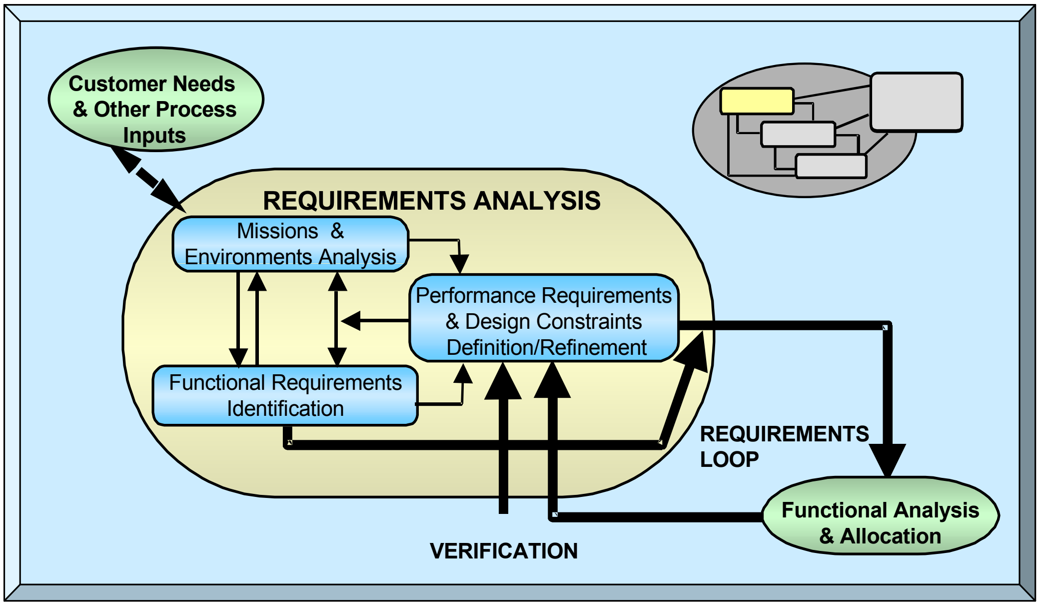


Figure 6. Requirement Analysis - Converting Customer Needs into System Requirements.

The activities of the Requirements Analysis are shown in Figure 6. The Missions and Environments Analysis firms the customers needs and states them in terms that can be used to establish system functions, performance requirements and design constraints. The output of this activity initiates Functional Requirements Identification and the Performance/Design Requirements Definition and Refinement. As these activities progress, the original assumptions and conclusions are checked against evolving details. Usually this results in some modification of the original thinking, and may even reflect back to the customer's needs where certain ones may be impractical or excessively costly. The output of the Requirements Analysis is a set of top-level functional definitions and accompanying performance and design requirements which become the starting point of the Functional Analysis and Allocation. The Requirements Loop serves to refine the requirements and initiate re-evaluation to determine how firm the requirements are for items that prove to be major cost, schedule, performance or risk drivers. Later in the overall process, detailed system characteristics are compared against the established requirements to verify that they are being met. At this point there is usually little change to the requirements due to the verification feedback, but occasionally some minor changes are considered when the payoff is significant.

Detailed descriptions of the activities of the Requirements Analysis are provided below.

Missions and Environments Analysis — The Systems Engineer helps the customer refine his needs, objectives, and measures of effectiveness in light of the initial and evolving results of the Requirements Loop. Questions such as, "What is the minimum/maximum operating time required to accomplish the mission? Are alternate existing capabilities available to provide backup?: are posed and answered. Needs that are design drivers are identified and characterized as desirable or mandatory. Constraints that limit solutions are identified and defined in detail, e.g., mission or utilization environments (extremes of heat or cold, or continuous on-line operation) or adverse impacts on natural or human environments (pollution or radiation). While this analysis is performed early in the process, it is not a once-and-for-all activity. Throughout the life of the program, the validity of mission and environmental requirements are analyzed and assessed for mission deficiencies and are revisited whenever they exhibit adverse impact on cost, schedule, performance, or risk.

Quite often customers define requirements as “thresholds” or “goals.” Thresholds are minimum requirements customers need to perform their missions. Goals are advanced qualities that provide added benefit. Achievement of a threshold is of utmost importance, since the customer has indicated he may not be able to perform the mission without it. Goals are less critical and the System Engineer should make the customer fully aware of any cost, schedule, performance or risks involved in their attainment before proceeding. Find out if the customer is willing to accept the added penalty associated with the benefit. Maybe it makes sense to put the goal on hold for later implementation. This is the customer’s choice, but the System Engineer has an obligation to provide all the information necessary to make that decision.

Functional Requirements Identification — The major functions that the system needs to perform are identified and the appropriate system-level attributes (requirements) are assigned to them. In this activity, a system hierarchy is established and a system-level specification tree developed. Where a system attribute involves more than one function, the requirement is apportioned over the affected functions. For example, to achieve the overall system reliability or weight, each functional element is assigned a specific reliability and weight requirement. Similarly measures of effectiveness are allocated to the functions. In some cases, a derived attribute is assigned to a function because the system-level attribute cannot be allocated directly. An example might be system measurement accuracy, which must be translated into such requirements as receiving function probability-of-detection, transmitting function stability, and all other functional requirements that contribute to system measurement accuracy. The assembly of all allocated or derived functional requirements must equate to the originating specific and overall system requirements, and the traceability of functional-to-system requirements must be recorded and maintained. Individual requirements must be characterized in terms of the degree of certainty in their estimate, criticality to system success, and relationship to other requirements. Again, this is not a one-time process. Re-balancing of functional requirements may be necessary when system requirements change or when analyses indicate that requirements assigned to a specific function might be more advantageously met in another.

Performance Requirements and Design Constraints Definition/Refinement — The mission/environments analysis and the functional requirements identification result in an initial set of performance requirements and design constraints assigned to major system functions. In the Functional Analysis and Allocation activity, this set is further divided and allocated as the first step in arriving at specifications suitable for the acquisition of hardware and software, and for recruiting and training of necessary personnel. These requirements are documented in a System Requirements Document (SRD). As this process of decomposition to lower levels progresses, the nature and validity of the original assignment of attributes to the functions is more fully understood. With this understanding, more efficient or effective functional divisions and requirements assignments may become apparent, necessitating a reassessment and modification of the original assumptions of the Requirements Analysis. This feedback completes the Requirements Loop.

Functional Analysis and Allocation

The Functional Analysis and Allocation bridges the gap between the high level set of system requirements and constraints (from the Requirements Analysis) and the detailed set required (in Synthesis) to develop or purchase systems and implement programs. It is an integral part of both the Requirements Loop and the Design Loop. (See insert at top right of Figure 4.) During this activity, an integrated functional architecture is defined in sufficient depth to support the synthesis of solutions in

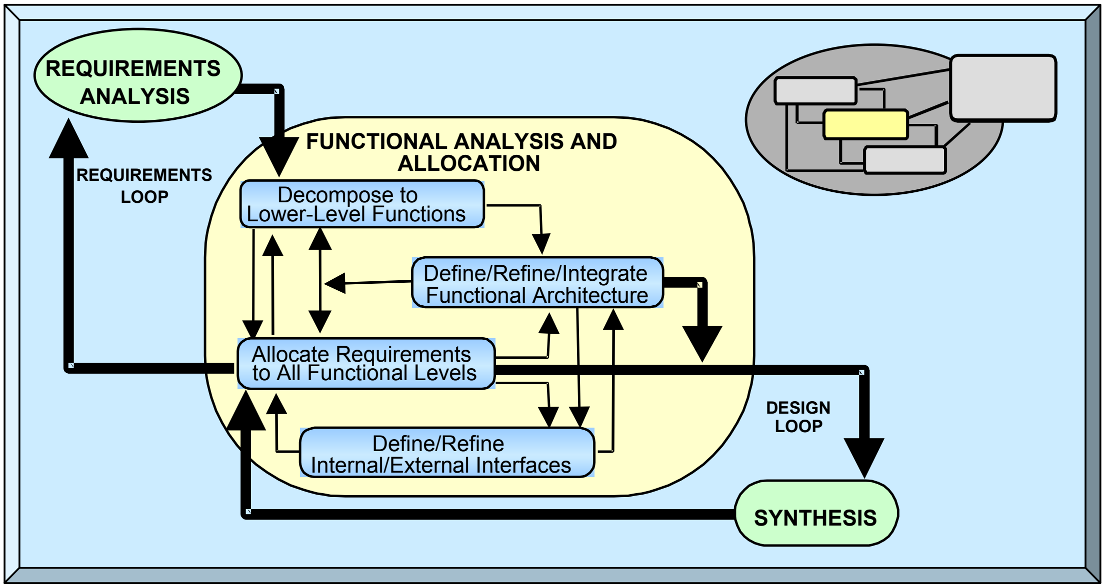


Figure 7. Functional Analysis and Allocation – Creating the Lower Level Requirements that Allow Synthesis of Solutions.

terms of people, products, and processes, and to allow identification and management of attendant risk. It is an iterative process, interacting and reacting to the on-going activities in the both the Requirements and Design Loops.

The initial step (Figure 7) is to identify the lower-level functions required to perform the various system functions. As this is accomplished, the system requirements are allocated and functional architecture(s) are developed. These activities track and interact so that as details evolve, they are continually validated against each other. Should anomalies occur — for example, GPS user equipment signal processing might require greater receiving sensitivity — or should a different decomposition appear more advantageous — say detection might be more easily accomplished with increased processing rather than greater signal strength, then re-evaluation of the driving requirements might be undertaken. Decisions may not be clear-cut. Consequently, alternate architectures and allocations may be carried through early stages of this activity until the optimum approach becomes apparent. The internal and external functional interfaces are defined as the architecture matures. The functional architecture(s) and their companion functional requirements are the input to the Synthesis activity. Completing the Design Loop, the detailed results of the Synthesis are compared to the candidate architecture(s) and allocated requirements to help zero in on the optimum approach and to assure that all proposed solutions meet established requirements.

Detailed descriptions of the activities of the Functional Analysis and Allocation are provided below.

Decomposition — Decomposition to lower-level functions is the incoming interface for the Requirements Loop. The functions identified in the Requirements Analysis are analyzed to define successively lower-levels of functions that accomplish the higher-level functional requirements. Alternate lower-level functional solutions covering all anticipated operating modes are proposed and evaluated to determine which provides the best fit to the parent requirements and best balance between conflicting ones. The initial decomposition is the starting point for the development of the functional architecture and the allocation of requirements to the lower functional levels. Adjustments to the decomposition strategy may be necessary as details are developed.

Allocation — All requirements of the top-level functions must be met by the aggregate of those for all lower-level functions. This is often difficult to prove when an upper-level performance requirement is achieved through a number of derived requirements. (For instance, system accuracy is composed of derived functional attributes that in sum determine its value.) Consequently it is extremely important not only that higher-level requirements are allocated properly, but also that **traceability** to the originating requirement, and rationale for the allocation be recorded and maintained. Traceability is an on-going record of the pedigree of requirements imposed on system and subsystem elements. Expressed in terms of "parents" and "children" and recorded on a suitable database, Traceability allows the System Engineer to ascertain rapidly what effects any proposed changes in requirements may have on related requirements at any system level.) Because requirements are derived or apportioned among several functions, they must be traceable across functional boundaries to parent and child requirement. Design constraints defined in the Requirements Analysis must also be flowed down to the lower functions. The allocated requirements must be defined in measurable terms, contain applicable go/no go criteria, and be in sufficient detail to be used as design criteria in the subsequent Synthesis activity.

Time dependent operations are also allocated to the functions. If the total time required for the system to perform an operation is critical, the time allowed for each function to perform its portion of the process must be allocated and the sequence specified. For each sequence, the characteristics of the inputs and outputs between functions must be identified.

In completion of the Requirements Loop, as the functional allocations are established they are continually evaluated against the original requirements. In addition, the functional allocations are one of the criteria used in parallel activities of functional architecture and interfaces definition. If required, the allocations may be modified as a result of these activities. In some cases this may reflect into reassessments of the Requirements Analysis results.

The allocated requirements along with the associated architecture form the input to the Synthesis activity. Results of the Synthesis are validated against the allocated requirements and occasionally necessitate re-allocation.

Functional Architecture — The functional architecture defines how the functions will operate together to perform the system mission. Generally, more than one architecture can satisfy the requirements. Usually each architecture and its set of associated allocated requirements have different cost, schedule, performance, and risk implications. Not only is it difficult at this point to ascertain which is the optimum solution, it is usually prudent to carry along low-cost, low-risk, lower-performance alternatives as insurance in case the higher-performance solution proves not feasible, too costly, or not possible to achieve in time for the need. In the Design Loop, synthesized designs are compared with the originating architectures and allocated requirements to assure compliance or to initiate re-evaluation. Figure 8 is an example from the NAVSTAR GPS Program of a second level functional architecture depicted as a functional flow block diagram.

Inherent in the process of establishing the architecture is the definition of the boundaries of the various functions and subfunctions. This leads to the definition of the internal and external interfaces.

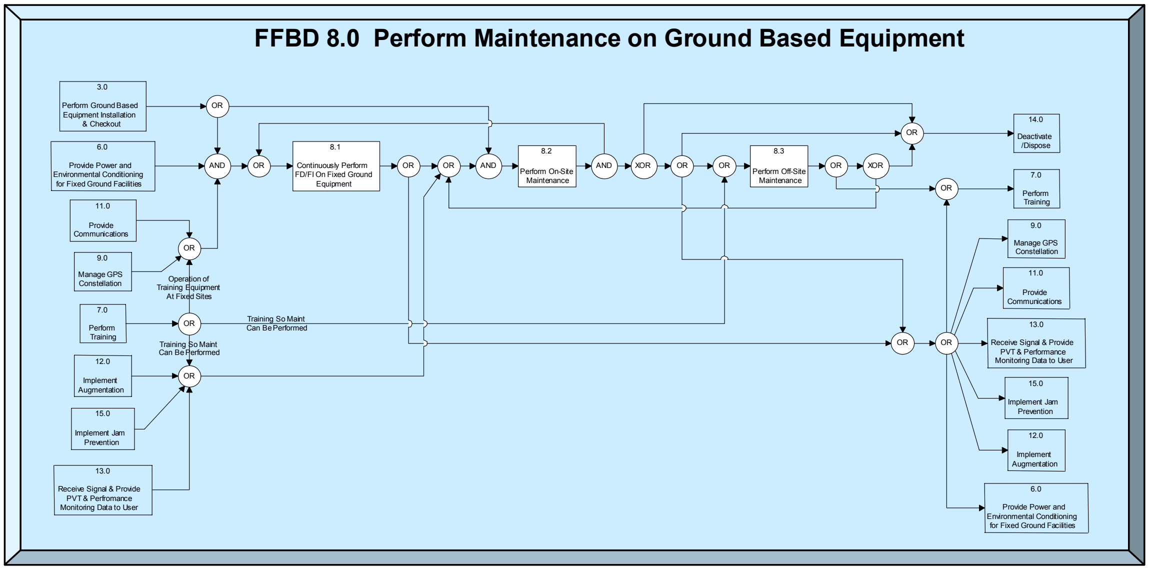


Figure 8. Functional Architecture – An example from GPS Program

Interfaces — System interfaces are both physical and functional. Included are such items as physical support and connectivity as well as the "gozintas" and the "gozoutas." Interface control is a Systems Engineering activity that begins in parallel with the development of functional architectures and continues for the life of the program. An exact accounting of interface status and change records is essential to an effective program. Not only is this important from an engineering standpoint, but these records are also the basis for letting contracts and subcontracts during the development and production phases. Without accurate interface accounting, components may not play together when the products of diverse suppliers are assembled into a system.

Interfaces also have an impact on many of the specialty engineering disciplines. Specific examples of specialties impacted by interfaces include:

- **Producibility** – The ability to produce the product may depend upon how it is housed, packaged and interconnected.
- **Maintainability –** The ease of fault isolation and repair depends upon modularity and the physical location of interfaces.
- **Logistics** – Provisioning is highly influenced by commonality and costs of modules.
- **Support** – Supply, training, documentation, and support equipment requirements depend on the match of the new system to existing support systems.

For these reasons, specialty disciplines are highly concerned with the way interfaces are drawn and specified. Interface requirements are incorporated into the functional architectures used by the Synthesis activity.

Synthesis

Synthesis is the process whereby the functional architectures and their associated requirements are translated into physical architectures and one or more physical sets of hardware, software and personnel solutions. It is the output end of the Design Loop. As the designs are formulated, their characteristics are compared to the original requirements, developed at the beginning of the process, to verify the fit. The output of this activity is a set of analysis-verified specifications which describe a balanced, integrated system meeting the requirements, and a database which documents the process and rationale used to establish these specifications.

The first step of Synthesis (Figure 9) is to group the functions into physical architectures. This high-level structure is used to define system concepts and products and processes, which can be used to implement the concepts. Growing out of these efforts are the internal and external interfaces. As concepts are developed they are fed back in the Design Loop to ascertain that functional requirements

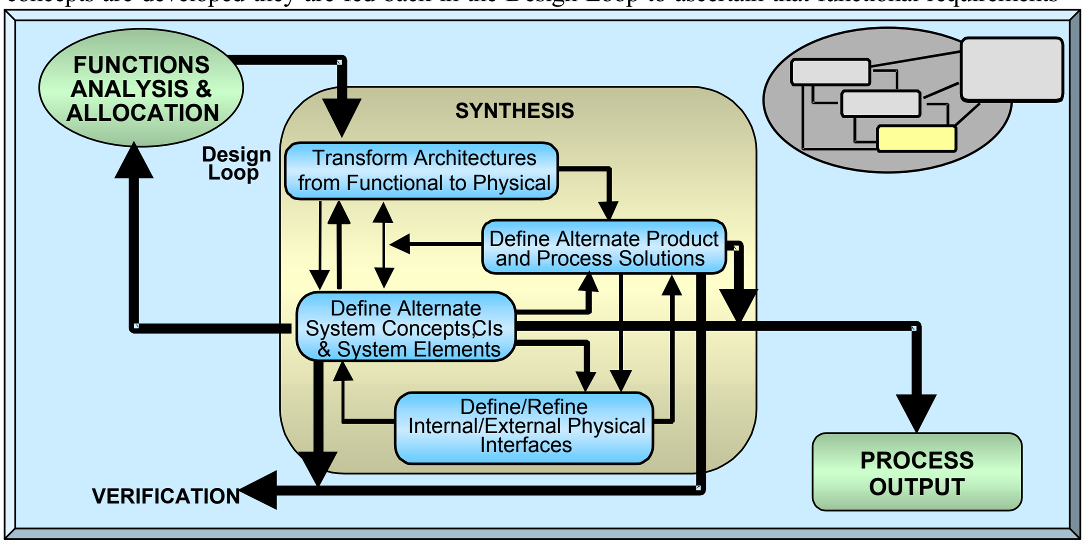


Figure 9. Synthesis – Developing Detailed Solutions

have been satisfied. The mature concepts, and product and process solutions are verified against the original system requirements before they are released as the Systems Engineering Process product output.

Detailed descriptions of the activities of Synthesis are provided below.

Architecture Transformation — Until this point, the emphasis has been on identification of functions with lesser consideration of how they may be implemented. For each set of inputs from the Functional Analysis and Allocation, like functions are grouped together to form major physical system elements, an integrated physical system architecture is developed, and the interaction of the elements established. As a part of this process, the completeness and adequacy of the input functional and performance requirements are established and if additional ones are necessary, the Functional Analysis and Allocation is revisited. The physical architectures are used as the basis for defining system concepts. Data fed back from the concept development may result in "tweaking " of the architecture.

In the development of physical architectures (and composite physical and functional architectures) it is important to retain and enhance any open-systems features built-in during Functional Analysis and

Allocation. Failure to do so may result in sub-optimized design, loss of opportunity to incorporate on-going technology advancements or replacements during development or subsequent sustainment, and even reduce the effective life of the system. Recent emphasis has been placed on open systems architectures. Such architectures facilitate use of COTS solutions for system implementation, later incorporation of advanced or replacement technologies, expansion of system capabilities, and interoperability with exiting or prospective related systems. The flexibility provided by open systems architecture during all phases of system development, recommends its consideration in making all Systems Engineering decisions.

C4ISR Architecture Framework -- The principal objective of the C4ISR architecture framework is to define a coordinated approach for DoD architecture development, integration, and presentation. The framework is intended to ensure that architecture descriptions can be compared and relate across organizational and system boundaries. In February 1998, the DoD Architectural Coordination Council mandated the use of this framework for all C4ISR architecture descriptions. It behooves the architectural system engineer to understand this methodology.

The framework prescribes three views of an architecture: operational view, system view, and technical view. The operational view is a description of tasks and activities operational nodes, and informational exchange between nodes. The system view is a graphical and textual description of systems and interconnections used to satisfy operational needs. The technical view is the minimum set of rules governing the arrangement, interaction, and interdependence of system parts and elements. Refer to Appendix C-7 C4ISR Architecture Framework for more detailed discussion on the C4ISR subject.

Through iterations of the Design Loop, some architectures are discarded because they do not satisfy the requirements, because others satisfy them more completely, or because they are solutions that differ only slightly or offer little advantage over others. For those few that survive, the architectures are used to derive/refine work breakdown structures (WBSs), specification trees, specifications, and configuration baselines that define the system and the effort needed to develop it. For the verified design(s), these defining documents become part of the Process Output database.

Alternate System Concepts and Elements Definition — The elements of the various architectures must be developed in sufficient detail to permit verification of the design against the requirements and constraints of the Requirements Analysis, and to eventually lead to detailed system design. In defining system implementation concepts, functions are assigned to "black boxes" which will be the subsystems and components that will be developed to perform the system functions. Functions might be distributed among several black boxes. Likewise, there may be several ways in which the boundaries of each box are defined, i.e., pre-amplifiers might be mounted with an antenna, or included in a receiver. Consequently several system implementations are usually proposed and further analysis performed in the Design Loop to determine which best fits the requirements.

Another important aspect of this activity is identification of the critical parameters of each alternate concept, and the sensitivity of the concept's performance, cost, schedule or risk to each parameter. The sensitivity may weigh heavily in trade studies performed in the System Analysis and Control activity and may help decide which concepts are carried further in the Design Loop.

The output of this activity is integrated logical sets of systems, configuration items (CIs), and system element solutions. As they are developed, they are evaluated repeatedly in the Design Loop to shake out those that do not meet the requirements. The remaining sets are further verified to arrive at the optimum solution(s). The concepts are handed off for definition of the interfaces and product/process solutions. Results from these parallel activities are fed back to refine the system concepts.

Physical Interfaces Definition — This a continuation and extension of the work began in the Functional Analysis and Allocation and is the foundation of the Configuration Management operations that continue through the life of the program. The functional and physical characteristics of the inputs and outputs at the boundaries identified during Synthesis activities must be identified and documented in a set of Interface Control Documents (ICDs). In addition to this accounting, methods must be established for tracing requirements across the interfaces and aggregating them as necessary to permit comparison with the original driving requirements and constraints resulting from the Requirements Analysis.

Again, this activity has both engineering and legal ramifications. The interfaces are an important factor in establishing contracting and subcontracting agreements and in assuring that items made by various suppliers play together as a system.

The interface definition is iterated as the system concepts are developed, and as alternate product/process solutions are defined. For each surviving system definition, the associated final set of interfaces is included in the database of the process output.

Alternate Product and Process Definition — Just as there are several ways to implement system configurations, there are also many ways in which these configurations may be accomplished. The Alternate Product and Process activity addresses such questions as the use of COTS (commercial off-the-shelf) products versus new or modified development, LSI (large scale integration) versus discrete or hybrid circuitry, human versus machine operations, and new versus existing technology. As alternates are developed, design simplicity approaches are incorporated to take maximum advantage of standardization, modularity, existing support equipment and facilities, and production techniques. Much of the output of system concept definition activity is fodder for the cost/benefit and risk analyses performed as part of the System Analysis and Control (Figure 11).

Another major consideration in this activity is the determination of how much automation to incorporate. Where the man-machine interface is drawn may cause large variations on the workloads on both sides of the interface. This could have considerable impact on the cost, performance, schedule and/or risk of alternate configurations. Many times the decision is deferred until later in the program. Costs of automation for all possible configurations may be prohibitive, so human operations may be incorporated during the concept demonstration phase of the program with the idea of automating later when the system has been defined in more detail.

The Alternate Product and Processes activity reacts interactively with the architecture development, systems concept definitions, and interfaces definition activities. Where appropriate, the results, complete with all applicable tolerances and variables, are included with the associated system concept in the process output database.

As described earlier, Systems Engineering has both technical and management aspects. One of the management tasks of the Synthesis function is developing a Work Breakdown Structure (WBS), which is used in managing the development of the system described in Synthesis.

Work Breakdown Structure (WBS) — The WBS is a means of organizing system development activities based on system and product decompositions. It is a product-oriented family tree composed of hardware, software, services, data, and facilities, which result from systems engineering efforts during the development and production of the system and its components, and which completely defines the program. The WBS is prepared from both the physical and system architectures, and identifies all necessary products and services needed for the system. This top-down structure provides a continuity of flow down for all tasks. Enough levels must be provided to properly define work packages for cost and schedule control purposes.

Because the WBS is a derivative of the physical and systems architectures, it is a direct output of the systems engineering process. It can also be considered part of the synthesis process since it helps to define the overall system architecture. The DSMC *Systems Engineering Fundamentals* Book, December 2000, includes the WBS in the System Analysis and Control process as a tool to help represent and control the overall process. The WBS is not just about hardware or software but also is used to structure development activities, identify data and documents, organize integrated teams, and is used for non-technical program management purposes such as scheduling, and measurement of progress.

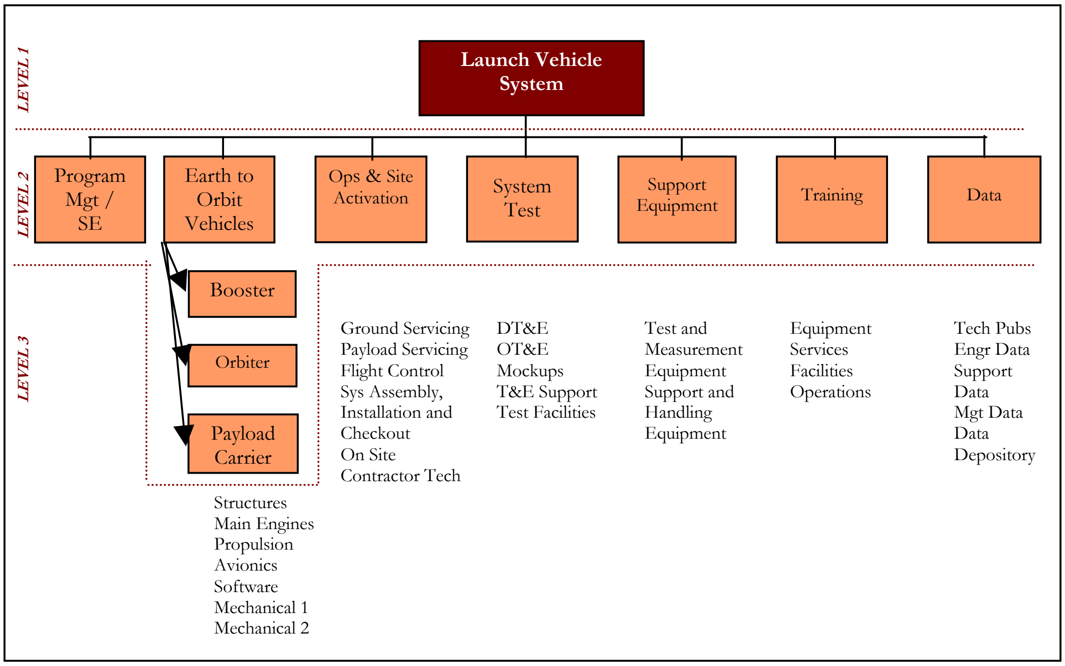


Figure 10. Launch System Work Breakdown Structure (WBS)

A program WBS is established to provide the framework for program and technical planning, cost estimating, resource allocation, performance measurement, and status reporting. The WBS defines the total system of hardware, software, services, data, and facilities, and relates these elements to each other and to the end product. Program offices develop a program WBS tailoring the guidance provided in MIL-HDBK-881. The WBS is also an integral part of preparation of the Cost Analysis Requirements Description (CARD). A sample WBS of a launch system is shown in Figure 10. Program Offices usually have the responsibility to develop an overall program WBS and to initiate development of contract WBSs for each contract in accordance with common DoD practice established in MIL-HNBK-881. The program WBS is the WBS that represents the total system and, therefore, describes the system architecture. The contract WBSs are part of the program WBS and relate to deliverables and tasks on a specific contract. The Program Office with the support of systems engineering develops the first three levels of the program WBS, and to provide contractors with guidance for lower-level WBS development. As with most standards and handbooks, use of MIL-HDBK-881 cannot be specified as a contract requirement. Though WBS development is a systems engineering activity, it impacts costing, scheduling and budgeting professionals, as well as contracting officers. An integrated team representing these stakeholders is needed to support WBS development.

The first three Work Breakdown Structure Levels are organized as:

- Level 1 – Overall System
- Level 2 – Major Element (Segment)
- Level 3 – Subordinate Components (Prime Items)

Levels below the first three represent component decomposition down to the configuration item level. In general, the government is responsible for the development of the first three levels, and the contractor(s) for levels below three.

System Analysis and Control

System Analysis and Control is the welding that holds all the other Systems Engineering Process activities together, the steering wheel that gives them direction, and the map that shows where the process is going and where it has been. It is the activity that spans the whole life of the program. It involves the initial analysis of system requirements to prepare the work views discussed in Chapter 1, the management of the activities shown in those views and their interactions, the review and measurement of work progress, and the documentation of work actions and results.

System Analysis and Control (Figure 11) interacts with all the other activities of the Systems Engineering Process. (Because it is so extensive, this interrelationship has been mentioned only briefly in the previous discussions of the other activities to allow a more comprehensive review at this point.) The initial analyses performed in this activity are the basis for the Systems Engineering Management Plan (SEMP) and the systems engineering entries in the Integrated Master Plan (IMP) which define the overall systems engineering effort. The SEMP is a process-oriented document, which describes what has to be done; the IMP is event oriented, identifies the significant accomplishments to complete each event, and defines the criteria for successful completion of each accomplishment. From the SEMP and IMP, the Integrated Master Schedule (IMS) is developed to relate the IMP events and SEMP processes to calendar dates.[7] Once the SEMP, IMP, and IMS are in place, the control and manage activity shown in Figure 11 directs their accomplishment.

7. The IMP and IMS are used by programs applying Integrated Product and Process Development (IPPD) to plan the systems engineering activity as an integrated part of the overall work necessary to complete program. The draft MIL-STD 499B and the early EIA/IS-632 and IEEE P1220 standards (all issued in the mid 1990s) used the term Systems Engineering Master Schedule (SEMS) for a plan equivalent to the IMP but covering only systems engineering and Systems Engineering Detailed Schedule (SEDS) for a schedule equivalent to the the systems engineering elements of the IMS. In the ANSI/EIA-632-1998, the SEMP is called an Engineering Plan. In the IEEE Std 1220-1998, the corresponding terms are the system engineering management plan or engineering plan, the master schedule, and the detailed schedule.

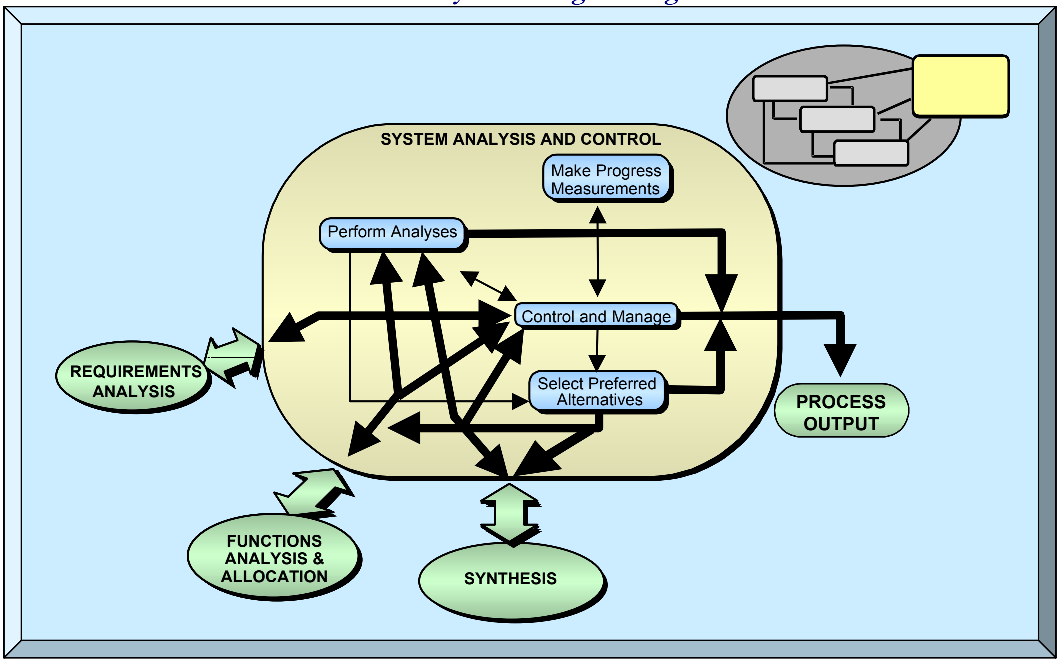


Figure 11. System Analysis & Control

As the process progresses, trade-off studies and system/cost effectiveness analyses are performed in support of the evaluation and selection processes of the other activities. Risk identification/reduction studies are conducted to aid in risk management. Analyses also identify critical parameters to be used in progress measurement.

The management activity directs all operations and also performs configuration management (CM), interface management (IM) and data management (DM). It specifies the performance parameters to be tracked for progress measurement. It conducts reviews and reports progress.

The information from the System Analysis and Control activity is a major part of the systems engineering process database that forms the process output. The control and manage activity contributes a record of the process as well as CM, IM and DM data. The analysis activity provides the results of all analyses performed, identifies approaches considered and discarded, and the rationales used to reach all conclusions. The selected preferred alternatives are recorded with the associated criteria and methodology for selection. Detailed descriptions of the activities of System Analysis and Control are provided below.

Perform Analyses – Initial analyses identify the salient factors of the program and its requirements providing the basis for planning the Systems Engineering effort. Subsequent analyses support the selection and refining operations of the other activities of the Systems Engineering Process. These analyses include trade-off studies, system/cost effectiveness analyses, and risk identification. Trade-off studies analyze the differences between alternate approaches. System analyses look at aggregate systems solutions and determine their performance characteristics. Cost effectiveness analyses establish the costs and associated benefits of candidate system concepts, functional configurations, products and processes. Risk identification analyzes all parts of candidate approaches and their associated program elements to isolate and evaluate the risk involved in their use. As the Systems Engineering Process advances from Requirements Analysis through Synthesis, the analyses become more detailed.

The trade-off studies supporting the other System Engineering activities are as follows:

Requirements Analysis — trade-off studies establish alternate performance and functional requirements. Often these studies identify major cost drivers to assist the customer in refining his requirements to obtain the most effective cost/performance mix.

Functional Analysis and Allocation — trade-offs provide evaluations of alternate functional architectures, help define derived requirements and resolve their allocation to lower levels, and aid in selecting the preferred set of performance requirements at functional interfaces.

Synthesis — trade studies support decisions on use of new versus non-development products and processes; establish system and CI configurations; assist selection of system concepts, designs, and solutions (based on people, parts and materials availability); support materials/processes selections and Make-or-Buy decisions, examine proposed changes; investigate alternate technologies for risk/cost reduction; evaluate environmental and cost impacts; establish standardization to reduce life-cycle costs; and evaluate and select preferred products and processes.

System Analyses are performed to assist in the development of candidate functional and physical configurations and to determine the performance of each candidate. The analyses also provide a methodology and mechanism to establish, track and control analytical relationships and measures of effectiveness, and permit traceability across functional and physical interfaces. Integral to this process is the identification of critical factors to support decisions and permit technical performance measurement.

Cost-effectiveness analyses determine the cost/benefit characteristics of candidate systems approaches to assist in selecting the preferred alternative(s). These analyses support the three other Systems Engineering Process activities and are a major factor in selecting the preferred alternative(s).

Risk analyses identify critical parameters that might be risk drivers. Potential sources include both individual items and groups of items where interrelationships may contribute to risks. For example, a product might itself be low risk, but because it must be matched to a high-risk new development item, use of the product might be high risk also. Risks are quantified for cost, schedule and performance impact. Also examined are design, cost and schedule uncertainties, and the risk sensitivity of program, product, and process assumptions. The analyses pinpoint areas that require risk management in the control and management activity.

Control and Manage – This activity interfaces with all other activities of the process. It plans and manages the activities, monitors and reports status, coordinates actions, and documents in the process output database all progress, results, decisions, and rationales for decisions. It promulgates the SEMP, and the systems engineering entries in the IMP and IMS, and any lower order plans or schedules required to implement them. It also includes the activities of Risk Management, Interface Management, Data Management, and Configuration Management. It is responsible for the conduct of technical reviews and audits. It identifies the items to be tracked for technical performance measurement. The Control and Manage activities are addressed in more detail in Chapter 5, What is Systems Engineering Management?

Select Preferred Alternatives – Based on analyses performed within the System Analysis and Control activity and within the Functional Analysis and Allocation and the Synthesis activities, preferred alternates are selected. The selections are made at increasingly fine-grained levels of system description. In support of the Functional Analysis and Allocation activity, these selections are made to determine which functional architecture and definitions should undergo continued development and which should be discarded. In support of Synthesis, the selection revolves around selection of physical systems architectures, product and process specifications, and determinations as to which technologies will be used initially to prove concepts and which will be inserted later as technology evolves and designs mature.

Make Progress Measurements – The Control and Manage activity determines which measures of effectiveness will be tracked and reported. Once this has been accomplished, the other activities are directed to supply the requisite data. The Progress Measurement compiles and analyzes the data for use by the Control and Manage activity to direct the program and report progress.

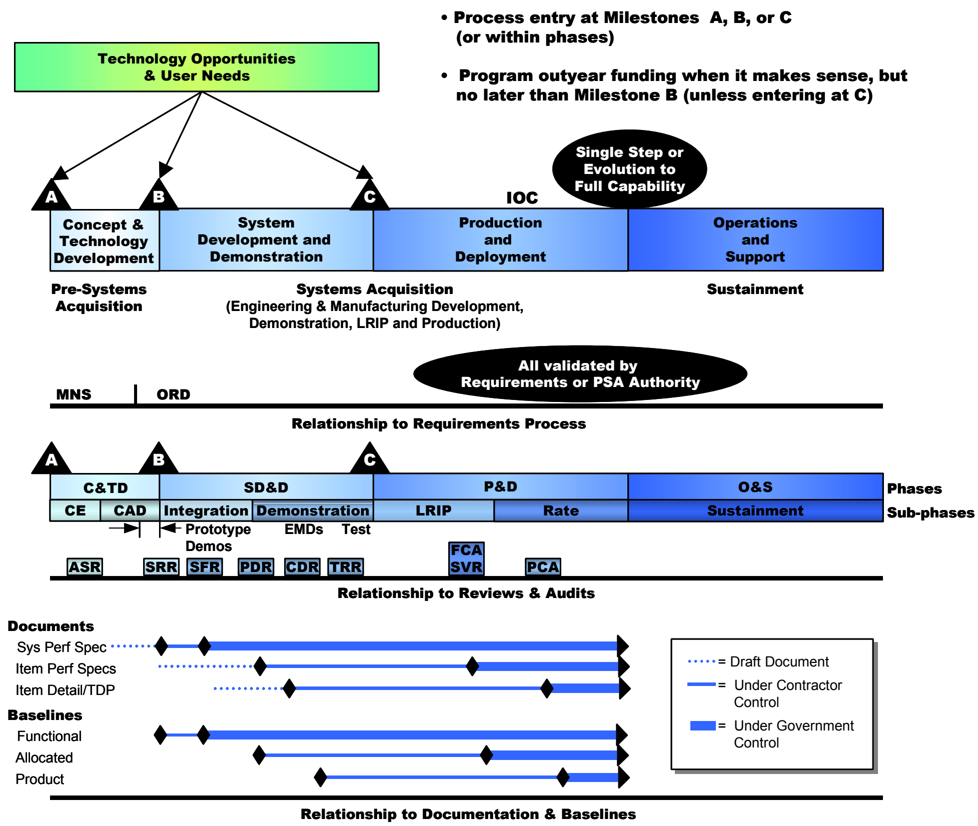
Systems Engineering Below the System Level

Because of its name, there is a tendency to think of the Systems Engineering process as something that is conducted only at the system level. Don't be misled. Even if not specifically imposed, there is a need to perform Systems Engineering even down to the component. While the scope of the effort decreases at lower levels of system complexity, requirements analysis and the other Systems Engineering activities have to be addressed whether the subject is the development of a sophisticated radar subsystem or the selection of a suitable fastener for a shipping container.

Again we encounter the recurring Systems Engineering theme of reiteration—a wheel within a wheel. Systems Engineering, in some form, permeates all activities at all levels and in all phases!

A Few Words About Time

The process described above is event-driven, that is, it is concerned only with how activities flow from one to another, in what order activities are accomplished, what predecessor tasks are required as prerequisites, and what subsequent activities are affected. DoD Instruction DODI 5000.1 describes the acquisition model used in developing DoD systems. Figure 12 relates this model to the Systems Engineering functions of documentation, baselining, and review/audit, and to the requirements documents driving these functions. The Acquisition model is currently under consideration for revision. Hence, the following text boxes provides, practices, products, reviews in the context of the interim acquisition framework.



LEGEND:

ASR = Alternate Systems Review	MNS = Mission Need Statement
CAD =Component Advanced Development	ORD = Operational Requirements Document
CDR = Critical Design Review	PCA = Physical Configuration Audit
CE = Concept Exploration	PDR = Preliminary Design Review (including System Software Review)
EMD = Engineering & Manufacturing Development	SFR = System Functional Review
FCA = Functional Configuration Audit	SRR = System Requirements Review
IOC = Initial Operational Capability	SVR = System Verification Review
LRIP = Low Rate Initial Production	TRR = Test Readiness Review

Figure 12. DoD 5000 Model Acquisition – System Engineering's Changing, Continuous Role

Life Cycle Phases of a Major System

Pre-Phase A—Advanced Studies

Advanced Projects groups usually perform Advanced Studies, Their purpose is to uncover, invent, create, concoct and/or devise a broad spectrum of ideas and alternatives for missions from which new projects (programs) can be selected. Typically, this activity consists of loosely structured examinations of new ideas, usually without central control and mostly oriented toward small studies. Its major product is a stream of suggested projects based on the identification of needs and the discovery of opportunities that are potentially consistent with stated missions.

Pre-Phase A—Advanced Stu ies

Goal: Produce a broad spectrum of id s and alternatives for missions from which new pro-gran / projects can be selected.

Common Practices and Products:
Identify missions consistent with chart
Identify and involve users
Perform preliminary evaluations of po ble missions
Prepare program/project proposals, wl ch include:
· Mission justification and objectives
· Possible operations concepts
· Possible system architectures
· Cost, schedule, and risk estimates.
Develop master plans for existing prog m areas

Reviews:
Mission Concept Review
Informal proposal reviews

Phase A—Concepts Phase

The Concept Phase further examines the feasibility and desirability of a suggested new major system before seeking significant funding. The major products of this phase are a formal Mission Needs Statement (MNS) and one or more credible, feasible designs and operations concepts.

The *USECAF Space Acquisition Policy 02-01* provides the following instruction for Phase A. "Concept Definition Studies, involves extensive study of the complete range of alternatives for meeting the requirements. A Program Manager manages each individual study. For multiple study efforts, the DSAB Executive Secretary, through the USECAF Space Staff and in conjunction with the Program Managers, will establish common assumptions, criteria, and measures of effectiveness. Discussion and review with all appropriate players are extremely important during this phase. The DSAB Executive Secretary will manage all such discussions."

In Phase A, a larger team, often associated with an ad hoc program or program office, readdresses the mission concept to ensure that the project justification is sufficient to warrant a place in the budget. The team's effort focuses on analyzing mission requirements and establishing a mission architecture. Activities become formal, and the emphasis shifts toward establishing optimality rather than feasibility. The effort addresses more depth and considers many alternatives. Goals and objectives are solidified, and the project develops more definition in the system requirements, top-level system architecture, and operations concept. Conceptual designs are developed and exhibit more engineering detail than in advanced studies. Technical risks are identified in more detail and technology development needs become focused.

Phase A—Concept Studies

Goal: Determine the feasibility and sirability of a suggested new major system and its c npatibility with Air Force strategic plans.

Common Practices and Products::
Prepare Mission Needs Statement
Define Operational and threat enviro nents
Develop desired operational capabilit s & requirements
Identify key performance parameters
Identify alternative operations and lo stics concepts
Identify constraints and system boun ries
Consider alternative design concepts, cluding: feasibility and risk studies, cost and sc edule estimates, and advanced technology requiremen
Demonstrate that credible, feasible d ign(s) exist
Acquire systems engineering tools an models
Initiate environmental impact studies
Establish assumptions, criteria, and M Es
Document Phase A exit criteria in(AI M)
IPA evaluates concept descriptions & eqts set

Reviews
Mission Definition Review
Preliminary Non-Advocate Review
Preliminary Program/Project Approv Review

Phase B – Pre-acquisition

In this phase, the Program Manager normally conducts a competition for detailed feasibility studies and awards one or more study contracts to qualified hardware or software development contractors. Phase B is designed to increase confidence in the selected alternative(s) by assessing the risk levels and performance envelope at a detailed, engineering level. Additionally, Phase B provides the data necessary for system source selection in Phase C.

Phase B – Pre-acquisition

Goal: Increase confidence in the select l alternative> Provide the data necessary for system s urce selection.

Common Practices and Products:
Perform Strategic Planning
Prepare SAMP / AP Development
Initiate the Acquisition Strategy Panel (SP) Process
Source Selection Planning
Develop Evaluation Criteria
Develop Evaluation standards
Select the Source Selection team and in iate process
RFP Preparation
Prepare SOO, SOW, CDRLS
Prepare contract technical compliance
Prepare Section L Sect M
Draft IMP, IMS, WBS
Perform Proposal Reviews/ Source Sel :tion
Information Baselined:
Program Baselining (APB)
SOO, SOW, CDRLS

Reviews:
Source Selection
Proposal Reviews

Phase C – Acquisition & Ops

This phase begins with source selection for the prime and/or associate contractors for system acquisition and operations. Phase C includes building, testing, and delivering the space-related system elements (e.g., satellite, booster and ground segments) and ensuring that necessary interfaces with the user elements function smoothly. Unless otherwise directed, the Program Manager also conducts studies to ensure the long-term reliability and maintainability of the system, to resolve emerging hardware or software problems, and to maintain mission performance over the planned life of the system. As the program moves into operations, the Program Manager is responsible for maintaining the system to accomplish those requirements allocated during the KDP-C process, as well as others that may be assigned by the USECAF. The Program Manager is expected to track these requirements closely as they evolve over time.

Phase C1 Systems Definition

The Systems Definition Phase establishes an initial project baseline, which includes a formal flow-down of the operational performance requirements to a complete set of system and subsystem requirements and design specifications for space/flight and ground

Phase C1 Systems Definitio

Goal: Define the system baseline suffi :ntly to initiate detailed design. Define a system baseli : capable of meeting mission needs.

Common Practices and Products:
Prepare a Systems Engineering Manag ient Plan
Prepare a Risk Management Plan
Initiate configuration management
Prepare engineering specialty program ans
Develop system-level cost-effectivenes model
Translate mission needs as functional r juirements
Establish the initial system requiremen and verification requirements matrix
Perform and archive trade studies
Select a baseline design solution and a ıncept of operations
Define internal and external interface r juirements
(Repeat the process of successive refin nent to get "design-to" specifications and drawing verifications plans, and interface docur :nts to lower levels as appropriate)
Define the work breakdown structure
Define verification approach end polic s
Identify integrated logistics support rec irements
Establish technical resource estimates d firm life-cycle cost estimates
Develop statement(s) of work
Initiate advanced technology developr nts
Revise and publish a Project Plan
Reaffirm the Mission Needs Statement

Technical Information Baselined:
System requirements
Verification requirements
Requirements traceablity
System architectures (functional, physi l, interface)
Work breakdown structure
Concept of operations
Complete set of specifications necessa to initiate detailed design

Management Information Baseline
Program plans, including schedule, res ırces, acquisition strategies, and risk management

Formal Reviews
System Requirements Review(s)
System Definition Review
Preliminary Design Review
Safety review(s)

elements and corresponding preliminary designs. The technical requirements should be sufficiently detailed to establish firm schedule and cost estimates for the project.

Actually, the Phase C baseline consists of a collection of evolving baselines covering technical and business aspects of the project: system (and subsystem) requirements and specifications, designs, verification and operations plans, and so on in the technical portion of the baseline, and schedules, cost projections, and management plans in the business portion. Establishment of baselines implies the implementation of configuration management procedures.

Early in Phase C, the effort focuses on allocating functions to particular items of hardware, software, personnel, etc. System functional and performance requirements along with architectures and designs become firm as system trades and subsystem trades iterate back and forth in the effort to seek out more cost-effective designs.

Phase C2—Design

The Design Phase establishes a complete design ("build-to" baseline) that is ready to fabricate (or code), integrate, and verify. Trade studies continue. Engineering test units more closely resembling actual hardware are built and tested so as to establish confidence that the design will function in the expected environments. Engineering specialty analysis results are integrated into the design, and the manufacturing process and controls are defined and validated.

Phase C2 -- Design

Goal: Complete the detailed design of e system

Common Practices and Products:
Add remaining lower-level design spec cations to the system architecture
Refine requirements documents
Refine verification plans
Prepare interface documents
(Repeat the process of successive refin nent to get "build-to" specifications and drawings, erification plans, and interface documents at all levels)
Augment baselined documents to refle growing maturity of system: system architecture verification req'ts matrix, work breakdown structure, pro ct plans
Monitor project progress against proje plans
Develop system integration plan and sy tem ops plan
Perform and archive trade studies
Complete manufacturing plan
Develop the end-to-end information sy tem design
Refine Integrated Logistics Support Pl
Identify opportunities for p3 improven nt
Confirm science payload selection

Information Baselined:
All remaining lower-level requirements nd designs, including traceability to higher levels
"Build-to" specifications at all levels

Reviews:
Subsystem (and lower level) Critical D gn Reviews
System-level Critical Design Review

Configuration management continues to track and control design changes as detailed interfaces are defined. At each step in the successive refinement of the final design, corresponding integration and verification activities are planned in greater detail. During this phase, technical parameters, schedules, and budgets are closely tracked to ensure that undesirable trends (such as an unexpected growth in spacecraft mass or increase in its cost) are recognized early enough to take corrective action.

Phase C culminates in a series of critical design reviews (CDRs) containing the system-level CDR and CDRs corresponding to the different levels of the system hierarchy. The CDR is held prior to the start of fabrication/production of end items for hardware and prior to the start of coding of deliverable software products. Typically, the sequence of CDRs reflects the integration process that will occur in the next phase— that is, from lower-level CDRs to the system-level CDR. Projects, however, should tailor the sequencing of the reviews to meet their individual needs. The final product of this phase is a "build-to" baseline in sufficient detail that actual production can proceed.

Phase C3 – Development, Manufacturing & Verification

The purpose of this phase is to build and verify the system designed in the previous phase, deploy it, and prepare for operations. Activities include fabrication of hardware and coding of software, integration, and verification of the system. Other activities include the initial training of operating personnel and implementation of the Integrated Logistics Support Plan. For flight projects, the focus of activities then shifts to pre-launch integration and launch. For large flight projects, there may be an extended period of orbit insertion, assembly, and initial shake-down operations. The major product is a system that has been shown to be capable of accomplishing the purpose for which it was created.

Phase C3—Development, Mfg & Ve ification
Goal: Build and verify the system desig ed in the previous phase.
Common Practices and Products: Fabricate (or code) the parts (i.e., the lc rest-level items in the system architecture) Integrate those items according to the i tegration plan and perform verifications, yielding veri ed components and subsystems (Repeat the process of successive integ tion to get a verified system) Develop verification procedures at all l els Perform system qualification verificatic (s) Perform system acceptance verification) Monitor project progress against projec plans Archive documentation for verification performed Audit "as-built" configurations Document Lessons Learned Prepare operator's manuals Prepare maintenance manuals Train initial system operators and main iners Finalize and implement Integrated Log tics Support Plan Integrate with launch vehicle(s) and lau ch, perform orbit insertion, etc., to achieve a deployed sy em Perform operational verification(s)
Information Baselined: "As-built" and "as-deployed" configura on data Integrated Logistics Support Plan Command sequences for end-to-end cc nmand and telemetry validation and ground data pı cessing Operator's manuals Maintenance manuals
Reviews: Test Readiness Reviews (at all levels) Acceptance Reviews System functional and physical configu tion audits Flight Readiness Review Operational Readiness Review

Manufacturing/Production/Construction includes the fabrication of engineering test models and "brass boards," low rate initial production, full-rate production of systems and end items, or the construction of large or unique systems or sub-systems.

At Production Readiness and LRIP system-level demonstrations have been accomplished and the product baseline is defined (although it will be refined as a result of the activities undertaken during this phase). The effort is now directed toward development of the manufacturing capability that will produce the product or system under development. When a manufacturing capability is established, a LRIP effort begins. The development of a LRIP manufacturing capability has multiple purposes. The items produced are used to proof and refine the production line itself, items produced on this line are used for Initial Operational Test and Evaluation (IOT&E) and Live Fire Test and Evaluation (LFT&E), is also the means by which manufacturing rates are ramped upward to the rates intended when manufacturing is fully underway.

Test reports, and a full-rate production decision by the MDA, the system enters the Rate Production and Deployment stage. After the decision to go to full-rate production, the systems engineering process is used to refine the design to incorporate findings of the independent operational testing, direction from the MDA, and feedback from deployment activities. Once configuration changes have been made and incorporated into production, and the configuration and production is considered stable, Follow-on Operational Test and Evaluation (FOT&E), if required, is typically performed on the stable production system. Test results are used to further refine the production configuration. Once this has been accomplished and production

Phase C4 – Deployment

Deployment (Fielding) includes the activities necessary to initially deliver, transport, receive, process, assemble, install, checkout, train, operate, house, store, or field the system to achieve full operational capability. As the system is produced, individual items are delivered to the field units that will actually employ and use them in their military missions. Careful coordination and planning is essential to make the deployment as smooth as possible. Integrated planning is absolutely critical to ensure that the training, equipment, and facilities that will be required to support the system, once deployed, are in place as the system is delivered. The systems engineering function during this activity is focused on the integration of the functional specialties to make certain that no critical omission has been made that will render the system less effective than it might otherwise be. Achieving the user's required initial operational capability (IOC) schedule demands careful attention to the details of the transition at this point. Furthermore, as the system is delivered and operational capability achieved, the system transitions to the Sustainment and Disposal phase of the system life cycle—the longest and most expensive of all phases.

Phase C5 – Operations

The purpose of this phase is to meet the initially identified need or to grasp the initially identified opportunity. The products of the phase are the results of the mission. This phase encompasses evolution of the system only insofar as that evolution does not involve major changes to the system architecture; changes of that scope constitute new "needs," and the project life cycle starts over. Phase C4 encompasses the problem of dealing with the system when it has completed its mission; the time at which this occurs depends on many factors.

Phase C5—Operations

Goal: Meet the initially identified need r to grasp the opportunity.

Common Practices and Products:
Train replacement operators and maint ners
Conduct the mission(s)
Maintain and upgrade the system
Document Lessons Learned

Information Baselined:
Mission outcomes, such as:
· Engineering data on system, subsyster and materials performance
· Mission data returned
· Accomplishment records ("firsts")
Operations and maintenance logs
Problem/failure reports

Reviews:
Regular system operations readiness re ews
System upgrade reviews

Phase C6 – Disposal

For a flight system with a short mission duration, such as a launch vehicle payload, disposal may require little more than de-integration of the hardware and its return to its owner. Alternately, planned disposal may include orbital maneuvers to a predetermined location. On large flight projects of long duration, disposal may proceed according to long-established plans, or may begin as a result of unplanned events, such as accidents. Alternatively, technological advances may make it uneconomic to continue operating the system either in its current configuration or an improved one. In addition to uncertainty as to when this part of the phase begins, the activities associated with safely decommissioning and disposing of a system may be long and complex. Consequently, the costs and risks associated with different designs should be considered during the project's earlier phases

Phase C6 -- Disposal

Goal: Dispose of the system in a respo sible manner.

Common Practices and Products:
Dispose of the system and supporting ocesses
Document Lessons Learned

Reviews:
Decommissioning Review

The DoD 5000 acquisition model stresses flexibility in the process to bring effective systems on line as quickly and affordably as possible. It fosters evolutionary development, whereby new system requirements are met by building on existing Government and commercial systems, equipments and technologies. Fielded systems may achieve full capability in a single step, or improvements may be added incrementally in subsequent blocks of production. The baseline model comprises: Pre-System Acquisition – having a single Concept and Technology Development phase; System Acquisition – having a System Development and Demonstration phase and a Production and Deployment phase; and Operations and Support – having a Sustainment phase.

System Engineering has a continuing but changing role in each phase. In the Concept and Technology Development (C&TD) phase, emphasis is on the Requirements Analysis activities in the definition/refinement of general requirements and overall feasibility. System Engineering assists the User in articulating his requirements in a Mission Need Statement (MNS) and also identifies needed research for technologies that will reduce the system development risk. Competing concepts are developed as possible system solutions. The Systems Engineering Requirements Loop is exercised to convert User requirements to system requirements and possible functional implementations. Analyses and trade studies are conducted to help select preferred alternatives. The costs of efforts at this stage are relatively small. Often several contracts are let to allow the procuring agency to choose two or three of the best among those proposed for further development. Prototypes are built to demonstrate the feasibility of components or complete equipment sets. Designs are implemented with existing technology or discrete components with the intent of substituting such items as advanced devices or large-scale integration (LSI) in later phases. Alternate Systems Review(s) (ASRs) evaluate the efficacy of each concept. If applicable, a System Threat Assessment Report (STAR) provides an evaluation of any threats, which could affect the performance of the system mission. Using the results of the ASR(s) and the STAR (if applicable), the User's requirements are refined and detailed in an Operational Requirements Document (ORD).

Usually only a single concept survives to System Development and Demonstration (SD&D) phase. However, the Procuring Agency may occasionally retain more than one concept if funding is available. In this way the Agency can pursue a highly innovative concept that promises greater performance but entails greater risk while maintaining as insurance, a more conservative alternate approach that uses proven technology. Similarly, the Agency may wish to maintain cost competition into the next phase. Early in the phase a System Requirements Review (SRR) is held to assure that all parties (User, Procuring Agency and Contractors) are aware and agree on the requirements for the system under development. During SD&D, the System Engineering activities begin to transition from the Requirements Loop to the Design Loop with analyses and trade studies performed to assist in selecting preferred solutions. The Functional Analysis and Allocation tasks become more prominent and the functional baseline and system specification for each concept are developed. When the functional baseline is sufficiently mature, a System Functional Review (SFR) is held. At the SFR the system specification and functional baseline for the concept is reviewed to determine whether the proposed system meets requirements, is achievable, and is ready to proceed to preliminary design. The ORD is updated based on the SFR results and any updated STAR. Engineering and Manufacturing Development (EMD) follows. The name is significant because not only does it indicate that the system is under development, but also anything needed to manufacture and test the system. Rarely is there more than one concept at this point, but occasionally another contractor is retained as a second source. During EMD the emphasis is on the synthesis activities with trade studies and analyses to narrow the selection of ways in which the hardware and software might be implemented. Configuration Item (CI) requirement allocation is finalized and design solutions are translated into system hardware and software that meet the User's need. In addition, during EMD all the things necessary to manufacture and support the system are developed -- manufacturing processes, technology, equipment and facilities; special test equipment and facilities; support equipment; training for production workers, system maintainers and system operators; etc. In EMD, System Engineering is also engaged in developing test requirements which will indicate that the system design meets User needs (qualification testing) and that individual systems meet established performance norms (acceptance testing).

Three major system reviews occur during EMD: Preliminary Design Review (PDR), Critical Design Review (CDR) and a Test Readiness Review (TRR). The PDR confirms that the system detailed design approach satisfies the functional baseline, that risks are under control, and that the system is ready for detailed design. If applicable, a Software Specification Review (SSR) is usually held with the system PDR. CDR demonstrates that the total system design is complete and meets requirements, that hardware elements are ready for initial builds, and that software elements are ready for coding. The complete allocated baseline and development specifications for all CIs are reviewed and approved prior to committing to hardware. It confirms readiness for full-scale production. Also during EMD, similar reviews (PDRs and CDRs) are conducted on individual CIs and Computer Software Configuration Items (CSCIs). A Test Readiness Review (TRR) is held before system and CI testing is initiated. The test results and any new STAR information are used to update the ORD.

In the Production and Deployment phase the system is produced and fielded. A Functional Configuration Audit (FCA) and a System Verification Review (SVR) is conducted on the product specification, all associated process and material specifications, and on a representative system from the first production run. When the system has been approved for production, a system Physical Configuration Audit is conducted to establish the product baseline for subsequent production systems. PCAs on all constituent CIs are completed prior to the system PCA and reviewed as part of the audit. Preceding or concurrent with the system deliveries support equipment and facilities are provided along with operation/maintenance training.

Changes occur throughout the operational life of the system. Missions change or are augmented. Threats change or new threats appear. Deficiencies are uncovered. New devices or technology provide improved performance, reliability or availability. Parts of the system become obsolete or are no longer supportable. All these factors lead to product improvements in the fielded system. During the Operations and Support (O&S) phase, System Engineering's role is to evaluate competing implementations and their relative effect on other elements of the system, chose the best, foster their development, orchestrate the changes, and maintain the evolving configuration baseline. Each change or group of changes is handled as a new development. For small changes, the process may be fairly informal. However, for major or critical changes, the complete formal review structure with SRR, PDR, CDR and PCA may be invoked. Throughout the remainder of the program, including the safe and secure disposal, System Engineering is responsible for the integrity of the system.

Milestones occur at major decision points in the acquisition model (Figure 9) with a Milestone Decision Authority (MDA), whose DoD level is dependent upon the size and criticality of the program, making a determination as to whether continuation into the next phase is warranted:

Milestone A – At start of Concept and Technology Development phase, authorizes initiation of concept studies. Requirements for these studies and related activities are documented in a Mission Need Statement. The MDA defines the goals of the activities in exit criteria that indicate what must be accomplished to support continuation into the System Development and Demonstration phase. A favorable Milestone A decision is not an authorization of a new acquisition program, merely a go-ahead to explore system concepts and underlying technology development.

Milestone B – At start of System Development and Demonstration phase, authorizes initiation of an acquisition program. Requirements for these activities are documented in an Operational Requirements Document. Since this is the DoD's commitment to a systems acquisition, in making the Milestone B decision the MDA must consider the validated ORD, the System Threat Assessment, an independent assessment of technology status and issues, early operational assessments or test and evaluation (T&E) results, analyses of alternatives, independent cost estimates, system affordability and funding, proposed acquisition strategy, cooperative opportunities, and infrastructure and operational support. At Milestone B the MDA confirms the acquisition strategy, the development acquisition baseline, low-rate initial production quantities (if applicable) and the System Development and Demonstration exit criteria.

Milestone C – At the start of the Production and Deployment phase, authorizes entry into low-rate production (for Major Defense Acquisition Programs – MDAPs, and major programs) into production or procurement (for non-major systems that do not require low-rate production) or into limited deployment for Major Automated Information Systems – MAISs, or software-intensive systems with no production components. In making the Milestone C decision the MDA must consider the independent cost estimate, manpower estimate, System Threat Assessment, Critical Program Information protection and anti-tamper recommendations, and the program for National Environmental Policy Act compliance. At Milestone C the MDA confirms the acquisition strategy, the development acquisition baseline update, exit criteria for low-rate initial production (LRIP), if applicable, or limited deployment.

As shown in Figure 9, phases of DoD 5000 Acquisition Model have sub-phases. In the Concept Exploration (CE) sub-phase of the Concept and Technology Development (C&TD) phase, paper studies of alternate concepts for meeting the MNS are developed and evaluated. At the end of CE, a decision review may be conducted to determine if additional component development is necessary before key technologies are sufficiently mature to enter the System Development and Demonstration (SD&D) phase. If such development is deemed unnecessary, the program could advance directly to Milestone B. Otherwise, the Component Advanced Development (CAD) sub-phase is conducted for subsystems/components requiring demonstration before integration into a system, and for demonstrating new system concepts and technology. This flexibility is built into the DoD 5000 model to allow the decision makers to take advantage of existing systems and technology and to speed acquisition when critical needs dictate.

System Development and Demonstration (SD&D) has two sub-phases. In the System Integration (SI) sub-phase, subsystems and components are integrated and evaluated to reduce integration risk. An Interim Progress Review (IPR) may be held before progressing to the System Demonstration (SD) sub-phase. The IPR confirms that the program is progressing as planned within the phase, or allows adjustment of the plan to better accommodate progress made to date or changed circumstances, or both. There are no hard requirements for the IPR and it can be eliminated if not required. When the system has been demonstrated in prototype articles, the utility of the system in its intended environment is demonstrated and that it meets validated requirements. In addition, the availability of industrial capabilities is determined.

Operations and Support (O&S) has a Sustainment sub-phase during which operational support is provided throughout the system's useful life. At the end of Sustainment, the system is de-militarized and safe disposal of non-useful equipment, components and parts.

Not all acquisitions follow the entire baseline model. Depending on the status of implementing technology and criticality of user's need, a program may enter the model at any of the three milestones

(Figure 9), and advance through sub-phases as required. This flexibility takes full advantage of prior government and commercial investment in Commercial-Off-the-Shelf (COTS) and Non-Developmentally Items, and to facilitate rapid and effective transition from Science and Technology to Products, and from Acquisition to Deployment and Fielding.

This section started as “a few words about time.” However, while we addressed phases, we have not discussed timing. Timing is identified in the system’s master schedule and subordinate schedules. Schedules are one of the control tools the System Engineer uses. These and other System Engineering tools are discussed in the next chapter.

Checklists for each of the reviews and audits mentioned above are provide in Appendix A.

Systems Engineering – Software Development

Systems Software Development

Software development is a labor intensive, costly, and often high-risk effort. We choose software in our designs to provide greater system performance, versatility, and flexibility of those functions that can be implemented through programmable processing. In recent years, our greatest challenges to finalize system design or major upgrades have been centered on software problems. For these reasons, emphasis on software development and test is as important as hardware. Though software is addressed throughout this SMC *Systems Engineering Textbook*, we provide more focused software discussion in this section.

Evolution of Software Development Standards

The DoD approach to managing software development efforts has changed dramatically over the last 10 years. As embodied in DoD 5000.1, DoD 5000.2, and DoD 5000.2-R, the emphasis in acquisition management had shifted from government development of detailed specifications of system parameters to more performance-based measures of system requirements, allowing the designer more freedom to define the most appropriate means of implementing these requirements. This is also true of the current DoD interim guidance and instruction. Though the software related military standards have been cancelled, some of the older active contracts may still impose requirements from these standards. DOD-STD-2167A DEFENSE SYSTEM SOFTWARE DEVELOPMENT was the first software standard to establishes uniform requirements for software development that are applicable throughout the system life cycle. The software development process prescribed by this standard included major activities that are applied iteratively or recursively:

System Requirements Analysis/Design

Software Requirements Analysis

Preliminary Design

Detailed Design

Coding and CSU Testing

CSC Integration and Testing

CSCI Testing.

System Integration and Testing.

MIL-STD-498, SOFTWARE DEVELOPMENT AND DOCUMENTATION replaced DOD-STD-2167A on 5 December 1994. Several years later in 27 May, 98, MIL-STD-498 was cancelled. These software standards are still available at various DoD web sites.

The international standard for the development, acquisition, maintenance, supply, and operation of software, ISO/IEC 122071, was approved in 1995. A joint working group of the Institute of Electrical and Electronics Engineers (IEEE)/ and the Electronics Industries Association (EIA) adapted the ISO/IEC 12207 standard to be more in line with United States software lifecycle practices. The IEEE/EIA standard, IEEE/EIA 12207 "Information technology-Software life cycle processes", is packaged in three parts. The three parts are: IEEE/EIA 12207.0, "Standard for Information Technology-Software life cycle processes"; IEEE/EIA 12207.1, "Guide for ISO/IEC 12207, Standard for Information Technology-Software life cycle processes-Life cycle data"; and IEEE/EIA 12207.2, "Guide for ISO/IEC 12207, Standard for Information Technology-Software life cycle processes-Implementation considerations."

There are three fundamental differences between the DoD standards and these industry standards.

1. The industry standards are designed to be implemented on a voluntary basis while the DoD standards were imposed contractually.

2. The new standards are intended to be applied to the full software lifecycle: development, acquisition, maintenance, supply, and operation.

3. IEEE/EIA 12207 is written at a much higher level than the DOD predecessor standards and avoids dictating particular software development approaches and life cycle models. IEEE/EIA 12207 does not provide detailed specification of 'how to' perform the software development tasks.

Highlights of the IEEE/EIA 12207 industry standards are provided here:

Covers the system lifecycle development, acquisition, maintenance, supply, and operation of software

Written to be compatible with the ISO 9000 approach to quality systems, quality management, and quality assurance

Includes references to other applicable industry standards

Complies with the international version of the standard, ISO/IEC 12207

Software Acquisition Strategy Considerations

Mandatory and discretionary acquisition information pertaining to software development are located in Department of Defense Instruction 5000.2 and the Defense Acquisition Deskbook. Since the Milestone Decision Authority (MDA) approval at these Milestones are dependent on software related criteria being met, some of the mandatory directives are provided in this handbook. Of course it is prudent to be familiar with all software related mandates and Milestone criteria. Hence, a thorough review of the latest 5000 series instructions and directives is necessary.

Two acquisition strategy approaches are frequently used to structure a program to achieve full capability: evolutionary and single step. An evolutionary approach is preferred. Evolutionary acquisition is an approach that fields an operationally useful and supportable capability in as short a time as possible. This approach is particularly useful if software is a key component of the system and the software is required for the system to achieve its intended mission. Evolutionary acquisition delivers an initial capability with the explicit intent of delivering improved or updated capability in the future.

The approach to be followed depends on the availability of time-phased requirements in the ORD, the maturity of technologies, the relative costs and benefits of executing the program in blocks versus a single step, including consideration of how best to support each block when fielded. The most recent practice requires that the rationale for choosing a single step to full capability, when given an ORD with time-phased requirements, be addressed in the acquisition strategy. Similarly, the rationale for choosing an evolutionary approach, when given an ORD with no time-phased requirements, must be addressed in the acquisition strategy.

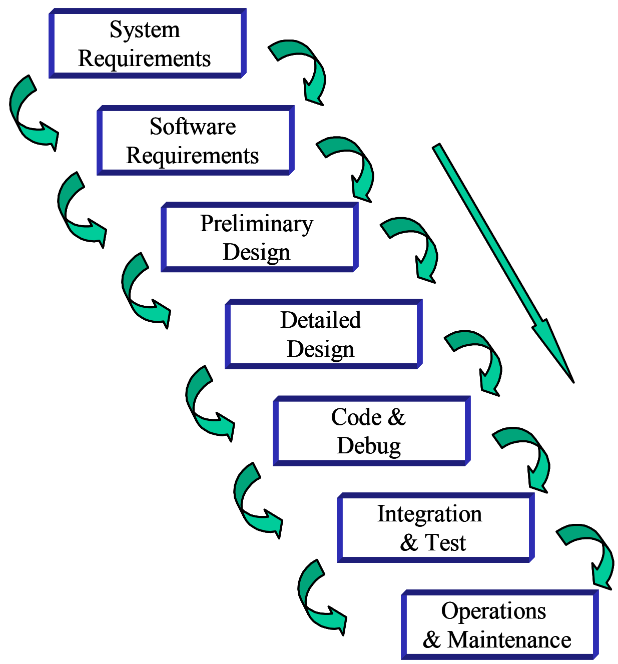


Figure 13. Waterfall Model

The latest mandates that, for both the evolutionary and single-step approaches, software development and integration shall follow an iterative spiral development process in which continually expanding software versions are based on learning from earlier development. In addition, programs with software components must be capable of responding to emerging requirements that will require software modification or periodic enhancements after a system is deployed.

Software Development Lifecycle

Mil-Std-2167A provided the DOD approach to software development and was based on the waterfall model, Figure 13, of software development. Two major shortcomings were recognized with the waterfall model. First, the characteristic sequential evolution includes phases of software development activities that allow only iterations between adjacent phases. Second, the iterations between the waterfall phases are often too long which tends to lengthen the time period from statement of *User* needs to production of a system. Barry Boehm introduced the spiral development model in 1986 to shorten the software development lifecycle. See Figure 14.

As we previously mentioned the Program Manager is required to plan a spiral development process for both evolutionary and single-step-to-full-capability acquisition strategies. DODI 5000.2 characterizes spiral development as a cyclical, iterative build-test-fix-test-deploy process to yield continuous improvements in software. Boehm [1], on the other hand, describes his model as a risk driven approach rather than a document or code driven process. The spiral applies equally to new development and upgrades/enhancements. The spiral has four phases. Starting from the first quadrant clockwise:

determine objectives, alternatives, and constraints; identify and resolve risks; develop and verify design and products; and plan next phase.

Determine Objectives, Design Alternatives and Constraints

The stakeholders' initial emphasis is to determine the performance objectives of the software and possibly the ability of the software to be upgraded. They also identify constraints (e.g., cost, schedule, security, environments, etc) that apply to each of the alternatives under review. The objectives and constraints provide the basis for the software requirements.

Identify and Resolve Risks

The emphasis for risk considerations distinguish Boehm's model from the rest. He cites numerous risk mitigation/resolution techniques such as prototyping, simulation, benchmarking, reference checking, user questionnaires, and analytical modeling. This development model includes software mock-up or prototyping activities [1]. The prototyping effort is initially for concept proofing and supports the Users/Operators to define their requirements. Subsequent evolutions of the spiral support prototyping of detailed design, and finally operational design. Rapid prototyping (not an element of the Boehm model) is intended to produce partially operational mock-ups/prototypes early in the design (initiated during preliminary design phase).

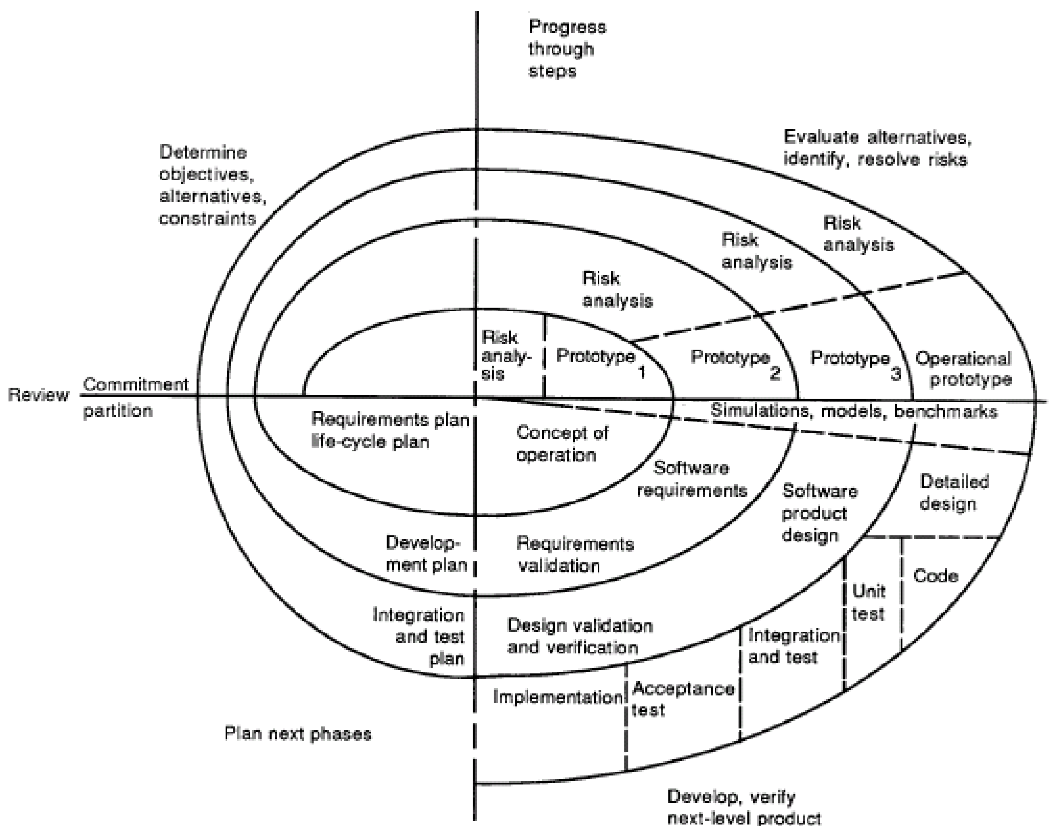


Figure 14. Spiral Model of Software Development

Develop And Verify Design And Products

The *Develop and Verify* phase has recognizable elements that are included in the waterfall model: requirements, design, code, integrate and test. For the spiral model, the *Develop and Verify* phase is entered four times. Planning, alternative assessments, and risk analysis are performed each time preceded by requirements development and validation, preliminary/product design, and detailed design.

SOFTWARE REQUIREMENTS ANALYSIS AND DEFINITION

Software requirements analysis involves defining and baselining requirements for each Computer Software Configuration Item (CSCI) based on the system specification and operational User needs. The requirements analysis should concentrate on the capabilities and performance of the entire system. This includes the software and the environment in which the system is to operate.

Approaches for performing software requirements development are the same for hardware and have been touched upon in a previous section of this handbook. IEEE/EIA 12207 also describes this process for allocating requirements in a top-down fashion. The standard requires first that system-level requirements be allocated to software items and that the software requirements then be documented in terms of functionality, performance, and interfaces. The standard also specifies that software item requirements be traceable to system-level, and be consistent and verifiable. Of course, the requirements analysis includes a cost/benefit analysis to determine the costs associated with developing, maintaining and operating the software of each design alternative under study.

PRELIMINARY DESIGN

During Preliminary Design the overall software structure is developed. The software developer decomposes each software item into software components/modules. If a functional design approach is taken, the developer then identifies the functional interrelationship of the modules as well as the functions within each module.

However, the software developer may elect to use the object-oriented approach to design the software. Objects of this model are physical resources or elements that perform activities and transmit messages to other objects. Object oriented models of a system provide three views: the object (physical or data repository) view, the functional view (processing or data flow), and the behavior (state transition or nested state) diagrams. Data and functionality are localized within the objects rather than being scattered as occurs using functional decomposition methods. This method produces more robust modularity with fewer interfaces. One drawback to object oriented design is that functional compatibility and traceability to system requirements is difficult to assess.

DETAILED DESIGN -- CODING

During Detailed Design, requirements are allocated from item level, to component, and eventually to unit level when using the functional design approach. IEEE/EIA 12207 requires that lower level of requirements allocations are documented or described. (See the IEEE/EIA standard for documentation requirements. The level of documentation detail required varies depending on project needs.) Obviously, all detailed design activities prior to coding are not complete when using the spiral development approach. Selected partitions/modules of the software will have been completed as a result of prototyping in the previous phase.

CSCI INTEGRATION & SYSTEMS TESTING

CSCI testing involves testing an element of a system to ensure that it meets the requirements defined during system requirements review. System integration and test ensures that the software works within the system environment as specified. Over the last few years, much emphasis is being placed on interoperability of weapon systems. Interoperability is the ability of systems, units, or forces to provide data, information, materiel, and services to and accept the same from other systems, units, or forces, and to use the data, information, materiel, and services so exchanged to enable them to operate effectively together. DODI 5000.1 January 4, 2001 Paragraph 4.1.addressed the goals to achieve interoperability within and among United States forces and U.S. coalition partners so that the Department of Defense has the ability to conduct joint and combined operations successfully. Interoperability is certainly applicable to SMC systems in many respects. Hence, the SPOs' systems engineers need to ensure the full set of satellite, control, and user interoperability requirements are identified and met. They must consider the use of standardized data to facilitate interoperability and information sharing with other systems. To the highest extent possible, systems and software must be designed, consistent with U.S. export control laws and regulations, to permit use in a multi-national environment with provision made for current and future information disclosure guidance and constraints. Improved interoperability of weapon systems is expected to be achieved through the C4ISR architecture framework as well. The framework is intended to ensure that architecture descriptions can be compared and relate across organizational and system boundaries. Refer to Appendix C-7 C4ISR Architecture Framework for more detailed discussion on the C4ISR subject.

Spiral Development Method Summary

Some final words on software spiral development -- specific *Interim Regulation* was provided in DODD 5000.2-R regarding the spiral development process. Excerpts from Paragraph C5.2.3.5.6.2. are provided here. Again, it is prudent to investigate the latest instruction and guidance on this subject.

The spiral development process shall accomplish the following:

- Facilitate requirements changes resulting from operational mission needs, technology opportunities, experimentation results, and technology obsolescence.
  - Incorporate T&E of operational effectiveness, suitability, and supportability using experimentation, demonstration, rigorous testing, or certification.
  - The T&E process shall be continuous throughout the system life cycle and involve the user, contractor, program office, and test community.
  - The T&E process shall consider the near continuous nature of change in the baseline and use techniques such as regression testing to ensure that existing functionality has not been compromised.
- The PM shall consider the risks and extent of change impacts to enable a cost-effective, yet rigorous T&E process.
  - Implement configuration, change, and data management.
  - Documented actual deployed capability provides the starting point for development of the next improvement release and provides a baseline for verification, training, etc.

- The PM shall implement a configuration control board to include the user, program office, development contractor, integration contractor or agency, and any other critical stakeholder.

- For legacy systems, the configuration control board shall include the appropriate support and sustainment organizations.

Software Engineering Institute (SEI) Capability Maturity Model (CMM)

You have likely heard of the SEI's CMMI unless you are relatively new to the world of software development. A short discussion on this subject follows for those who are not familiar with CMMI. The premise underlying the CMMI is that, if an organization that develops systems retains organizational maturity in controlling and managing software and hardware development efforts, that organization retains low risk to develop and deliver the products within cost. There are five levels of maturity associated with this model.

Level 1. Initial Process

The process is ad hoc and chaotic and depends on individual efforts. There are neither project plans nor formal procedures. Change control is limited or lacking. Senior management is not aware of software development issues.

Level 2. Repeatable Process

Basic project controls are in place to repeat project successes. An organization has in place, predefined software development procedures. No environment or commitment for process improvements. Edward Yourdon [2] suggests the following processes for software: software Planning, software cost estimating, configuration management, and management commitment.

Level 3. Defined Processes

Organization wide software development processes are standardized. An Engineering Process Group is in place. Youdon provides then following necessary to achieve Level 3: formal standards, formal process models, formal processes for testing, inspections, configuration control, and establishment of an engineering process group.

Level 4. Managed Process

This level emphasizes detailed quantitative methods to measure product and process quality. In other words, an emphasis is placed on quality to identify and correct deficiencies.

Level 5. Optimized Process.

The organization continually invests in process automation and improvements. This level is measured quantitatively in terms of innovative process adaptations and new technologies. A rigorous defect causal analysis and prevention program is in place.

Based on surveys and assessments, SEI estimates that approximately 80% of the software development organizations are at level 1. This model may very well be a solid indication of software development risks. However, it is under discussion that measures of personnel capability and performance are also important to identify and assess potential risks. Marian Myerson [3] provides more discussion on several modified CMMs which do take into consideration personnel capability and performance.

- The Carnegie Mellon Software Engineering Institute latest CMMI now combines 3 source models:
- Capability Maturity Model for Software (SW-CMM) v2.0 draft C
- Electronic Industries Alliance Interim Standard (EIA/IS) 731
- Integrated Product Development Capability Maturity Model (IPD-CMM) v0.98

There are 4 categories of CMMI Process Areas, which include Process Management, Project Management, and Engineering Support. Within each process area goals and practices are defined as reflected in Figure 15. SMC is currently assessing CMMI application for Center and SPO activities. Hence, the next version of this text will include much more in-depth discussion of this model as it applies to SMC.

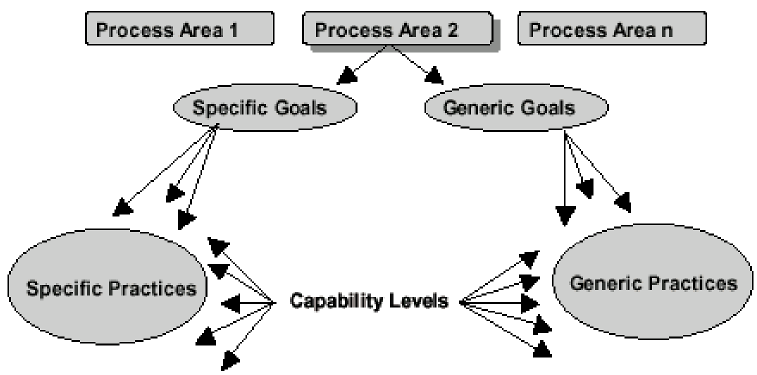


Figure 15. CMMI Model Components

CHAPTER 4

What Are The System Engineer's Tools?

No workman is effective without the proper tools . . . and so it is with the System Engineer. However, his or her tools are not the kind made of steel, wood and plastic, but of techniques, software programs, computers, and ideas. For our discussion, they can be categorized roughly as being of two types: 1) analyses tools, and 2) control/management tools. Analysis tools are used to understand the underlying factors contributing to the system requirements and to evaluate and select candidate approaches and solutions to Systems Engineering problems. Typical analysis tools comprise such things as functional analysis diagrams, cross correlation charts, joint application design techniques, and a large sub-category of tools called models and simulations. The control/management tools assist the System Engineer in planning, tracking and measuring progress along the Systems Engineering process. Typical control/management tools include plans and schedules. e.g., the Integrated Master Plan (IMP), Integrated Master Schedule (IMS), and the System Engineering Management Plan (SEMP) and progress assessment techniques, e.g., technical performance measures (TPMS) and earned value measurements. Some tools do dual service. They may be used to establish metrics used to evaluate candidate approaches, and are then used during the program to determine how well the development is progressing. Another type of dual service systems engineering tool includes application software that aids in performance of a wide range of requirements definition and allocation activities and also provides a relational database.

A discussion of some of the tools available to the System Engineer and an introduction to their use is provided in this chapter. The treatment here is not exhaustive. What is presented is a broad brush description of the tools/techniques and their general application. Reams of literature are available on each and the reader is directed to his closest friendly librarian for help in locating the information desired. The summary descriptions provided are meant only to permit you to assess which tools might be useful and to allow you to ask enough intelligent questions to get the detail you need. In no way will these paragraphs make you an expert in the application of these tools.

PRODUCT FEATURES

CUSTOMER REQUIREMENTS

	A	B	C	D	E	F	G	H	I	J
1		●				●	●			●
2	●		●							●
3							●	●		
4				●					●	
5				●						
6		●						●		●
7										
8										
9	●					●				
10			●				●	●	●	

Figure 16. Cross Correlation Charts -- Graphic Checklists

Analysis Tools and Techniques

The analysis tools and techniques introduced in this section are listed in Table 2.

Correlation Charts

Correlation charts are a graphic means of displaying interrelationships between various analytical factors (e.g., requirements and features). The three types discussed here are the cross-correlation chart, the self-interaction matrix, and the N x N chart.

Cross-Correlation Chart — Figure 16 is an example of a cross-correlation chart. It allows the analyst to relate customer

Table 2. Available Systems Engineering Analyses Tools -- *A Shopping List to Help You Determine What You Need in Your Tool Belt.*

TOOL/TECI NIQUE	USE
Correlation Charts	A means of identifying the relationships between technical fact rs such as design features and system requirements
Value System Desi 1	A technique for quantifying objectives and developing measure of utility
Functional Analysi Tools	Means of representing sequential functions and their interrelati nships to assist in defining system solutions
Quality Function I :ployment	A methodology for decomposing top-level Quality requiremen
Pugh's Controlled onvergence	A peer process for optimizing system design
Models and Simula ons	Software, hardware or combinations of both that allow elemen of a system and its intended environment to be exercised in the laboratory
Scenarios	Sample situations representative of those to be encountered by he system, used to analyze the response of candidate configurations
Joint Application I :sign	A technique for gathering inputs from all disciplines in a joint fort to establish the application design
Non-Customer Interactive Analysi	Techniques for finding available data on related system design ıd application
Allocations, Trace: ility & Decomposition	A methodology to establish a relational database that also aids requirements allocation and decomposition
Baselining	Tools to document configurations as the basis to control desig efforts and

requirements to product features to assure that all requirements are being met and that unneeded features are not included without be addressed. In Figure 16, a dot at an intersection indicates that a particular feature contributes in part or in whole to the achievement of a customer requirement. Notice that Customer Requirement 8 is not satisfied by any product feature. The analyst should determine how important Requirement 8 is and whether it is sufficiently important to launch a design effort to incorporate it. Likewise, Product Feature E has no corresponding customer requirement. The analyst should determine whether Feature E is required for performance of the system now or in the future and the additional costs incurred. If Feature E is expensive, tends to lower reliability, or is a commonality feature that would be costly to remove from present production, and the feature has no immediate requirement, the analyst might decide to eliminate it or incorporate it in a later version when the need arises.

Self-Interaction Matrix — Figure 17 is a typical self-interaction matrix. It shows how different requirements impinge on each other, either positively or negatively. For example, an improvement in performance may adversely affect reliability or availability. Likewise, incorporation of a Built-In Test (BIT) may reduce Mean Time To Repair (MTTR). In Figure 17, Requirement 1 affects or is affected by Requirements 2, 3, 5, 7, and 9. On the other hand, Requirement 4 interacts only with Requirement 9. From such a chart, the analyst is reminded that when designing to satisfy one requirement, he must be aware of the effects on those related requirements.

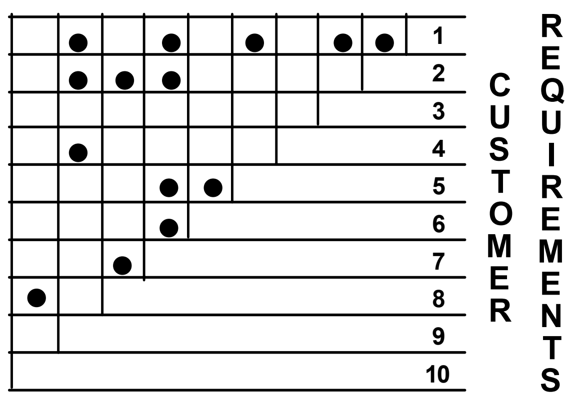
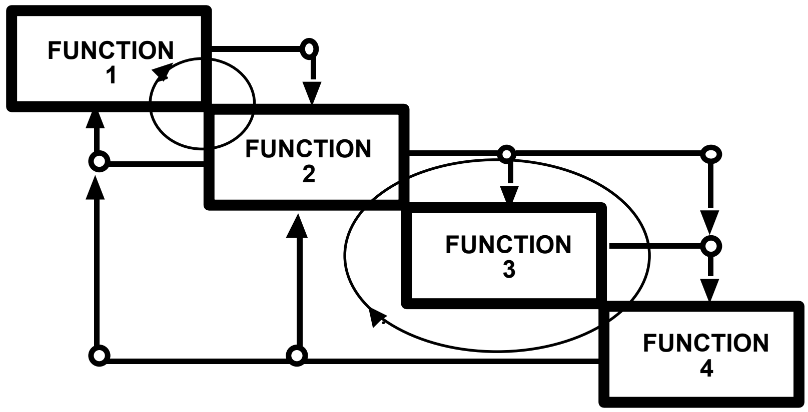


Figure 17. Self-Interaction Charts -- Showing which factors affect others.

N x N Charts — These charts show both interfaces and relationships. Figure 18 is an example of an N x N chart used to show functional flow. The four functions represented form the diagonal of the chart. The block atop the second column shows

FUNCTION 1	F1→ F2		
F1← F2	FUNCTION 2	F2→ F3	F2→ F4
		FUNCTION 3	F3→ F4
F1← F4	F2← F4		FUNCTION 4

a. Standard N x N (4x4) Chart



b. N x N Chart showing interrelationship graphically

Figure 18. N x N Charts - Making Functional Interfaces and Interrelationships More Easily Analyzed.

that Function 1 feeds Function 2. Similarly, the blocks in the third and fourth column of the second row show that Function 2 feeds both Functions 3 and 4. The first block in the second row shows that Function 2 also feeds back to Function 1. Completing the picture, Function 3 feeds Function 4, and Function 4 feeds back to both Functions 1 and 2.

In Figure 18 - b, we have a more graphic representation of the interrelationships. Notice that the diagram shows two complete feedback loops — between Function 1 and Function 2, and between Function 4 and Function 2 . (the feedback between Function 4 and Function 1 does not constitute a loop since there is no direct connection from Function 1 to Function 4). The analyst can see that Function 2 is complex since it is the intersection of two feedback loops. This will warn him to be extra careful in the design of Function 2 or to consider other interfacing that might eliminate this complexity. This type of chart is excellent to represent the states and modes of a system. See Appendix C-6 States & Modes

Value System Design

Value System Design is a technique for establishing the system requirements in a fashion that can be easily understood and measured by all who contribute to the design. It essentially takes the requirements in the user's language and translates into goals in the designer's language. Value system design looks at five areas that define what is desired of the product: objectives; objective measures; criteria and weighting; and utilities.

Objectives — Objectives include requirements but may also include goals above requirements or in areas not specifically stated in requirements. For example, you may want a faster processor because its needed on a collateral project, or you may want to develop the capability to advance a product out of the lab and into production. Setting objectives has strong elements of the creative dimension. Objectives must be stated in terms of what is needed, not how to implement them. Presupposing solutions eliminates initiative and innovation. Objectives are often stated as maximization, minimization, or closet fit to a target. The English language with its ambiguities and slanted meanings can be a hindrance. Therefore, be sure each objective is simply stated and is measurable. Also objectives must be consistent with user requirements and lower-level objectives must be consistent with higher-level ones. Otherwise, efforts are wasted on objectives of no import. Establishing the right objectives is crucial for product success. Wrong objectives lead to wrong solutions. Using the right objectives, you have a better chance of selecting the right solution even if it is less than optimal.

Objectives Measures — Objectives measures are sometimes called Measures of Effectiveness (MOEs). A product's effectiveness determines its "worth." Systems Engineering seeks the greatest possible "worth" at an acceptable cost. A measure of effectiveness has these characteristics:

- Relates to performance.
- Is simple to state.
- Is complete.
- States any time dependency.
- States any environmental conditions.
- Can be measured quantitatively (if required, may be measured statistically or as a probability).
- Is easy to measure.

An example of an MOE for an automobile is fuel consumption in miles per gallon under specified environmental conditions.

Effectiveness at a system level may have several definitions. A typical definition comprises these factors:

- Performance — the probability that the product will perform its mission.
- Availability — the probability that a product is ready for use when needed.
- Dependability — the probability that a product behaves reliably in use.
- Utilization — the actual use of the product versus its potential.

Measures of effectiveness have many factors. To help you identify critical contributing factors you may wish to show them graphically as a performance hierarchy tree traceable from the original user requirements, through the system objectives, to the subsystem and lower-level objectives. Be sure the measures of effectiveness have quantitative expressions. Analyze the measures of effectiveness to develop supporting measures of performance. Make the measures of performance specific, and derive lower-level measures from these. The complete hierarchical structure thus formed shows the critical technical performance measures.

Criteria — Criteria differ from constraints. Constraints are the "musts," the restrictions, the limitations that have to be met and are generally not available for trade-offs. Constraints can be used for screening to filter out alternatives, however, once screening is accomplished, constraints can no longer help determine the best alternative. Constraints establish boundary conditions within which the developer must remain while allocating performance requirements and/or synthesizing system elements and are generally pass or fail.

Criteria are continuous. They provide a means of judging feasible alternatives. Examples might be lowest cost, most range, fastest acceleration, or closest flow rate to 10 gallons per minute.

Sometimes, a measure can be both a constraint and a criterion. For example, as a constraint, the product must cost no more than $10,000., but the customer prefers the lowest cost below that point. A cost of $10,000 is the constraint; costs below $10,000 are criterion.

Sources of criteria are:

- The customer.
- Quality Function Deployment charts.
- Functions or behaviors.
- Measures of effectiveness.
- Measures of performance.
- Contractual costs.
- Contractual schedules.
- Manufacturing.
- Product Support.
- Project and organization objectives.
- Other considerations.

Weighting — Criteria are not of equal importance. Weighting factors are assigned as a means of identifying relative importance. In evaluating alternatives, criteria weighting seeks a closer problem-to-solution match.

Weighting can be established empirically or subjectively. The empirical method derives weights by determining how much each elementary measure contributes to a general outcome. Large numbers of measures require statistical analysis. The scenarios and environments for the studies must be chosen carefully. The sensitivity of measures of success or stated customer desires to changes in individual criteria drives the weighting of those criteria.

Subjective weighting relies on the judgment of experts. One widely used method gives raters a fixed number of points, 100 or 1000, to allocate to the criteria. The distribution of points reveals each criterion's relative importance. In another technique, experts score existing alternatives and then the criteria and weighting factors are derived by analyzing the preferred alternatives. This latter method is used more for establishing values for subsequent design efforts rather than selection candidate approaches.

You should be aware of some of the concerns with weighting methods. The empirical techniques are sensitive to the specific conditions for which they were measured. The subjective techniques depend on the judgment of the experts. New products might not have strongly identified criteria. If you depend entirely on the rating method you ignore the inherent uncertainties. Scoring should always be challenged, and recursion often occurs as the program matures.

		Alternatives					
		1		2		3	
Criteria	Wt	Score	Wt'd Score	Score	Wt'd Score	Score	Wt'd Score
Cost	40	3	120	4	160	5	200
Performance	30	3	90	4	120	5	150
Reliability	10	2	20	3	30	3	30
Maintainability	5	1	5	4	20	3	15
Ease of Mfg	5	2	10	3	15	4	20
Ease of Use	5	5	25	4	20	4	20
Safety	3	4	12	5	15	5	15
Ease of Test	2	3	6	3	6	2	4
Total	100		288		386		454

Figure 19. Criteria Weighting -- An example of comparison using weighted criteria.

Figure 19 is an example of a scoring chart using weighting. Cost, performance and reliability are the major factors, accounting for 80% of the total weighting. Scores in the range zero to five are assigned by criterion to each alternate and then multiplied by the weight. After the weighted scores are summed, Alternate 3 is the clear winner. Early in a program, Alternate 2 may also be carried along as insurance in case the criteria or their weighting change, e.g., Alternate 3 does not live up to expectations, or Alternate 3 depends heavily on unproven or immature technology.

As with any Systems Engineering technique or tool, it is necessary to understand the underlying principles that contribute to Value System Design results. In the example in Figure 19, it is prudent to analyze the sensitivity of each of the Alternates 2 and 3 to changes in requirement values. It may be that a small but acceptable change could radically change the outcome. Utility curves are one means of checking sensitivity.

Utilities — Utility curves describe the relative value of a criterion for different levels of performance. They are graphs of a characteristic versus its relative numeric value. In the examples show in Figure 20, utility ranges from 0-5. Calculating loss is one way to plot a utility. In Figure 20 the schedule is insensitive to time for the first six months, but missing that schedule results in a total loss. For mean time between failures (MTBF), loss decreases nearly linearly as the MTBF increases out to about 10,000 hours. Conversely, loss is fairly insensitive for mean times to repair (MTTR) less than 15 minutes, but drops sharply after that point. Battery life shows little loss of utility for all plotted values. Estimating the loss at intervals results in points that can be graphed. Such graphs show sensitivities in easily understandable form.

A final word of caution: do not use Value System Design in isolation as the sole basis for selection. The application of another tool/technique might provide insight missed by blindly accepting the results shown. Also results should be evaluated in light of your own experience…do they seem reasonable? There is no substitute for common sense!

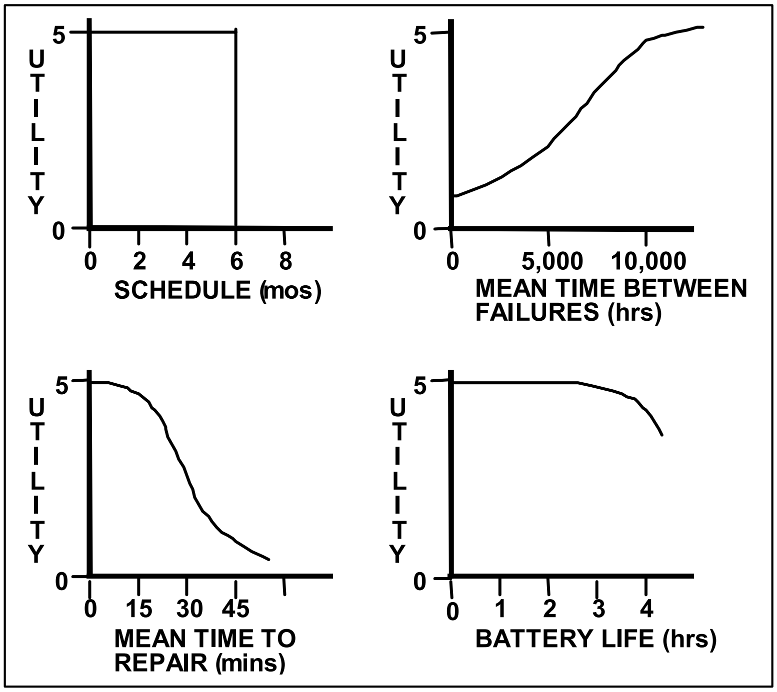


Figure 20. Utility Curves – Providing Insight into Criteria Sensitivity.

Functional Analysis Processes

Functional Analysis is one of the major Systems Engineering activities/processes. Two extremely important benefits of Functional Analysis are that it discourages single-point solutions, and it aids in identifying the desired actions that become lower-level functions/requirements. Design teams typically include experts in the product field. Their knowledge makes for a better design. The drawback to that approach is that those with extensive design experience tend to start designing items before sufficient requirements have even been identified. It's like a reflex; they can't help it. Designers often drive towards single-point solutions without sufficiently considering/examining alternatives. Functional analysis yields a description of actions rather than a parts list. It shifts the viewpoint from the single-point physical to the unconstrained solution set. Although this may sound like functional flows deal only with the abstract, that is not the case. The set of functional flows eventually reflects the choices made in how the system will accomplish all the user's requirements. This characteristic is more apparent as you progress to the lower levels of the functional hierarchy.

Products have desired actions associated with them. These are usually actions that are visible outside the system/product, and directly relate to satisfying the customer's needs/requirements. Those that are internal to the system/product reflect functional and physical architectural choices made to implement the higher-level functions/requirements. Actions/functions are of interest in Systems Engineering because they really reflect requirements. Requirements associated with subordinate functions, themselves, will have to be accomplished by subordinate system elements. Functions, their sequential relationships, and critical timing need to be determined clearly to derive the *complete set* of performance requirements for the system or any of its subordinate system elements. For more information and example approaches to performing functional analyses, see APPENDIX C-2 FUNCTIONAL ANALYSIS TECHNIQUES.

Function Analysis Limits — Unfortunately, function analysis by itself does not adequately describe a product. Function analysis does not describe limitations, iteration, complete information flow, performance, or environments. However, it is a significant and essential tool is systems engineering activities. One method of relating these attributes to functions is the Quality Function Deployment (QFD) tool.

Quality Function Deployment

Quality Function Deployment (QFD) is an excellent tool for both planning and requirements flowdown. It combines elements of the cross-correlation chart and the self-interaction matrix. QFD is also useful in decomposing requirements to lower levels of the system. It integrates many of the systems engineering activities and tools. Interestingly, Quality Function Deployment began in Japan about the same time that J. Douglas Hill and John Warfield published a paper called "Unified Program Planning" in 1972 that describes linking correlation and self-correlation matrices. QFD might be based in systems engineering, but it integrates the planning and flowdown beautifully. It provides information including answers to:

What is important to the customer?

How can it be provided?

What relationships are there between the "WHATs needed" and "how accomplished?"

How much must be provided by the "HOWs" to satisfy the customer?

The most popular QFD tool (Figure 21) utilizes a series of connected correlation matrices to graphically represent interrelationships for analyzing requirements and allocating them to system elements. The graphic is called the "House of Quality" because the self-correlation matrix at the top resembles a roof. Individual areas within the graphic are called "rooms." The core of the house is a cross-correlation matrix which shows the relationship of the driving requirements (the WHATs) to the implementing requirements (the HOWs).

At the top-product level, the WHATs are taken directly from the customer. Information such as "must work a long time without breaking" is organized into categories. An importance rating is assigned to each demanded quality. Prioritizing is one of the most important activities in Quality Function Deployment. In identifying and weighting top-level WHATs it is imperative to ensure that they reflect the customer's/user's viewpoint and not internal biases. Failure to do so results in products that everyone in the project thinks are great, but may not serve user's needs. Beware of the "Edsel Effect." When you begin to develop lower level HOWs, internal customers (e.g., Manufacturing, Quality, Test, etc.) may be able to contribute to the method of implementation, but not at the top level.

With the WHATs organized and rated, the next step is to describe the HOWs. The HOWs are allocated and derived requirements. At the top-product level, the HOWs describe the product features and characteristics. The WHATs and HOWs are linked by a cross-correlation matrix. Initially there may be no one-for-one relationship between the WHATs and HOWs. The matrix allows you to see unfulfilled customer demands and also features that are expensive yet do not serve the customer. Eventually, there should be a HOW for each WHAT to ensure that all customer requirements are met.

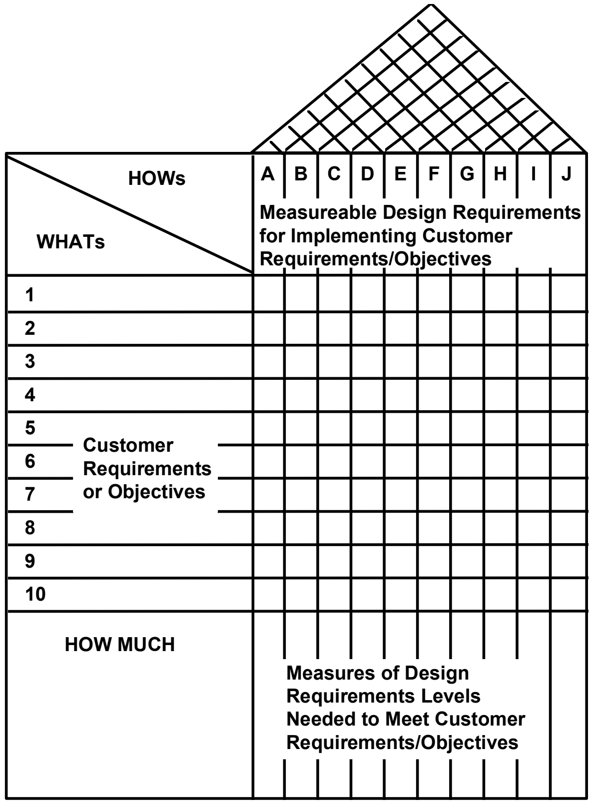


Figure 21. QFD Graphic Representation – The "House of Quality"

The HOWs have their own self-correlation matrix at the roof of the house. It identifies how requirements might reinforce, oppose, or not affect each other. The HOWs are given target values for "how much." Engineering can then do a competitive assessment on the "HOW MUCH" against benchmarks. If the value of a HOW MUCH is initially unknown, record the measure but leave the value open until it can be established.

Figure 22 illustrates the organization of a sample Quality Function Deployment chart for an automobile. Charts should be kept small, 30 x 30 or less. Use the Pareto 80/20 rule (80% of the total requirements are reflected in 20% of the possible factors). Don't ask customers about things they don't know, but be sure to capture all relevant information. An incomplete chart does more harm than good.

In relating the WHATs and HOWs, the following symbols can be used to indicate the strength of the relationship:

◉	Strong	Nominally valued at 9
○	Medium	Nominally valued at 3
△	Weak	Nominally valued at 1
Blank	None	Nominally valued at 0

The nominal value is an arbitrary weighting to allow comparison of features' worth.

Figure 21 has two new rooms. Relative Importance allows each WHAT to be assigned a value between one and five indicating how important it is to achieving the customer's perceived need. When this weighting is multiplied by the strength of the relationship to each HOW and then summed, the result recorded in the Weighted Importance lets you determine the contribution of each HOW to the overall satisfaction of the customer. In the sample chart, Manufacturing Hours and MTBF are the two principal drivers. This exercise shows how important it is to be talking to the right customer. Another group may consider comfort and luxury as the most important requirements, which would change the ratings and even cause replacement of some of the requirements.

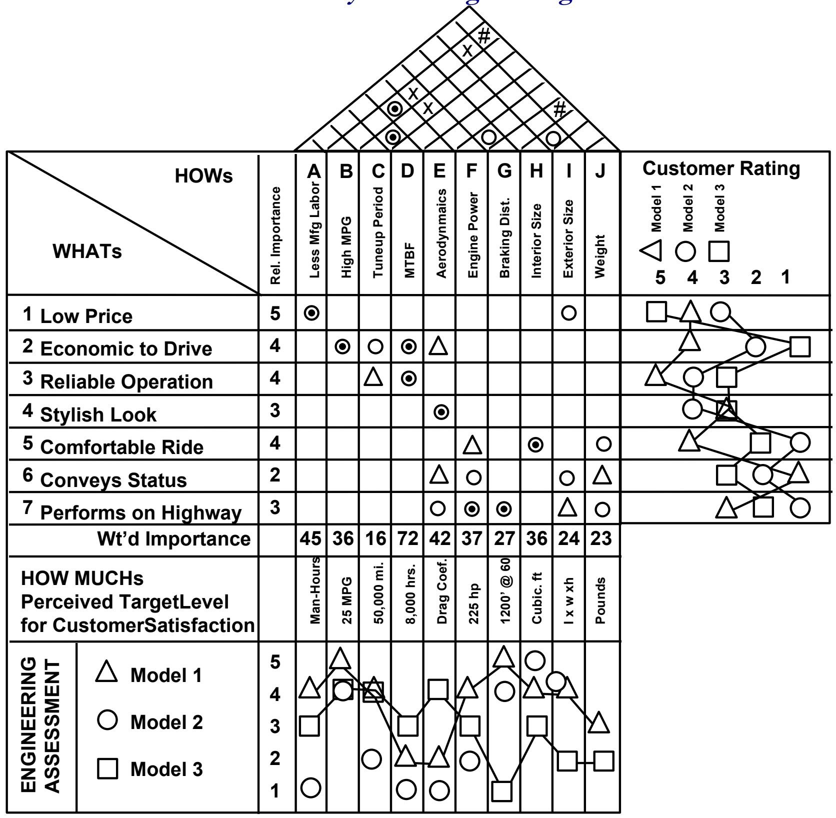


Figure 22. A Full House of Quality - Added Rooms for Greater QFD Definition

Figure 22 also adds symbols to the HOWs' self-correlation chart in the roof. These are:

◉ **Strongly** support each other

○ Support each other

X Adversely affect each other

※ **Strongly** oppose each other

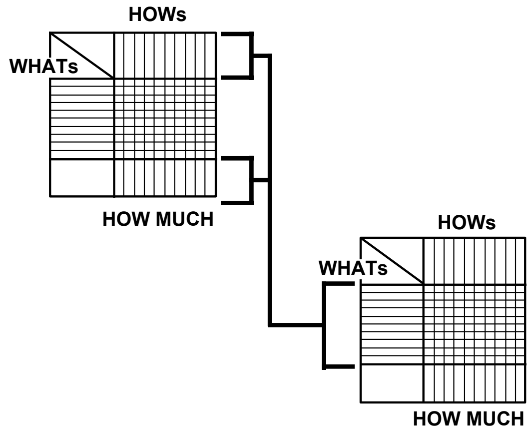
Blank Do not affect each other

Efforts should be made to eliminate or reduce HOWs that strongly oppose. Such relationships might be used to direct trade studies and research.

Other rooms added in Figure 22 show engineering assessment of how well candidate approaches meet HOW goals and also how well candidates meet customer requirements (WHATs).

At the next lower hierarchical level, the WHATs come from the higher-level HOWs and HOW MUCH (Figure 23). The requirements flow down in this manner. Quality is deployed throughout the system from the voice of the customer speaking through marketing, engineering, manufacturing, and supporting organizations.

You can purchase software that organizes and prints the charts. There is a standard symbology for relationship and correlation's. If you are a first-time user, your goal might be to just get through the first time. Mastery of the chart technique takes practice.



Pugh's Controlled Convergence

Evaluating alternatives requires a common means of measure. You must compare on a basis of equivalent standards. In addition to the weight scoring method, you can evaluate alternatives by the Pugh controlled convergence method.

Stuart Pugh of Great Britain developed a technique of selecting the best alternative by controlled convergence. In a sense, you are describing a benchmark and then improving on it. In the process of evaluating alternatives, you also generate new ones.

Pugh's controlled convergence method involves team effort. Pugh's experience is that the method makes it difficult for strong-willed people to push their own ideas for irrational reasons. The peer process is both analytic and synthetic in that both selection and creativity happen. Pugh believes that a disciplined approach leads to improvements in the product development.

The process is recursive, going through several phases to improve the initial concepts. A synopsis of the steps is:

1. Outline each alternative concept approach to the same level of detail.
2. Make a concept evaluation and comparison matrix (Figure 24) and enter approaches in the matrix.
3. Choose the criteria for the selection evaluation.
4. Choose a benchmark from the alternatives.
5. Comparing the alternatives to the benchmark, sticking to one criterion at a time. Record an evaluation for each criterion/concept pair as follows:

   \+ decidedly better

Concepts / Criteria	Concept #1	Concept #2	Concept #3	
A	+	-	+	
B	+	-	-	
C	+	-	+	
D	S	S	S	
Total +	3	0	2	
Total -	0	3	1	

Figure 24. Pugh Evaluation Matrix – Incorporating the Best Features of Competing Candidates

- \- decidedly worse
- S about the same

6. Abstain from modifying alternatives during the comparison.
7. Add pluses and minuses for each alternative.
8. Look at the negatives of the strongest alternatives. Can they be changed into pluses? Do not change the existing alternatives on the matrix, but add those modified as new additions to the matrix.
9. Look at the weakest alternatives. Can they be saved? If not, delete them from the matrix.
10. Look at all alternates for direction in improving the best, not worrying about numerical scores.
11. Repeating the steps until the design converges to a single acceptable and optimum solution.

Modeling and Simulation

Models and simulations allow you to study the effects of choices without actually building and testing a product. A model is a representation of a process or product that shows the effects of significant design factors. Simulation uses models to explore the results of different inputs and environmental conditions. Models or simulations may be actual hardware or scale replicas, mathematical programs that emulate system operation or processing response, or combinations of both hardware and programs. Often models are built to prove critical technology or to hone configurations. Simulations are used to optimize man/machine interfaces. Operational data may be fed into processing simulators to ensure proper data processing prior to committing to production software and firmware.

Models can be as simple as a picture or sketch. They can also be mathematical and statistical. Beginning models are simple and become more complex as time and understanding increase. The first step in modeling is identifying inputs that can be manipulated and determining what outputs result for the process or product under study. Then examine the effects of the environment on the product's performance. Last, the internal transfer function of the product or process to complete the model is represented. When these are tied together, your model is ready.

Traditional optimization theory uses differential calculus, the simplex method, and other mathematical techniques. Computing power is readily available through desktop computers and spreadsheets. Spreadsheets have built-in numerical functions and iteration capabilities, making them ideal for small models. The references listed in the Further Reading section are good starting points.

Scenarios

Scenarios are often used in conjunction with models and simulations. A scenario describes expected situations in which the system might operate. Applying these situations to a simulation will allow you to see the system's response and change or augment the system to improve it. Using Monte Carlo techniques and multiple runs, it is possible to simulate closely the expected environment in which the candidate system will operate.

Scenarios include outlines and synopses of proposed events concerning a customer's problem. One of the most common descriptions is the operations concept. The operations concept is a time-sequence description of event and functions in the use of a product. The term mission profile is sometimes used to include both operations concept and environmental profile. The questions answered by the operations concept include:

Why must these things happen?

What is supposed to happen?

Who or what is doing these functions or behaviors?

When do these things happen, and in what order?

The scenarios can be outlined in charts. A single chart is too confining for comprehensive information. Several charts typically show the overall operations and the details for each major operation. The information is then available for derivation of further requirements.

Joint Application Design

Joint Application Design (JAD) is a common effort performed by the system users and system designers. It centers about a structured workshop called the JAD session. The workshop has a detailed agenda, a moderator/leader, and a scribe who records the agreed-upon requirements. The beauty is in the short time it takes to arrive at requirements, agreed to by the user/customer, and recorded in real time!

Credit for the JAD concept goes to Chuck Morris of IBM who started with it about 1977. In 1980, IBM Canada adapted and refined the tool. JADs have since spread outside IBM through training courses and are now used for all types of applications, including the original management information systems. JAD tasks include:

Project definition:

- Interviewing users.
- Creating the participation list.

Research:

- Interviewing designers.
- Learning about the system.

Preparation:

- Preparing the Working Document.
- Preparing the session script.

- Scheduling the meeting.

JAD session

Final Document:

- Reviewing and updating the draft.
- Getting signatures on the document.

Using neutral, trained moderators and scribes works best. The key is preparation. For the meeting to be focused, the designers must have a good idea of the requirements for which they are looking. JAD sessions are an excellent way to converge diverse groups to an agreed specification or set of requirements. They can shorten the development time of a product dramatically by forcing all the key players into one room without disturbances.

Non-Customer Interactive Analysis

Not all requirements analysis is customer interactive. Other sources of requirements include:

Literature research.

Computerized databases.

Trade journals.

Trade shows.

Market research.

User characteristics databases (for example, anthropometrics).

Forecasting.

Modeling.

Baselining

Baselining your decisions means taking a snapshot of the design at a given point in time. Engineering is recursive and changes will happen. Having a baseline makes it easier to understand the effects of new changes, both in requirements and design. There are a variety of baseline/requirements change control tools available on the market. Currently, the SMC Program Offices of Configuration Management employ several tools to assist in baseline control activities. For example, the JPO GPS WEB contains current baseline control products including configuration lists, upcoming CCB activities, and some baseline documentation review and comment workflow capabilities.

Requirements Definition/Traceability/Decomposition Tools

One of the most important tasks of the Systems Engineer is to establish a structured requirements development process and maintain a requirements trail that traces the pedigree of every allocated and derived requirement to the lowest level. Surely somewhere along the line someone in the design/production chain is going to question the need for a particularly sticky requirement that he would just as soon not have to meet. He may be right! But even if he is, unless you know how the requirement originated you can't tell feel safe in granting relief unless you can determine its origin. Then too, he may be wrong!! Likewise, without a secure guide, extraneous requirements tend to creep in when someone thinks it would be a "good idea," or "the way we did it last time." Traceability tools help alleviate this problem.

Such tools usually employ relational databases. SMC has developed RDAV (Requirements Development and Validation) to more effectively perform requirements definition and change management. This is a Government owned tool developed by SMC, LAAFB, CA. As the system evolves from the top down, requirements, specifications, and constraints are attributed to each portion of the lower-level requirements and recorded in the database. Related trade studies, research, and analyses that lead to derived requirements are also registered. As the system design matures, designers and production management can validate or challenge any requirement. In this way, only those requirements that contribute to mission performance affect final design.

Risk Analysis and Optimization

According to DoD 5000.2-R, The PM shall identify the risk areas of the program and integrate risk management within overall program management. Systems Engineering evaluates the risk, or potential loss, of selecting an alternative as a solution. Even if a solution is the best technically, if the possible drawbacks cannot be accepted, the alternative must be discarded or modified. The need for risk analysis is not confined to the beginning of a project, but it is a continuing effort. The process of risk management is an organized method of identifying and measuring risk and developing, selecting, and managing options for handling these risks. The types of risk include, but are not limited to, schedule, cost, technical feasibility, threat, risk of technical obsolescence, security, software management, dependencies between a new program and other programs, and risk of creating a monopoly for future procurements.

Robust Design vs. Optimization

One consideration in the process is the robustness of the design. Optimal design is not always the best solution. Figure 25 illustrates this fact. Shown is a design characteristic with two possible design points. Point B is optimal because it produces the maximum Utility. However, the sensitivity of point B is such that small changes in x cause wild swings in Utility. Point A provides lower values, but it is more robust. Fairly wide variations of x cause very little change in Utility. If x is an unknown or uncontrollable factor, design point A is more desirable from an engineering and producibility viewpoint, because of its lower sensitivity to uncontrollable parameters.

Analyzing Sensitivity

Analyzing sensitivity means the sensitivity of the proposed solution to changes in the value system, requirements, or functions, as well as identifying changes in weights or scoring that might reverse decisions. Utility curves often point out peaks of optimization that might not be stable, and analyzing sensitivity can prevent selecting an unstable design.

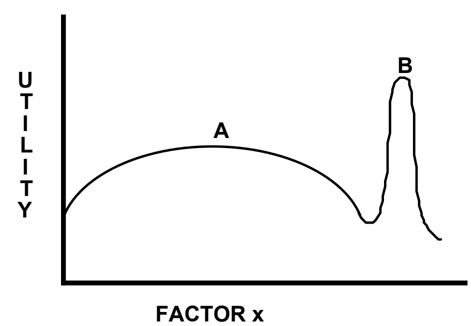


Figure 25. Robust design is Often Better than Optimum.

You might want to use optimization methods and designed experiments to determine sensitivities to changing environments and other noise. Manufacturing methods are another area you might want to cover.

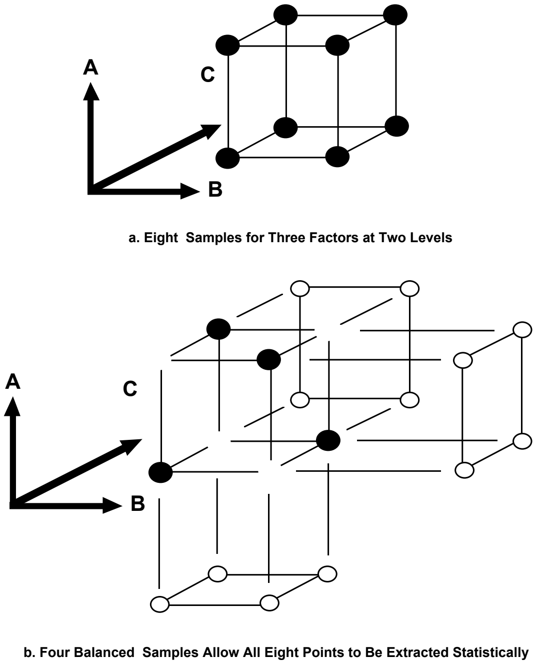


Figure 26. Balanced Experiments can Reduce Experimentation Costs and Schedule.

Optimization Through Experiments

If experiments are used to obtain optimization data, using statistical methods can reduce experimentation time. The term factor is used to denote any feature of the experiment that can be varied, such as time, temperature, or pressure. The levels of a factor are the actual values used in the experiment. Experiments can be designed for best capture of data and reduced number of experiments required. Most engineers are taught to vary one factor at a time in an experiment or simulation, holding everything else constant. This allows observation of each factor's contribution. However, if the number of factors is great, this process requires much time and does not show interactions directly.

For an example of how a designed experiment might save time and cost, suppose two sample levels are proposed in a simulation involving three factors. A three-dimensional, orthogonal representation of the testing is shown in a, Figure 26. If each of the factors A, B, and C are exercised at every point, a total of eight simulation runs is required.

In an experiment of four balanced runs (Figure 26-b), you can extract the other information statistically. The four samples can be projected onto three planes. Each of the planes contains the necessary information to extract other desired data. There are three advantages of designed experiments:

1. It takes less time to run the simulations or experiments.

2. Unknown biases are avoided.

3. Variation from day-to-day and batch-to-batch are balanced out.

The statistical techniques are not difficult. For engineering work, you can use a cookbook approach to performing the necessary mathematics. Consider asking an experienced person in experiment design for help so that you measure the factors properly.

Optimization Using the Taguchi Method

Dr. Genichi Taguchi's methodology for quality engineering optimization has been used in Japan for more than 30 years. It uses two tools, the Signal-to-Noise Ratio and the Quality Loss Function. The idea is to develop high-quality, low-cost products that incorporate robust designs that are insensitive to variability factors encountered in manufacturing and the field. This approach differs from the Go/No Go design and test methods normal to American operations. The Taguchi method borrows the Signal-to-Noise Ratio concept from communications engineering. Products with good signal-to-noise ratios are impervious to noise.

In this context, noise factors are anything over which the engineer has no control. Noise causes quality characteristics to deviate from the target, which results in a loss. The three types of product noise are:

1. External noise - variables in the environment or conditions of use.

2. Internal noise - changes that occur when a product deteriorates or ages.

3. Unit-to-unit noise - differences between individual units that are manufactured to the same specification (manufacturing noise).

The engineer does not attempt to control the noise factors. Such control is usually expensive and may be impossible. The engineer designs around the noise factors, choosing parameters and values that minimize the effects of the noise.

The Taguchi method is not aimed at identifying cause-and-effect relationships. It is not necessary to understand the causes in order to produce a robust design that is not sensitive to variations. However, the method does place strong reliance on the product knowledge of the engineer. The Quality Loss Function describes the loss to the customer for deviation from the target values. American specifications call for a pass/fail test for conformance. Taguchi shows that ANY deviation from target is a loss to the customer, EVEN an <u>increase</u> in quality if it comes at a price that is higher than the customer wants to pay. Taguchi uses a loss curve to establish the loss to the customer. The on-target loss is zero. The costs as the product moves away from target are based on tangible costs such as warranty costs. The curve can be fitted to pass through such identifiable cost points. The objective of the method is to minimize loss to the customer.

Systems Engineering minimizes loses by selecting a low-cost system design. The key parameters that allow the least variation in the presence of noise are identified using experiments, usually in orthogonal arrays. The levels of the parameters are set for least variation, again using orthogonal arrays as previously described. The results are confirmed before engineering release. Concentrating on the "vital few," only those parameters that can be controlled in a cost-effective manner are used. The designer has to find solutions to quality and cost problems caused by many factors, including those about which he knows nothing. Statistics are used to analyze the main parameters to determine how to use of their interactions to minimize the effects of unknown causes. Mathematicians fault Taguchi methods as not mathematically rigorous. Taguchi's response is that engineering differs from science, using problem-solving short cuts to get practical, not perfect answers.

The Taguchi method requires low cost as a precondition to any increase in quality. Dr. Taguchi believes that price is the primary arena of competition. Even perfect quality cannot compete if the price is too high. His three-step process to producing a product is: a) design to lower product cost; b) improve quality as much as possible through parameter design (adjusting parameters for best combination of robustness and quality); and c) perform tolerance design (similarly adjusting tolerances) as necessary. Steps b and c allow the true costs of quality to be calculated. From these data it is possible to determine the best quality obtainable at the lowest cost. Taguchi considers the three steps in the engineering of both the product, and the manufacturing system to build the product.

In engineering the manufacturing system for the product the steps are:

System design - selecting the manufacturing processes from available technology.

Parameter design - establishing the operational conditions, including materials and purchase parts sources.

Tolerance design - setting the tolerances of the process conditions and sources of variability.

The results of the Taguchi methods have also been proven in the market place and are a potent Systems Engineering tool for cost reduction and increased customer satisfaction.

Summary

This chapter presented an overview of some of the tools available to the Systems Engineer. Appendix B has a list of references for further reading providing additional information.

CHAPTER 5

What is Systems Engineering Management?

Systems engineering is both an engineering discipline and a management discipline. The engineering involves most of the activities described in Chapters 2 through 4 above. The management involves the "Control and Manage" aspect of Systems Analysis and Control discussed in Chapter 3. The technical or engineering management is similar to other management challenges in that it involves planning the work; creating, staffing and directing an organization to complete the work; monitoring progress against the plan; and taking corrective action to control the work when the plan is not realized – such is typically enough of a challenge. It is different in that it must address the highly technical and iterative processes of systems engineering as described earlier in the context of the extraordinarily complex processes that are used by the DoD to define the capabilities to be provided, to acquire the systems that provide the capabilities, to budget for the associated acquisition programs, and to manage the associated finances – the differences make it challenging indeed.

What is Management?

The classical management tasks are planning, organizing, staffing, directing, monitoring, and controlling. The tasks must usually be carried out interactively and iteratively as the system to be acquired is better defined, especially given the complexities of DoD acquisition programs. One complexity not yet addressed is that the systems engineering process described in this handbook is the overall responsibility of the SMC SPO but many of the activities are assigned to support contractors and/or the prime system contractor through one or more contracts. The allocation of responsibilities between the SPO, the prime Contractor, and the support Contractors varies from program to program.

What is Integrated Product and Process Development (IPPD)?

One management philosophy that the Air Force has used to address the complexities is Integrated Product and Process Development (IPPD). In that approach, the product-oriented Work Breakdown Structure (WBS) introduced earlier under the Work View becomes the outline for planning, organizing, and directing. The Integrated Master Plan (IMP), Integrated Master Schedule (IMS), and Earned Value Management System (EVMS) also introduced under the Work View form much of the planning. The organization mirrors the upper levels of the WBS. The IMP, IMS, and EVMS supplemented by Technical Performance Measures (TPMs) and other specific risk monitoring devices are the basis for monitoring. Controlling is via immediate action plans and longer-term updates to the IMP, IMS, EVMS, and TPMs.

Managing the Systems Engineering Process

As it is applied to systems engineering, planning has two aspects: definition of the process and organization responsibilities for implementing the process ("how") and identification and flow of tasks to apply the process to the program at hand ("what"). "How" is typically defined in either process narratives in the IMP or in a Systems Engineering Management Plan (SEMP) (or both with an overview in the IMP and details in the SEMP). In an IPPD program, "what" is defined in increasing detail in the IMP, IMS, and EVMS. In a program not applying IPPD, less integrated program management plans and schedules, such as water-fall (Gantt) or critical-path charts, may be used in lieu of the IMP and IMS.

Systems engineering management interacts with all other activities of the systems engineering process as discussed in Chapter 3 under the "Control and Manage" element of Systems Analysis and Control. It functions as the planner, manager, judge, traffic cop and secretary of the process. It plans and manages the activities, monitors and reports status to the program management, coordinates and controls technical work such as trade-off analyses and interface definition between elements of the organization, and ensures documentation of all progress, results, and decisions along with the rationale for each decision in the system decision database. It integrates the outputs of the other activities and conducts independent studies to determine which of alternate approaches is best suited to the application. It is responsible for the conduct of technical reviews and audits. It includes the planning and day-to-day activities of Risk Management, Interface Management, Configuration Management (CM), and Data Management (DM) carried out as staff functions for the Program Director or Program Manager. It identifies the items to be tracked for technical performance measurement as part of risk monitoring.

The success of the systems engineering management can be measured by the completeness and accuracy of the decision database and the degree of balance among capabilities, cost, schedule, and risk in the system solution. The decision database includes:

- trade-off and other analyses,
- requirements and requirements allocations,
- specifications,
- verification requirements, and
- all the decisions made to arrive at the design,
- the design, and
- traceability of design features to imposed specifications, requirements, constraints, and standards.

See the Glossary for a more complete definition.

The balanced system solution meets all the final requirements and is one for which all driving design decisions were made by Government or Contractor managers at a level that encompassed all products and factors affected by the decision based on comprehensive trades of cost, schedule, and risk.

Relationship of Systems Engineering Management to Program Management

The systems engineering process described in this handbook governs the technical effort on the program as a subsidiary process to the program management process. The Government program director or program manager is responsible for the implementation of both processes. He or she in turn holds SPO personnel responsible and delegates to them certain authority (1) to ensure that the technical requirements in the Contract accurately reflect the capabilities to be provided based on the decisions of the program Milestone Decision Authority and are complete and verifiable and (2) to monitor the Contractor's progress. Via the Contract, he or she also holds the Contractor program manager responsible to meet all the requirements of the contract to include the technical requirements.

Within the SPO as well as within the Contractor's organization, it is important to distinguish between the systems engineering process and the systems engineering organization. Typically, most to all of the organization has responsibilities associated with implementation of the systems engineering process while only one to a few organizational entities have systems engineering in their title. For example, in an organization implementing IPPD, teams within the SPO and the Contractors organization with names along the lines of Systems Engineering and Integration Team (SEIT) may be directly responsible to the Government program director/program manager and the Contractor program manager, respectively. The SEITs may be held responsible for day-to-day management of the overall process as well as conducting certain tasks such as allocation of the system level requirements to the teams responsible for the products at the next lower level in the product tree. Lower tier SEITs (or individuals) may have the analogous responsibilities to the corresponding integrated product team leaders or similar organizational entities at lower levels.

Relationship of Systems Engineering Management to Overall Program Cost, Schedule and Risk

The point has been made many times in this handbook that one of the desired outcomes of systems engineering is balanced capabilities, cost, schedule, and risk. In addition, the overall program cost, schedule and risk reflect the technical plan and technical execution of the plan for the program. Verification that the design provides the needed capabilities (or meets the requirements), estimating all elements of program cost, monitoring adherence to the schedules, and assessing and monitoring the risk are, therefore, all essential systems engineering management tasks, no matter how responsibility for them is assigned in the Government and Contractor organizations. Stated a different way, the assessment of all those factors is essential to monitoring the implementation of the systems engineering process on the program and the contract(s).

Earlier, the Government management systems for establishing capabilities (the Requirements Generation System), for overseeing the acquisition programs (the Defense Acquisition System), and for establishing the budget (the Planning, Programming, and Budgeting System, PPBS) were described. Other Government agencies also provide key data to the program including the threat assessment provided by the intelligence community and environmental data/phenomenology from a variety of laboratories and agencies. Obviously, for capabilities, cost, schedule, and risk to be balanced, then the capabilities set by the Requirements Generation System, direction given by the Defense Acquisition System (including acquisition strategy, schedule, and the like), the budget set by the PPBS, and other program inputs must be in balance. Typically, the relationship between these factors is as shown in Figure 27 below.

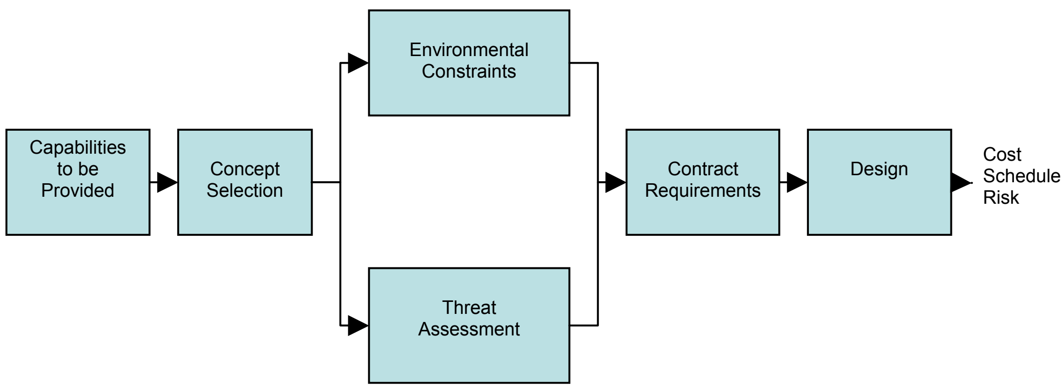


Figure 27. Typical relationship of capabilities and other program inputs to cost, schedule, and risk.

The concept selection is usually made during a program phase prior to detailed design. The environmental constraints are then predicted and the threat is then assessed based on the concept selected, and the systems engineering process prepares the contract requirements accordingly. The design that complies with the contract requirements then follows from the systems engineering process. Cost, schedule, and risk are all consequences of the development, verification, manufacture, deployment, support, and disposal of the design – none can be predicted with any certainty until the basic parameters of the concept and its design are understood. In other words, a different design will result in a different cost, schedule, and risk. Furthermore, the relationship between cost and the budget is a significant contributor to the risk – if the predicted cost rises above the budget, the risk obviously increases apace. It should be clear, therefore, that the systems engineering process has to interact closely with the Requirements Generation System, the Defense Acquisition System, the intelligence and environmental communities, and the PPBS to balance capabilities, cost, schedule, and risk. In a program where such interactions are not effective, cost growth and schedule slippage is almost certain as is an adverse impact on the careers of those involved, and program cancellation is a real possibility.

To help you understand the evaluation of capability (or performance), cost, and risk, later subsections of this Chapter address systems analysis, cost estimating, and risk management.

Planning and Organizing

The steps in planning and organizing for systems engineering include the following:

- selection of a proven process and the tailoring of that process to the next phase of the program life cycle to include the processes for risk management, interface management, configuration management (CM), and data management (DM),
- assigning responsibilities for implementing the process,
- outlining the work via the product-oriented Work Breakdown Structure (WBS),
- defining the scope of the work via the Contract Statement of Work (CSOW),
- structuring the next program phase to include the selection of major events such as reviews and audits,
- establishing an organization to carry out the work (such as Integrated Product Teams or IPTs for each major product or work area in the WBS),
- identifying what must be accomplished by each major event (such as in an Integrated Master Plan or IMP),
- scheduling the tasks to achieve complete each major event (such as in an Integrated Master Schedule or IMS),
- and planning and authorizing the detailed work/work packages to complete each task (such as in an Earned Value Management System or EVMS).

In most programs, the first and third items in the above list are specific to the systems engineering process and its output and will be treated next. The remainder are usually conducted in an integrated fashion for all work and organizational elements and heavily tailored to both the management philosophy and the objectives of the next program phase so only a few additional points will be made in the subsequent discussions.

Systems Engineering Process Selection

Selecting a proven process is the critical first step described above. Considerable attention has been given to process development since the early 1990s starting with the publication of the draft MIL-STD-499B in 1994 which details requirements for both Government SPOs and Contractors. Soon after, two standards-issuing organizations, the EIA and IEEE, issued standards based heavily on the draft MIL-STD-499B (EIA/IS-632 and IEEE P1220). Subsequently, both EIA and IEEE issued standards more attune to the general industrial setting, i.e., not specific to Government contracting. These were ANSI/EIA-632-1998[8] and.[9] Since then, many industrial firms including defense contractors have put in place corporate processes based on one or the other of these standards. It is important to note that the SPO cannot enforce compliance with such corporate processes unless such is required by the contract.

---

8. ANSI/EIA-632-1998, **Processes for Engineering a System**, available from Global Engineering Documents, 1-800-854-7179.
9. IEEE Std 1220-1998, **IEEE Standard for Application and Management of the Systems Engineering Process**, The Institute of Electrical and Electronics Engineers, Inc.345 East 47th Street, New York, NY 10017-2394.

As discussed earlier, the systems engineering process and responsibilities for its implementation are usually described in an IMP Narrative and/or SEMP. An outline for a SEMP showing the kinds of data that can be considered for inclusion is in Appendix C.1.

All required technical specialties should be addressed as an integrated part of the systems engineering process. At times, some of these are covered in separate plans but, if so, the IMP Narrative or SEMP should show how they are integrated with and support the overall technical effort on the program. To support review of the Contractor's plans and programs in those areas, Risk Management, Interface Management, Configuration Management (CM), Data Management (DM), and Operational Safety, Suitability, & Effectiveness (OSS&E) are addressed in a separate subsections below. Still other specialties are covered in Chapter 6 and verification and validation are covered in Chapter 7 below.

The Work Breakdown Structure

As noted in earlier discussions, the product-oriented Work Breakdown Structure (WBS) evolves with and reflects the physical design that is a product of the systems engineering effort so it is discussed further here. The WBS is a means of organizing system development activities based on system and product decompositions. It is a product-oriented family tree composed of hardware, software, services, data, and facilities, which result from systems engineering efforts during the development and production of the system and its components, and which completely defines the program. The WBS is prepared from both the physical and system architectures, and identifies all necessary products and services needed for the system. This top-down structure provides a continuity of flow down for all tasks. Enough levels must be provided to properly define work packages for cost and schedule control purposes.

Because the WBS is a derivative of the physical and systems architectures, it is a direct output of the systems engineering process. It can also be considered part of the synthesis process since it helps to define the overall system architecture. The DSMC *Systems Engineering Fundamentals* Book, December 2000, includes the WBS in the System Analysis and Control process as a tool to help represent and control the overall process. The WBS is thus not just about hardware or software but also is used to structure development activities, identify data and documents, organize integrated teams, and is used for non-technical program management purposes such as scheduling, and measurement of progress. A sample WBS is shown under the discussion of the Work View in Chapter 1.

The Interim Guidebook for the DoD 5000 series directives (formerly DoD 5000.2-R) lists a program WBS as a best practice to provide the framework for both program and technical planning, cost estimating, resource allocation, performance measurement, and status reporting. The WBS defines the total system of hardware, software, services, data, and facilities, and relates these elements to each other and to the end products. Program offices develop a Program WBS (or PWBS) tailoring the guidance provided in MIL-HDBK-881. The WBS is also an essential step in the preparation of the Cost Analysis Requirements Description (CARD) which is used as a basis for independent cost and other assessments. The Series 5000 Interim Guidebook suggests that Program Offices develop an overall PWBS and to initiate development of a contract WBS (CWBS) for each contract in accordance with common DoD practice established in MIL-HDBK-881.[10] The program WBS represents the total system and, therefore, reflects the system architecture. The contract WBSs relate to deliverables and tasks on a specific contract. The SPO usually develops the first three levels of the program WBS to provide contractors

10. MIL-HDBK-881, **DoD Handbook -- Work Breakdown Structure**, 2 January 1998

with guidance for lower-level WBS development. As with many standards and most handbooks, use of MIL-HDBK-881 cannot be specified as a contract requirement. Though WBS is a product of the systems engineering process, it impacts costing, scheduling, and budgeting professionals as well as contracting officers. An integrated effort including these stakeholders should be applied to develop the program WBS and monitor its application in the contract WBS.

A top level program WBS for a space system is in Appendix C.2. It can be tailored to the program at hand.

Staffing and Directing

Staffing the SPO is primarily a responsibility of the Air Force manpower and personnel systems. Direction for the program usually comes in the form of decision memoranda approved by the Milestone Decision Authority for the program and program direction memoranda from the Air Force.

Staffing by the Contractor is usually carried out by a human resources function with little oversight needed unless staffing is not as planned or personnel are unqualified. Directing by the Contractor is unique to each corporation, but should be formal. It is often keyed to the Earned Value Management System and includes formal authorization to open or close work packages.

Monitoring and Controlling

Day-to-day monitoring of the Contractor's progress is by comparing progress against the plans and schedules. The IMP, IMS, and EVMS can be particularly effective for this purpose. Though formal EVMS reports can be a lagging indicator, the contractor may collect and be able to make available data that is timelier. For example, resources such as total manpower are usually available for a given week by early in the following week. Manpower levels higher than planned, especially if part of a trend, can be an indication of a technical problem. Levels lower than planned can be an indication of a staffing problem.

Reviews and Audits

Requirements reviews, design reviews, and configuration audits provide an opportunity to assess program status in considerable detail. In particular, requirements and design reviews can be essential to monitoring at points in the program prior to the availability of test and other verification data that provide a direct indication of contract compliance. MIL-STD-1521 provides a generic summary of what to look for at each review. The System Engineering Critical Process Assessment Tool (CPAT) provides more detail on what to look for – see http://ax.losangeles.af.mil/axm/axmp/CPAT/cpat.html

Metrics & Measurement Assessments

The Carnegie Mellon Capability Maturity Model – Integration (CMMI) provides systems engineering process metrics that can provide an appraisal of the contractor's process – see http://www.sei.cmu.edu/cmmi/

Technical Performance Measures (TPMs) provide an assessment of key capability values in comparison with that expected over time. A simple example is shown in the figure below. TPMs can be valuable for risk monitoring – levels below that forecast can indicate the need for an alternate approach.

TPMs can be used in conjunction with schedule analysis and EVMS data in an integrated assessment. The schedule may provide the most timely indication, indicating a task that is taking longer than planned. TPMs can indicate the nature of the technical problem, and the EVMS data can be used to forecast a cost and schedule impact.

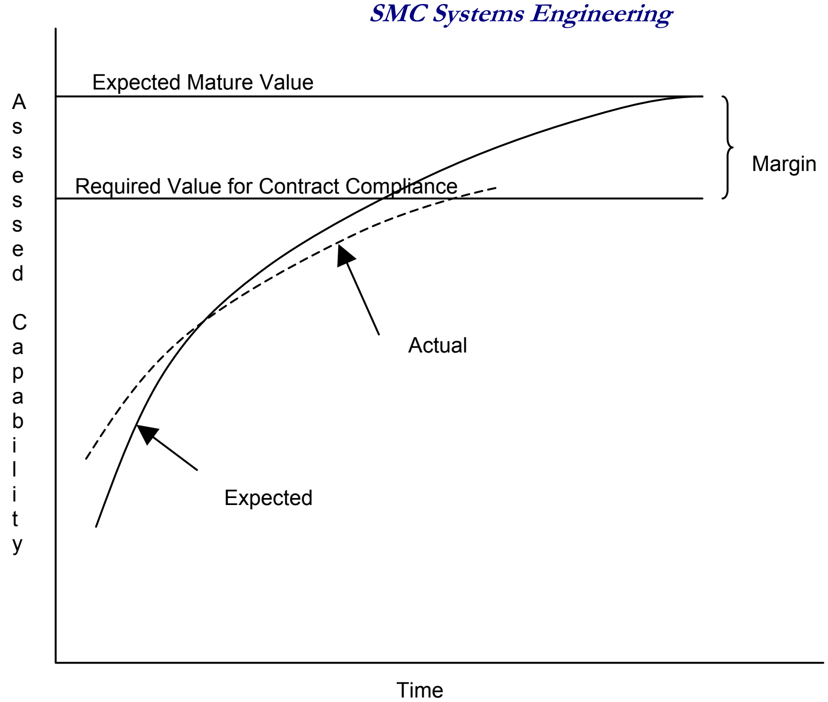


Figure 28. Notional Technical Performance Measure

Systems Analysis

THE TRADE STUDY PROCESS

Controlling the Trade Study Process

Using Models

Selecting the Selection Rule

Trade Study Process: Summary

EFFECTIVENESS DEFINITION AND MODELING

Strategies for Measuring System Effectiveness

System Effectiveness Measures

Availability and Logistics Supportability Modeling

Probabilistic Treatment of Cost and Effectiveness

Sources of Uncertainty in Models

Modeling Techniques for Handling Uncertainty

Cost Estimating

In any Systems Engineering selection process, reliable cost estimates are critical in avoiding expensive design solutions. There are presently several commercially available cost models that give fairly accurate relative hardware and software cost indications of competing approaches with even the most fragmentary design information. These models have been supplemented with more customized models developed by individual organizations and aimed at the types of systems with which they have specific interest. Most models require some training in their use and experience in interpreting results. While there is much disagreement on their absolute accuracy in predicting costs, models are especially useful to Systems Engineers in establishing relative costs in order to choose between candidate approaches. Running several models and then comparing outputs can increase confidence in model results.

Cost estimators can provide meaningful results soon after candidate system architectures begin to emerge. As the designs firm, models become less important and the estimating function turns increasingly to those in manufacturing versed in process and materials estimating. The SE should be aware of this transition. As the development phase of a project ends and EMD begins, cost estimates should be firmly based on actual cost data.

Life-Cycle Cost and Other Cost Measures

Controlling Life-Cycle Costs

Cost Estimating

Risk Management

Good risk management is based on the following criteria:

- Planned procedures – risk management is planned and systematic
- Prospective assessment – current and potential future problems are considered
- Explicit attention to technical risk
- Documentation – all aspects are recorded and data maintained
- Continuous process throughout acquisition

Successful risk management programs generally have the following characteristics:

- Feasible, stable, and well understood user requirements and threats
- A close relationship with user, industry, and other appropriate participants
- A planned and structured risk management process, integral to the acquisition process
- Continual reassessment of program risks
- A defined set of success criteria for performance, schedule, and cost
- Metrics for monitoring effectiveness of risk reduction strategies
- Effective test and evaluation program
- Documentation

Program guidelines that ensure management programs possesses the above characteristics include:

- Assess program risk using a structured process
- Identify early and manage intensively those design parameter which affect cost, capability and readiness
- Use technology demonstrations/models/simulations and prototypes to reduce risk ### GAO/NSIAD-99-162, page 68, for definitions of the TRLs
- Use test and evaluation as a means of quantifying the results of the risk handling process
- Include industry and user participation in risk management
- Establish a series of risk assessment review to evaluate the effectiveness of risk handling against clearly defined success criteria
- Establish the means and format to communicate risk information and to train participants in risk management including the program office.
- Obtain risk management buy-in at all appropriate levels of management

Types of Risks -- There is really only one type of risk: cost. Other risks such as technical and schedule are manifested in cost. That is to say, technical problems and schedule delays can usually be fixed at the expense of cost. Therefore evaluate risks against the cost baseline.

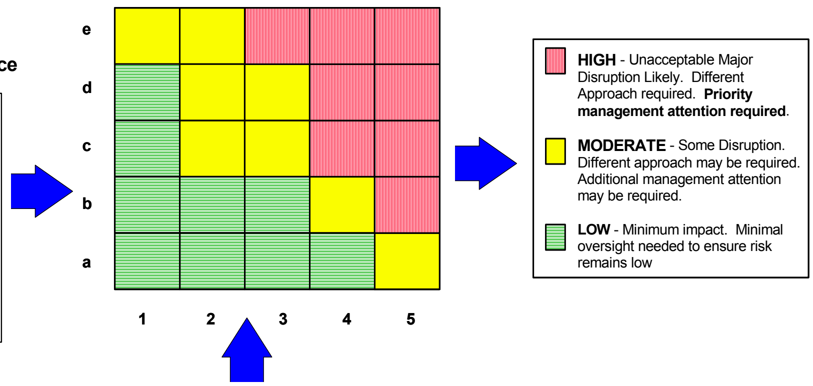
Six-Step Process of Risk Management

1. Identify the Hazard. A hazard can be defined as any real or potential condition that can cause cost, schedule or performance degradation. Experience, common sense, and specific risk management tools help identify real or potential hazards.

2. Assess the Risk. Risk is the probability and severity of loss from exposure to the hazard. The assessment step is the application of quantitative or qualitative measures to determine the level of risk associated with a specific hazard. By combining the probability of occurrence with consequence, a matrix is created where intersecting rows and columns define a Risk Assessment Matrix. The Risk Assessment Matrix* forms the basis for judging both the acceptability of a risk and the management level at which the decision on acceptability will be made. See Figure 27 below:

Probability of Occurrence

Level	Likelihood the Risk will Happen
a	Remote
b	Unlikely
c	Likely
d	Highly Likely
e	Near Certainty



Consequence: Given the Risk is Realized, What is the Magnitude of Impact

Level	Technical	Schedule	Cost	Impact on Others
1	Minimal Or No Impact	Minimal Or No Impact	No Impact	No Impact
2	Acceptable With Some Reduction In Margin	Additional Resources Required; Able To Meet Need Dates	Minor cost growth absorbable within budget	Some Impact
3	Acceptable With Significant Reduction In Margin	Minor Slip In Key Milestone; Not Able To Meet Need Dates	Cost growth exceeds budget, Mgmt. Reserves available	Moderate Impact
4	Acceptable with No Remaining Margin	Major Slip In Key Milestone Or Critical Path Impacted	Cost growth exceeds budget. Mgmt reserve is inadequate.	Major Impact
5	Unacceptable	Can't Achieve Key Team Or Major Program Milestone	Cost growth greatly exceeds budget. Large funding increase necessary	Significant Impact

Figure 27. A Risk Assessment Approach

* Note: The risk assessment matrix can be as simple as 3X3 or as large as 5X5. Furthermore, which blocks in the risk matrix are low, medium or high is a matter of discretion.

3. Analyze Risk Control Measures. Investigate specific strategies and tools that reduce, mitigate, or eliminate the risk. Effective control measures reduce or eliminate one of the components (probability or severity) of risk.

4. Make Control Decisions. Decision makers at the appropriate level choose the best control or combination of controls based on the analysis of overall costs and benefits.

5. Implement Risk Controls. Once control strategies have been selected, an implementation strategy needs to be developed and then applied by management and the work force. Implementation requires commitment of time and resources.

6. Supervise and Review. Risk management is a process that continues throughout the life cycle of the system. Once controls are in place, the process must be periodically reevaluated to ensure their effectiveness and identify any new risks.

Risk Identification and Characterization Techniques

Figure 28 describes a disciplined process for identifying , characterizing, and monitoring program risks. It is based on dedicated IPT involvement and lead by the program manager who is the head of the Risk Management Board. If a risk is characterized as moderate or high, a risk mitigation plan must be developed and implemented. Progress towards reducing risk is monitored by the RMB. In the following sections, more specific techniques for characterization and monitoring are described.

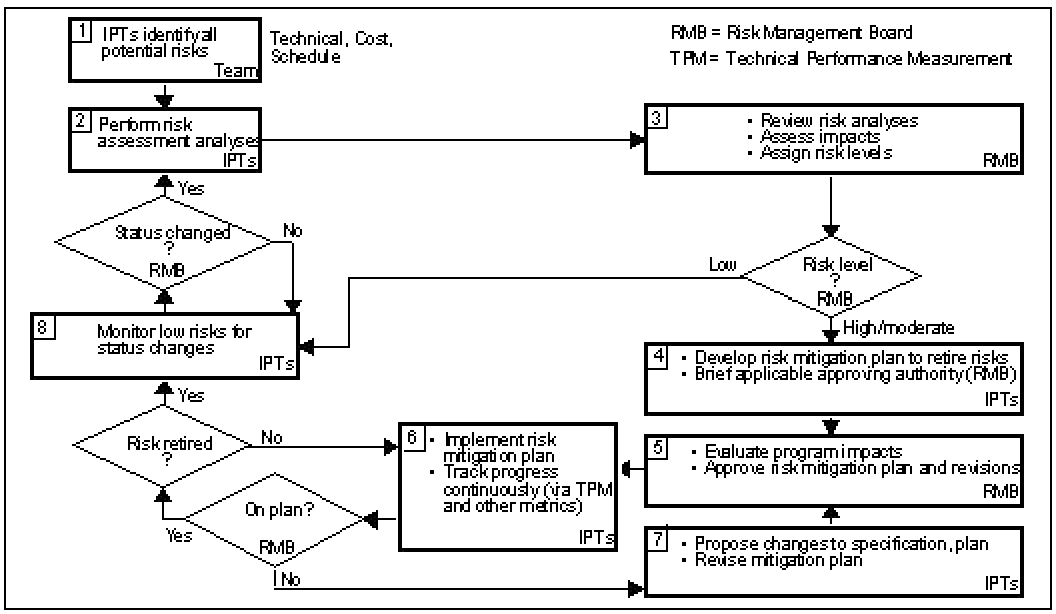


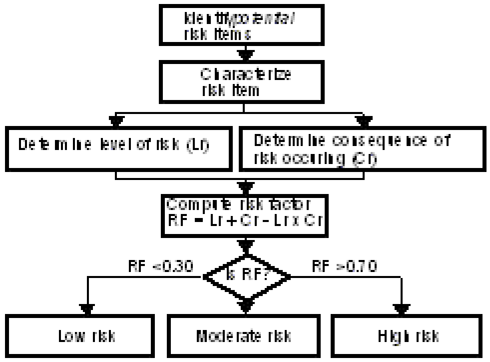
Figure 28. Process for evaluating and characterizing system risks

Figure 24A describes a disciplined process for identifying , characterizing, and monitoring program risks. It is based on dedicated IPT involvement and lead by the program manager who is the head of the Risk Management Board. If a risk is characterized as moderate or high, a risk mitigation plan must be developed and implemented. Progress towards reducing risk is monitored by the RMB. In the following sections, more specific techniques for characterization and monitoring are described.

Risk Analysis Techniques

Figure 29 describes one potential approach to quantify risk using level of risk and consequence to the system. This particular method relies on being able to assign numerical scale factors to hardware and software attributes such as complexity, maturity, and dependency to define the level of risk posed to the system. In addition values are assigned to consequence attributes such as: performance, schedule, and cost. Through a relative simple mathematical technique, an overall risk factor is calculated. Weighting factors in equations a and b are best defined using the Delphi method. When this approach is being used to compare risks between choices during a trade study, the weights can be equalized. Using an arbitrary scheme that can be modified by each program office, low, moderate, and high risk is assigned based on the range of the risk factor. A key to using this approach successfully is the ability to clearly define a quantitative set of characteristics for each attribute factor. This approach works best at the configured item level. Attribute definitions can be changed or modified as appropriate for each system.

As with any other risk management process, once high and moderate risks are identified, mitigation plans must be developed, implemented, and tracked for success.



RISK LEVELS

High	Likely to cause serious disruption of schedule, increase cost, and/or degrade performance without extraordinary emphasis and monitoring
Moderate	Can potentially cause some disruption of schedule, increase cost, or degrade performance. However, special emphasis and monitoring will probably overcome difficulties
Low	Has little potential to cause disruption of schedule, increase cost, or degrade performance. Normal effort and monitoring will probably overcome difficulties

TYPICAL TOP-LEVEL ATTRIBUTE FACTORS CONTRIBUTING TO LEVEL OF RISK (Lr)

Magnitude	Attribute factors				
	Maturity factor (LM)		Complexity factor (LC)		Dependency factor (LD)
	Hardware LM_{hw}	Software LM_{sw}	Hardware LC_{hw}	Software LC_{sw}	
0.1	Existing unit, off the shelf	Existing, 100% reuse	existing manf. proc. and proven parts	<1000 lines of code	Independent of other development work
0.3	Minor design work to modify existing unit	>70% reuse of existing software	<25% new parts or manuf. processes	>1000<10000 lines of code	Depends on work being currently funded with no schedule overlap
0.5	Major work remains to complete new design	<70%>50% reuse of existing software	>25%<50% new parts or manuf. proc.	>10K<100K lines of code	Depends on work being currently funded with some schedule overlap
0.7	Technology available design work has not begun	<50% reuse; some new algorithm development req'd	>50% new parts or manuf. processes	>100K<500K lines of code	Depends on work not currently funded with no schedule overlap
0.9	State of art, some research needed	State of art, all new algorithm devel. bn	New manuf. tech or broad use of new parts	>500K lines of code	Depends on work not currently funded with some schedule overlap

TYPICAL TOP-LEVEL ATTRIBUTE FACTORS CONTRIBUTING TO CONSEQUENCE OF OCCURRENCE (Cr)

Magnitude	Attribute factor		
	Technical factor (Ct)	Cost factor (Cc)	Schedule factor (Cs)
0.1	Minimal or no consequences, unimportant	Cost estimates do not exceed budget	Schedule stable; satisfactory slack; Inchstone slips do not affect IPT milestones
0.3	Small reduction in perform-ance margins; all reqmts met	Cost estimates exceed budget by up to 5 percent	Inchstone slips (less than 1 month), minor adjustment in IPT milestones; positive schedule slack remains
0.5	Marginal or zero performance margin; reqmts still met	Cost estimates increased by 5 to 20 percent	Inchstone slips (between 1 and 3 months); significant adj. in IPT milestones; positive schedule slack remains
0.7	Negative performance margin for 1 key technical factor	Cost estimates increased by 20 to 50 percent	Inchstone slip in excess of 3 months or slip causes zero or negative schedule slack
0.9	Negative performance margin for >1 key technical factor	Cost estimates increased in excess of 50 percent	Schedule slip that affects other IPT milestones or has possible affect on system milestones

Equation a.: $Lr = \alpha1\ LMhw + \alpha2\ LMsw + \alpha3\ LChw + \alpha4\ LCsw + \alpha5\ LD$ where $\alpha1, \alpha2, \ldots$ are weighting factors whose sum equals one.

Equation b.: $Cr = \beta1\ Ct + \beta2\ Cc + \beta3\ Cs$ where $\beta1, \beta2, \ldots$ are weighting factors whose sum equals one.

Figure 29. A semi-quantitative approach to evaluating risk

Risk Mitigation and Tracking Techniques

Figure 27 above is an example of a technique used to assessment and status each system risk. This is a conventional approach that evaluates the likelihood of the risk causing a problem and the manifest consequences to the system to the problem. Colors identify the level of risk rising from green (low) to red (high). Cell A-1 is the lowest level of risk whereas Cell E-5 is the highest level. Risk mitigation plans must be developed for unacceptable risks. As these plans are worked down, the status of the level of risk will change according to the schedule developed in the plan. Eventually, all unacceptable risks will be lowered to the green range. This approach has an advantage of being able to quickly and clearly summarize results for presentation purposes.

Risk Management: Summary

How does a risk management approach work successfully?

- Senior project management must be involved to properly allocate priorities and to get performer buy-in. If the program manager does not think risk management is his/her job, the risk management efforts will not be very effective.
  - Project manager and system engineer lead the process -- learn the process, then train the team
- Integrate the risk mitigation plans into the project plan
- Celebrate the risk reduction victories
- Establish a *risk-reducing culture* on the project
  - Do not treat risk management as an add-on or parallel activity -- it is an integral part of the daily project discussion

A *Sample Risk Management Plan* outline is provided io Appendix C-.3. Other good sources on Risk Management include:

Defense Acquisition Deskbook – see http://deskbook.dau.mil/jsp/default.jsp

AFMC Pamphlet 63-101, **Risk Management**

SMC Risk Management Critical Process Assessment Tool (CPAT) – see http://ax.losangeles.af.mil/axm/axmp/CPAT/cpat.html

Air Force Pamphlet 91-215, **Operational Risk Management (ORM) Guidelines And Tools**

DoD 4245.7-M, **Transition From Development To Production**

Interface Management

The important steps in interface management include (1) development and documentation of the interface constraints in interface specifications, interface control drawings or documents (ICDs), or the like, (2) review and approval of the documentation by those responsible for all affected products including the responsible systems engineers, and (3) a means to control changes. Documents such as ICDs are often published with data that is yet to be determined, confirmed, or fully resolved. As a result, a means is needed to reach agreement by all affected on the date for the item to be completed and to monitor timely completion. These responsibilities are often assigned to a team with a name such as Interface Control Working Group (ICWG). In an organization that does not fully reflect the WBS or product tree, the ICWG or equivalent must be situated at the top of the organization and managed by the program manager or someone acting with his or her authority. In a product-oriented organization, ICWGs can **also** operate at lower levels but still need adequate oversight from leaders and systems engineers at higher levels to ensure that their actions do not affect requirements or interfaces outside their purview. Alternatively, the function of the ICWG can be included as part of the configuration management process discussed next.

Change Management/Configuration Management

Change management is an important responsibility of any acquisition program. Generally, SPOs put in place formal change procedures for all requirements that are to be placed on contract such that the initial RFP and all subsequent contract changes are approved by the program director or program manager, the chief systems engineer, the director of financial management, and the contracting officer. Such change procedures would normally handle changes to the system requirements documents such as system specifications and system-level (system-of-systems) interface specifications. These are the top-level configuration documents for the system.

The contract may also require that the Contractor manage and control the configuration of lower-tier products. To understand configuration management, some key definitions, extracted here from the Glossary in Appendix A, may be helpful:

configuration	The functional and physical characteristics of an item as documented in a baseline and ultimately achieved in a product or process.
configuration item	An item that satisfies a documented set of requirements and is designated for separate configuration management to include any item required for logistic support or designated for separate procurement. A configuration may be either a hardware or computer software.
configuration baseline	The configuration document(s) or database(s) that record the initially approved set of requirements and/or product solutions and all approved changes thereto and that is changed only by formal, documented procedures.
configuration management	For configuration items, (1) the identification and documentation of the configuration, (2) the control of changes to the items or their documentation, (3) configuration status accounting, and (4) the auditing to confirm that conformance to all requirements has been verified.
configuration control	Formal change control for configuration items.
configuration status accounting	For configuration items, the recording and reporting of (1) the approved configuration baseline and identification numbers, (2) the status of proposed changes, deviations, and waivers, (3) the implementation status of approved changes, and (4) the configuration of all units of the configuration item owned by the Government.
component	An item that is viewed as a separate entity for purposes of design, manufacturing, software coding, testing, maintenance, contracting, reprocurement, record keeping, or configuration management. A configuration item is a component, but all components are not necessarily configuration items, i.e., they may be controlled by other than formal configuration management procedures. Hardware components may be further divided into additional components; software components may be further divided into additional components and/or software units.
computer software	The complete set or any item of the set of computer programs or instructions in the physical hierarchy and the associated documentation.
computer software unit	A subdivision of a computer software component.

Where the contract requires the formal identification and control of the configuration of certain products, the contractor should have procedures in place, as part of the systems engineering process, for determining the corresponding configuration items and their configuration baseline as well as for managing their configuration in accordance with the contract. For all other products, the contractor's decision database should identify the configuration and include means for controlling changes.

Since most space systems, once launched, cannot be retrieved and repaired, it is critical that they be designed and verified to survive the environments they will experience during storage, transportation, launch, and operation in space. The design and test for environments such as vibration, shock, and acoustic environments is addressed in MIL-STD-1540 which is based on the premise that space hardware should be verified to tolerate the expected worst case for such environments. Such testing is not always practical at the satellite level and is seldom practical at the launch vehicle level. Accordingly, MIL-STD-1540 establishes design and verification margins for the key environments at both the **component** level, as defined in the above table, and, where practical such as for satellite thermal vacuum design and testing, at higher levels.

Certain configuration items are **critical** such as because they represent single point failures or incorporate one or more parts such as a battery with a limited life. The contract may require that such items be identified and given special management attention. An example of a list of such Critical Items is in Appendix C.7

CM Monitoring and Control

Baseline Management and Evolution

Status Reporting and Assessment

Data Management

Much of the data produced on a program is technical in nature and describes a technical result, a plan to achieve the result, and/or the basis for the result. Hence, the content, the control, and the archiving of the data should be managed as a part of the systems engineering process and with the oversight of the responsible systems engineers acting under the authority of the program manager. Specifically, data should always reflect the balanced consideration of all the products in the product tree that could be affected by the matters under consideration to include the interfaces between those products.

Data often has to meet other requirements and so may also come under the purview of contract, data, and other specialists. Such other requirements and oversight should never be allowed to detract from the technical content and timeliness of the data.

Operational Safety, Suitability, & Effectiveness (OSS&E)

The OSS&E Assurance program implements Air Force Policy Directive (AFPD) 63-12, Air Force Instruction (AFI) 63-1201, Air Force Materiel Command Instruction (AFMCI) 63-1201, and Space and Missile Systems Center Instruction (SMCI) 63-1201. It also implements AFPD 63-13, “USAF Space Flight Worthiness Certification”. This program applies to developmental, operational and fielded SMC

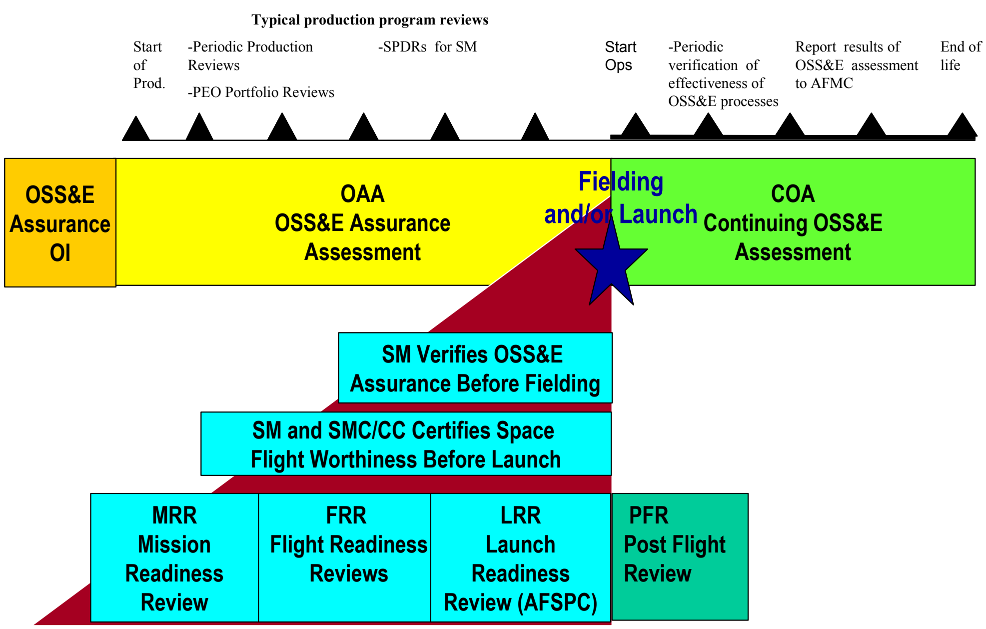


Figure 28. OSS&E Process for Operational and Fielded Systems

assets

The purpose of the OSS&E assurance program is to implement a process for establishing and preserving the OSS&E space, launch, and ground/ user baselines or end items over their entire operational life.

The Program Office structures and manages the implementation of the OSS&E assurance process throughout the life cycle of the system. Prior to fielding a new system, the Program Office verifies that the system is operated in an operationally safe, suitable, and effective manner and that the OSS&E baseline is adequately maintained throughout its operational life. The Program Office also certifies that the Space Flight Worthiness of the system at the Flight Readiness Review (FRR). Certification is made to the SMC/CC in accordance with established criteria The JPO documents the method of compliance with these criteria.

The OSS&E Assurance Process for an SMC mission consists of two major portions; an initial assurance assessment and a continuing assessment. The OSS&E Assurance Assessment (OAA) includes processes leading up to the fielding of a system, end item or launch of a satellite. The Continuing OSS&E Assessment (COA) is concerned with continuing OSS&E activities throughout the operational life of the fielded asset. The OAA is a phased assessment of the system and consists of a series of programmatic and independent assessments performed during the acquisition, manufacturing, and mission preparation phases. The scope and type of reviews are based on a program level of maturity. Specific Program Reviews, System Program Director Reviews, and PEO/DAC portfolio reviews are conducted for these modernized systems or end items.

The readiness and mission reviews are conducted before launch is shown in Figure 28. Specific readiness and mission reviews will be tailored to meet program needs. The Space Flight Worthiness Certification will be accomplished at the FRR. The PFR provides a connection between OAA and COA as lessons-learned from missions are fed back to subsequent pre-flight preparation activities. Descriptions of the reviews are found in SMCI 63-1201.

CHAPTER 6

What are the Companion Disciplines to Systems Engineering

Some of the Systems Engineering interfaces with the design team are obvious. What Systems Engineer would ever try to develop a system without checking with the Design Engineers to assure that it can be implemented with existing or projected designs and within the capabilities and resources of his company? Many of the other interfaces may seem less compelling, but are no less important. In the past, some enlightened Systems Engineers might even have had a passing conversation with Manufacturing or Reliability, but it was a rare day when the poor Logistician ever got a chance to see the product very long before it hit the field. Often the race was to finish the design before those Cost Estimators could get in and muck things up.

Times have changed. In a world where there is strong competition for limited resources, performance is no longer the only criterion. It has been recognized that manufacturing, quality and support costs heavily outweigh those of development. Integrated Product Teams (IPTs) are now common at SMC. In IPTs, specialists in manufacturing, testing, quality, reliability, maintainability, support, costing, and personnel subsystems have a chance to influence system design in the early stages while modifications are relatively inexpensive to implement. One of the most important jobs of the Systems Engineer is to ensure communication among all disciplines. Only when all voices are heard can a truly balanced system be achieved. It is up to the Systems Engineer to orchestrate these disciplines to achieve a product that is producible, reliable, supportable, economical and meets the users' needs. Programs that survive provide an optimum blend of performance and these factors.

This chapter is an overview of the interfaces a Systems Engineer might generally encounter. As with the preceding chapter on tools, it is not meant to be exhaustive. What is intended is to give you a feel for the disciplines and what you might expect from their practitioners.

Design

Good Systems Engineers work hand-in-hand with designers. If you develop a good set of requirements and several reasonable system architectures, designers can tell you how they might be implemented and the risk associated with each approach. Be flexible and allow innovation. Brainstorming can be effective in stimulating both you and the designers. As the designers begin hanging meat on your system's bones, requirements might come into sharper focus, requiring further refinement or modification. Trade studies and analyses using the tools of the previous chapter help select viable candidates and establish firmer requirements. Such requirements are usually recorded in System Requirements Documents (SRDs) that initially may have only "TBDs" (To Be Determined), but which fill up as the design matures.

In modern systems, software is as important as hardware, and because it is so labor intensive, often more expensive. Usually also greater system performance versatility and flexibility is realized if certain functions are implemented through programmable processing. For these reasons, when forming IPTs ensure that a software design representative is included for systems where processing is a formidable part of the expected candidate systems.

It's a given that Designers like to design. For this reason, the Systems Engineer must be ever watchful of design for design's sake. Market analyses may uncover commercial products fully capable (or reasonably so) of fulfilling some system functional needs with little or no modification. Only in rare instances requiring unusual performance do engineers design their own power supplies or RF plumbing. But complete subsystems, such as receivers, may easily be adapted to the intended use with great savings in time and money. The use of COTS (Commercial Off The Shelf) equipment is becoming more popular in this era of declining funds for development. COTS often provides savings not only in development, but also in support areas such as data, training, provisioning support equipment, etc. Similar savings may be derived through use of NDI (Non-Developmental Items). These are system elements developed on other programs which can be used in original or modified form in the proposed system. As with COTS, NDI avoids many of the costs and headaches associated with new design. Systems Engineers should actively search for COTS and NDI. The savings realized may eventually save the program itself! There is a caution that must be stated with regards to use of COTS and NDI. **COTS items must be carefully evaluated to establish that they can really satisfy the requirements**. Environments, life cycles, and overall reliability, for example, may not be met. Consider an evaluation similar to that used for "qualification by similarity" before baselining COTS or NDI.

Design Engineers are involved in most programs nearly as long as the Systems Engineer. Starting soon after concepts are first identified, they contribute throughout development and into the production phase. After deployment, designers are called upon to provide fixes for field problems and modifications as changing needs, environments, or threats surface. During this period, it is the Systems Engineer's responsibility to assess the system effects of any change, maintain tight configuration control, ensure that proper consideration is given to other disciplines affected, and oversee the introduction of the change.

The relationship between Systems Engineering and Design is close and generally well understood. Many Systems Engineers have extensive prior design experience and hence are conversant with both areas. For this reason the interface will not be belabored here.

Research

Systems Engineers may interface with Research as users and/or patrons. As the requirements firm, Research may be asked if there is anything in the pipeline that might provide advantages over present technology in accomplishing the required functions. Research help need not just come from internal departments. Literature searches, trade or scientific journals, trade or industry shows and seminars, etc. may identify work conducted elsewhere that might provide complete solutions, or at least clues to some of your pressing systems problems. On the other hand, Systems Engineering may commission a research project to determine the feasibility of a critical component or process. In commissioning research it is imperative that you clearly define your requirements and a timetable for the results. Also, be practical in your requests in terms of performance and schedule. If your design relies on travel at light speed, don't expect a working model before your next birthday. Consider also the possibility of later infusion of updated technology. Often you can design around an area requiring advanced technology and then incorporate the new research product later in the development, in subsequent production, or even retrofitting in the field. All these approaches have their own costs and risks, so beware and don't design yourself in a corner where a major breakthrough is your only salvation. Not that many people win the lottery! There are levels of risk associated with using new technology. Obviously the most risky level is depending on technology being developed by another program that has yet to begin. The least risky approach is for your own program to assume the development responsibility. It is very important that budget and schedule be coordinated with key program milestones along the way.

Research has limited representation on IPTs. They are seldom directly involved unless new technology is THE major factor for success. When they do participate, it is usually only in the initial phases of development or modification. Rarely do they follow a program into production except for exotic technologies where "art' is not easily transformed to "practice."

Manufacturing & Producibility

One of the major goals of IPTs is to develop products that can be efficiently manufactured. For this reason it is essential to have early manufacturing representation in the IPT. Manufacturing can identify cost, schedule and production difficulties that can aid in the trade offs of requirements in the Requirements Loop and candidate approaches in the Design Loop. Interaction with Systems and Design Engineering can result in minor changes in system/subsystem/unit design that have major impact on the cost and ease of production. The roots of the Manufacturing Plan should be in IPT participation and the plan should grow in concert with the system design.

Often those things which enhance the producibility of a product also have beneficial impact on testing, reliability and support but not always. Certain means of functional division, interconnection or assembly may improve producibility but adversely affect testability or reliability, or add to the problems of maintenance, servicing, provisioning or even operation. Achieving balanced system design requires that the other disciplines in the IPT be recognized as important contributors to the finalization of manufacturing decisions. Manufacturing involvement grows from early design through production. They are also involved in spares manufacture, in modifications and in producing retrofit assemblies and kits.

Reliability and Maintainability

Many times you will see Reliability lumped with Maintainability (i.e., R&M). While these disciplines are related, interactive and often performed by the same personnel, their perspective is different. Reliability is directed toward assuring that the given design attains the longest possible continued operation (high Mean Time Between Failures — MTBF) and operating life. Maintainability is directed toward achieving the reliability inherent in the design through servicing and maintenance, and efficiently restoring the system to operation should failures occur.

Engineers working in the R&M field deal with a number of Reliability and Availability terms and concepts with which the Systems Engineer must be conversant. Reliability is the probability that a product will perform without failure over a stated period of time and under a given set of conditions. The inherent Availability (A_I) of a product is a measure of the designed-in probability that the product is ready for mission use. It is based on the reliability of the product, reduced by factors related to the time required for maintenance actions (servicing, preventive maintenance, troubleshooting and failure repair). Operational Availability (A_O) is A_I further reduced by factors related to down times caused by such items as administrative delays (e.g., not having the right part or person available to complete a maintenance action) or longer than expected mission times. Inherent and operational dependability are similar terms used to measure the ability of a system to complete its mission once it starts. In space systems, dependability usually applies to the space element while availability and dependability can both apply to the ground element. A_I and D_I are essentially within the control of the Systems Engineers and Reliability and Maintainability engineers. However, they and the ILS engineers must work closely with the customer/user to assure that the A_O/D_O achieved is as close as possible to the inherent Availability/Dependability (A_I/ D_I). Appendix E provides an example of how a system engineer can apply these principles to a real world problem.

Reliability

Reliability and Availability/Dependability goals are usually specified in the contract or the user/customer requirements. Reliability engineers can review candidate approaches and give some indication to the SE of the relative chances of each candidate meeting the MTBF goals. As requirements are firmed, Reliability can comment on the feasibility of requirements, techniques (redundancy, fault tolerance, HiRel, etc.) that must be employed to meet them, and the methodology and cost involved in verifying achievement through tests and demonstrations. This kind of information is essential to the SE in selecting viable system candidates. Consequently, Reliability should be involved as approaches are formulated and functional analyses are performed in the Requirements Loop. Their involvement increases in the detailed design phase and decreases as the system enters production. After deployment, Reliability monitors field reports to ascertain the need for changes to improve system reliability and/or fix areas where unexpected reliability problems occur.

Designing a reliable space-based system requires use of proven techniques and engineering discipline. They include the following:

- use of redundancy at the unit level
- fault detection, isolation, and correction at the unit level
- rigorous thermal control of electronic units
- selection of electronic piece parts which are resistant to; degradation in the expected radiation environment to be encountered, and latch-up due to single event upsets
- adequate derating of electronic piece parts for electrical stresses and radiation environments encountered
- systematic approach to evaluating the design for potentially mission-catastrophic single point failure modes
- adequate margins for wearout items and consumables
- adequate margins for structural and thermal loads

There are standard reliability analysis tools and techniques available. Some examples include:

- reliability models, analyses, and predictions as defined in. The foundation of a reliability model is the reliability block diagram (RBD). It is a top down symbolic logic model generated in the success domain. Simple RBDs are constructed of series, parallel, and a combinations of series and parallel elements. Blocks may depict events or elements in a system. By applying appropriate probabilistic success functions to each block, an overall value for system success, over a defined time period, can be calculated. Mil-Hdbk-217 describes analytical approaches that may be used to evaluate system designs. It includes methods for applying electrical and thermal stress to electrical, electronics, and electro-mechanical (EEE) parts to further refine reliability models. Failure data in the handbook corresponds with recent statistics for EEE parts. Failure data for mechanical parts are more difficult to obtain. A potential source is the non-electronic parts reliability database (NRPD-25) collected and maintained by the Reliability Analysis Center in Rome, NY.

Note: reliability models using Mil-Hdbk-217 data usually result in conservative predictions of mean mission duration.

- failure modes, effects, and criticality analysis (FMECA) as defined in Mil-Std-1629 – The FMECA process is a disciplined approach to identifying the failure modes of a system. It is a bottoms up tabular technique that explores the ways or modes in which each system element can fail and assesses the consequences of these failures. The FMECA also addresses the criticality and risk of each failure. Countermeasures can be defined and consequent reduction in risk can be evaluated. FMECA is a valuable tool for cost and benefit studies, and to implement effective risk mitigation and countermeasures. Of particular interest are those mission catastrophic modes which my be the result of a single failure in the system. For each single point failure mode resulting in serious consequences to the system, a critical item control plan should be developed. The implementation of this plan should mitigate that failure mode.

- fault tree analysis (FTA) – It is a top down symbolic logic model generated in the failure domain. This modeling technique traces the failure pathways from a predetermined undesirable condition or event (top event) of a system to failures or faults that could act as causal agents. FTA includes generating a fault tree. It is very useful in graphically depicting the aggregate of failure modes for a system. It is also very helpful in identifying significant cut-sets and path sets. A cut set is any group of initiators that will, if they all occur, cause the top event to occur. A path set is a group of fault tree initiators, if none of them occur, will guarantee the top event cannot occur. It is particularly useful for high-energy systems (i.e., potential high severity events) to ensure that an ensemble of countermeasures adequately suppresses the probability of mishap. An FTA is a powerful diagnostic tool for analysis of complex systems and is used as an aid for design improvement.

- event tree analysis (ETA) – It is a bottoms up symbolic logic model generated in both the success and failure domains. This modeling technique explores system responses to an initiating challenge and enables assessment of the probability of an unfavorable or favorable outcome. The system challenge may be a failure or fault, an undesirable event, or a normal system operating command. The event tree presents all plausible system operating alternative paths from the initiating event. The ETA is particularly useful for analyzing command start or stop protective devices, emergency response systems, and engineered safety features.

Maintainability

Maintainability Engineers need a working understanding of Reliability concepts because they must build on the Reliability results in identifying Maintainability needs and approaches. Maintainability must work to satisfy the Availability and Dependability requirements. Prime among the factors contributing to Availability/Dependability is the MTBF, which establishes the frequency of need for corrective maintenance. Once a failure has occurred, Availability/Dependability is dictated by the amount of time necessary to return the system to operation (Mean Time To Restore Function — MTTRF). This in turn is affected by the time required to isolate the problem and to repair, switch to a backup, or replace the defective component(s). Rapid isolation is enhanced by the manner in which functional interfaces are drawn, by the inclusion of test ports for insertion and measurement of signals, and by the use of selfdiagnostic or Built-In-Test (BIT). Some factors that reduce MTTRF also have a negative effect on MTBF (BIT usually adds components and numerous interfaces may increase connections, both tend to reduce Reliability). Such lower Reliability normally places additional stress on meeting Maintainability goals. The Systems Engineer must be aware of these tradeoffs and strive for a balance that approaches both Reliability and Maintainability targets.

Maintainability also has major interaction with Logistics and Support and in many cases may be handled under the umbrella of Integrated Logistics Support (ILS). Maintainability decisions greatly affect other ILS functions and likewise, some ILS constraints (expected deployment, isolation of locations, maintenance echelons, training of available personnel, etc.) may steer Maintainability approaches. Some common goals, such as modularity and BIT, may be mutually supportive. Others, such as commonality, the need for special tools or equipment, etc., may be disparate. Often the SE is called upon to make judgments as to which approach provides the proper blend of these goals.

Maintainability Engineering starts with a general review of requirements to ascertain there are no "show stoppers." The Maintainability effort increases during the transition from the Requirements Loop to the Design Loop begins to decrease through the development effort, and usually ends after Maintainability goals have been demonstrated.

The System Maintenance Concept and Maintenance Plan

Designing Maintainable Space-Based Systems

Maintainability Analysis Tools and Techniques

Mass Properties

How much does it weight? Always the big question. Just as in life, it is always easier to add weight than to reduce it. Over the life of a system's development cycle, system engineering must carefully manage the weight budget and other mass properties of the system. Probably most critical is the throw weight of the space vehicle. But also important is the weight of transportable and mobile elements of the system. Will it fit into that C5B? Will all the equipment fit into the ISO container?

Weight estimates must be established early in the development cycle. For a new space element, it is wise to plan on having a 25 percent weight contingency at PDR and 15 percent by CDR. This is in addition to any contingency held by the program office for future capability growth. Of course these can be adjusted depending on the maturity of the hardware to be used by the system. Weight is a parameter that should be managed using a program level metric.

For the space element moments of inertia and center of mass are also important properties to be understood as the design matures, although a metric is usually not needed.

Environments and Survivability

AF Space Command and US Stratcomm place high importance on protection of space systems. The US warfighter is highly dependant on space systems to successfully complete his mission. Imagine a battlefield without the capability for early attack warning, protected communications, Intelligence, Surveillance, and Reconnaissance (ISR), GPS navigation, or weather prediction and reporting. This implies the need to design USAF space systems to operate under extreme space and terrestrial weather and in a weapons environment.

The natural environment of space is demanding. During launch the satellite must survive acceleration, vibration, acoustics, depressurization, thermal, radiated RF emissions, and separation shock. While on orbit temperature, geomagnetic radiation, solar flare particles, galactic cosmic rays, and orbital debris play together to degrade satellite performance and lifetime. During wartime, weapons effects become a significant driver. A high altitude nuclear detonation contributes significantly to the total radiation dose received by the satellite. In fact a yield in the 10s of kilotons can essentially use up the entire radiation lifetime of a low Earth orbiting satellite in the matter of several months. X-rays fluence from a single device can destroy the electronics of any satellite in line of sight in a matter of seconds. Scintillation in the atmosphere, although not life threatening, can block communications for extended periods of time. There are many other man-made threats to space assets including: high and low energy lasers; kinetic energy kill vehicles; and ground and air based RF jammers to name a few.

The natural environment on the ground can be equally challenging. If a ground element is based in a fixed facility at Schriever AFB or Buckley AFB, the system deals with normal local weather conditions through normal environment management systems. In addition, buildings and antennas need to be designed to survive thunderstorms, wind storms, ice storms, snow, attack by wildlife, fungus, fog, blowing sand and dirt, lightning, and seismic events. If the facility is being attack during wartime, it may have to endure long enough to switch control over to an alternative facility or to survivable mobile elements. This implies protection and countermeasures against attack by Special Forces or terrorists, and airborne systems that may deliver nuclear or conventional, biological, or chemical weapons. If the facility is located OCONUS, attack by ground forces and local agitators is a consideration. Mobile, survivable ground elements may require protection measures from all types of threats depending on basing.

There are numerous mil-stds defining the threats resulting from natural and space weather environments. The "System Threat Assessment Report" (STAR) is developed by the National Aerospace Intelligence Center (NAIC) for each system to be fielded. This document includes the definitive sets of manmade threats to a space system. The system engineer must be familiar with this information and be prepared to make decisions on countering threats to the system under development. It is too costly, however, to have a countermeasure for every threat. It is systems engineering job to perform a threat evaluation and CAIV study to determine reasonable, cost effective countermeasures. An approach to threat evaluation and CAIV is outlined in Appendix D. Other approaches are certainly feasible.

Environmental, Health and Safety

It is required by federal law and the responsibility of each program to perform an environmental assessment and prepare a report to be reviewed by the EPA. It is possible to get a waiver but to avoid risk, it should be worked at the beginning of the program. It is important to understand that an assessment of impacts to the environment is not just a review of what chemicals used during manufacturing will pollute the ground water. It includes factors such as: Will the system create a burden on the community where production or operations will occur?; Will traffic patterns be impacted due to new employee hiring?; Will construction of new facilities be necessary to product and/or operate the system?; Will RF signals, noise, or other factors interfere with the surrounding environment?; What types of materials and processes will be used to manufacture and operate the system?; and a thousand more questions! And every question must be answered and supported by a stack of documentation. It is very desirable and efficient to hire an independent contractor to perform this specific work. The system prime contractor usually is not the correct choice. There are a number of available contractors with very good qualifications and experience successfully dealing with the EPA. It will be necessary for the system prime contractor to provide extensive information for the preparation of the report. Ensure that this interface is planned for and part of the prime contract.

In addition, there is a plethora of federal and state laws and regulations regarding the health and safety of personnel and facilities. Federal and Air Force regulations include OSHA and AFOSHA. Compliance is mandatory and should be managed by the prime contractor.

Human Engineering

Personnel Subsystems

Personnel Subsystems addresses the factors affecting the man-machine interface. Considerations include Human Engineering and the associated field of Ergonomics, man-in-the-loop requirements, decision processes and automated situation reporting, and understanding of the intelligence, experience and training of the expected operators. The SE must include such analysis in candidate system selection and development. If you require an operator who is less than four feet tall, has three arms and no regard for bodily functions, your chances of widespread acceptance of the system are nil. Personnel Subsystems should have a chance to review and comment on requirements to identify any potential problem areas, however, their expertise is not regularly needed until specific designs begin to emerge. They are particularly helpful in the layout and arrangement of controls. They should also look at maintenance functions to ensure they are workable.

Training

Closely allied to Personnel Subsystems is the Training activity. Early system tests require a cadre of trained operators, so consideration of training must begin soon after PDR. What must be decided is the kinds of personnel required, types of training to be used and the need for any training equipment. In fact, some training equipment, such as mock-ups and simulators, may even be an integral part of the testing itself to tie down proposed operating procedures and control layout. As the system advances to Operational Test and Evaluation (OT&E) training requirements increase. Many of the operators and maintenance personnel who conduct these tests have little or no prior contact with the system, and must be brought quickly up to speed. This requires prior planning of training and the training of trainers who can pass on the requisite information to the troops who will be doing the work. Training planning for OT&E and beyond should start about the time of CDR. Methodology should be established—lectures, computer based instruction (CBI), workshops, briefings, and demonstrations. Required resources must be identified—personnel (instructors and students), data (manuals, drawings, workbooks, and interactive computer programs), equipment (familiarization trainers, simulators, mock-ups, complete systems, and support equipment), and facilities (classrooms, labs, and computers). It is the System Engineer’s responsibility to blend these requirements and activities with all the other activities clamoring for recognition and resources. Shot changing Training is short sighted. Unless your system works autonomously, you’re going to need the cooperation of a knowledgeable user to accomplish the system’s mission and maintain a satisfied customer.

Quality Assurance

Quality has a dual function. It operates as a policeman to ensure that all contractually imposed specifications, standards, processes, and other design requirements are met. It also acts as an in-process and final check of workmanship, test and overall production functions. Through its role as policeman, Quality understands the legal and contractual ramifications of design and planning options and is therefore valuable as a counsel to steer away from future problems inherent in proposed implementation approaches. Quality is also helpful in establishing test programs to verify that the design meets user requirements. Once design-proofing tests are complete, Quality assures that planning and design are carried out properly in the production system. Quality may also identify slight changes in the design of individual candidate approaches that could make their job easier and less costly. As software has attained a greater importance in modern systems, so to has the need for Software Quality Engineering and Control. The Systems Engineer must be mindful of this requirement and assure that Software Quality is involved appropriately in the program.

Special problems for both the Quality Engineer and the Systems Engineer stem from the growing movement away from imposing MIL Specs on contracts. It used to be that you could just specify MIL-Q-9858A and let the chips fall where they may. The cost of this approach is now recognized and new contracts tend to specify industry, international, or even company standards in an effort to reduce overall development and production costs. As a result, the QE and SE must evaluate these standards in light of the system mission and identify any areas where those standards will not produce the required performance. Additional contractual language or particular statements in the system technical specification may be required.

Another area of growing interest is the use of statistical quality control and related techniques. The reduction in inspection touch labor support that such approaches provide recommend them highly in today's cost-conscious environment. The Quality Engineer and Systems Engineer should investigate the use of these techniques as much as possible.

Because they can be an important aid in avoiding pitfalls and future problems, Quality must be involved in the program from its inception through final disposal. Obviously then, Systems Engineers should promote a good working relationship with Quality personnel and listen well to their suggestions. If Quality concerns cause problems, it is not due just to the intractability of Quality, but more often a need to reevaluate some assumptions and requirements. It may even be a sign that the Systems Engineer should confer with the user/customer to ascertain that they are willing to assume the costs involved in reaching certain goals. Trust your Quality associates. Few Systems Engineers have gone wrong listening to their advice.

Integrated Logistics Support

Logistics and Support, or more properly Integrated Logistics Support (ILS), contains ten elements which are mini-disciplines in their own right. These elements are:

Maintenance Planning (MP) — the determination of what maintenance operations are required and the organizational level at which they will be performed.

Manpower and Personnel (M&P) — the numbers of personnel and kinds of training required at each level to support the maintenance planning.

Supply Support (SS) — provisioning and the development of data to support provisioning.

Support Equipment (SE) — planning, design and development of equipment to test, handle and service the system in the field.

Technical Data (TD) —planning and development of manuals, drawings, and related documents required to operate and maintain the system equipment at all planned maintenance levels.

Training and Training Support (T&TS) — planning development and execution of training required to implement the maintenance planning and of all the devices, mock-ups, and documentation necessary to conduct training.

Computer Resource Support (CRS) — planning and support of efforts to maintain and upgrade fielded system software/hardware.

Facilities (FA) — plan and implement the modification or upgrade of existing facilities, or the development of new facilities to support the system.

Packaging, Handling, Storage & Transportation (PHS&T) — planning the modification or upgrade of existing containers, equipment, or facilities, or the development of new ones to enclose, handle, warehouse or move complete systems or their components.

Design Interface (DI) — sum of all efforts to ensure transfer of the latest design information to those performing ILS analyses and related work, and to ensure that the results of ILS operations properly influence system design. Often these efforts result in establishment of a central database of design and support data that can be accessed electronically by all those involved in the development and use of the data.

In the past there was a tendency not to address logistics issues until the developed system was about ready to be deployed. After all, "why worry about how you are going to support it if you're not yet sure it will work?" In this era of limited resources, we have come to recognize that if we must expend nearly everything supporting systems already in the field, there will not be much left over for the new starts necessary to keep us competitive. Cost of ownership has tilted heavily toward support. ILS involvement in early design decisions has greatly reduced support costs and facilitated some of the recent initiatives invoking greater reliance on Commercial Off-the Shelf (COTS) items, Non-Development Items (NDI), and joint usage.

ILS personnel should be involved from the earliest requirements analyses through development, production, deployment and continuing operations. Because of their long-range viewpoint, logisticians tend to keep their options open. This characteristic is extremely helpful to the SE in identifying and avoiding potential problem areas.

CHAPTER 7

Validation and Verification

Validation and Verification are important to the designer because they help keep him on track to meet the customer's requirements, and they give him incremental and final assurance that the product will pass the customer's acceptance criteria. They are important to the customer because they provide proof that the product performs as specified, and they provide an indication of how well the product will satisfy his operational needs. Because of the importance of Validation/Verification, careful planning and controlling of the processes is required.

Table 3. Validation & Verification Considerations

Type	Description	Comment
Inspection	Examination by the senses (sight, sound, smell, taste, or touch) to determine requirements compliance.	Might use gauges or simple measures. Some Physical Characteristics.
Analysis	Technical evaluation of data using logic or mathematics to determine compliance with requirements.	Used in Verification when given attribute is impossible or difficult/costly to test. Commonly used to extend test results beyond range of test.
Demon-stration	Un-instrumented test — compliance determined by observation (e.g., maintenance task performance time)	Used when compliance with requirement does not require measurement of a parameter. Some aspects of Maintainability.
Test	Using procedures and test/measuring equipment to verify compliance with requirements.	Most recognized method of Verification; used also to support Validation analyses.
Process Control	Process control values accepted as evidence of requirements compliance. Process factors known, measured, and held to predetermined targets.	Use growing. Used to show dependability/ consistency of process results. Cannot be used to show that a system/component design complies with requirements.

Table 3 lists some of the considerations involved in Validation/Verification planning. As to be expected, those associated with Validation tend to be oriented toward analysis, while those associated with Verification are oriented toward test. Planning should be documented in an integrated plan that identifies what will be validated and/or verified, the method(s) to be employed, and a schedule of events. To ensure a satisfactory conclusion to the V&V process, it is necessary to plan early in the development life of the program. V&V requirements must be established to provide adequate direction for system engineers to complete the process. As an example, the Advanced EHF program built requirements V&V plans prior to the signing of the EMD contract. These plans described in detail how each individual requirement was to be assured. Information in the plan included: the requirement and its identification number (traceable through a database tool to higher or lower level requirements); any other requirements which may be verified together; verification approach (i.e., analysis, test); which test series would be used to verify or what analysis tools would be used; for analyses, was information required from a particular test to support the analysis; assumptions; inputs; outputs or expected results; and test sets required. Eventually, when V&V is completed for each requirement the individual V&V plans will include links to analytical results or test data that satisfy the V&V of the requirement. This is a very good, well thought out approach to ensuring requirements are met.

Table 4 lists some of the considerations involved in Validation/Verification control. Those associated with Verification are fairly well integrated into engineering practices, since they have been in general use and are often contractually required. The Validation controls are less well understood and implemented. Their major thrust is to document results, to integrate the results into all design decisions, and provide traceability from the designs to the related analyses. This process ensures that anyone making future changes is aware of all the factors that shaped how particular designs evolved, and can avoid possible counter-productive decisions. Recently relational database tools have been developed which assist in this process. Making such databases available to all cognizant functions though an electronic network enhances the probability of arriving at an optimum design. Systems Engineering is often the instigator and curator of the database/network combination

Table 4. Validation/Verification Control Considerations.

Vali ation	*Verific tion*
Analyses prop ly identified and defined pr r to start	Document prepar ion properly supervised and ap roved.
Analysis resul documented and cataloged r traceability	Documents are ur ler configuration con ol.
Analysis resul disseminated to design/ spec lty disciplines	Non-conformanc identified and analyzed.
Design decisio s traceable to associated ana ses	Measuring/test eq ipment calibrated to trace ble standard.

Validation and Verification Methods

The five methods normally employed in Validation/Verification to establish compliance with requirements are listed in Table 5. Analysis is the primary method used in Validation while the others are used primarily in Verification. However, some testing is done to support Validation efforts, and occasionally Verification is accomplished by analysis where testing is difficult or prohibitively expensive, where expected operational environment cannot be created (all-out missile attack), or where testing costs can be effectively reduced because similar systems have been previously tested or have a history of use (compliance by similarity).

Inspections may be used to show compliance with some Physical Characteristics (size, weight, color), and along with Process Controls, may be used Quality and Manufacturing personnel to ensure/measure quality in production.

Demonstrations are used to show successful completion of an action, either by the system/component or upon the system/component, and may be associated with some aspects of some of the "Ilities," — Maintainability, Safety, Human Engineering, etc. The SE needs to know of them, but the special province and the major focus of the SE must be on analysis and test. Analysis has been discussed at length throughout this manual. Following are a few words about testing.

Testing

Testing increases confidence in meeting customer requirements and is part of overall risk reduction. Testing is of two types: a) developmental tests; and b) qualification/acceptance tests. Developmental tests are conducted to obtain data on the operational characteristics of the test subject for use in design decisions, and are a primary part of Validation. Qualification/acceptance tests are conducted to show proof that particular designs or particular units meet design specifications and are the purview of Verification.

Validation/Verification testing of performance surfaces:

- Designs and design changes that fail to meet requirements.
- Manufacturing defects.
- Component failure or non-conformance.

Types of tests include:

- Burn-in and stress screening.
- Environmental testing.
- Variable and Go/No Go testing.
- Hierarchical level testing.
- Production assessment.
- Destructive and nondestructive testing.

Burn-In Tests are meant to get components past their infant mortality stage. By weeding out failures in this manner, the remaining test samples exhibit a higher level of reliability. Often burn-in is combined with temperature, vibration and vacuum stressing of the samples. Temperature cycling stresses the product to allow identification, replacement, or even redesign, of components that are particularly sensitive to thermal effects. Random vibration causes loose screws and parts to work free. Vacuum reduces outgassing of finishes that would otherwise contribute to contaminating surfaces in space. Such screening finds:

- Parts failure.
- Manufacturing defects.
- Marginal design.

Environmental Testing simulates the expected operating environment. In design proofing, the product may be subjected to levels greater than expected to prove design margins and as insurance that it can handle overstress conditions should they be encountered. Environments typically tested include:

- Atmospheric pressure or vacuum
- Temperature
- Solar radiation
- Rain
- Humidity
- Fungus
- Corrosive Atmosphere(s) (Salt fog)
- Sand and dust
- Explosive atmosphere
- Water immersion
- Acceleration
- Vibration
- Acoustic noise
- Shock
- Icing and freezing rain

- Electromagnetic Radiation

Variable testing records the actual value of the measurement.

Go/No Go compares the measured value against predetermined limits and determines whether or not the item is acceptable.

Hierarchical Level Testing refers to the evaluation performed at varying levels of assembly. As stated previously, it is more economical to surface problems at the lowest possible level. However, some problems that might not appear until elements are aggregated at higher levels. Such problems include tolerance build-up, race conditions, sneak paths, and stored energy hazards. For example, paralleling relays without isolation diodes will cause "chattering" relays because of stored charge in the relay coils. Hierarchical testing is especially important in software development programs.

Production Assessment Testing is done on sample products drawn periodically from production. This is an on-going verification of the production process. An example is verification of weight when the product is under configuration control. Production assessment is a check on processes and parts that might change over time, and otherwise go undetected.

Destructive Tests are performed to determine the stress level that causes the item to fail, and renders the test object unfit for its intended use. These tests must be done as samples, or nothing would be left to ship. Destructive tests are done on objects such as fuses, flash bulbs, and metallic materials.

Test and Evaluation

Test and evaluation is an adjunct of Validation. It provides confidence that the product will work before it is assembled. It identifies areas of risk for elimination or reduction during the product's development. It is also a validation of the Systems Engineering process. Test and evaluation generates information and knowledge on the developing product. It is deliberate and rational. System engineering compares and evaluates results of testing against the requirements. Test and evaluation includes physical testing, modeling and simulations, experiments, and analyses. "Test" means the actual testing of the product and components. "Evaluation" is the review and analysis of the information. The distilled information allows system engineering to:

- Define requirements.
- Manage the system engineering process.
- Identify risk
- Discover new alternatives.
- Improve product robustness.
- Find constraints.
- Decide the allocation of resources.

Design for Testing

Efficient test and evaluation demands design for testing during product development. Systems Engineering must, in its design, address the need to:

- Collect data during the development process.
- Enable easy measurement, including:
  - — Partitioning
  - — Controllability
  - — Observability

- Enable rapid and accurate assessment of the information.

Integrating Test and Evaluation

Test and evaluation must be integrated with the rest of the Systems Engineering effort. Documented decisions for test and evaluation are called the Test and Evaluation Master Plan (TEMP). The testing program in the TEMP must be consistent with the overall program management plan. The test program in the TEMP must provide the technical performance measurements required for review, audits, and risk management. Other documents integrated with the TEMP include the:

- Configuration management plan.
- Functional analysis documents.
- Requirements Allocation Sheets (RASs) and Design constraint Sheets (DCSs).
- Test Requirements sheets.
- Specifications.

Test and evaluation is not limited to the primary product. The facilities and support system need to be considered by risk reduction efforts also. For example, supportability can and must be measured.

Reducing Integration and Test Time

In this era of cost competition and short schedules, reducing integration and test time has major benefits. Of all the considerations listed in Table 6, careful attention to the first two will provide maximum return. Paying attention to what requirements must be tested, and accommodating the need for future testing to the fullest practical extent will lower costs and shorten schedules. It will also make you a hero to your test engineering, manufacturing, and quality associates. Equally important is ascertaining the level at which you will verify requirements. Attention here will avoid the use of convoluted testing arrangements or the need to tear down the product to make certain measurements.

Table 6. Considerations for Reducing Integration and Test Time

Easily verifiable requirements.
Clear identification of the system level at which each requirement will evaluated
Interface definition.
Peer walkthroughs.
Models and simulations.
Robust design to component parameter variation, manufacturing process
Robust inputs, targets outputs.
Commonality, standardization.
Simplicity.
Testability.
Reliability.
Maintainability.
Test equipment and facilities available.
Independence of components.
Hardware emulator for untested software; tested software for untested hardware.
Modular, bottom-up testing.
Understanding of the critical path.
Have test plan, test procedures ready.

CHAPTER 8

Summary

Not Commandments. Not Rules. Not even Guidelines. Just 12 undeniable facts of Systems Engineering:

1. **It ain't over 'til it's over.** — Systems Engineering is not a once-through-the-process-and-forget-it routine. It is a continuous, evolving, ever-different, program-tailored course that starts at program inception and progresses to product disposal after useful life is expended.

2. **There's no such thing as a stupid question.** — Encourage your associates to question anything they don't comprehend. If they can't understand, they can't implement your ideas. Or they may implement them incorrectly. You need to rephrase your ideas in a way that all associates understand. Then too, occasionally a question brings up something you overlooked, and the stupid question saves you from disaster!

3. **Everybody's a QA man.** — The product will be better if all are focused on product quality. Encourage everyone involved in the process to be on the lookout for potential problems. You can't be everywhere at once, and sometimes someone else's perspective uncovers items that may never occur to you.

4. **There's got to be an easier way.** — This is the essence of all engineering. Be ever mindful of the power of innovation and open to the great revelation that leads you to the better mousetrap.

5. **There's no easy way to do anything.** — On the surface this looks like a contradiction to the previous fact, but it isn't really. What this says is there's no substitute for hard work and beware of treacherous shortcuts.

6. **Humans are the only animal to invent and use tools. Be human!** — This is an admonition to make maximum use of available tools and look for ways to adapt them to the present use. (Actually some birds and sea otters use primitive tools but it's hard to work that into the basic fact statement.)

7. **We're all in this together.** — The practice of Systems Engineering is an interdisciplinary process. The development of superior products requires that all specialties have timely knowledge of all design decisions and a chance to air their views.

8. **Listen to your instincts.** — We've become so dependent on computers that we have a tendency to accept their outputs without challenge. Don't get so wound up in the process that you don't occasionally step back and look where you're going and where you've

been. Also, weigh things in light of your experience and listen to your intuition. If something doesn't look right, it may not be!

9. There's probably an upper limit to the number of times you should check your results, but you'll never reach it in any practical problem. — Check your inputs. Check your outputs. Check your checks. It's incredible how persistent some errors are. You may exorcise them out of one version of the program and find someone using a previous version. That slipped decimal point will come back to haunt you 'til they give you the gold watch.

10. A good Systems Engineer is humble. — Don't think you have all the answers. If you do, you'll just end up with the same system you designed last time. Be open to suggestions.

11. Yesterday's solutions may not be the answer — but it's the best place to start. — Don't get pulled into that "We did it this way last time " syndrome. On the other hand, the wheel is a pretty basic device that has worked well for some time now and probably needs little re-engineering. Spend your energy where it will provide the most return. You usually have to have something that works before you can make something that works better.

12. The good Systems Engineer knows when to kick it out the door. — There will always be a new device on the horizon that will give you 3 db more. Or a new technique in development that will speed processing. But if it's not needed now to make your product meet requirements, don't hold off deployment to chase that extra bit of performance. If the product as is meets today's need, it should be in the customer's hands. Add the new item when the need arises. Besides, you may learn more from a few weeks of field experience than you might get in years of experiment and test.

Congratulations if you have gotten this far!! But don't think you've got this Systems Engineering thing completely in hand. This booklet was not intended as the last word on all things related to Systems Engineering. What we hoped to do was provide some background for those who are encountering Systems Engineering for the first time, or provide a reprise of the latest thinking for those who have been away from it for a while. We hoped we have peaked your interest to the extent that you seek additional information, and with the aid of some practicing professionals, implement the SE principles in your programs. The suggested additional readings in Appendix B would be a good place to start in gathering more information. Your friendly librarian will also help you find suitable books, articles, and journals that might help and interest you. One of the main purposes of this booklet is to aid you in forming the right questions in your search for additional knowledge.

INCOSE

The International Council on Systems Engineering (INCOSE) is an organization formed to develop and enhance multi-disciplinary system development under the title of Systems Engineering. INCOSE is the one of the only professional associations dedicated entirely to systems engineering. INCOSE currently has more than a dozen working groups covering issues such as best practices, policy review, process description, tools, etc. INCOSE has national meetings annually with professional papers and other information of interest to systems engineers. INCOSE was created to:

- Foster the definition, understanding and practice of world class systems engineering in industry, academia, and government,
- Provide a focal point for dissemination of systems engineering knowledge,
- Promote collaboration in systems engineering education and research, and
- Assure the existence of professional standards for integrity in the practice of systems engineering.

So Many Interfaces, So Little Time

After reaching this point you're probably wondering how you'll ever be able to meet and deal with all these people on a daily basis. Fortunately, the problem is fairly bounded. While it's essentially true that the Systems Engineer has to interface with the world, he doesn't have to do it all the time and all at once. He will have a close long-term relationship with the designers, Quality and the logisticians, but they will get together to make interim decisions and then each will go off to perform the analysis, synthesis and design work necessary for the next set of decisions. Interfaces with the others are on a similar basis, but over a shorter period of time. The most important point for the SE to understand is that each of these disciplines has a specific contribution to make and successful projects properly blend these inputs.

Appendix A – Glossary

(Sources used in the preparation are in parentheses following each definition)

Accomplishment	See "significant accomplishment."
Accomplishment criteria	See "significant accomplishment criteria."
acquisition program	Within the DoD, an approved and funded activity that defines the skill and manpower levels for the people, develops and produces the products, and develops the processes that make up a system.
Affordable	An acquisition program for which the life-cycle cost of is in consonance with the long-range investment and force structure plans of the Department of Defense or individual DoD Components.
Allocated (design-to) Baseline Completion	During the Engineering and Manufacturing Development (EMD) or similar phase, the contract status in which (1) the functional baseline and any changes since it was established have been approved by the Government, (2) the functional architecture reflects all eight primary system functions and has been extended to the point that all decomposed functional requirements can be and have been mapped one-to-one to a physical hierarchy to form an allocated baseline, (3) the functional architecture, physical hierarchy, and allocated baseline have been balanced with respect to performance, cost, schedule, and risk, (4) the allocated baseline for each component is complete including complete and compatible interface design constraints between the items and between the items and other systems, facilities, and personnel, (5) it has been verified that the allocated baseline can satisfy the approved functional baseline, (6) the design-to-cost and life cycle cost estimates have been updated and remain consistent with any contract cost goals, constraints, or requirements and (7) the two-way traceability has been demonstrated via the decision data base from each element in the functional architecture and allocated baseline to the corresponding (a) source of the functional baseline and (b) requirement reference. During the Program Definition and Risk Reduction (DEM/VAL) or similar phase, the contract status in which the design requirements for each prototype has been verified to align with (1) the evolving functional architecture and allocated baseline including compatibility of the physical and functional interfaces between the item and other items, facilities, and personnel and (2) the planned risk handling approach.
allocated baseline	The approved design-to requirements for each system component (hardware or computer software) or computer software unit. The requirements include the allocations from the functional architecture and higher level elements, interface constraints with interfacing elements, additional design constraints, and the verification method required to demonstrate compliance.
allocation	(1) All or part of a requirement for a higher level system element that has been designated to be satisfied by a lower tier element or item. (2) The process of decomposing the requirements for a system among the elements or items of the system. (3) The results of (2).
Alternative Systems Review (ASR)	A formal review, usually conducted during the Concept Exploration Phase (Phase 0) of the acquisition life cycle, (1) to make a preliminary assessment that the preferred concept(s) can provide an affordable, timely solution that meets the operational requirements in the intended environment at acceptable risk and (2) to define the risks for the preferred system concept(s) that should be addressed during subsequent phases.
analysis	(1) The performance and assessment of calculations (including modeling and simulation) to evaluate requirements or design approaches or compare alternatives. (2) The verification method of determining performance (a) by examination of the baseline, (b) by performing calculations based on the baseline and assessing the results, (c) by extrapolating or interpolating empirical data of collected using physical items prepared according to the baseline, or (d) by a combination of all of the above.
approved	The formal acceptance of an item, data, or document by the management level required by the contract or contract plan. If the level is the Government, the Government has notified the Contractor that it is acceptable through a contractual letter.
architecture	A structure that shows the elements and their relationship for a set of requirements or a system concept or both.
article	An individual copy of item.
as-built configuration	A production-representative article built or fabricated in accordance with the design baseline.
audit	An independent examination of the results of work to assess compliance with a specification, standard, or contract, or other criteria.

balance	The act of assessing and comparing performance, cost, schedule, and risk for alternative requirements, requirements allocations, and/or design solutions.
balanced	A set of requirements, requirements allocations, and/or design solutions for which the performance, cost, schedule, and risk have been assessed and found to be acceptable in the context of the program that is to satisfy the requirements.
baseline	*noun*: Document(s) or database(s) that record a set of requirements and/or product solutions and that can be changed only by formal, documented procedures. *verb:* To formally approve a baseline.
build-to requirements	Drawings, manufacturing or assembly instructions, process specifications and instructions and/or any other data required to manufacture an item.
change	A modification of an approved requirement, baseline, or item as documented in a decision data base, specification, or any other configuration item documentation.
change control	The engineering management function of (1) limiting change to a baseline or item to that which has been (a) assessed for impacts to performance, cost, schedule, and risk and (b) approved by formal, documented procedures and (2) assuring implementation of all changes so assessed and approved.
change proposal	A proposed change to the currently approved configuration baseline for a configuration item and the documentation by which the change is described, justified, and, if required by the contract, submitted to the Government for approval or disapproval.
commercial off the shelf (COTS)	An item that is available in the commercial marketplace that does not require unique Government modifications or maintenance over its life-cycle to meet the requirements.
compatibility	The capability of two or more items to exist or function in the same system or environment without mutual interference.
component	An item that is viewed as a separate entity for purposes of design, manufacturing, software coding, testing, maintenance, contracting, reprocurement, record keeping, or configuration management. A configuration item is a component, but all components are not necessarily configuration items, i.e., they may be controlled by other than formal configuration management procedures. Hardware components may be further divided into additional components; software components may be further divided into additional components and/or software units.
computer software	The complete set or any item of the set of computer programs or instructions in the physical hierarchy and the associated documentation.
computer software unit	A subdivision of a computer software component.
configuration	The functional and physical characteristics of an item as documented in a baseline and ultimately achieved in a product or process.
configuration baseline	The configuration document(s) or database(s) that record the initially approved set of requirements and/or product solutions and all approved changes thereto and that is changed only by formal, documented procedures.
configuration control	Formal change control for configuration items.
configuration item	An item that satisfies a documented set of requirements and is designated for separate configuration management to include any item required for logistic support or designated for separate procurement.
configuration management	For configuration items, (1) the identification and documentation of the configuration, (2) the control of changes to the items or their documentation, (3) configuration status accounting, and (4) the auditing to confirm that conformance to all requirements has been verified.
configuration status accounting	For configuration items, the recording and reporting of (1) the approved configuration baseline and identification numbers, (2) the status of proposed changes, deviations, and waivers, (3) the implementation status of approved changes, and (4) the configuration of all units of the configuration item owned by the Government.
control	The engineering management function of ensuring that plans are having the intended effect and that work is being completed according to the plans. Controlling is one of the basic functions of engineering management -- the others are planning, organizing, staffing, directing, and monitoring.
Cost Analysis Requirements Document (CARD)	The common description of the salient programmatic and technical features of the program (and the system it is to provide) that is used by the teams preparing the program office cost estimate, component cost analysis, and independent life-cycle cost estimates.
cost engineering	The art of analyzing and estimating the cost of a design solution and relating those costs to the requirements.
cost goals, cost constraints, or cost requirements	The financial objectives or thresholds for the program or contract and their allocation to items. Often expressed in terms of development, design-to-cost (DTC), unit production cost (UPC), operations and support (O&S), and life cycle cost (LCC) thresholds, targets, or goals. Cost goals and requirements are a reflection that fiscal constraints are a reality in defense acquisition.

Critical Design Review (CDR): (1) During Engineering and Manufacturing Development (EMD) or similar phase, the review by the Contractor and the Government of (1) the status of any changes to the functional baseline and architecture and allocated baseline since they were established, (2) the design baseline for each configuration item including the completeness and compatibility of interfaces between the items and between the items and other systems, facilities, and personnel, (3) the basis for each element in the design baseline in terms requirements and objective, comprehensive, quantitative design trades, (4) the balance between performance, cost, schedule, and risk for each element in the selected design baseline, (5) the two-way traceability from the source of the functional baseline to the design baseline and back, and (6) the verification that the design baseline can meet the contract requirements. The data available for CDR should document or demonstrate these six items and reside in the decision data base.
(2) During the Program Definition and Risk Reduction (DEM/VAL) or similar phase, a review conducted on each prototype (1) to evaluate the progress, technical adequacy, and risk resolution of the detailed design and (2) to determine its alignment with the evolving functional architecture and allocated baseline including compatibility of the physical and functional interfaces among the item and other items, systems, facilities, and personnel.

data accession/internal data list: An evolving list, prepared and maintained by the Contractor, of data acquired or prepared under the contract and accessible by the Government either by access to a management information system or by PCO direction.

decision database: The linked and readily retrievable collection of data (including inputs and intermediate and final results) that provide the audit trail of decisions and their rationale from initially stated needs and requirements, the system threat assessment, other program documents, and DoD policy, AF practice, and public law to the current description of the system requirements and the products, processes, facilities, and personnel requirements that collectively satisfy the requirements. It includes, as they evolve, (1) the functional baseline, the functional architecture, the physical hierarchy, and the allocated, design, and product baselines; (2) life-cycle verification, manufacturing, support, deployment, training, operations, and disposal data, procedures, and plans (including but not limited to test plans and procedures, drawings, manufacturing instructions, logistics support plans, common [Government-inventory] support equipment requirements, spares requirements, training programs [or training program requirements for training programs not developed under the contract], technical manuals, and required Government personnel skill and manpower levels applicable to both OT&E and the operations phase); (3) the embedded software; (4) remaining risks and corresponding risk monitoring (including TPMs and metrics) and mitigation steps; (5) cost estimates and their bases; (6) data, models, and analytic techniques used to verify that an evolving solution can meet its requirements; (7) the verification results that verify compliance of designs or delivered products with the contract requirements; (8) the approval authority and rationale for any changes to the data; and (9) any other decision support data developed under the contract linked to its basis in the rest of the data base. It provides for the efficient traceability through the architectures, baselines, and the physical hierarchy from any element up to the Government sources of the functional baseline or down to the lowest elements of the allocated, design, and product baselines; from any element to the corresponding requirement reference; from any requirement to the corresponding verification method and verification plans, procedures, and data; from any component in the physical hierarchy to its design-to and build-to requirements, product description, and supportability data; and from any element to its change history.

demonstration: The verification method of determining performance by exercising or operating the item in which instrumentation or special test equipment is not required beyond that inherent to the item and all data required for verification is obtained by observing operation of the item.

deployment function: Tasks to be performed to take the elements of a system or system upgrade from the completion of development, training, manufacturing, and verification to a state of operational readiness.

derived requirements: Requirements not explicitly stated in the operational requirements and which are inferred from the nature of the proposed solution, the environment, policy, law, best engineering practice, or some combination of the above.

design: *verb:* Architecting and selecting products (including processes) and corresponding personnel manpower, skill levels, and specialized training that satisfy all requirements and describing them so that the products can be manufactured or coded, verified, deployed, operated, supported, and disposed of and so that the personnel can be selected and trained.
noun: The result of designing.

Design (build-to) Baseline Completion	During Engineering and Manufacturing Development (EMD) or similar phase, the contract status in which (1) any changes to the functional baseline have been approved by the Government, (2) any changes to the functional architecture and allocated baselines since they were established have been approved as required by the contract, (3) the design baseline is complete including the interface designs between the baseline components and between the components and other systems, facilities, and personnel, (4) the functional architecture and allocated and design baselines are balanced with respect to performance, cost, schedule, and risk, (5) it has been verified that the design baseline can satisfy the approved functional baseline, (6) the design-to-cost and life cycle cost estimates have been updated and remain consistent with any contract cost goals, constraints, or requirements and (7) the two-way traceability has been demonstrated from each element in the functional architecture and the allocated and design baselines to the corresponding (a) source of the functional baseline and (b) requirement reference via the decision data base. During the Program Definition and Risk Reduction (DEM/VAL) or similar phase, the contract status in which the design for each prototype is verified to align with (1) the evolving functional baseline architecture and allocated baseline including compatibility of the physical and functional interfaces among the item and other items, systems, facilities, and personnel and (2) the planned risk handling approach.
design baseline	The documented requirements for (1) material ordering ("buy-to" requirements), (2) hardware fabrication and manufacturing process setup and operation for developmental hardware ("build-to" requirements), (3) software coding ("code-to" requirements), (4) verification, training, deployment, operations, support, and disposal ("verify-to, train-to, deploy-to, operate-to, support-to, and dispose-to" requirements) and (6) personnel skill and manpower levels that collectively satisfy the functional baseline. The design baseline includes separable documentation for each hardware and software component. For programs that will transition to production, the design baseline forms an initial or preliminary product baseline. The complete product baseline will usually be formalized near the end of development or early in production. If the Event "Design (build-to) Baseline Completion," Critical Design Review (CDR), or the equivalent is held, the design baseline is usually formalized as part of the Event close-out.
design constraints	Requirements that form boundaries within which other requirements must be allocated and items must be designed. The constraints may be externally imposed or result from decisions internal to the program or contract. Design constraints include interface, environmental, physical mass and dimensional, reliability, maintainability, human factors, logistics support, personnel resource (skill levels and manpower) and training, standardization, design and construction practices, and fiscal (cost) requirements.
Design to Cost (DTC), Design-to-Cost	*noun:* An acquisition management technique in which cost design constraints are derived and allocated to the items to be designed. *adj.:* Derived by applying the DTC technique.
development function	Tasks to be performed to take a system or system upgrades from the statement of the operational requirement to readiness for verification, manufacturing, training, deployment, operations, support, and disposal.
Developmental Test & Evaluation (DT&E)	Test and evaluation activities to (1) support technology selection, requirements analysis and allocation, and design and (2) verify compliance with the contract requirements.
deviation	A specific written authorization, granted prior to the manufacture of an item, to depart from one or more particular requirements of an items approved configuration baseline for a specific number of units or a specified period of time.
disposal function	Tasks to be performed to ensure that the disposition of products and by-products that are no longer needed or no longer useful complies with applicable security classification guidance and environmental laws and regulations. The function addresses the short and long term impact to the environment and health hazards to humans and animals as well as recycling, material recovery, salvage for re-utilization, demilitarization, and disposal of by-products all other functions, i.e., across the life cycle.
documented	Recorded on paper or in electronic or other media in accordance with the contract.
eight primary system functions	The essential tasks that must be accomplished so that a system will satisfy the operational needs, DoD policy, and the law over the life cycle. Any defense acquisition program must complete eight primary functions: development, manufacturing, verification, deployment, operations, support, training, and disposal.
element	For a system, baseline, or architecture, any requirement, any function or sub-function, any item (any product to include any process or facility), any material, or any personnel requirement.

environment	The natural and induced conditions experienced by a system including its people and products (including its processes) during operational use, stand-by, maintenance, transportation, and storage. The natural conditions include space (exo-atmospheric), atmospheric (weather, climate), ocean, terrain, and vegetation. Induced conditions includes manufacturing (process conditions, clean room, storage), test, transportation, storage, normal operations (thermal, shock, vibration, electromagnetic, the range of power inputs), maintenance, combat (dust, smoke, nuclear-chemical-biological), and the threat (existing and potential threat systems to include electronic warfare and communications interception).
environmental constraints or requirements	The expected worst case impact of the environment on the system or item as well as the system or items allowed impact on the environment.
equipment	Hardware, hardware and software, or an assembly of hardware or hardware and software
event	A point in a program or contract defined by significant accomplishments and accomplishment criteria (or metrics) in the IMP. The goal for the calendar date to complete an event is documented in the IMS.
external interface	A design constraint imposed on a system by another system or facility.
Follow-on Operational Test and Evaluation (FOT&E)	See "Operational Test & Evaluation (OT&E)."
formal	An act that follows a documented procedure and that is approved by the signature of an authorized individual recorded in a readily retrieved archive.
function	A task to be performed to achieve a required outcome or satisfy an operational need.
functional analysis and allocation	The decomposition of each of the top-level functions to sub-functions to the point that each sub-function can be related to the elements of a physical hierarchy, the allocation of the top-level performance requirements and design constraints to the functions and sub-functions, and the capture of the aggregation in a functional architecture.
functional architecture	The hierarchical arrangement of functions and their decomposition to sub-functions and the allocation of the top level performance requirements and design constraints to functions and sub-functions.
functional baseline	The initially approved documentation describing a system's top level functional and performance requirements and design constraints and all changes thereto. The functional baseline can be changed only with Government approval. The functional baseline is usually initially approved near the end of the *Program Definition and Risk Reduction* Phase (Phase I, formerly called DEM/VAL), as part of the procurement process for Engineering and Manufacturing Development (EMD or Phase II), or soon after the start of EMD. See the definition for the Event, *Functional Baseline Completion*, in this Annex.
Functional Baseline Completion	Contract status in which (1) the planned risk reduction efforts under the contract have been completed, (2) the functional baseline has been approved by the Government, (3) the preliminary functional architecture maps to the preliminary physical hierarchy and both are balanced with respect to performance, cost, schedule, and risk, (4) the design-to-cost and life-cycle-cost projections have been updated and compared to the contract cost requirements or objectives, (5) it has been verified that the preliminary allocated baseline can satisfy the functional baseline, (6) the decision data base (a) provides two-way traceability from the sources of the functional baseline to any element in the approved functional baseline and evolving functional architecture and allocated baseline and from any element to the rationale for that element and (b) archives the rationale and approval authority for all changes, (7) the preliminary physical hierarchy maps to the proposed CWBS for the next phase, and (8) the significant accomplishments and accomplishment criteria have been planned in the IMP for at least all technical activity required prior to the next event on the contract, if any.
Functional Configuration Audit (FCA)	For each configuration item, the formal examination of its functional characteristics to verify that it has achieved the requirements in its allocated baseline. For a system, the formal examination of its functional characteristics to verify that it has achieved the requirements in the functional baseline.
functional requirement	A task that must be accomplished to satisfy an operational need or set of requirements. The top-level functional requirements are the eight primary system functions stated and linked as they apply to the operational need or requirements.
hardware	Items made of a material substance but excluding computer software and technical data packages.
Initial Operational Test and Evaluation (IOT&E)	See "Operational Test and Evaluation (OT&E)."
inspection	The verification method of determining performance by examining (1) engineering documentation produced during development or modification or (2) the item itself using visual means or simple measurements not requiring precision measurement equipment.

Integrated Logistics Support (ILS)	A disciplined, unified, and iterative approach to the management and technical activities necessary to (1) integrate support considerations into system and component design; (2) develop support requirements that are consistently related to readiness objectives, to design, and to each other; (3) acquire the required support; and (4) provide the required support during the operational phase at minimum cost.
Integrated Master Plan (IMP)	A description, usually contractual, of the applicable documents, significant accomplishments, accomplishment criteria, events, and critical processes necessary to satisfy all contract requirements. The completion of each significant accomplishment is determined by measurable accomplishment criteria. The significant accomplishments have a logical relationship to each other and, in subsets, lead up to events. Each event is, in turn, complete when the significant accomplishments leading up to it are complete. The critical processes are described by narratives that include Objectives, Governing Documentation, and an Approach. The IMP includes an indexing scheme (sometimes called a single numbering system) that links each significant accomplishment to the associated CWBS element, event, significant accomplishment criteria, and tasks presented in the Integrated Master Schedule (IMS). The data in the IMP defines the necessary accomplishments for each event both for each IPT and for the contract as a whole. See also Integrated Task and Management Plan (ITAMP).
Integrated Master Schedule (IMS)	The schedule showing the time relationship between significant accomplishments, events, and the detailed tasks (or work packages) required to complete the contract. The IMS uses (and extends if necessary) the same indexing (or single numbering system) as used in the Integrated Master Plan (IMP).
Integrated Process Team (IPT)	Team composed of specialists from all appropriate functional disciplines working together (1) to develop and operate processes that affordably meet all program requirements and (2) to enable decision makers to make the right decisions at the right time. For Acquisition Category I and II (ACAT I and II) space programs, the IPT is chaired by a senior individual in the office of the Air Force Mission Area Director for Space (SAF/AQS).
Integrated Product and Process Development (IPPD)	A management technique that simultaneously integrates all essential acquisition activities through the use of multi-disciplinary Integrated Product or Process Teams (IPTs).
Integrated Product Team (IPT)	Team composed of specialists from all applicable functional disciplines working together (1) to deliver products and processes that affordably meet all requirements at acceptable risk and (2) to enable decision makers to make the right decisions at the right time by timely achievement of the significant accomplishments in the Integrated Master Plan (IMP).
Integrated Task and Management Plan (ITAMP)	A single document that combines and fulfills the purposes of the Statement of Work (SOW) and the Integrated Master Plan (IMP). The Task Section of the ITAMP replaces the SOW and the other sections are identical to the IMP.
integration	The merger or combining of two or more parts, computer software units, components, or other items into a still higher level item to ensure that the functional requirements and design constraints for the higher level item are satisfied.
interface	The boundary, often conceptual, between two or more functions, systems, or items or between a system and a facility at which interface requirements are set.
interface constraint	See interface requirement.
interface control	The process of identifying, documenting, and controlling all interface requirements on a system or the elements of a system.
Interface Control Document (ICD), Interface Control Drawing	Drawing or other documentation that depicts interface designs or elements of interface designs that satisfy interface requirements.
Interface Control Working Group (ICWG)	A group with representation from all sides of an interface that seeks agreement on mutually compatible interface requirements and controls the documentation of the resulting interface agreements. ICWGs that address external interfaces will usually be chaired by the Government. ICWGs that address internal interfaces, if separate, may be chaired by the Contractor.
interface requirement	The functional and physical design constraints imposed on each other by two or more functions, items, or systems or between a system and a facility. Functional interfaces include signal, electrical, electromagnetic, and software. Physical interfaces include keep-out volumes and mating surfaces and connections.
interface requirements specification (IRS), interface specification	A repository for interface requirements that details the functional and physical connection between systems or system elements or between systems and facilities.
internal interface	The functional and physical design constraints imposed on an item resulting from the designs selected for other items in the same system. (Also, see interface requirement and external interface.)

interoperability	The ability of systems, units, or forces to provide services to or accept services from other systems, units, or forces and to use the services so exchanged to operate effectively together.
item	Any product (where products include processes and facilities).
life cycle	The scope of a system or upgrade evolution beginning with the determination of a mission need or identification of a system deficiency through all subsequent phases through disposal of the system.
Life Cycle Cost (LCC)	The total cost to the Government of acquisition and ownership of the system over its useful life. It includes the cost of development, production, operations & support, and disposal.
Logistics Support Analysis (LSA)	Engineering efforts, as part of the systems engineering process, to assist in: causing support considerations to influence design; defining support requirements that are related optimally to design and to each other; acquiring the required support; and providing the required support during the operational phase at minimum cost.
manufacturing function	Tasks to be performed to convert materials and parts into a product ready for verification, training, and/or deployment.
metric	A measure used to indicate progress or achievement.
milestone	(1) A point in a program or contract at which some team member or leader is held accountable and at which progress toward completion of the program or contract is measured. Also, see event. (2) Major decision points that separate the phases of defense acquisition programs. Phases include, for example, engineering and manufacturing development and full-rate production.
Milestone Decision Authority (MDA)	The individual designated in accordance with criteria established by DoD 5000.2-R to approve entry of a defense acquisition program into the next phase.
Mission Need Statement (MNS)	A statement of the need for a material solution to perform an assigned mission or to correct a deficiency in existing capability to perform the mission.
modification	The act of changing a system or component after delivery to improve some characteristic, to adapt it to function in a changed environment, or to respond to a change in the law. Also, see upgrade.
Non-Developmental Item (NDI)	Any item that is (1) available in the commercial marketplace or (2) previously developed and in use by a department or agency of the United States, a State or local Government, or a foreign Government with which the United States has a mutual defense cooperation agreement and that does not require unique upgrades or maintenance over its life-cycle to meet the current requirements. In some cases NDI may be extended to include items that (a) have been developed but are not yet available in the commercial marketplace or in use by a Government entity or (b) require only minor modification or upgrade. In other cases, items meeting these latter criteria are termed Near-NDI or N-NDI.
objectives	Operationally significant desired levels of performance or functionality above the requirement that are goals for the program or contract but not a requirement.
operational effectiveness	The overall degree of mission accomplishment of a system when used by representative personnel in the environment planned or expected (e.g., natural, electronic, threat etc.) for operational employment of the system considering organization, doctrine, tactics, survivability, vulnerability, and threat (including countermeasures, initial nuclear weapons effects, nuclear, biological, and chemical contamination (NBCC) threats).
operational requirements	Requirements generated by the Operator/Users, normally in terms of system capabilities or characteristics required to accomplish mission tasks, and documented in a Mission Needs Statement (MNS) that evolves into an Operational Requirements Document (ORD) and associated Requirements Correlation Matrix (RCM).
Operational Requirements Document (ORD)	Usually prepared during Phase 0, Concept Exploration, the ORD will be based on the most promising alternative determined during the Phase 0 studies. The ORD documents how the system will be operated, deployed, employed, and supported by describing system-specific characteristics, capabilities, and other related operational variables. The ORD will be updated for Milestones II and III. The CSAF approves all Air Force and Air Force-led ORDs.
Operational Test & Evaluation (OT&E)	Independent test and evaluation to determine the effectiveness and suitability of the weapons, equipment, or munitions for use in combat by typical military users; and the evaluation of the results of such tests. Can be either Initial (IOT&E) or Follow-on (FOT&E). IOT&E is conducted on production or production representative articles, to support a decision to proceed such as beyond low-rate initial production. It is conducted to provide a valid estimate of expected system operational effectiveness and operational suitability. FOT&E is conducted during and after the production period to refine the estimates made during IOT&E , to evaluate changes, and to reevaluate the system to ensure that it continues to meet operational needs and retains its effectiveness in a new environment or against a new threat.
operations function	Tasks to be performed subsequent to verification and deployment to accomplish defined missions in either the expected peacetime or wartime environments excluding training, support, and disposal.

performance	A measure of how well a system or item functions in the expected environments.
performance requirement	The extent to which a mission or function must be executed, i.e., a functional requirement that is stated in terms of quantity or quality such as range, coverage, timeliness, or readiness.
physical architecture	The physical hierarchy and the functional requirements and design constraints for each element in the hierarchy. It can be viewed as an intermediate step between the functional architecture and the physical hierarchy, on the one hand, and the allocated baseline, on the other hand. It is not directly addressed in this CPAT.
Physical Configuration Audit (PCA)	For each configuration item (CI), the formal comparison of a production-representative article with its design baseline to establish or verify the product baseline. For the system, the formal comparison of a production-representative system with its functional and design baseline as well as any processes that apply at the system level and the formal examination to confirm that the PCA was completed for each CI, that the decision data base represents the system, that deficiencies discovered during testing (DT&E and IOT&E) have been resolved and changes approved, and that all approved changes have been implemented.
physical hierarchy, product physical hierarchy	The hierarchical arrangement of products, processes, personnel skill levels, and manpower levels that satisfy the functional baseline. The top entry in the hierarchy is the system. The hierarchy extends to include all components and computer software units necessary to satisfy the functional baseline whether deliverable or not. It includes the prime operational hardware and software, Contractor-supplied support equipment, Government-inventory support equipment, technical manuals, training programs for both Government and Contractor personnel, Government personnel skill and manpower levels, spare parts requirements, and factory support equipment and tooling which collectively result in the system that satisfies the functional baseline.
physical requirement	A physical characteristic, attribute, or distinguishing feature that a system or item must possess.
plan	Documented approach, resources, and schedule necessary to complete a task.
planned profile	The time-phased projection, usually in graphical form, of the values for a technical parameter.
planned value	The predicted value of a technical parameter at the planned time of measurement based on the planned profile.
Preliminary Design Review (PDR)	During Engineering and Manufacturing Development (EMD), the review by the Contractor and the Government of (1) any changes to the functional baseline since it was established, (2) the functional architecture, (3) the physical hierarchy, (4) the allocated baseline for each configuration item including the completeness and compatibility of interfaces between the items and between the items and other systems, facilities, and personnel, (5) the basis and the balance between performance, cost, schedule, and risk for each element in the architectures and each requirement in the baseline, (6) the two-way traceability from the source of the functional baseline to the allocated baseline and back, and (7) the verification that the allocated baseline can meet the system requirements. The primary PDR data is the Decision Data Base documenting or demonstrating these seven items. During the Program Definition and Risk Reduction (DEM/VAL) or similar phase, a review conducted on each prototype to evaluate the progress, technical adequacy, and risk resolution of the selected design approach; to determine its alignment with the evolving functional baseline and architecture and allocated baseline including compatibility of the physical and functional interfaces among the item and other items, facilities, and personnel.
primary functions, primary system functions	See the entry, "eight primary system functions."
procedure	A documented description of a sequence of actions to be taken to perform a given task.
process	A set of steps or activities that bring about a result and the criteria for progressing from step to step or activity to activity.
product	What is delivered to the customer (e.g., hardware, software, test reports, RFPs, data...), as well as processes (e.g., system engineering, design, manufacturing, test, logistics, acquisition security...) which make the product possible.
product baseline	Build-to requirements for each physical element to be manufactured; software code for each software element that has been separately designed or tested; and buy-to requirements for each other physical element, part, or material to be procured from a subcontractor or vendor.

Product Baseline Completion	For each configuration item (CI), the contract status in which a production-representative article and any associated processes have been formally demonstrated to satisfy the corresponding design baseline to establish or verify the product baseline for the CI. For the system, the contract status in which (1) a production-representative system and any processes that apply a the system level have been formally demonstrated to satisfy the system functional and design baseline, (2) it has been formally confirmed that (a) the Product Baseline is complete for each CI, (b) that the decision data base represents the system, (c) that deficiencies discovered during test and evaluation (DT&E and IOT&E) have been resolved and changes approved, and (d) that all approved changes have been implemented.
product physical hierarchy	See physical hierarchy in this Annex.
REQUIREMENT REFERENCE	A higher level requirement or an analysis, test, or other justification for a requirement, requirement allocation, or other architectural element. Abbreviated Req. Ref.
requirements	Characteristics, attributes, or distinguishing features that a system or system element must have within a stated environment or set of conditions in order to meet an operational need and comply with applicable policy and practices. Also, see operational requirements and program technical requirements.
requirements analysis	The determination of the system specific functional and performance requirements and design constraints based on analyses of the operational need, requirements, objectives (or goals), and measures of effectiveness; missions; projected utilization environments; DoD policies and practices; and the law.
Requirements Analysis Completion	Contract status in which (1) the operational requirements have been translated into technical requirements and captured in the evolving functional baseline, (2) the functional baseline and the plans to complete it account for the eight primary system functions and all design constraints, (3) the preliminary functional architecture is consistent with the functional baseline and maps to the preliminary physical hierarchy, (4) design-to-cost and life-cycle-cost projections have been updated and compared to any cost requirements or objectives, (5) the decision data base captures the completed work on the baselines and architectures and provides two-way traceability from the sources of the functional baseline to any element and from any element to the rationale for that element and archives the rationale and approval authority for all changes, (6) the risk reduction efforts have resulted in the planned progress, remain applicable to the preferred system concept(s), and address all the risks that can be handled at this point in the program, and (7) the significant accomplishments and accomplishment criteria have been planned for at least all technical activity required prior to the next event.
risk	A measure of the uncertainty of attaining a goal, objective, or requirement and the consequences of not attaining it. The uncertainty is the result of one or more undesirable events that could occur during the system life cycle for which insufficient resources and time are programmed to overcome them. The consequences are inability to satisfy the operational military need and exceeding the programmed budget and directed schedule.
risk management	A documented process for the prospective (looking ahead) and recurring identification of what can go wrong, assigning a level of risk (e.g., High, Moderate, Low) to each risk, and planning and implementing mitigation steps for each commensurate with the level of risk. Also, see the Risk Management CPAT.
schedule, schedule requirements	Progress characteristics imposed on the completion of program phases, on contract events and deliveries, and operation and support parameters such as time between failures and repair time.
significant accomplishment	A specified step or result that indicates a level of progress toward completing an event and, in turn, meeting the objectives and requirements of the contract.
significant accomplishment criteria	Specific, measurable conditions that must be satisfactorily demonstrated before a significant accomplishment listed in an Integrated Master Plan (IMP) is complete and before work dependent on the accomplishment can proceed.
simulation	The process of conducting experiments with a model (an abstraction or simplification) of an item and/or part or all of its operating environment for the purpose of assessing its behavior under selected conditions or of evaluating various strategies for its operation within the limits imposed by developmental or operational criteria. Simulation may include the use of analog or digital devices, laboratory models, or "test bed" sites. Simulations are usually programmed for solution on a computer; however, in the broadest sense, military exercises and war games are also simulations.
Software Development Plan (SDP)	A management plan for the software development activities on a contract, usually prepared by the developer.
software, software product	See computer software.

solution, solution set	Products (including processes) and corresponding personnel manpower, skill levels, and specialized training that satisfy all requirements and balance performance, cost, schedule, and risk.
spares, spare parts	Maintenance replacements for replaceable parts, components, or assemblies in deployed items of equipment.
specification	A description of the essential technical requirements for items (hardware and software), materials, and processes that includes verification criteria for determining whether the requirements are met.
specification tree	The hierarchical depiction of all the specifications needed to formally control the development, procurement, manufacture, integration, verification, and/or reprocurement during any part of the life cycle.
subsystem	A grouping of items satisfying a logical group of functions within a system.
support equipment	All equipment (mobile or fixed) required to support the operation and maintenance of a materiel system. This includes associated multi-use end items, ground-handling and maintenance equipment, tools, meteorology and calibration equipment, test equipment, and automatic test equipment. It includes the acquisition of logistics support for the support and test equipment itself.
support function	Tasks to be performed to provide support for operations, maintenance, and training. The tasks include the acquisition and supply of spares, depot level maintenance, and the acquisition and maintenance of the facilities and selection and training of personnel to carry out the support function.
supportability	The degree to which planned logistics support (including system design; test, measurement, and diagnostic equipment; spares and repair parts; technical data; support and facilities; transportation requirements; training; manpower; and software support) allow meeting system availability and wartime usage requirements.
survivability	The capability of a system to avoid or withstand man-made hostile environments without suffering an abortive impairment of its ability to accomplish its designated mission.
system	An integrated composite of people, products, and processes that satisfy an operational requirement or objective. An acquisition program defines the skill and manpower levels for the people, develops and produces the products, and develops the processes.
System Concept Assessment Completion	The contract status in which (1) the performance in the intended environment(s) relative to the operational requirements and objectives; the cost relative to program cost objectives, if any; the schedule relative to the operational need, if stated; and the risk have been assessed for the preferred system concept(s) and (2) the risks to be handled during subsequent phases have been identified.
System Design Review	See System Functional Review.
system element	See element.
system engineering	**As a process**, an interdisciplinary effort to recursively and iteratively (1) support the evolution of, first, the operational need, and then later, the operational requirements and objectives, (2) translate the requirements and objectives into, first, a functional baseline, second, an allocated baseline, third, a design baseline, and, finally, a product baseline, (3) to maintain those baselines over the life cycle of the system, and (4) verify initially that the requirements can be met by the evolving baselines and ultimately that the requirements have been met. **As a team or organizational entity**, a group that is directly responsible for certain activities in the process and for facilitating or monitoring others as a staff function to a program or product manager. Note: **All** of the *technical* organizations involved in a program or contract have a role in the system engineering process so the it is much more than what the system engineering team or office does. Also, see Section 1.1.

System Functional Review (SFR) — A review, usually held during the Program Definition and Risk Reduction or similar phase (Phase I), by the Contractor and the Government to confirm that (1) the planned risk reduction efforts have been completed and the results reflected in the proposed functional baseline and preliminary functional architecture and allocated baseline, (2) the proposed functional baseline is accurate and comprehensive (though perhaps with TBDs, TBRs, and TBSs), (3) the preliminary functional architecture and allocated baseline reflect the proposed functional baseline and is balanced with respect to performance, cost, schedule, and risk, (4) the decision data base supports two-way traceability from the source of the functional baseline to the preliminary allocated baseline and from any element to the rationale for that element and shows the rationale and approval authority for all changes, (5) the verification that the evolving allocated baseline can satisfy the functional baseline, (6) the preliminary physical hierarchy, the planned (or approved) PWBS, and the proposed CWBS are all consistent, (7) the life cycle cost for the evolving design is consistent with the program affordability constraints, and (8) the remaining risks have been identified and can be handled in the context of the planned next phase. The primary SFR data is the Decision Data Base documenting or demonstrating these eight items.

System Requirements Review (SRR) — A review, usually held near the end of the Program Definition and Risk Reduction or similar phase (Phase I), by the Contractor and the Government to confirm that (1) the planned risk reduction efforts are making adequate progress and reflect the technologies envisioned to implement the preferred system concept(s), (2) the operational requirements and objectives have been accurately and comprehensively translated into technical requirements and are reflected in the preliminary functional baseline, (3) the preliminary functional baseline and the plans to complete it account for the eight primary functions and all design constraints on the system design, (4) the preliminary physical hierarchy is consistent with the preliminary functional baseline, (5) life cycle cost projections remain consistent with the program affordability constraints, (6) the decision data base supports two-way traceability from the source of the functional baseline to the functional baseline and from any element to the rationale for that element and shows the rationale and approval authority for all changes, and (8) the significant accomplishments and accomplishment criteria have been planned for the next wave of technical activity on the contract. The primary SRR data is the Decision Data Base documenting or demonstrating these eight items.

system threat assessment report, System Threat Assessment Report (STAR) — Describes the threat to be countered and the projected threat environment. The threat information should reference DIA or Service Technical Intelligence Center approved documents.

System Verification Completion and Readiness for Production, Deployment, Operations, and Support — The contract status in which (1) the system has been verified to satisfy the functional, allocated, and design baselines and the assumptions and methods used in verification by analysis have been demonstrated to be consistent with the operational, threat, and other system requirements and environments, (2) the decision data base has been demonstrated to represent the system, (3) that deficiencies discovered during test and evaluation (DT&E and IOT&E) have been resolved and changes approved as required by the contract, (4) all approved changes have been designed and verified, (5) the life cycle cost projections have been updated and shown to be consistent with any contract affordability or cost goals, constraints, or requirements, (6) planning is complete (including significant accomplishments and corresponding criteria for the next contract event, if any) and procedures, technical manuals, resources, and other requisite systems or facilities are available (or, if not planned to be complete by this event, are on schedule to be available when needed) to initiate production, verification, training, deployment, operations, support, and disposal , and (7) the remaining risks have been identified and can be handled in the context of the planned next program phase.

System Verification Review (SVR) — A review, usually held near the end of Phase II, EMD, by the Contractor and the Government to confirm that (1) the system has been verified to satisfy the functional, allocated, and design baselines including an assessment of the assumptions and methods used in verification by analysis, (2) that the decision data base has been maintained and represents the system, (3) that deficiencies discovered during testing (DT&E and IOT&E) have been resolved and changes approved, (4) that all approved changes have been designed and verified, (5) the life cycle cost projections remain consistent with the program affordability constraints, (6) planning is complete and procedures, resources, and other requisite systems or facilities are available to initiate production, verification, training, deployment, operations, support, and disposal, and (7) the remaining risks have been identified and can be handled in the context of the planned next phase. The primary SFR data is the Decision Data Base documenting or demonstrating these eight items.

tailoring	The process by which sections, paragraphs, and sentences of specifications, standards, and other requirements or tasking documents are evaluated to determine the extent to which they are applicable to a specific acquisition contract and then modified to balance performance, cost, schedule, and risk.
task	A unit of work that is sufficiently well defined so that, within the context of related tasks, readiness criteria, completion criteria, cost, and schedule can all be determined.
team	A group of people that collectively have the necessary knowledge, skills, and resources and are assigned the **Responsibility** and **Authority** and are held **Accountable** (RAA) to perform a task or function.
technical data package (TDP)	The evolving data needed for implementing the acquisition strategy, engineering, production, verification, deployment, training, operations, logistics support, and disposal for an item. It defines the configuration and procedures to ensure that the item meets requirements. It consists of performance requirements and the associated development and product specifications, standards, quality assurance provisions, drawings, associated lists, process instructions, packaging details, training program, and technical manuals. The technical data package is a part of the decision data base.
technical manual (TM)	Instructions for the deployment, operation, maintenance, training, support, and disposal of weapon systems, weapon system items, and support equipment. Technical Orders (TOs) that meet this definition may also be classified as Technical Manuals.
Technical Performance Measure (TPM)	A parameter that is related to progress toward meeting the program or contract functional requirements or goals and is assessed periodically and at certain events to estimate the degree to which the final value will meet the anticipated or required level. See Figure 1.7 of AFMC Instruction 63-XXX for more detail.
program technical requirements	Verifiable requirements and objectives restated or derived by the acquisition community from the program operational requirements, the program threat assessment, applicable DoD and DoD-Component practices and policies, and program decisions to achieve all program requirements and objectives. Technical requirements include all program functional and performance requirements, design constraints, and, ultimately, personnel tasks, numbers and skills of personnel, quantities of equipment, spares, repair parts, and consumables. Government program technical requirements are usually initially documented in a Systems Requirements Document (SRD) or similar record and evolved by the Government or the prime Contractor into the System Specification. Technical requirements for the elements of the system are allocated from the Government program technical requirements to the components of the system and documented consistent with the management and contracting structure and support plans.
test	The verification method of determining performance by exercising or operating the system or item using instrumentation or special test equipment that is not an integral part of the item being verified. Any analysis of the data recorded in the test and that is needed to verify compliance (such as the application of instrument calibration data) does not require interpretation or interpolation/extrapolation of the test data.
test plan	Documented approach, resources, and schedule to verify compliance of a system or one of its elements by test.)
test report	Documentation of compliance with the test plan and the compliance or non-compliance of the items under test.
threat	(1) Countries or groups that are considered to have a potential adverse impact on the national security of the United States. (2) Weapon systems that must be defeated by U.S. systems in battle and the environment in which those systems operate. Note: Threat information, to include the target data base, shall be validated by the Defense Intelligence Agency (DIA) for acquisition programs subject to review by the Defense Acquisition Board (DAB).
time-line analysis	The analysis of the time sequencing of the elements of the functional architecture and the operation of the elements of a design response to define any resulting time or sequencing requirements.
To Be Determined (TBD)	When used in a Government controlled requirements document or Interface Control Drawing, an item that has not been determined and for which a determination is to be recommended by the Contractor (by a System Engineering or Integrated Product Team in which the Government participates) for final Government approval.
To Be Resolved (TBR)	When used in a Government controlled requirements document or Interface Control Drawing, an item that is preliminary and for which a final resolution is be recommended by the Contractor (by a System Engineering or Integrated Product Team in which the Government participates) for final Government approval.

To Be Supplied (TBS)	When used in a Government controlled requirements document or Interface Control Drawing, an item that has not been determined and for which a determination is to be formally supplied by the Government to the Contractor (though it may be studied by the System Engineering or Integrated Product Teams on which both Contractor and Government personnel participate).
traceability	The ability to relate an element of the functional baseline, functional architecture, physical hierarchy, allocated baseline, design baseline, and product baseline (or their representation in the decision data base) to any other element to which it has a master-subordinate (or parent-child) relationship.
trade-off study	**An objective comparison with respect to performance, cost, schedule, risk, and all other reasonable criteria of all realistic alternative** requirements; architectures; baselines; or design, verification, manufacturing, deployment, training, operations, support, or disposal approaches.
training function	Tasks to be performed to achieve and maintain knowledge and skill levels necessary to perform the operations, support, and disposal functions efficiently and effectively over the system life cycle.
unit	A subdivision of time, fabrication or production quantity, or some other system or program parameter. For software, a subdivision of a component.
unit production cost (UPC)	The cost of a single, specified unit (such as first or average) under a defined set of production ground rules (such as schedule and quantity).
upgrade	A change from previously delivered items because of obsolescence of a part; a change in the military need or threat; an operational, supportability, or training deficiency is identified; the system life must be extended; a change in the law occurs; or an unsafe condition is detected. Also, see modification.
users	The personnel who operate, maintain, support, or dispose of an item delivered to the Government inventory or those who train such personnel.
variation	The difference between the planned value of a technical parameter and the current assessed value.
verification	The task of determining whether a system or item meets the requirements established for it.
verification function	Tasks to be performed to evaluate the compliance of the evolving system (people, product, and processes) with the program or contract requirements. Includes analysis, demonstration, test, inspection, and special methods. The function includes technology assessments and demonstrations and all test and evaluation such as Development Test and Evaluation (DT&E) and Operational Test and Evaluation (OT&E). Also includes the evaluation of program or contract risks and monitoring the risks.
verification method	A way to verify that a solution meets a requirement. The usual verification methods are test, demonstration, inspection, and analysis. Other, special methods are also sometimes applied. The verification method for each requirement should be included in the baseline containing the requirement.
waiver	A written authorization to accept an item which, subsequent to the start of manufacture, is found to depart from specified requirements but nevertheless is considered suitable for use "as is" or after repair by an approved method.
Work Breakdown Structure (WBS)	A product-oriented hierarchical tree composed of the hardware, software, services (including cross-product tasks such as systems engineering), data, and facilities that encompass **all** work to be carried out under the program or contract along with a dictionary of the entries in the tree. The WBS for the entire program is called the Program or Project WBS (PWBS). The WBS for the work under the contract is called the Contract WBS (CWBS) and is prepared in accordance with the contract.

Appendix B – Acronyms

Note: most terms are defined in Appendix A.

ACAT	Acquisition Category
	Air Force Material Command
ASR	Alternative Systems Review
B	(1) Section of an RFP or model contract that specifies supplies or services and prices/costs (2) Blue evaluation ranking
BCD	Baseline Concept Description
BPPBS	Biennial Planning, Programming, and Budgeting System
C/SCS	Cost/Schedule Control System
C/SSR	Cost/Schedule Summary Report
CAID	Clear Accountability in Design
CAM	Cost Account Manager
CARD	Cost Analysis Requirements Document
CCA	Critical Capability Area
CDR	Critical Design Review
CDRL	Contract Data Requirements List
CE	Concept Exploration (Phase 0)
CE&D	Concept Exploration and Definition
CFSR	Contract Funds Status Report
CI	Configuration Item
CLIN	Contract Line Item Number
COTS	Commercial off the Shelf
CPAT	Critical Process Assessment Tool
CPR	Cost Performance Report
CSOW	Contract Statement of Work
CWBS	Contract Work Breakdown Structure
DAD	Defense Acquisition Deskbook
DEM/VAL	Demonstration and Validation (Phase I)
DIA	Defense Intelligence Agency
DID	Data Item Description
DoD	Department of Defense
DPML	Deputy Program Manager for Logistics
DT&E	Development Test and Evaluation
DTC	Design to Cost (See also DTUPC, UPC)
DTUPC	Design to Unit Production Cost (See also DTC, UPC)
EBB	Electronic Bulletin Board
ECP	Engineering Change Proposal
EELV	Evolved Expendable Launch Vehicle
EMD	Engineering and Manufacturing Development (Phase II)
F	Section or an RFP or model contract that specifies delivery schedules
FCA	Functional Configuration Audit
FFBD	Functional Flow Block Diagram
FFP	Firm Fixed Price
FOT&E	Follow-On Operational Test and Evaluation
FRD	Functional Requirements Document
G	Green evaluation ranking
H	(1) Section or an RFP or model contract that specifies special contract requirements or provisions (2) High Risk
I	Section or an RFP or model contract that specifies contract clauses
ICD	Interface Control Document
ICWG	Interface Control Working Group
ILS	Integrated Logistics Support
IMP	Integrated Master Plan
IMS	Integrated Master Schedule
IOT&E	Initial Operational Test and Evaluation
IPD	Integrated Product Development -- see IPPD

IPPD	Integrated Product and Process Development
IPT	Integrated Product Team
IRS	Interface Requirements Specification
ITAMP	Integrated Task and Management (or Master) Plan (ITAMP)
ITO	Instructions to the Offerors
J	List of attachments to an RFP or model contract
L	(1) Section of an RFP that includes the Proposal Preparation Instructions (2) Low Risk
LAAFB	Los Angeles Air Force Base
LCC	Life Cycle Cost
LOE	Level Of Effort
LRU	Line Replaceable Unit
LSA	Logistics Support Analysis
M	(1) Section of an RFP that includes the evaluation criteria and factors (2) Moderate Risk
MDA	Milestone Decision Authority
MIL-Spec	Military Specification
MIL-STD	Military Standard
MIS	Management Information System
MNS	Mission Need Statement
MSSRP	Military Specifications and Standards Reform Program
MTBF	Mean Time Between Failure
NBCC	Nuclear, Biological, and Chemical Contamination
NDI	Non-Developmental Item
O&S	Operations and Support
ORD	Operational Requirements Document
OT&E	Operational Test and Evaluation (IOT&E and/or FOT&E)
PCA	Physical Configuration Audit
PCO	Procuring Contracting Officer
PDR	Preliminary Design Review
PPI	Proposal Preparation Instructions
PWBS	Program or Project Work Breakdown Structure (WBS)
R	Red evaluation ranking
RAA	Responsibility, Authority, and Accountability
RCM	Requirements Correlation Matrix
RFP	Request for Proposal
SAF	Secretary of the Air Force
SDCE	Software Development Capability Evaluation -- see AFMC Pamphlet 63-103, Volumes 1 and 2
SDP	Software Development Plan
SDR	System Design Review
SEIT	System Engineering & Integration Team
SEMP	System Engineering Management Plan
SERD	Support Equipment Requirements Data (SERD)
SFR	System Functional Review
SMC	Space and Missile Systems Center
SOO	Statement of (Government) Objectives
SOW	Statement of Work
SPD	System Performance Document
SPO	System Program Office
SRD	System Requirements Document
SRR	System Requirements Review
SRU	Shop Replaceable Unit
SSA	Source Selection Authority
SSS	System/Subsystem Specification
STAR	System Threat Assessment Report
SVR	System Verification Review
TBD	To Be Determined (see definition in Annex 1)
TBR	To Be Resolved (see definition in Annex 1)
TBS	To Be Supplied (see definition in Annex 1)
TDP	Technical Data Package
TM	Technical Manual
TO	Technical Order
TPM	Technical Performance Measure

TRD	Technical Requirements Document
UPC	Unit Production Cost (See also DTC, DTUPC)
WBS	Work Breakdown Structure (see also CWBS and PWBS)
Y	Yellow evaluation ranking

Note: most terms are defined in Appendix A.

Appendix C - Systems Engineering Templates and Examples

Appendix C.1 - A Sample SEMP Outline

- Title Page
  - Systems Engineering Management Plan
  - System Name or Identifier
- Table of Contents
- Scope
  - Purpose of the System
  - Summary and Purpose of SEMP
  - Relation to other plans and schedules such as the Integrated Master Plan (IMP), Integrated Master Schedule (IMS), and Earned Value Management System (EVMS)
  - The following statement: "This SEMP is the plan for the complete, integrated technical effort. Nothing herein shall relieve the Contractor of meeting the requirements of the Contract."
- Applicable Documents
  - Government Documents to include contractual requirements documents or specifications
  - Non-government Documents to include any applicable from independent standards organizations
  - Corporate Documents
- Systems Engineering Process and Responsibilities for its Implementation
  - Description of the Contractor's systems engineering process activities to be accomplished during the contract to include the iterative nature of the process application in the form of narratives, supplemented as appropriate by graphical presentations, detailing the contractor's processes and procedures for completing the systems engineering effort
    - Requirements Analysis
    - Functional Analysis and Allocation
    - Synthesis
    - Systems Analysis and Control to include Control and Manage to include trade studies, cost-effectiveness analyses
    - Risk Management
    - Configuration Management
    - Interface Management
    - Data Management
    - Technical Performance Measurements (TPMs) – initial list, criteria for changing the list, update schedule, responsibility for monitoring, and relationship to risk management
    - Technical Reviews and Audits
  - Description of products and results
    - Decision Database – describe development, implementation, life-cycle accessability, and life-cycle maintenance including how traceability of the information will be accomplished
    - Specifications (or equivalent) and configuration baselines – describe development, measures of completeness, verifiability, traceability, and how and when controlled
  - Verification Planning – planning for verifying all requirements to include identification, configuration control, and maintenance of accuracy/precision of all verification tools

- Organizational responsibilities, authority, and means of accountability for implementing the process under the Contract
- Work authorization – methods for opening work packages under the EVMS, closure, and authorization of changes
- Subcontractor technical effort – description of the level of subcontractor participation in the technical effort as well as the role of systems engineering in subcontractor and vendor selection and management

Transitioning Critical Technologies

- Criteria for assessing and transitioning technologies
- Evolutionary/spiral acquisition strategies

Integration of the Systems Engineering Activities

- How management plans and schedules (such as the IMP and IMS) and the EVMS will be used to plan, organize, direct, monitor and control the systems engineering activities
- Systems Engineering Tools
- Approach and process for system integration and test

Additional Systems Engineering Activities

Notes

- Glossary of terms used in the SEMP

Appendices – each appendix shall be referenced in the main body of the SEMP where the data would otherwise have been provided.

Appendix C.2 - A "Tailored" WBS for a launch & satellite System

Level 1	Level 2	Level 3
Space System		
	Launch Vehicle	
		Stage I
		Stage II . . . n (as required)
		Strap-on boosters (as required)
		Fairing (shroud)
		Guidance and Control
		Integration, Assembly, Test, and Checkout
	Space Vehicle	
		Spacecraft (bus)
		Payload (I . . . n)
		Orbit injector/dispenser
		Integration, Assembly, Test, and Checkout
	Ground Command, Control, Communications, and Mission Equipment	
		Telemetry, Tracking and Control
		External Communications
		Data Processing Equipment
		Auxiliary Equipment
		Facilities (Control, Communications, Mission)
		Integration, Assembly, Test and Checkout
	Systems Engineering/Program Management	(See Definitions below)
	System Test and Evaluation	Development Test and Evaluation
		Operational Test and Evaluation
		Mock-ups
		Test and Evaluation Support
		Test Facilities
	Training	
		Courseware
		Equipment
		Services
		Facilities
	Data	(See Definitions below)
	Peculiar Support Equipment	
		Test and Measurement Equipment
		Support and Handling Equipment
	Operational/Site Activation	
		System Assembly, Installation, and Checkout
		Contractor Technical Support
		Site Construction
		(See Definitions below for others)
	Flight Operations and Services	
		Assembly, Mate, and Checkout
		Mission Control
		Telemetry, Tracking, and Control
		Launch Equipment
	Storage	
		Planning and Preparation
		Storage
		Removal and Transportation
	Initial Spares	(See Definitions below)

Definitions

Space System

The complex of equipment (hardware/software), data, services, and facilities required to attain and/or maintain an operational capability in space. This operational capability requires the ability to develop, deliver, and maintain mission payload(s) in specific orbit, which further requires the ability to place, operate, and recover manned and unmanned space systems.

Includes:

- launch vehicles, orbital transfer vehicles, payload fairings (shrouds), space vehicles, communications, command and control facilities and equipment, and any mission equipment or other items necessary to provide an operational capability in space.

Launch Vehicle

The primary means for providing initial thrust to place a space vehicle into its operational environment. The launch vehicle is the prime propulsion portion of the complete flyaway (not to include the orbital transfer vehicle and space vehicle). The launch vehicle may be single-stage or multiple-stage configuration.

Includes:

- the structure, propulsion, guidance and control, and all other installed equipment integral to the launch vehicle as an entity within itself
- the design, development, and production of complete units (i.e., the prototype or operationally configured units which satisfy the requirements of their applicable specification, regardless of end use)
- Sub-elements to the launch vehicle

Note: ***All effort directly associated with the remaining level 3 WBS elements and the integration, assembly, test and checkout of these elements into the launch vehicle is excluded.***

Stage I

The launch vehicle stage which provides initial lift-off propulsion for the complete launch vehicle (flyaway) and cargo.

Includes, for example:

- structure, propulsion, controls, instrumentation, and all other installed subsystem equipment integral to Stage 1 as an entity
- design, development, production, and assembly efforts to provide Stage I as an entity

Excludes:

- strap-on units

Note: ***All effort directly associated with the remaining level 3 WBS elements and the integration, assembly, test and checkout of these elements into the launch vehicle is excluded.***

Stage II…n (as required)

The second and subsequent launch vehicle stages (if applicable) used to place a space vehicle into its operational environment.

Includes, for example:

- propulsion following separation of the first stage and subsequent stages (if applicable)
- structure, propulsion, controls, instrumentation, separation subsystems, and all other installed subsystem equipment integral to the stage as an entity
- design, development, production, and assembly efforts to provide each individual stage as an entity

Excludes:

- strap-on units

Note: ***All effort directly associated with the remaining level 3 WBS elements and the integration, assembly, test and checkout of these elements into the launch vehicle is excluded.***

Strap-On Boosters (as required)

Solid or liquid propulsion assemblies that provide additional thrust or propellant to assist the launch vehicle in placing a spacecraft into its operational orbit if strap-on units are employed.

Includes, for example:

- complete set of strap-on units -- case, nozzle, igniter, tanks, mounting structure, cordage, etc.
- design, development, production, and assembly efforts to provide the strap-on units as an entity

Note: ***All effort directly associated with the remaining level 3 WBS elements and the integration, assembly, test and checkout of these elements into the launch vehicle is excluded.***

Payload Fairing (Shroud)

The protective covering and equipment mated to the launch vehicle which protects the cargo (i.e., orbital transfer vehicle or space vehicle/orbital transfer vehicle combination) prior to and during the launch vehicle ascent phase.

Includes, for example:

- structure -- the shroud structure, mechanisms and hinges
- instrumentation -- the hardware and software required to measure the environment and loads being experienced by the shroud during the ascent phase until shroud separation and deployment
- separation subsystem -- the sequencers, ordnance, and other necessary mechanisms to assure a successful shroud separation from the launch vehicle and cargo
- power system -- the necessary generation, storage, and distribution of electrical power and signals, hydraulic power, and any other power required by the shroud
- thermal control systems -- thermal paint, insulation, heat shield tiles, or any other active or passive means necessary to maintain appropriate temperature of the shroud and mission equipment within it
- integration, assembly, test and checkout

Note: ***All effort directly associated with the remaining level 3 WBS elements and the integration, assembly, test and checkout of these elements into the launch vehicle is excluded.***

Guidance and Control

The means (hardware/software) for generating or receiving guidance intelligence, conditioning the intelligence to produce control signals, and generating appropriate control forces.

Controllers may interface with the structure by actuating moveable aero surfaces or with the propulsion system to produce control reaction forces or may independently produce reaction forces for control.

If the design is such that electronics are packaged into a single rack or housing as an assembly, this rack or housing will be considered part of the guidance and control system.

Includes, for example:

- guidance intelligence system, computer, sensing elements, etc.

Note: ***All effort directly associated with the remaining level 3 WBS elements and the integration, assembly, test and checkout of these elements into the launch vehicle is excluded.***

Integration, Assembly, Test, and Checkout.

In those instances in which an integration, assembly, test, and checkout element is used (Appendices A through G), this element includes all effort of technical and functional activities associated with the design, development, and production of mating surfaces, structures, equipment, parts, materials, and software required to assemble the level 3 equipment (hardware/software) elements into a level 2 mission equipment (hardware/software) as a whole and not directly part of any other individual level 3 element.

Includes:

- the development of engineering layouts, determination of overall design characteristics, and determination of requirements of design review
- the set up, conduct, and review of testing assembled components or subsystems prior to installation
- the detailed production design, producibility engineering planning (PEP), and manufacturing process capability, including the process design development and demonstration effort to achieve compatibility with engineering requirements and the ability to produce economically and consistent quality
- inspection activities related to receiving, factory and vendor liaison
- design maintenance effort
- quality planning and control
- tooling (initial production facilities, factory support equipment) including planning, design, and fabrication
- administrative engineering
- the joining or mating and final assembly of level 3 equipment elements to form a complete prime mission equipment when the effort is performed at the manufacturing facility
- integration of software (including loading and verification of firmware)
- conduct of production acceptance testing

Excludes:

- all systems engineering/program management and system test and evaluation which are associated with the overall system

Note: When an integration, assembly, test, and checkout element is utilized at lower levels of the contract work breakdown structure, it will be summarized into the next higher level equipment (hardware/software) work breakdown structure element and should never be summarized directly into a level 3 integration, assembly, test, and checkout element.

Space Vehicle

The satellite.

Includes:

- the structure, propulsion, thermal control, power and power conditioning, and all other installed equipment integral to the space vehicle as an entity within itself
- the design, development, and production of complete units (i.e., the prototype or operationally configured units which satisfy the requirements of their applicable specification, regardless of end use)
- Sub-elements to the space vehicle

Note: ***All effort directly associated with the remaining level 3 WBS elements and the integration, assembly, test and checkout of these elements into the space vehicle is excluded.***

Spacecraft

The principal operating element of the space vehicle which serves as a housing or platform for carrying a payload and other mission-oriented equipments in space.

Includes, for example:

- structure, power, attitude determination and control, and other equipments characteristic of spacecraft
- all design, development, production, and assembly efforts to provide the spacecraft as an entity

Payload

The equipment provided for special purposes in addition to the normal equipment integral to the spacecraft or reentry vehicle.

Includes, for example:

- experimental equipment placed on board the vehicle and flight crew equipment (space suits, life support, and safety equipment)
- communications, displays and instrumentation, telemetry equipment and other equipments specifically to collect data for future planning and projection purposes

Note: ***All effort directly associated with the remaining level 3 WBS elements and the integration, assembly, test and checkout of these elements into the space vehicle is excluded.***

Orbit Injector/Dispenser

The function of placing orbiting objects in the planned orbital path.

Includes, for example:

- structure, propulsion, instrumentation and stage interface, separation subsystem, and other equipment necessary for integration with other level 3 elements

Note: ***All effort directly associated with the remaining level 3 WBS elements and the integration, assembly, test and checkout of these elements into the space vehicle is excluded.***

Integration, Assembly, Test, and Checkout

The integration, assembly, test, and checkout element includes all efforts as identified above to provide a complete space vehicle.

Ground Command, Control, Communications, and Mission Equipment

The ground hardware/software equipment used for communicating between control and tracking facilities, monitoring the health and status of space vehicles, commanding the space vehicle's hardware, and adjusting the space vehicle's orbit as required for space vehicle health or mission purpose.

Two configurations for the ground command, control, communications and mission equipment are the parabolic dish-based antenna system and the phased array-based antenna system.

If a ground site has multiple antenna configurations, each will have its own separate command and control equipment, communications equipment, data processing equipment and test equipment.

Includes:

- the design, development, and production of complete units -- (i.e., prototype or operationally configured units which satisfy the requirements of their applicable specifications, regardless of end use)
- sub-elements to the ground command, control, communications, and mission equipment

Telemetry, Tracking and Control

The hardware/software elements that facilitate launch decisions and command and control of the aerospace vehicle.

Includes, for example:

- supplementary means for guidance of those aerospace vehicles not having completely self-contained guidance and control and means to command destruct
- control and check-out consoles, data displays, and mission records

External Communications

The hardware and software components that allow the ground station to communicate with any external data link or source like telephone (analog) lines, digital data lines, nonsatellite radio receivers. While the terrestrial data lines may connect to radio of other satellite communications stations, the external communications subsystem ends where these links physically connect to the secure communications, modulation/demodulation (modem) or coder/decoder equipment.

Data Processing Equipment

The hardware and software components that provide the activities and means to condition data generated at the launch site or aboard the space vehicle, or data received from associated systems to accommodate the needs of command and control or mission data processing.

Includes, for example:

- central processing unit (computer), peripheral equipment, and the software required to operate the data processing equipment.

Auxiliary Equipment

The general purpose/multi-usage ground equipment utilized to support the various operational capabilities of the command and launch equipments.

Includes, for example:

- power generators, power distribution systems, environmental control, cabling, malfunction detection, fire prevention, security systems, and other common-usage items not applicable to specific elements of the ground based equipment

Facilities

The special construction necessary to accomplish ground system objectives.

Includes, for example:

- modification or rehabilitation of existing facilities used to accomplish ground system objectives

Excludes:

- installed operational ground equipment
- the brick and mortar-type facilities identified as industrial facilities – see Operational/Site Activation

Integration, Assembly, Test, and Checkout

The integration, assembly, test, and checkout element includes all efforts as identified above to provide a complete ground system.

Systems Engineering/Program Management

The systems engineering and technical control as well as the business management of particular systems and programs. Systems engineering/program management elements to be reported and their levels will be specified by the requiring activity.

Includes:

- the overall planning, directing, and controlling of the definition, development, and production of a system or program including supportability and acquisition logistics, e.g., maintenance support, facilities, personnel, training, testing, and activation of a system

Excludes:

- systems engineering/program management effort that can be associated specifically with the equipment (hardware/software) element

Systems Engineering

The technical and management efforts of directing and controlling a totally integrated engineering effort of a system or program.

Includes but not limited to:

- effort to define the system and the integrated planning and control of the technical program efforts of design engineering, specialty engineering, production engineering, and integrated test planning

- effort to transform an operational need or statement of deficiency into a description of system requirements and a preferred system configuration

- technical planning and control effort for planning, monitoring, measuring, evaluating, directing, and replanning the management of the technical program

- (all programs, where applicable) value engineering, configuration management, human factors, maintainability, reliability, survivability/vulnerability, system safety, environmental protection, standardization, system analysis, logistic support analysis, etc.

Excludes:

- actual design engineering and the production engineering directly related to the WBS element with which it is associated

Examples of systems engineering efforts are:

1) System definition, overall system design, design integrity analysis, system optimization, system/cost effectiveness analysis, and intra-system and inter-system compatibility assurance, etc.; the integration and balancing of reliability, maintainability, producibility, safety, human health, environmental protection, and survivability; security requirements, configuration management and configuration control; quality assurance program, value engineering, preparation of equipment and component performance specifications, design of test and demonstration plans; determination of software development or software test facility/environment requirements.

2) Preparation of the Systems Engineering Management Plan (SEMP), specification tree, program risk analysis, system planning, decision control process, technical performance measurement, technical reviews, subcontractor and vendor reviews, work authorization, and technical documentation control.

3) Reliability engineering -- the engineering process and series of tasks required to examine the probability of a device or system performing its mission adequately for the period of time intended under the operating conditions expected to be encountered.

4) Maintainability engineering -- the engineering process and series of tasks required to measure the ability of an item or system to be retained in or restored to a specified condition of readiness, skill levels, etc., using prescribed procedures and resources at specific levels of maintenance and repair.

5) Human factors engineering -- the engineering process and the series of tasks required to define, as a comprehensive technical and engineering effort, the integration of doctrine, manpower, and personnel integration, materiel development, operational effectiveness, human characteristics, skill capabilities, training, manning implication, and other related elements into a comprehensive effort.

6) Supportability analyses -- an integral part of the systems engineering process beginning at program initiation and continuing throughout program development. Supportability analyses form the basis for related design requirements included in the system specification and for subsequent decisions concerning how to most cost effectively support the system over its entire life cycle. Programs allow contractors the maximum flexibility in proposing the most appropriate supportability analyses.

Program Management

The business and administrative planning, organizing, directing, coordinating, controlling, and approval actions designated to accomplish overall program objectives which are not associated with specific hardware elements and are not included in systems engineering.

Includes for example:

- cost, schedule, performance measurement management, warranty administration, contract management, data management, vendor liaison, subcontract management, etc.

- support element management, defined as the logistics tasks management effort and technical control, and the business management of the support elements. The logistics management function encompasses the support evaluation and supportability assurance required to produce an affordable and supportable defense materiel system

- planning and management of all the functions of logistics. Examples are:

  - maintenance support planning and support facilities planning; other support requirements determination; support equipment; supply support; packaging, handling, storage, and transportation; provisioning requirements determination and planning; training system requirements determination; computer resource determination; organizational, intermediate, and depot maintenance determination management; and data management

System Test and Evaluation

The use of prototype, production, or specifically fabricated hardware/software to obtain or validate engineering data on the performance of the system during the development phase (normally funded from RDT&E) of the program.

Includes:

- detailed planning, conduct, support, data reduction and reports (excluding the Contract Data Requirements List data) from such testing, and all hardware/software items which are consumed or planned to be consumed in the conduct of such testing

- all effort associated with the design and production of models, specimens, fixtures, and instrumentation in support of the system level test program

Note: Test articles which are complete units (i.e., functionally configured as required by specifications) are excluded from this work breakdown structure element.

Excludes:

- all formal and informal testing up through the subsystem level which can be associated with the hardware/software element

- acceptance testing

Note: These excluded efforts are to be included with the appropriate hardware or software elements.

Development Test and Evaluation

This effort is planned, conducted and monitored by the developing agency of the DoD component. It includes test and evaluation conducted to:

- demonstrate that the engineering design and development process is complete.

- demonstrate that the design risks have been minimized.

- demonstrate that the system will meet specifications.

- estimate the system's military utility when introduced.

- determine whether the engineering design is supportable (practical, maintainable, safe, etc.) for operational use.

- provide test data with which to examine and evaluate trade-offs against specification requirements, life cycle cost, and schedule.

- perform the logistics testing efforts to evaluate the achievement of supportability goals, the adequacy of the support package for the system, (e.g., deliverable maintenance tools, test equipment, technical publications, maintenance instructions, and personnel skills and training requirements, etc.).

Includes, for example:

- all contractor in-house effort
- (all programs, where applicable) models, tests and associated simulations such as wind tunnel, static, drop, and fatigue; integration ground tests; test bed aircraft and associated support; qualification test and evaluation, development flight test, test instrumentation, environmental tests, ballistics, radiological, range and accuracy demonstrations, test facility operations, test equipment (including its support equipment), chase and calibrated pacer aircraft and support thereto, and logistics testing
- (avionics) avionics integration test composed of the following:
  - test bench/laboratory, including design, acquisition, and installation of basic computers and test equipments which will provide an ability to simulate in the laboratory the operational environment of the avionics system/subsystem
  - air vehicle equipment, consisting of the avionics and/or other air vehicle subsystem modules which are required by the bench/lab or flying test bed in order to provide a compatible airframe avionics system/subsystem for evaluation purposes
  - flying test bed, including requirements analysis, design of modifications, lease or purchase of test bed aircraft, modification of aircraft, installation of avionics equipment and instrumentation, and checkout of an existing aircraft used essentially as a flying avionics laboratory
  - avionics test program, consisting of the effort required to develop test plans/procedures, conduct tests, and analyze hardware and software test results to verify the avionics equipments' operational capability and compatibility as an integrated air vehicle subsystem
  - software, referring to the effort required to design, code, de-bug, and document software programs necessary to direct the avionics integration test

Operational Test and Evaluation

The test and evaluation conducted by agencies other than the developing command to assess the prospective system's military utility, operational effectiveness, operational suitability, logistics supportability (including compatibility, inter-operability, reliability, maintainability, logistic requirements, etc.), cost of ownership, and need for any modifications.

Includes, for example:

- Initial operational test and evaluation conducted during the development of a weapon system
- such tests as system demonstration, flight tests, sea trials, mobility demonstrations, on-orbit tests, spin demonstration, stability tests, qualification operational test and evaluation, etc., and support thereto, required to prove the operational capability of the deliverable system
- contractor support (e.g., technical assistance, maintenance, labor, material, etc.) consumed during this phase of testing
- logistics testing efforts to evaluate the achievement of supportability goals and the adequacy of the support for the system (e.g., deliverable maintenance tools, test equipment, technical publications, maintenance instructions, personnel skills and training requirements, and software support facility/environment elements)

Mock-Ups

The design engineering and production of system or subsystem mock-ups which have special contractual or engineering significance, or which are not required solely for the conduct of one of the above elements of testing.

Test and Evaluation Support

The support elements necessary to operate and maintain, during test and evaluation, systems and subsystems which are not consumed during the testing phase and are not allocated to a specific phase of testing.

Includes, for example:

- repairable spares, repair of repairable, repair parts, warehousing and distribution of spares and repair parts, test and support equipment, test bed vehicles, drones, surveillance aircraft, tracking vessels, contractor technical support, etc.

Excludes:

- operational and maintenance personnel, consumables, special fixtures, special instrumentation, etc., which are utilized and/or consumed in a single element of testing and which should be included under that element of testing

Test Facilities

The special test facilities required for performance of the various developmental tests necessary to prove the design and reliability of the system or subsystem.

Includes, for example:

- test tank test fixtures, propulsion test fixtures, white rooms, test chambers, etc.

Excludes:

- brick and mortar-type facilities identified as industrial facilities

Training

Deliverable training services, devices, accessories, aids, equipment, and parts used to facilitate instruction through which personnel will learn to operate and maintain the system with maximum efficiency.

Includes:

- all effort associated with the design, development, and production of deliverable training equipment as well as the execution of training services

Excludes:

- overall planning, management, and task analysis function inherent in the WBS element Systems Engineering/Program Management

Courseware

Distinctive deliverable end items of training courses, assigned by either a contractor or military service, required to meet specific training objectives.

Includes, for example:

- operational training courses, maintenance training courses, and other training courses

Excludes:

- training equipment

Equipment

Distinctive deliverable end items of training equipment, assigned by either a contractor or military service, required to meet specific training objectives.

Includes, for example:

- operational trainers, maintenance trainers, and other items such as cutaways, mock-ups, and models

Excludes:

- training courseware

Services

Deliverable services, accessories, and aids necessary to accomplish the objectives of training.

Includes:

- training course materials; contractor-conducted training (in-plant and service training); and the materials and curriculum required to design, execute, and produce a contractor developed training program

- materiel, courses, and associated documentation (primarily the computer software, courses and training aids)

Excludes:

- deliverable training data associated with the WBS element Support Data

Facilities

The special construction necessary to accomplish training objectives.

Includes, for example:

- modification or rehabilitation of existing facilities used to accomplish training objectives

Excludes:

- installed equipment used to acquaint the trainee with the system or establish trainee proficiency
- the brick and mortar-type facilities identified as industrial facilities

Data

The deliverable data required to be listed on a Contract Data Requirements List, DD Form 1423.

Includes:

- only such effort that can be reduced or avoided if the data item is eliminated
- (government-peculiar data) acquiring, writing, assembling, reproducing, packaging and shipping the data
- transforming into government format, reproducing and shipping data identical to that used by the contractor but in a different format

Technical Publications

Technical data, providing instructions for installation, operation, maintenance, training, and support, formatted into a technical manual. Data may be presented in any form (regardless of the form or method of recording). Technical orders that meet the criteria of this definition may also be classified as technical manuals.

Includes, for example:

- operation and maintenance instructions, parts lists or parts breakdown, and related technical information or procedures exclusive of administrative procedures
- data item descriptions set forth in categories selected from the Acquisition Management Systems and Data Requirements Control List (DoD 5010.12-L)

Engineering Data

Recorded scientific or technical information (regardless of the form or method of recording) including computer software documentation. Engineering data defines and documents an engineering design or product configuration (sufficient to allow duplication of the original items) and is used to support production, engineering and logistics activities.

Includes, for example:

- all final plans, procedures, reports, and documentation pertaining to systems, subsystems, computer and computer resource programs, component engineering, operational testing, human factors, reliability, availability, and maintainability, and other engineering analysis, etc.
- Technical data package (reprocurement package) which includes all engineering drawings, associated lists, process descriptions, and other documents defining physical geometry, material composition, and performance procedures

Excludes:

- computer software or financial, administrative, cost or pricing, or management data or other information incidental to contract administration

Management Data

The data items necessary for configuration management, cost, schedule, contractual data management, program management, etc., required by the government in accordance with functional categories selected from the DODISS and DoD 5010.12-L.

Includes, for example:

- contractor cost reports, cost performance reports, contract funds status reports, schedules, milestones, networks, integrated support plans, etc.

Support Data

The data items designed to document support planning in accordance with functional categories selected from DoD 5010.12-L.

Includes, for example:

- supply; general maintenance plans and reports; training data; transportation, handling, storage, and packaging information; facilities data; data to support the provisioning process and all other support data; and software supportability planning and software support transition planning documents.

Data Depository

The facility designated to act as custodian to maintain a master engineering specification and establish a drawing depository service for government approved documents that are the property of the U.S. Government. As custodian for the government, the depository, authorized by approved change orders, maintains these master documents at the latest approved revision level. This facility is a distinct entity.

Includes, for example:

- all drafting and clerical effort necessary to maintain documents

Excludes:

- all similar effort for facility's specification and drawing control system, in support of its engineering and production activities.

Note: When documentation is called for on a given item of data retained in the depository, the charges (if charged as direct) will be to the appropriate data element.

Peculiar Support Equipment

The design, development, and production of those deliverable items and associated software required to support and maintain the system or portions of the system while the system is not directly engaged in the performance of its mission, and which are not common support equipment (See H.3.7 below).

Includes:

- vehicles, equipment, tools, etc., used to fuel, service, transport, hoist, repair, overhaul, assemble, disassemble, test, inspect, or otherwise maintain mission equipment

- any production of duplicate or modified factory test or tooling equipment delivered to the government for use in maintaining the system. (Factory test and tooling equipment initially used by the contractor in the production process but subsequently delivered to the government will be included as cost of the item produced.)

- any additional equipment or software required to maintain or modify the software portions of the system

Excludes:

- overall planning, management and task analysis functions inherent in the work breakdown structure element, Systems Engineering/Program Management

- common support equipment, presently in the DoD inventory or commercially available, bought by the using command, not by the acquiring command

Test and Measurement Equipment

The peculiar or unique testing and measurement equipment which allows an operator or maintenance function to evaluate operational conditions of a system or equipment by performing specific diagnostics, screening or quality assurance effort at an organizational, intermediate, or depot level of equipment support.

Includes, for example:

- test measurement and diagnostic equipment, precision measuring equipment, automatic test equipment, manual test equipment, automatic test systems, test program sets, appropriate interconnect devices, automated load modules, taps, and related software, firmware and support hardware (power supply equipment, etc.) used at all levels of maintenance

- packages which enable line or shop replaceable units, printed circuit boards, or similar items to be diagnosed using automatic test equipment

Support and Handling Equipment

The deliverable tools and handling equipment used for support of the mission system.

Includes, for example:

- ground support equipment, vehicular support equipment, powered support equipment, nonpowered support equipment, munitions material handling equipment, materiel handling equipment, and software support equipment (hardware and software)

Common Support Equipment

The items required to support and maintain the system or portions of the system while not directly engaged in the performance of its mission, and which are presently in the DoD inventory for support of other systems.

Includes:

- acquisition of additional quantities of this equipment needed to support the item
- all efforts required to assure the availability of this equipment to support the item

Test and Measurement Equipment

The common testing and measurement equipment which allows an operator or maintenance function to evaluate operational conditions of a system or equipment by performing specific diagnostics, screening or quality assurance effort at an organizational, intermediate, or depot level of equipment support.

Includes, for example:

- test measurement and diagnostic equipment, precision measuring equipment, automatic test equipment, manual test equipment, automatic test systems, test program sets, appropriate interconnect devices, automated load modules, taps, and related software, firmware and support hardware (power supply equipment, etc.) used at all levels of maintenance

- packages which enable line or shop replaceable units, printed circuit boards, or similar items to be diagnosed using automatic test equipment

Support and Handling Equipment

The deliverable tools and handling equipment used for support of the mission system.

Includes, for example:

- ground support equipment, vehicular support equipment, powered support equipment, nonpowered support equipment, munitions material handling equipment, materiel handling equipment, and software support equipment (hardware/software)

Operational/Site Activation

The real estate, construction, conversion, utilities, and equipment to provide all facilities required to house, service, and launch prime mission equipment at the organizational and intermediate level.

Includes:

- conversion of site, ship, or vehicle

- system assembly, checkout, and installation (of mission and support equipment) into site facility or ship to achieve operational status

- contractor support in relation to operational/site activation

System Assembly, Installation, and Checkout on Site

The materials and services involved in the assembly of mission equipment at the site.

Includes, for example:

- installation of mission and support equipment in the operations or support facilities and complete system checkout or shakedown to ensure operational status. (Where appropriate, specify by site, ship or vehicle.)

Contractor Technical Support

The materials and services provided by the contractor related to activation.

Includes, for example:

- repair of repairable, standby services, final turnover, etc.

Site Construction
Real estate, site planning and preparation, construction, and other special-purpose facilities necessary to achieve system operational status.

Includes, for example:
- construction of utilities, roads, and interconnecting cabling

Site/Ship/Vehicle Conversion
The materials and services required to convert existing sites, ships, or vehicles to accommodate the mission equipment and selected support equipment directly related to the specific system.

Includes, for example:
- operations, support, and other special purpose (e.g., launch) facilities conversion necessary to achieve system operational status. (Where appropriate, specify by site, ship or vehicle.)

Industrial Facilities
The construction, conversion, or expansion of industrial facilities for production, inventory, and contractor depot maintenance required when that service is for the specific system.

Includes:
- equipment acquisition or modernization, where applicable
- maintenance of these facilities or equipment
- industrial facilities for hazardous waste management to satisfy environmental standards

Construction/Conversion/Expansion
The real estate and preparation of system peculiar industrial facilities for production, inventory, depot maintenance, and other related activities.

Equipment Acquisition or Modernization
The production equipment acquisition, modernization, or transferal of equipment for the particular system. (Pertains to government owned and leased equipment under facilities contract.)

Maintenance (Industrial Facilities)
The maintenance, preservation, and repair of industrial facilities and equipment.

Flight Support Operations and Services
Mate/checkout/launch; mission control; tracking; and command, control and communications (C^3); recovery operations and services; and launch site maintenance/refurbishment. This element supports the launch vehicle, orbital transfer vehicle, and/or space vehicle during an operational mission.

Sub-elements to the flight operations and services:

Mate/Checkout/Launch
The preflight operations and services subsequent to production and/or storage, and the actual launch of the complete system and payload.

Includes, for example:
- materials to conduct equipment receiving and checkout at launch site, preflight assembly and checkout, pre/post flight data reduction and analysis, and any prelaunch flight control/mission control planning

Mission Control
The personnel and materiel required to operate individual mission control centers and to perform ground command and control with the space vehicles.

Includes, for example:
- mission control centers such as Constellation Command Center, Battle Management/Command Control Center (BM/C^3), Space Asset Support System Control Center, and Space Transportation Control Center

Excludes:
- tracking and communications centers (these are included below)

Tracking and C3
The personnel and materiel required to perform the functions of telemetry, tracking, controlling, and data retrieval for the mission control systems.

Includes, for example:

- mission control systems, on the ground or in space, including Satellite Control Facility; Remote Tracking Station; Tracking, Data, Relay Satellite System; and other ground/space tracking systems

Excludes:

- initial acquisition of tracking and C^3

Recovery Operations and Services

The contractor effort and materiel necessary to effect recovery of the space vehicle or other mission equipment.

Includes:

- the launch site recovery forces, reentry site recovery forces, logistics support to recovery forces, logistics support to the recovery operations, communications, and transportation of recovered equipment to assigned facilities

Launch Site Maintenance/Refurbishment

The organization, maintenance, and management of launch vehicle facilities and mission equipment, and support at the launch base.

Includes, for example:

- requirements to clean up and refurbish each launch site after each launch

Storage

Those costs of holding portions of the space system while awaiting use of the system being stored, prepared for storage, or recovered from storage. Periods of holding result from schedule changes and/or technological problems exogenous to the portion of the space system.

Includes:

- Sub-elements to storage

Planning and Preparation

The planning and preparation costs for storage of all systems/subsystems associated with the launch vehicle, orbital transfer vehicle, and space vehicle equipment.

Includes, for example:

- generation of any storage or maintenance instructions and documents necessary for repairable systems or subsystems

Storage

The cost incurred while the systems or subsystems of the launch vehicle, orbital transfer vehicle, and space vehicle equipment are in storage.

Transfer and Transportation

The transfer and storage costs incurred when the systems/subsystems of the launch vehicle, orbital transfer vehicle, and space vehicle equipment are moved from one location to another.

Includes, for example:

- costs of relocation necessitated by mission requirements

Initial Spares and Repair Parts

The deliverable spare components, assemblies and subassemblies used for initial replacement purposes in the materiel system equipment end item.

Includes:

- repairable spares and repair parts required as initial stockage to support and maintain newly fielded systems or subsystems during the initial phase of service, including pipeline and war reserve quantities, at all levels of maintenance and support

Excludes:

- development test spares and spares provided specifically for use during installation, assembly, and checkout on site. Lower level WBS breakouts should be by subsystem.

Appendix C.3 - A Sample Risk Management Plan Outline

The sample Risk Management Plan will be included in the next release version.

Appendix C.4 – Risk Identification Trigger List

A set of starting questions for identifying potential program risks have been provided in different functional areas. These questions were derived from the Risk Management Critical Process Assessment Tool (CPAT) developed by SMC/AXD as part of the Military Specifications and Standards Reform Program (MSSRP). They are not meant as all-inclusive, but serve as a starting point of discussion by the management team, a team of experts, or an IPT.

Systems Engineering and Technical Risk Questions

1. Are the program requirements/objectives clearly defined? Have the stakeholders had opportunities to influence the objectives, requirements and design solution?

2. Have all system functions been identified and used to derive requirements?

3. Do design(s) or requirement(s) push the current state-of-the art?

4. Have vague requirements(s) been implemented in a manner such that a change has the potential to cause large ramifications?

5. Are the problems/requirements/objectives well understood?

6. Have the designs/concepts/components been proven in one or more existing system?

7. Is there adequate margin to meet system performance, reliability, and maintainability requirements?

8. Is the design easily manufacturable/producible/reworkable?

9. Are there environmental risks associated with the manufacturing or deployment of the system?

10. Were there governmental, environmental, safety constraints considered?

11. Are interfaces clearly defined? External? Internal?

12. Do the interfaces have clearly defined ownership to ensure adequate attention to details?

13. Are the external interfaces well defined and stable?

14. Is there adequate traceability from design decisions back to requirements to ensure the effect of changes can be adequately assessed?

15. Has the concept for operating the system been adequately defined to ensure the identification of all requirements?

16. Is there a clearly defined requirement verification plan?

17. Is there a clearly defined configuration management plan and is it being followed?

18. Are appropriate lessons learned from prior programs integrated into the design?

Cost Risk Questions

1. Are budgets adequate to handle the scope of program requirements/objectives?
2. Are the budgets adequate to handle the level of changes expected to occur?
3. Are there any state-of-the-art products for which the cost is very soft?
4. Are there any suppliers whose performance is potentially questionable?
5. Are there any products where a viable manufacturer must be developed?
6. Are the manufacturing processes unproven or partially unproven?
7. Can a single supplier hold the program hostage?
8. Are there any key suppliers whose financial health is in question?
9. What areas need to have at least two suppliers?
10. Are there areas of concern where the potential for delays in development, manufacturing, or demonstration of a product could result in a cascading effect (cost increases) on system costs?
11. Has the cost of complying with applicable security requirements been included in the budget?
12. Has the cost of regulatory, statutory, or environmental constraints been included?
13. Have ground rules and assumptions for cost modeling and cost risk assessments been clearly defined and documented?

Schedule Risk Questions

1. Does a complete, detailed schedule / IMS exist?
2. Has a schedule risk analysis been performed?
3. Are the schedules adequate to meet objective(s)?
4. Will GFE/GFI be available when needed?
5. Are there critical lead item concerns?
6. Has adequate schedule been provided to allow adequate schedule slack?
7. Are there technical or performance risks that lie on the critical path?

8. Are there resource limitations, e.g. personnel/staffing, facilities, manufacturing tools, simulators, test equipment, which could impact the critical path?

9. Is the schedule overly optimistic?

10. Is the schedule sub-optimal due to fiscal funding limitations and is it sensitive to potential funding changes?

Program Management Risk Questions

1. Are there risks associated with the teaming allocation of responsibilities?
2. Does geographical separation among team members potentially impact the program?
3. Does the program manager have previous management experience as a contractor?
4. Is the technical skill set in short supply?
5. Are there adequate resources for a management reserve?
6. Has pre-contract work been performed?
7. Is the organizational structure in place?
8. What controls are in place to manage subcontractors?

Software Risk Questions

1. Was a detailed operations concept used to derive requirements?
2. How well are the software requirements defined?
3. Are the algorithms to be programmed developed?
4. Is the software reuse? Realistically how much?
5. What is the interface complexity?
6. What is the stability of the interfaces?
7. What is the implementation difficulty?
8. What is the anticipated code size?
9. Is the hardware/software integration complex? Extensive?
10. Is the schedule for software development compressed?

11. How good is the documentation on reuse software?

12. Do simulators exist or need to be developed to check out the software?

13. Do simulators or prototype hardware exist for hardware/software integration?

14. Can the hardware/software handle the data rate input?

Manufacturing / Producibility Risk Questions

1. Are the design requirements well defined?

2. Are the design requirements stable?

3. Does a prototype exist?

4. Is the first article the flight article?

5. Does a manufacturing line exist?

6. Are there subsystems/components that must be produced in greater quantities than past experience?

7. What is the production failure rate?

8. Are there process steps prone to breakage?

9. Are metrics on some production steps such that the manufactured component is close to the tolerance of acceptability?

10. Are an adequate number of suppliers available for key components?

11. Are there integration and test issues?

12. Are there facility availability issues, particularly if a stressing production rate is required?

13. Is there slack in the schedule for unexpected problems?

14. Will the test equipment and special tooling be available when needed?

Systems Integration

1. Are there components for which the integration is complex or difficult?

2. What is the difficulty of hardware / software integration?

3. How well are hardware and software interfaces defined?

4. Are there prototypes, pathfinders and engineering models available for system integration testing?
5. Is the CONOPS modeled after existing systems?
6. Are CONOPS interfaces defined?
7. How well are space vehicle interfaces defined?
8. Is there a simulation environment ready to support assembly, integration and test? Is it adequate to the anticipated volume?
9. Does the ground segment exist, or must it be defined concurrently with the space segment development?
10. Does the ground segment need to be merged into an existing system? How stable is that design?
11. Is there a transition plan defined in going from an old system to the new?
12. Are requirements changing?
13. What is the potential of funding changes?
14. What is the impact of funding shortfalls and project stretch out?
15. Is the customer in the development pipeline, e.g. obtaining frequency allocation, concurrent risk reduction efforts, mandated GFE?
16. What external factors could impact the program?

Appendix C.5 - Techniques of Functional Analysis

Functional Analysis Processes

Functional Analysis is one of the major Systems Engineering activities/processes. Two extremely important benefits of Functional Analysis are that it discourages single-point solutions, and it aids in identifying the desired actions that become lower-level functions/requirements. Design teams typically include experts in the product field. Their knowledge makes for a better design. The drawback to that approach is that those with extensive design experience tend to start designing items before sufficient requirements have even been identified. It's like a reflex; they can't help it. Designers often drive towards single-point solutions without sufficiently considering/examining alternatives. Functional analysis yields a description of actions rather than a parts list. It shifts the viewpoint from the single-point physical to the unconstrained solution set. Although this may sound like functional flows deal only with the abstract, that is not the case. The set of functional flows eventually reflects the choices made in how the system will accomplish all the user's requirements. This characteristic is more apparent as you progress to the lower levels of the functional hierarchy.

Products have desired actions associated with them. These are usually actions that are visible outside the system/product, and directly relate to satisfying the customer's needs/requirements. Those that are internal to the system/product reflect functional and physical architectural choices made to implement the higher-level functions/requirements. Actions/functions are of interest in Systems Engineering because they really reflect requirements. Requirements associated with subordinate functions, themselves, will have to be accomplished by subordinate system elements. Functions, their sequential relationships, and critical timing need to be determined clearly to derive the *complete set* of performance requirements for the system or any of its subordinate system elements.

Functional analysis supports optimal functional and physical groupings to define interfaces. Verification, testability, and maintainability also improve through functional and interface analysis. Systems are less complicated and easier to support if the inputs and outputs of the subsystems and the interactions between subsystems are minimized.

Functional Analysis, alone, does not yield requirements**.** It <u>does</u> provide the *essential framework* for deriving the performance requirements for the system/product. Functional Analysis, working in tandem with requirements analysis provides a different approach for developing requirements for subordinate system elements. Other approaches flow requirements down to subordinate elements in the spec tree. Functional (requirements) analysis, on the other hand, by decomposing functions to produce the next level functional diagrams (FFBDs, IDEFs, etc), initially flows *functions* down without regard to what system element will perform them. Following the initial decomposition, alternate functional groupings are assessed to minimize interface complexity and determine candidate physical elements/resources that may be required for each alternative functional grouping. Of course, technology, risk, and cost trades are performed on the viable functional/physical choices as necessary.

Requirements are then derived to accomplish the functions, and each requirement is allocated/assigned to the system element that will then perform it. This approach facilitates system integration because as the requirements are derived, those that identify a need to *receive inputs from,* or identify a product that needs to be *output* to, another entity can be worked to find a solution with minimal impact. In this way, functional analysis allows better functional and physical groupings for interfaces. Verification, testability, and maintainability improve through function and interface analysis. Systems are less complicated and easier to support if the inputs and outputs of subsystems and the interactions between subsystems are minimized.

The first step in this process is identifying the system's functions. For any system/product, while there may be relatively few functions that can be identified from analysis of system-level user requirements and desired behaviors; there may be a larger number of possible functional architectures. There is no single right answer. Some approaches will be more productive in supporting the derivation of requirements than others. If the architecture selected starts to become a hindrance, go back and regroup. Knowing the shortcomings of the present architecture will help in developing its replacement. Contrary to what some texts may indicate, the customer's concept of the system's functions may not be the one on which to base your functional analysis, but a sound understanding of what is really wanted is. This is not license to ignore the customer's wants, merely an invitation to explore other alternatives. The odds are that the functions chosen by the customer may not have been well thought out, and the functions' boundaries and scope are more than a little fuzzy. Sometimes the customer's description of the system provides more insight as to what is wanted than does their concept of the functions, or the requirements portion of their requirements document. The functions ultimately developed/chosen must accurately model the system's performance. Usually the architecture chosen is presented to the customer in a design review to make sure there is comfort with your choice.

Most engineers have little difficulty identifying primary or active functions of the product. For any communications system it's easy to recognize the need for a data transmitting, a data receiving, and an operations control function. Supporting functions seem to be harder to grasp. Although not specified by the user, it may be customary (or mandated by overlooked directives) to archive data transferred. The archiving and retrieval would have to be captured by the functional architecture. The fact that the user wants the product to be continuously available, operable in an automobile, and transportable on his wrist is a little harder to work into lower-level functional requirements. These are design constraint requirements, and with the exception of the "continuously available", would not even need to be reflected in lower level flows. The means of *achieving* the availability would eventually have to be reflected in the much lower level flows. If there were redundant components, the automatic switching from the failed component to the operable spare would need to be portrayed in the flows, as would the sensing that a failure had even occurred.

The application of Functional Analysis is not limited to the system as a whole. It can be applied at any given level of product hierarchy within the system. Similarly, Functional Analysis is not limited to the Operational System; it may, and should, be applied to the development of requirements for the support equipment, training equipment, and facilities. These functions interrelate with the Operational System functions and coexist with them.

Functional Analysis Methodologies.

No single functional analysis methodology is sufficient by itself. Different types of requirement related information may be handled by the various implementation methodologies. Discussed below are two of the common methodologies widely used, the functional flow block diagram and timeline analysis.

Functional Flow Block Diagrams (FFBDs)

FFBDs* portray the sequential relationships among functions at each given level, and provide a framework for deriving performance requirements for the system and/or all subordinate system

---

* NOTE: FFBDs may also be referred to as functional flow diagrams. Some may even refer to them as Functional Block Diagrams, but that term has alternate interpretations. One common meaning of functional block diagrams refers to diagrams describing the relationships among functional areas (or physical elements) of a system. The relationships/interactions among

elements. FFBDs are the means used to document the Functional Analysis. Figure C.5-1 shows the typical symbology used in block diagrams. A detailed discussion of the symbology/conventions used follows.

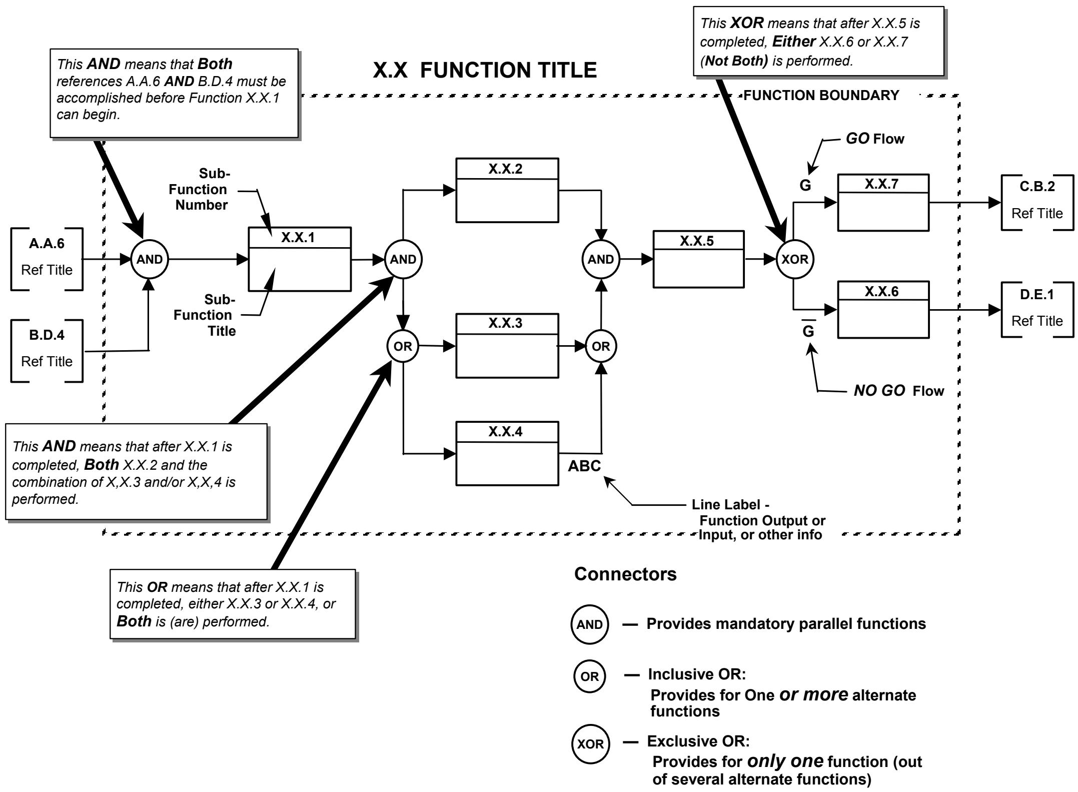


Figure C.5-1. Sample Functional Flow Block Diagram (FFBD) – typical symbols used in FFBDs

Function Blocks on a FFBD are shown as a solid box having a number and a title. The traditional form contains the number in a separate "banner" at the top of the box, and the title in the major portion of the box. The number is unique to that function, and has nothing to do with the sequence in which the functions may be performed; it identifies the function's level within, and relationship to, the functional hierarchy. For example, the top-level system flow, FFBD 0.0, shows the sequential relationships among Functions 1.0, 2.0, 3.0, 4.0, 5.0, etc. When Function 5.0 is decomposed (i.e., broken into its component parts), relationships among Functions 5.1, 5.2, 5.3, 5.4,etc., and the functions/entities external to function 5.0 would be shown. Decomposing Function 5.4 would portray relationships among Functions 5.4.1, 5.4.2, 5.4.3, 5.4.4, 5.4.5, etc., and the functions/entities external to function 5.4. Using titles without a numbering scheme would make it extremely difficult to recognize where a particular function/FFBD would fit in the functional hierarchy. Function titles must consist of an active verb and a noun. (Other parts of speech are optional and may be used to narrow or clarify the scope to the function). Ideally, the noun should be a measurable attribute, and the verb-noun combination something verifiable. Nouns should not be a part or activity. This can prove difficult at first. For example, "provide power" is better

---

the prime items of a segment might be shown in this form of a Functional Block diagram. This particular application of the term *Functional Block Diagrams* is also known as *Schematic Block Diagrams* (SBDs) stated as "power electronics." Active verbs are something that can be demonstrated. Keep it functional, and avoid describing physical parts.

External Reference Blocks represent other entities or functions that are external to the function depicted by the diagram. On the 0.0 FFBD, the reference blocks are all entities that interact with the system but are external to it. These are shown as dotted boxes on the left and right sides of the FFBD. An alternate, and more traditional way, is to use "brackets" instead of a dotted box.

When a function is decomposed, it is important to depict accurately the preceding and succeeding functions and reference blocks that appear on the higher level FFBD as external reference blocks on the decomposed FFBD. Since the external reference blocks on the 0.0 FFBD (Top-Level System Flow) are shown to interact with the system functions on the 0.0 FFBD, that interaction must also be captured when those functions are decomposed. All of the external reference blocks on the 0.0 FFBD must appear on at least one of the FFBDs depicting decomposition of the 0.0 FFBD functions, and on down through the hierarchy. If they have no relationship to the parts of the decomposed functions, they could not have had any relationship to the functions at the 0.0 FFBD. On lower level FFBDs, functions from the higher level FFBD *must* appear as reference blocks on the left and/or right sides of the subject FFBD, and be linked by sequencing arrows to the appropriate sub-function(s), *if* they are precursors or successors to the subject function on the higher level diagram. Maintaining the relationships portrayed on higher level FFBDs at the next lower level is essential to ensuring the integrity of the functional analysis. If this is not done, the process breaks down. Functions do not exist in isolation; there is always at least one function or one reference (function or external entity) that precedes it, and almost always at least one that follows it. That is why functional flows *flow*. (The one exception that forces the use of "almost always" might be the function: *Disposing of the System/Components*.)

There is another instance where *external* reference blocks are used. That is when you utilize a function from an existing FFBD rather than identify a new function with the same performance as the already existing function on the other diagram. When this is done, it is essential to go back to the FFBD on which the reference block originally appears as a *function* block, and show the functions with which it interacts (from the FFBD where it is "borrowed" as a reference) as reference blocks on the left and/or right sides of the flow, as appropriate. This is necessary so that all functions with which the "borrowed" function interacts are portrayed in one location, its primary usage location.

Internal Reference Blocks also appear as dotted boxes or brackets. There are instances where, for the sake of clarity, a function within a FFBD is used in more than one location. This enables a clearer depiction of the functional relationships. The first time it appears it appears as a normal function block; for any subsequent uses on the diagram, it appears as a reference block.

Floating Block may be either a Function Block or a Reference Block. It is called a Floating Block because no sequencing arrows (see below) connect it to any other Function Block on that diagram. It may be used when the subject block is a precursor to, and/or a successor to, ***all*** the other Function Blocks on the diagram. In either use, the key consideration is that it relates to ***all*** the other functions.

1. As a Reference Block:

    a.) If it appears as a Reference Block on the left edge of the diagram (along with the other Reference Blocks on the left side), it is a precursor to all the Function Blocks in the diagram.

b.) If it appears as a Reference Block in the right edge of the diagram (along with the other Reference Blocks on the right side), all the Function Blocks in the diagram are precursors to it,

c.) If it appears as a reference block in the bottom center of the diagram, it is both a precursor to, and a successor to all the Function Blocks in the diagram.

2. As a Function Block: Although a Floating Function Block cannot have any sequencing arrows connecting it to any other Function Block on the diagram, it may have sequencing arrows connecting it to reference blocks on either the left or right side of the diagram but NOT both.

a.) If it appears as a Function Block towards the bottom-left of the diagram, it is a precursor to all the Function Blocks in that diagram.

b.) If it appears as a Function Block towards the bottom-right of the diagram, all the Function Blocks in the diagram are precursors to it.

c.) If it appears as a Function Block in the bottom-middle of the diagram, it is both a precursor to, and a successor to all the Function Blocks in the diagram. NOTE: Other programs may use the bottom-middle positioning to indicate that the Floating Function Block is only a precursor to all Function Blocks on the diagram.

Sequencing Arrows indicate the sequence in which functions are performed. An arrow leaving one function and entering another indicates that the function into which the arrow enters is performed after the one from which it exited. An arrow entering a function almost always enters from the left (never from the right) and almost always exits from the right (never from the left). The above statement is qualified with "almost always" because there are rare instances where arrows enter the top of a function block and/or exit from the bottom. *Arrows are unidirectional; they never have two heads.*

FFBDs *are not data flow diagrams*; they <u>do</u> indicate the sequence in which the functions are performed. If some of the functions being performed are involved with the processing or transferring of data (or some other product), some of the function sequences would correspond to a data (or product) flow. On a FFBD there is often a mix of functions that process/transfer product, and functions that perform other activities. So, in some instances the sequencing arrows may indicate an actual product transfer from one function to another; in other instances nothing more than an implication that "this function is/may be performed next." This duality is sometimes difficult to grasp.

To help clarify the relationship of the functions connected by a sequencing arrow, *arrow/line* labels may be used. The label could indicate the "product" transferred from one function to the next function, or describe the conditions associated with each of the alternate paths. Both uses (the "GO – NO GO" alternatives, and "ABC Function Output/Input") are portrayed within Figure C-1.

Connectors. Any time it is intended to show that more than one function may be performed before a function, or may be performed after a function, a connector is utilized to join the sequence arrows linking the functions. The type of junction must be defined, and connectors are the means used to define the junction. The approach described here is not universal; some approaches do not distinguish between inclusive and exclusive ORs, while others do not use inclusive ORs at all. The former approach is workable, but may lose clarity; the latter is not really workable. It is not possible to describe all possible function relationships without the use of some form of inclusive OR.

There are three types of connectors used: the AND, the OR, and the XOR. On a FFBD they appear as small circles with AND, OR, or XOR inside. The OR represents an *inclusive or*; the XOR represents an *exclusive or*. There are seven basic rules/conventions governing the use of ANDs, ORs, and XORs:

1. If two or more arrows enter an AND, **all** functions they originate from are **always** performed before the function following the AND is performed.

2. If there are two or more arrows originating from an AND, **all** functions to which they go to are **always** performed after the function preceding the AND is performed.

3. If there are two or more arrows entering an OR, **at least one** of the functions from which they originate is **always** performed before the function following the OR is performed.

4. If there are two or more arrows originating from an OR, **at least one** of the functions to which they go is **always** performed after the function preceding the OR is performed.

5. If there are two or more arrows entering an XOR, **only one** of the functions from which they originate is performed before the function following the XOR is performed.

6. If there are two or more arrows originating from an XOR, **only one** of the functions they go to is performed after the function preceding the XOR is performed.

7. Multiple inputs **and** multiple outputs to/from the same connector (AND, OR, or XOR) should not be used.

Function Descriptions may not be visible on the FFBD, itself, but are an essential aspect of Functional Analysis. The function description is a much more thorough explanation of what the function does than the title, alone. It bounds the function by limiting what is included within it: when it begins, when it ends, and what happens in the interim. It can also serve as an outline or checklist for the requirement developer(s) to insure that all aspects of the function are addressed by requirements.

Figure 18 illustrates the decomposition of functions, producing functional flow block diagrams at succeedingly lower levels of the functional architecture. This process provides the systems engineer with a hierarchy of functions that provides the framework for deriving performance requirements that will completely define the system and all its components. At any lower level, the sub-function numbering system carries a reference to the next higher level so that the functional hierarchy is easily discernible.

TIMELINE ANALYSIS

Time-line analysis supports developing requirements for the product operation, test, and maintenance. The analysis shows:

Time-critical paths,

Sequences,

Overlaps, and

Concurrent functions.

Time-critical functions affect reaction time, downtime, or availability. Performance parameters can be derived, in part, from time-critical functions. Figure C.5-2 is a sample time-line sheet for a maintenance function and illustrates that functional analysis applies to support systems as well as the prime product.

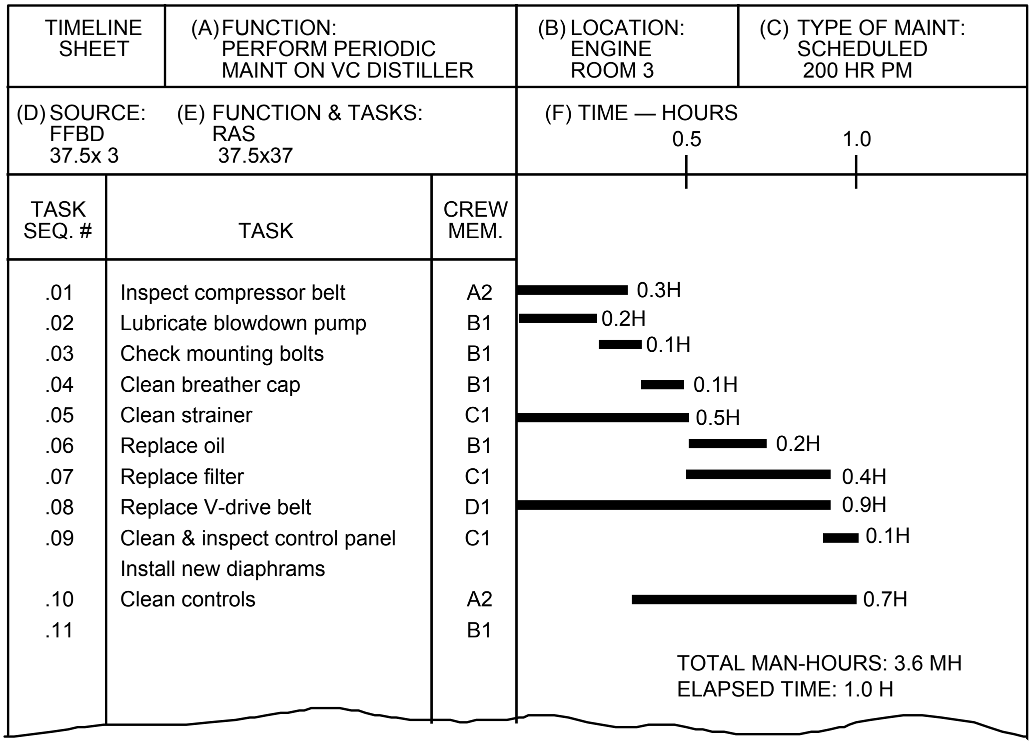


Figure C.5-2. Timeline Sheets – Show Sequence of Operations and Concurrent Action.

For simple products, most functions are constant and have a fixed relationship to their physical components. This is not the case in more complex products. Here, functions are variables with peak demands and worst-case interactions. The time-line analysis is valuable in identifying overload conditions. A matrix of function needs versus component capabilities to perform the functions can be constructed. The matrix is best left to the analysis activities after the functions have been identified.

Function Analysis Limits — Unfortunately, function analysis by itself does not adequately describe a product. Function analysis does not describe limitations, iteration, information flow, performance, or environments. However, it is a significant and essential tool is systems engineering activities. One method of relating these attributes to functions is the Quality Function Deployment (QFD) tool. See Chapter 4

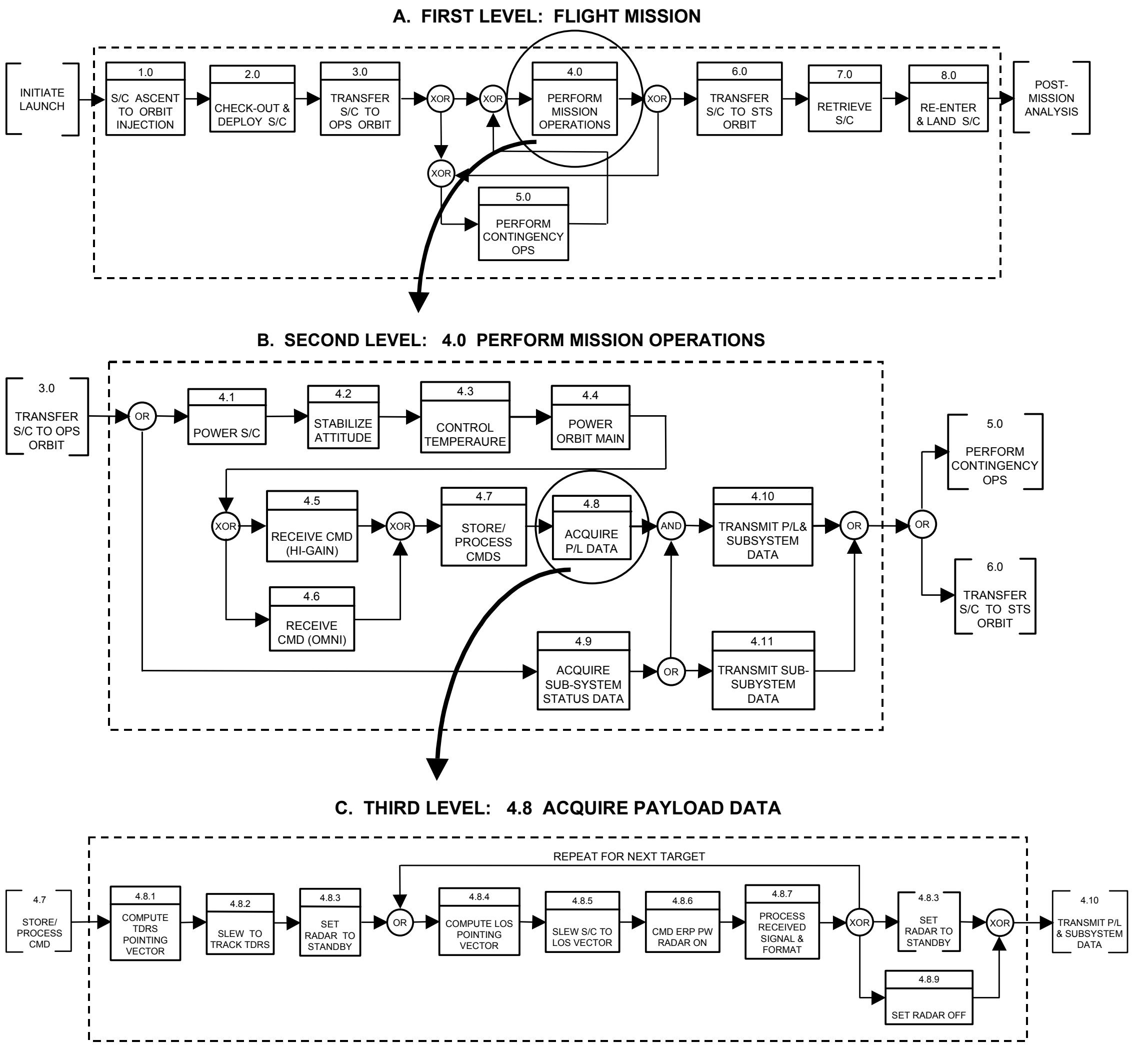


Figure C.5-3. Sample Functional Flow Diagram – showing interrelation of various levels.

Appendix C.6 – Example of System D_o Allocation and Assessment Process

Lets take an example of a two satellite system with redundant ground facilities. The customer only requires one of the two satellites to operate to meet the minimum mission requirements. The requirement for mission life is one year with a desire to continue it for at least four or more years. Of course there is a strong desire that both satellites operate throughout their lifetimes. The required probability of success of completing the one year mission is 0.9 with a goal of 0.97. An assumption is made that the launch is successful.

Preliminary Requirements Allocations: Step one is to assign a preliminary set of reliability and maintainability requirement that meet the system requirement usually based on engineering judgment.

Accepted goal of 0.97 as requirement

Mission payload equipment needed to perform mission defined in system specification to be in an up and operable state at least 97% of the mission time

Space Allocation
SV design life = 5 years
SV MMD = 4.5 years

Ground Allocation
Ground station A (MTBF = 450 hours; MTTR of any individual unit = 72 hours)
Ground station B (MTBF = 475 hours; MTTR of any individual unit = 72 hours)
MTTR of the satellite after a downing anomaly = 67 hours

Methodology for analysis

Develop reliability block diagrams using baseline design

- describe all satellite subsystems, radar payload, and ground
- identify redundancy and cross-strapping
- total number of units
- heritage of each unit
- software items

Space Segment
Develop reliability model for the spacecraft system based on block diagrams

- establish a design life and calculate mean mission duration (MMD)

Modify model to reflect a single string design for spacecraft availability prediction

- calculate mean time between failure (MTBF)

Develop a mean time to restore function (MTTR) model based on historical data from other space systems

Ground Segment
Estimate MTBF for each unit

- vendor supplied data

- comparison with equipment in standard reliability handbooks
- engineering estimates

Establish preliminary estimate of MTTR for each unit considering

- minimum sparing to support availability (formal provisioning analysis deferred)
- maximum use of commercial maintenance contracts with vendors
- assumes no logistics or administrative delays for this example

Figure C.6-1 presents the results of the reliability assessment using reliability block diagrams, statistics, and failure rates in Mil-Hdbk-217. Reliability functions are calculated for each major element of the satellite and combined into an aggregate curve. Integration of this function from time 0 to the design life determines the mean mission duration (MMD) or average satellite lifetime.

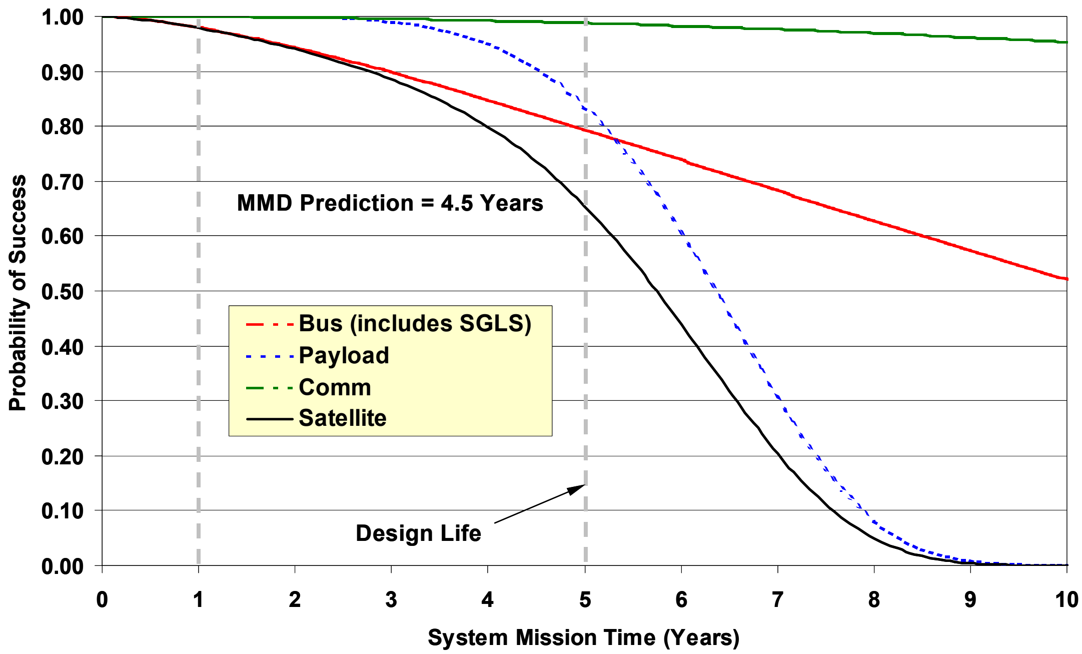


Figure C.6-1. Reliability functions calculated for each major element of satellite.

Satellite dependability is calculated using a standard equation. Mean time between failure (MTBF) is calculated by integrating the satellite reliability function from time 0 to infinity. Mean time to restore (MTTR) is based on historical information of known orbital anomalies.

Dependability = MTBF/(MTBF + MTTR)

Mean time between failure (MTBF) is 17852.8 hours (from above figure)

- Historical on-orbit anomaly resolution
- 80% of all anomalies are corrected by switchover to redundant unit in 3 days
- 15% are watch and see
- 5% require functional workaround, further analysis, software mods, etc. in 8 days
- Mean time to restore (MTTR) is 67.2 hours

Figure C.6-2 predicts the probability that either one or both the satellites will fail during the mission lifetime. The results conclude that the probability of loss of a single satellite is less than 4 percent in the first year of the mission. The loss of both satellites in the first year is much less than one percent.

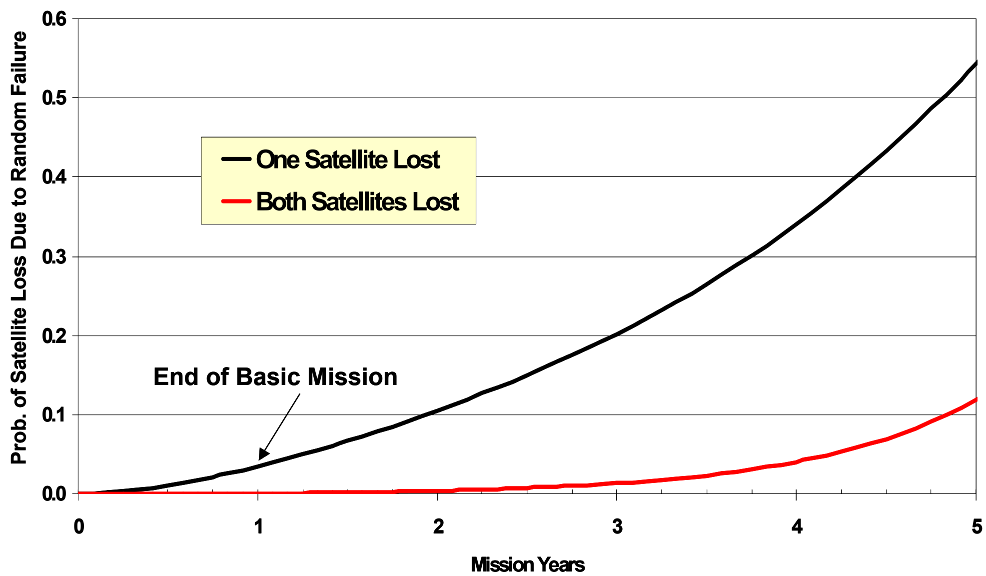


Figure C.6-2 Depiction of probability of loss of either one or both satellites due to random failure.

Figure C.6-3 is an example of a ground segment allocation. This figure provides a depiction of the probability of loss of either one or both satellites due to random failure. The assumption is made that there is no loss due to wear-out of components or expiration of design life. In this

Ground Elements	Number	Type of Redundancy	MTBF (hours)	MTTR (hours)	Individual Do
Operations Facility			475		
Antenna, trailer, Gimbal, and Electronics	1		6000	72	0.988142
Command &Telemetry Processor	1		9000	72	0.992063
Mission Data Archive (MDA)	1		9000	72	0.992063
Direct Demod/Bit Sync. (DDBS)	1		8265	72	0.991364
Data Formatter Unit (DFU)	1		75000	72	0.999041
IRIG-B	1		15000	72	0.995223
Adaptive Equalizer	1		15000	72	0.995223
Low Noise Amp.	2		9000	72	0.984190
SS High Power Amplifier	1		9000	72	0.992063
Common Imagery Processor (CIP)	1		5000	72	0.985804
Data Network	1		10000	72	0.992851
MYK-5	1		50000	72	0.998562
MYK-15	1		70000	72	0.998972
Fiber Optic Modem	2		15000	72	0.990469
SGLS Demodulator	1		9000	72	0.992063
SGLS Downconverter	1		9000	72	0.992063
SGLS Modulator	1		9000	72	0.992063
SGLS Upconverter	1		9000	72	0.992063
Dependability					**0.87252**

Figure C.6-3 Represents dependability of a single ground station.

example real equipment has been selected. MTBFs are based on historical data using the NPRD-25. MTTRs are based on engineering estimates.

Figure C.6-4 is the combined results of space and ground segment dependability. Either ground station can complete the mission without loss of data while the other is down. Combined availability for the ground segment is 0.98102. It can be seen that the mission can be successfully completed with one satellite out. This figure provides the summary results of a system dependability analysis. The conclusion is that the system will meet requirements.

Based on these results, the system engineer can allocate the preliminary requirements initially assumed to space and ground segment for implementation. The system engineer showed good engineering judgment at the beginning of this exercise. However, typically this is an iterative process to converge on an acceptable set of allocated requirements to meet the system requirement. Part of the iteration process is negotiations with segment managers to minimize their cost impacts.

System Dependability Summary	1 Sat Out	Both Operating
Space Segment	**0.99999**	**0.99251**
Ground Segment	**0.98102**	**0.98102**
System	**0.98101**	**0.97367**

Figure C.6-4 Summary results of a system dependability analysis

Appendix C.7 - An Example of a Critical Items List

The sample Critical Items List will be included in the next release version.

Appendix C.8 – States & Modes

States and Modes

States and Modes provide a means to identify different sets of conditions that will be encountered by the system/element, *and* the corresponding sets of performance requirements that the system/element must meet for each of them. They are only useful if they help clarify *what* performance is needed/expected *when*. As with other systems engineering terms used in this handbook, definitions and examples for the terms *state* and *mode* are provided below and are borrowed from James Martin's Systems Engineering Guidebook.

State: The condition of a system or subsystem when specific modes or capabilities (or functions) are valid.

Examples of states: Off, Start-up, Ready On, Deployed, Stored, In-Flight, etc.

Mode: The condition of a system or subsystem *in a certain state* when specific capabilities (or functions) are valid. Each mode may have different capabilities defined. Examples of modes within the Ready state: Normal, Emergency, Surge, Degraded, Reset, etc.

From the above definitions, it should be noted that according to this interpretation, modes are included within states. This is the most common and accepted relationship. However, the reverse convention is sometimes used. The important point is to be consistent in the use of the terms within the proper context.

Using States/Modes. The only reason for introducing states and modes into the requirements process and in the resulting specification is as a means to identify different sets of performance requirements for different sets of conditions that will be encountered by the system. It may not be obvious, but once states and modes are introduced, it is imperative that *all* the performance requirements for each mode (within each state) be delineated. Often the specification developer only thinks in terms of the requirements that may have driven him/her to identify the mode in the first place, and neglects to consider all the other requirements that would need to be performed in that mode. For example, while concentrating on the key requirements for the Autonomous Mode, the ability to receive, interpret, and execute commands needed to transition out of the mode may be overlooked. This is another instance of the "tip of the iceberg" approach that is seen all too often. The danger of not explicitly stating all the performance requirements for each and every state/mode should be readily apparent. If the requirement isn't clearly delineated, the finished system/element won't perform as expected.

Remember that once states and modes are introduced, ***all*** the performance requirements must be included within the states/modes structure; there cannot be any performance requirements that are not associated with at least one state/mode combination. Put another way, performance requirements cannot exist outside the state/mode structure. If the states/modes defined cannot include all the performance requirements, there is something fundamentally wrong with that set of states and modes, and they should be revised. In some instances, it may be that requirements that appear to exist outside the state/mode structure are really common to all states/modes, or common to some subset of the states/modes. If either is the case, it should be clearly stated that the requirements are common to whatever states/modes that share them. The author may know that the requirements are common to all or some subset of all and assumes everyone else would also. Such an assumption does not facilitate clear understanding of what the system/element is supposed to do. One shortcut sometimes employed to implement states and modes is, instead of organizing the performance requirements within the state/mode structure; a matrix is included in the specification that indicates the states/modes applicability for each performance requirement. That procedure does convey the information, but not as clearly as having all the requirements for a given mode in one place.

The use of states and modes in system level requirements documents probably came into widespread use as a result of Data Item CMAN 80008A. This was the document that specified the format, content, and structure for A-Specs (system and segment level specs). However, trying to apply states and modes to an entire system may not have been a great idea. Often, while states and modes may make sense for a subsystem or element of a system, they would be difficult to apply (or meaningless) to the entire system. Although no longer mandated, some engineers still use states/modes within their requirements documents. If states and modes are going to be used, the following structure prescribed by CMAN 80008A is still a good one to follow:

3.2.1 Performance Characteristics

3.2.1.1 State 1 Name

3.2.1.1.1 Mode 1 (within State 1) Name

3.2.1.1.1.1 Performance Capability (1)

3.2.1.1.1.n Performance Capability (n)

3.2.1.1.2 Mode 2 (within State 1) Name

3.2.1.1.2.1 Performance Capability (1)

3.2.1.1.2.n Capability (n)

3.2.1.1.n Mode n (within State 1) Name

3.2.1.1.n.1 Performance Capability (1)

3.2.1.1.n.n Performance Capability (n)

3.2.1.2 State 2 Name

3.2.1.2.1 Mode 1 (within State 2) Name

3.2.1.2.1.1 Performance Capability (1)

3.2.1.2.1.n Performance Capability (n)

In practice, the actual performance requirement title would replace "Performance Capability (n)" in the above outline. It should be readily apparent the intent of CMAN 80008A was to define all performance functions/capabilities within the structure of the states and modes. Even though CMAN 80008A may no longer be the governing directive for A- Specs, the concepts it put forth regarding states and modes are still valid.

Common/Shared Requirements. It is not uncommon for performance requirements to be applicable to more than one mode. A satellite operating in its Autonomous Mode would perform many (but not necessarily all) of the same functions that it would in its Normal Mode. In addition, it may perform some functions in the Autonomous Mode that it does not perform in its Normal Mode. Where capabilities/ requirements existed in more than one mode, CMAN 80008A prescribed identifying the performance requirement by title and referring back to the first appearance of the capability/requirement for the actual text, rather than repeating it.

Mode Transitions. Care must be exercised in considering transitioning between modes. It may not be necessary/possible to transition from each and every mode to each and every other mode. Allowable/ required transitions need to be specified. It is also necessary to consider that the transitioning begins from the current mode. Transitioning from the Autonomous Mode into the Normal Mode would be a function/capability required of the Autonomous Mode. The satellite is not in the Normal Mode until the transition is completed, so transitioning into the Normal Mode is not a capability, function, or requirement of the Normal Mode.

Appendix C.9 – C4ISR Architecture Framework

The principal objective of the C4ISR architecture framework is to define a coordinated approach for DoD architecture development, integration, and presentation. The framework is intended to ensure that architecture descriptions can be compared and relate across organizational boundaries. In February, 1998, the DoD Architectural Coordination Council mandated the use of this framework for all C4ISR architecture descriptions. It behooves the architectural system engineer to understand this methodology.

The framework prescribes three views of an architecture: operational view, system view, and technical view. The operational view is a description of tasks and activities operational nodes, and informational exchange between nodes. The system view is a graphical and textual description of systems and interconnections used to satisfy operational needs. The technical view is the minimum set of rules governing the arrangement, interaction, and interdependence of system parts and elements. Figure C.9-1 depicts the linkages between views.

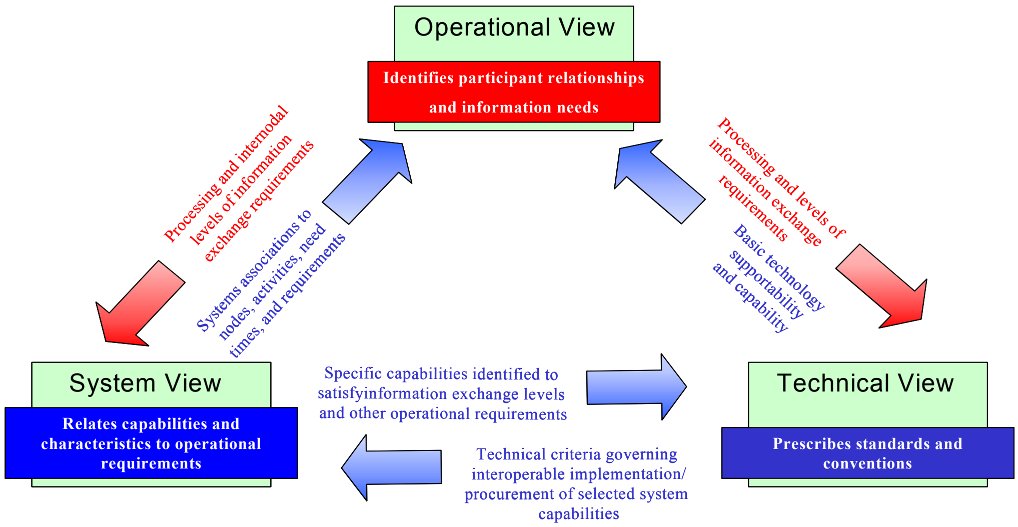


Figure C.9-1 Linkages among the three views

The framework describes a generic process for describing architectures. The six steps in this generic process are:

- Determine the intended use of the architecture description
- Determine the scope of the architecture
- Determine the characteristic to be captured
- Determine the views and products to be built
- Build the requisite products
- Use the architecture for the intended purpose

Figure C.9-2 lists the different types of views along with their general characteristics. Note that not all views are mandatory. References that provide further insight into this process include DoD Architecture Framework, Version 1.0: Volume 1, Definitions and Guidelines, Volume 2, Product Descriptions, Volume 3, Appendices

Applicable View	Product Reference	Architecture Product	Mandatory or Supporting	General Description
All Views (Context)	AV-1	Overview and Summary Information	Mandatory	Scope, purpose, intended users, environment depicted, analytical findings, if applicable
All Views (Terms)	AV-2	Integrated Dictionary	Mandatory	Definitions of all terms used in all products
All Views (Capabilities)	AV-3	Capability Maturity Profile	Supporting	Description of focus areas in terms of incremental capability levels, consistent with a standard capability maturity scale.
Operational	OV-1	High-level Operational Concept Description	Mandatory	High-level graphical/textual description of operational concept (high-level organizations, missions, geographic configuration, connectivity...)
Operational	OV-2	Operational Node Connectivity Description	Mandatory	Operational nodes, activities performed at each node, connectivities & information flow between nodes
Operational	OV-3	Operational Information Exchange Matrix	Mandatory	Information exchanged between nodes and the relevant attributes of that exchange such as media, quality, quantity, and the level of interoperability required.
Operational	OV-4	Organizational Relationships Chart	Supporting	Command, control, coordination, other relationships among organizations
Operational	OV-5	Activity Model	Mandatory	Activities, relationships among activities, inputs and outputs. Overlays can show cost, performing nodes, or other pertinent information.
Operational	OV-6a	Operational Rules Model	Supporting	One of the three products used to describe operational activity sequence and timing - identifies business rules that constrain operation
Operational	OV-6b	Operational State Transition Description	Supporting	One of three products used to describe operational activity sequence and timing - identifies business process responses to events
Operational	OV-6c	Operational Event/ Trace Description	Supporting	One of three products used to describe operational activity sequence and timing - traces actions in a scenario or sequence of events
Operational	OV-7	Logical Data Model	Supporting	Documentation of the data requirements and structural business process rules of the Operational View.
Systems	SV-1	System Interface Description	Mandatory	Identification of systems and system components and their interconnections, within and between nodes
Systems	SV-2	Systems Communications Description	Supporting	Systems nodes and their related communications laydowns
Systems	SV-3	Systems²Matrix	Supporting	Relationships among systems in a given architecture; can be designed to show relationships of interest, e.g., system-type interfaces, planned vs. existing interfaces, etc.
Systems	SV-4	Systems Functionality Description	Supporting	Functions performed by systems and the information flow among system functions
Systems	SV-5	Operational Activity/ System Function Traceability Matrix	Supporting	Mapping of system functions back to operational activities
Systems	SV-6	System Data Exchange Matrix	Supporting	Extends the Operational Information Exchange Matrix OV-3 to show source & destination systems and details of data being exchanged
Systems	SV-7	System Performance Parameters Matrix	Supporting	Performance characteristics of each system(s) hardware and software elements, for the appropriate timeframe(s)
Systems	SV-8	System Evolution Description	Supporting	Planned incremental steps toward migrating a suite of systems to a more efficient suite, or toward evolving a current system to a future implementation
Systems	SV-9	System Technology Forecast	Supporting	Emerging technologies and software/hardware products that are expected to be available in a given set of time frames, and that will affect future development of the architecture
Systems	SV-10a	Systems Rules Model	Supporting	One of three products used to describe systems activity sequence and timing -- Constraints that are imposed on systems functionality due to some aspect of systems design or implementation
Systems	SV- 10b	Systems State Transition Description	Supporting	One of three products used to describe systems activity sequence and timing -- Responses of a system to events
Systems	SV-10c	Systems Event/Trace Description	Supporting	One of three products used to describe systems activity sequence and timing -- System-specific refinements of critical sequences of events described in the operational view
Systems	SV-11	Physical Data Model	Supporting	Physical implementation of the information of the Logical Data Model, e.g., message formats, file structures, physical schema
Technical	TV-1	Technical Architecture Profile	Mandatory	Extraction of standards that apply to the given architecture
Technical	TV-2	Standards Technology Forecast	Supporting	Description of emerging standards that are expected to apply to the given architecture, within an appropriate set of timeframes

Figure C.9-2. Types of architecture views developed using the C4ISR methodology

Appendix C.10 – Example of Threat Evaluation and CAIV Study

The following is an example of how system engineering can perform a top-level assessment of threats to a system. Through the use of models associated with countermeasure implementation, a CAIV analysis can be performed to optimize threat protection and cost to the system.

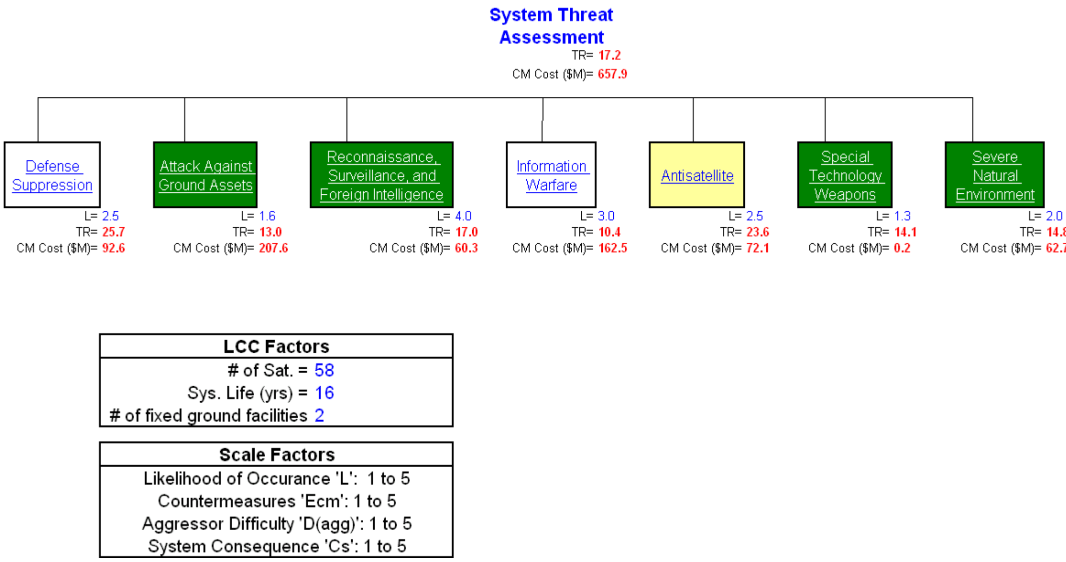


Figure C.10-1 Threat evaluation model based on generic threats in an applicable STAR. Numerical values for threat risk (TR), countermeasure cost (CM cost), and likelihood of occurrence (L) are rolled up from lower levels assessments.

Figure C.10-1 identifies top-level generic threats to a space system as defined in a STAR. A threat risk (TR) model has been developed as a direct function of consequences to the system (Cs) and an inverse

Scale Factor	Threat Imposed System Consequence (C_s)	Threat Difficulty for Aggressor (D_{agg}) Without Countermeasures	Effectiveness of Countermeasures (E_{cm})	Likelihood of Occurrence (L) Based on STAR
1	System continues to perform mission uninterrupted; Limited, minor damage (<$1M) which does not impact dependability	Threat capability currently exists in more than one potential enemy nation; Mature technology: Robust aggressor forces with worldwide operational capability; Natural defenses do not exist	No known countermeasure exists or no countermeasures have been implemented in the system against the specific threat; Expect total mission(s) failure	Very Low
2	Causes damage to facilities ($1M - $10M); System still performs its mission without impact to dependability	Threat capability exists in one potential enemy nation; Limited implementation of threat technology; Natural defenses would provide limited protection to minimize damage or compromise	Implemented countermeasures protect the system to the level which allows the primary mission to be completed while sacrificing secondary missions	Low
3	Some key elements of the system are out of commission for more than one month (or >$10M damage); Mission continues with impact to dependability	Threat technology being aggressively pursued by one or more potential enemy nation; current intelligence predicts implementation before 2010; Natural defenses would protect some key elements of the system from major damage or compromise; Moderate aggressor force with regional operational capability	Implemented countermeasures protect the system to the level which allows the primary and secondary mission(s) to be completed with degraded performance to both	Medium
4	System is partially compromised; Damaged or lost space asset; Some enemy actions can be missed	Threat technology being pursued by one potential enemy nation; current intelligence predicts implementation after 2010; Natural defenses would protect all key elements of the system from major damage or compromise; limited aggressor force with local capability only	Implemented countermeasures protect the system to the level which allows the primary mission to be completed with degraded performance to secondary mission(s) only	Medium High
5	System completely compromised; Mission halted; Most or all enemy actions can be completely missed	Threat technology does not exist; limited or no R&D being performed by potential enemy nations; Natural defenses would easily protect the system from any damage or compromise; No identified aggressor force	Implemented countermeasures are 100% effective against the enemy threat; All missions continue uninterrupted; No performance degradation	High

Figure C.10-2 Scale factors must be defined in a quantitative manner. Definitions can be changed depending on the nature of the space system analyzed.

function of countermeasure effectiveness (Ecm) and difficulty of an aggressor to impose the threat (Dagg). Likelihood (L) of the threat occurring is used as a weighted average factor when combining threats risks. A life cycle cost model is developed for each threat based on increasing effectiveness of the countermeasure. Figure C.10-2 defines scale factors used in the threat risk calculation.

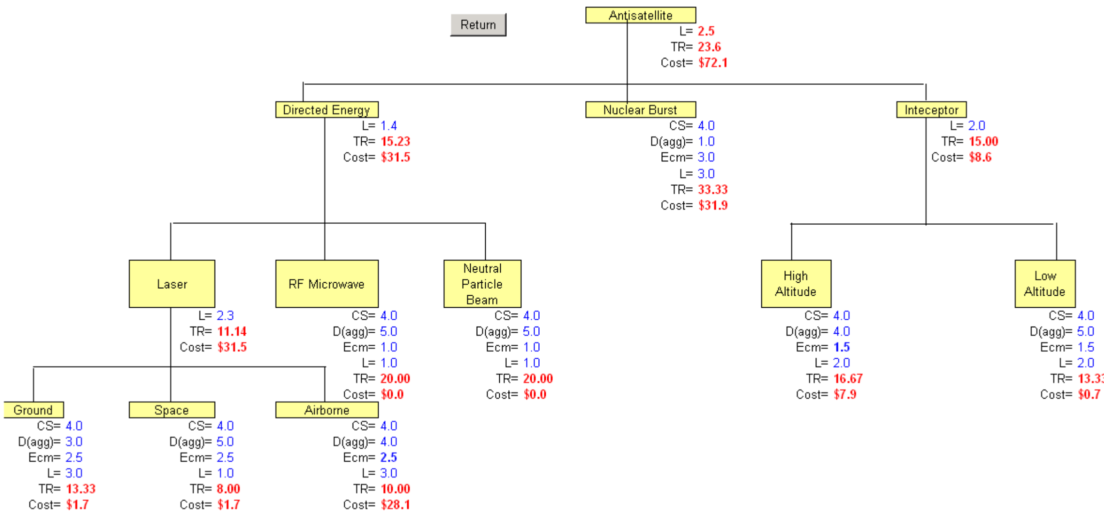


Figure C.10-3 Example of detailed threats from antisatellite weapons systems as defined in a STAR.

Figure C.10-3 breaks down the generic threats (for example antisatellite weapons) into specific threats (directed energy, nuclear burst, interceptor) as defined in the STAR. Threat risks and countermeasure costs are determined at the lowest level (i.e., airborne) and rolled up (laser). Likelihood of occurrence (L) provides a weighting factor to combine threats at the same level. The threat risk analysis begins at the lowest level of specific threats (i.e., space, ground, and airborne lasers) and is rolled up to the top level shown in D-1. A value for each threat risk parameter is determined from the definitions in Figure D-2. Effectiveness of countermeasures is defined at the lowest level of threat as are cost models for countermeasures. Results are rolled up to the system level.

Once a basic model is evaluated without countermeasures (Ecm = 1 for all threats) to give a baseline, a CAIV analysis can be performed. Through either linear programming techniques or manual manipulation in a spread sheet, the effectiveness of countermeasures on threat risk can be played off the cost of the countermeasures until a desired result is reached such as minimum threat risk or lowest threat risk for a fixed cost.

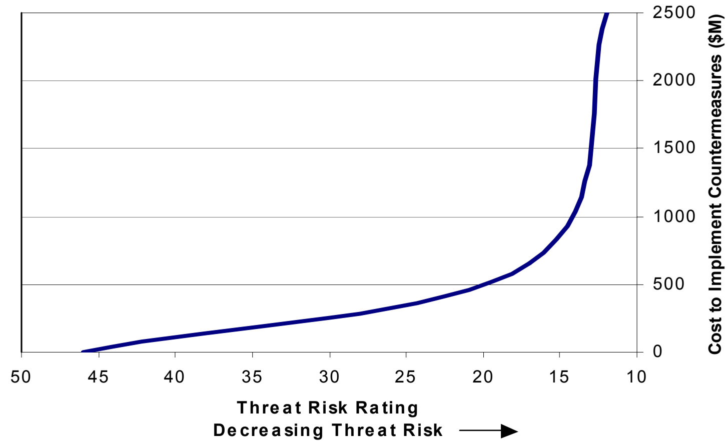


Figure C.10-4 Determine best manner of investment in countermeasure

Figure C.10-4 depicts an example of a CAIV study done for SBIRS Low. The Study performed for SBIRS Low was to determine the best manner of investment in countermeasure based on information in the NMD STAR and specifics of the system design. A system engineer should allocate requirements such that the cost of countermeasures stays well to the right of the cost cliff. Very little is to be gained by trying to reduce the threat risk below 20 or so in this example.

Appendix D – Planned Actions to Update for Final SMC SE Version (Oct 03)

- Continue to capture SMC SER initiatives as they have matured
    - SMC Common/Key Processes
    - Specs & Stds
    - Metrics
    - CMMI as it relates to SMC
    - New SMC SER policies, goals
- Provide latest on OSS&E, SFW
- Capture latest DOD And other SE related policy, directives, guidance
- Provide an example RAS with write-up in the existing appendix on allocations
- Include a SE concept and systems definition tools selection write-up
- Integrate improved Cost Estimating/EVMS discussion
- Expand examples of applied Systems Engineering … How tos
    - Develop ConOps
    - Integrate Conops into System Definition
    - Perform systems definition tasks (Add to the example methodologies already included.)
        - Development, analyses, and control of architectures
        - Development, analyses, and control Evolution of requirements
    - Generating Baseline Concept Descriptions
    - Generating CARDs
    - Generating Baseline Documents (SRDs, TRDs, Specs, ICDs)
    - SE inputs to SAMPs, APBs, and other program docs
    - Interoperability, OSA -- Implementing JTA, C4ISR
- Chapter 1 Short intro/subsection on architectures
- Chapter 3 Figure 8
    - Replace the Ground System example with a launch or satellite FFBD example
- Chapter 3, Systems Engineering – Software Development
    - Identify the latest mandates for both the evolutionary and single-step approaches, software development and integration.
- Chapter 4 Expand discussion on architecture, requirements development, change management tools
- Chapter 5 Expand discussion on systems analysis and cost estimating
- Chapter 6
    - Planning for ILS
    - ILS Tools and Techniques: The Logistics Support Analysis
    - Continuous Acquisition and Life-Cycle Support
- Appendix C-3 Sample Risk Management Plan
- Appendix C-7 Sample Risk Critical Item List
- Expand index
- Expand Bibliography

Touch up Figures 8, 28, 29, C9-2, C10-1, C10-2, C10-3

Appendix E – Customer Review & Feedback Form

This form is provided for your convenience.

	DATE: **23 March 2003**
TITLE OF DOCUMENT: **SMC Systems Engineering Primer & Handbook** **Concepts, Processes, and Techniques**	DATE OF DOCUMENT: ***Draft, 1st Edition*** **23 March 2003**

ITEM NO.	PAGE NO.	SECTION NO.	LINE NO.	COMMENT AND/OR RECOMMENDATION
	Gen			
	1			
	2			
	…			
	n			

NAME	RANK	TITLE
ORGANIZATION	DSN	E-MAIL

Request a WORD version of this form and forward completed form to:

Mr. Dave Davis david.davis@losangeles.af.mil

or

Barry Portner bportner@tor.bdsys.com .

Bibliography

[1] Boehm, Barry, A Spiral Model of Software Development and Enhancement, IEEE, May 1998
[2] Yourdon, Edward, D*ecline & Fall of the American Programmer*, Englewood Cliffs, New Jersey, 1993
[3] Myerson, Marian, *Risk Management for Software Engineering Models*, Artech House, Boston, MA, 1996

Index