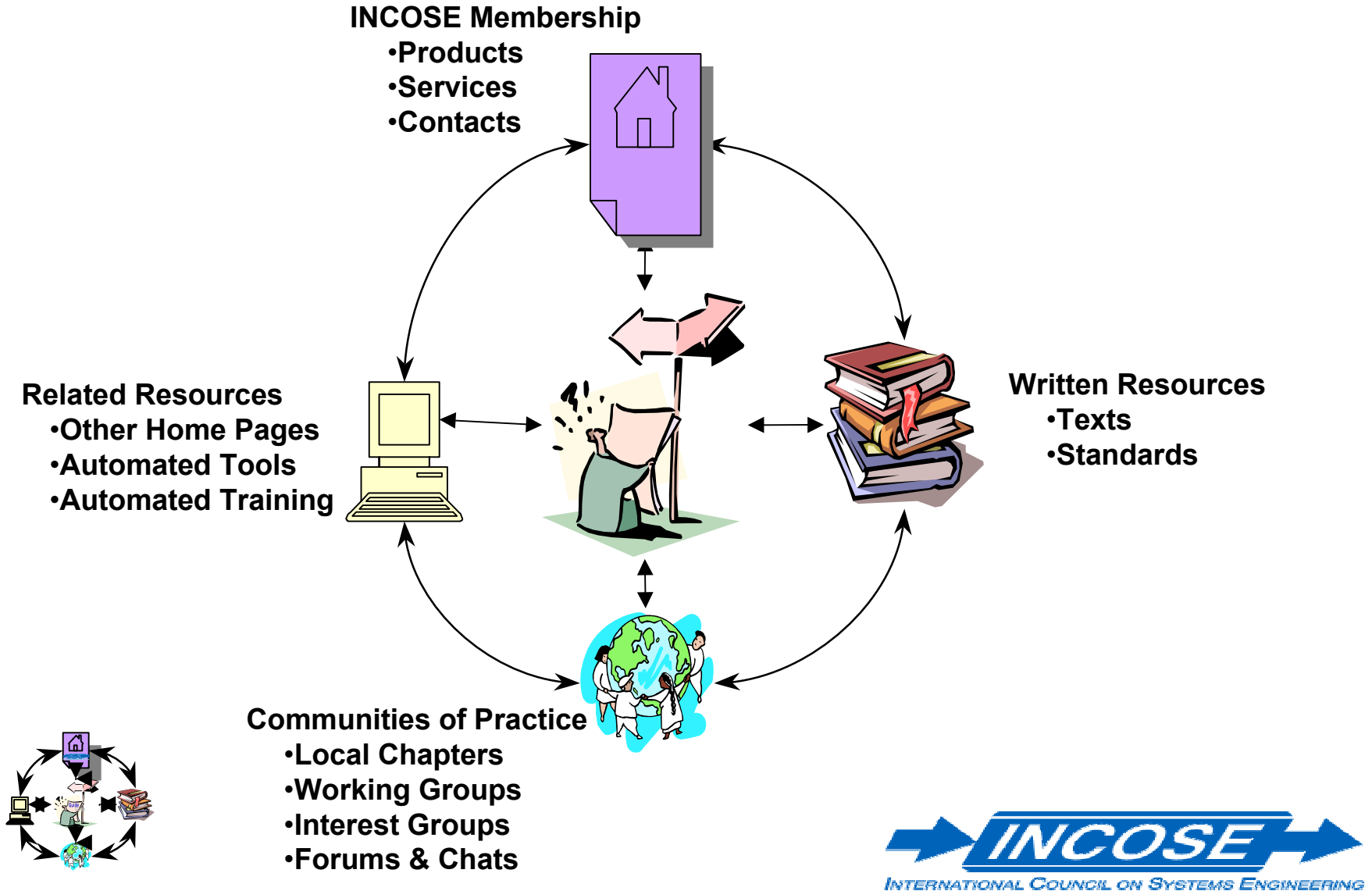


# Guide to the Systems Engineering Body of Knowledge (G2SEBoK)



# Entry into G2SEBOK

1. Systems Engineering Fundamentals

2. Systems Engineering Processes

3. Competency Development of SE Practitioners

4. SE Process Capability Assessment

What is/isn't the G2SEBoK Guide?

What is unique about the G2SEBoK?

When is the G2SEBoK useful?

Doesn't this knowledge already exist?

## Purpose

A comprehensive guide will provide a singular resource for understanding the extent of the practice of Systems Engineering for a spectrum of purposes. These purposes may include accomplished systems engineers seeking more in-depth information in a particular area of the discipline or just in time performance support or other engineering disciplines performing systems engineering tasks. Just as likely the interested neophyte or potential consumer of systems engineering services may wish to learn more about the discipline. It is critical for this guide to be both simplistic and comprehensive. The tools to accomplish this layering do exist and will be applied to intermediate and long-term evolutions of this guide.

The structure of the G2SEBoK is intended to foster inclusion rather than strictly defining what is and is not systems engineering. It would be very difficult and not entirely productive to reach consensus on a single process due simply to the diversity of the applications and the varying complexity of the systems to which the discipline is applied.

G2SEBoK Navigation Page

# Systems Engineering Fundamentals

Systems Engineering, like other engineering disciplines, is concerned with putting scientific knowledge to practical uses. SE is a knowledge creating process set in motion by a customer or sponsor defining an objective to be achieved, for example, flying a man to the moon. Output from the process is multilevel, highly precise, and highly consistent information characterizing the objective system.

## Definition of a system

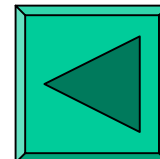
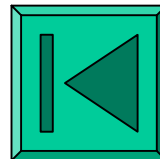
A system is a construct or collection of different elements that together produce results not obtainable by the elements alone. The elements, or parts, can include people, hardware, software, facilities, policies, and documents; that is, all things required to produce systems-level results. The results include system level qualities, properties, characteristics, functions, behavior and performance. The value added by the system as a whole, beyond that contributed independently by the parts, is primarily created by the relationship among the parts; that is, how they are interconnected (Rechtin, 2000). While the designers of a system typically address the purpose of the system that they are trying to achieve during the design phase, the purpose of the system (that is delivered) is what the system does (POSIWID).

**Systems Engineering**

**Primer on Systems Engineering Activities**

**Important Concepts of Systems Engineering**

**Key References**



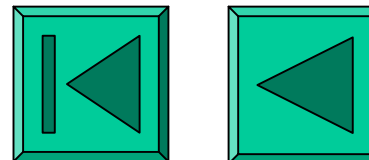
# Systems Engineering

Systems Engineering is a holistic, product oriented engineering discipline whose responsibility is to create and execute an interdisciplinary process to ensure that customer and stakeholder needs are satisfied in a high quality, trustworthy, cost efficient and schedule compliant manner throughout a system's life cycle.

***Science determines what  
IS...  
Component engineering determines what  
CAN BE...  
Systems engineering determines what  
SHOULD BE.***

***Project Management ensures that the trains run on time...  
Systems Engineering ensures that the trains run at all.***

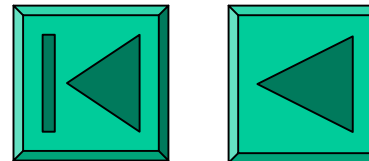
**Key References**



# *Systems engineering determines what SHOULD BE*

The role of science is to determine “what is” by determining the “laws” of science through the development of theoretical hypotheses and experimentation. All engineering is concerned with the application of knowledge and concepts from science and mathematics. Traditional engineering design of components uses scientific principles to create solutions to a specification or “what can be”. Systems engineering plays the interface between component engineering (the people in search of physical solutions) and society (the people with a need). These systems engineers work with both groups to determine “what should be” or the set of requirements for the solution to the need.

**Key References**



# Systems Engineering ensures that the trains run at all

Systems engineering brings two elements to a development project that are not usually present. First, systems engineering brings a disciplined focus on the *end product*, to include:

- (a) Its *enabling products*, and
- (b) Its internal and external *operational environment* (i.e., a system view).

The end product, of course, is the thing that performs the system mission. The enabling products are what enable the developer to produce, test, deploy, install, operate and support the end product. The operational environment involves the interactions that cross the system boundary.

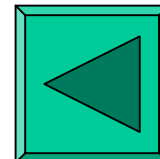
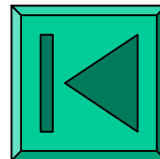
Second, systems engineering brings a consistent *vision of stakeholders' expectations* independent of daily project demands (i.e., the system's purpose or mission).

With this notion of engineering integration discussed above, it may appear that there is an inherent conflict between systems engineering and project management. In practice, this conflict often leads to confusion over roles and responsibilities and, as a result, the project suffers in terms of low morale, lower productivity, and inferior quality product.

In reality there should be no conflict. There is a need for both project management and systems engineering on development projects. The project manager should focus on the acquisition of resources (people, tools, facilities, funds, etc.) and protection of these resources from competing projects. He acts as the chief spokesman to upper management as to the importance and criticality of the project and how it fits into the overall strategic intent of the enterprise.

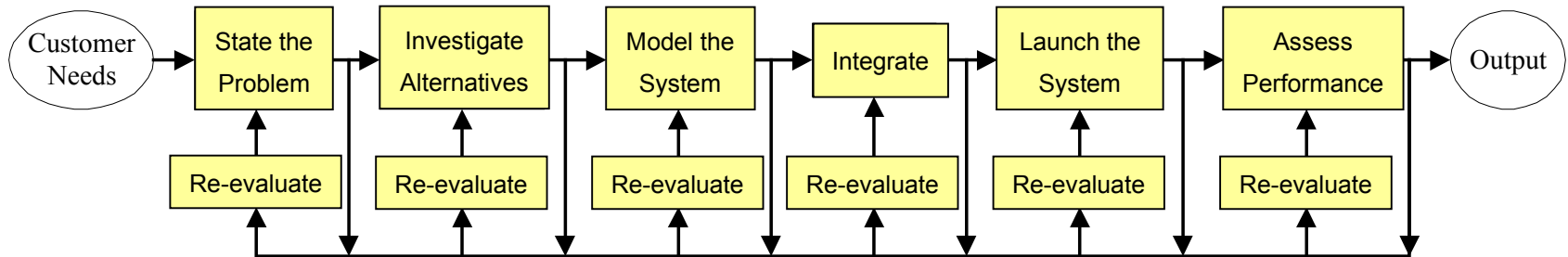
The systems engineer, on the other hand, is responsible for the efficient and effective use of these resources (working closely with the program manager), in addition to making sure the system meets the needs and expectations of stakeholders.

## Key References



# Primer on Systems Engineering Activities

One perspective is that this process consists of the following seven tasks: State the problem, Investigate alternatives, **M**odel the system, Integrate, Launch the system, **A**ssess performance, and **R**e-evaluate. These functions are captured in the acronym SIMILAR: State, Investigate, **M**odel, Integrate, Launch, **A**ssess and **R**e-evaluate. The SIMILAR process is diagrammed in Figure 1. The linearity of the functions depicted **does Not** represent sequential performance of the tasks. Functions are performed in parallel and repetitively, as better information on the objective system becomes available in any task area.

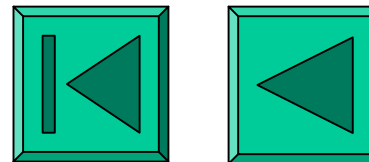


The Systems Engineering Process

from A. T. Bahill and B. Gissing, Re-evaluating systems engineering concepts using systems thinking, *IEEE Transaction on Systems, Man and Cybernetics, Part C: Applications and Reviews*, 28(4), 516-527, 1998.

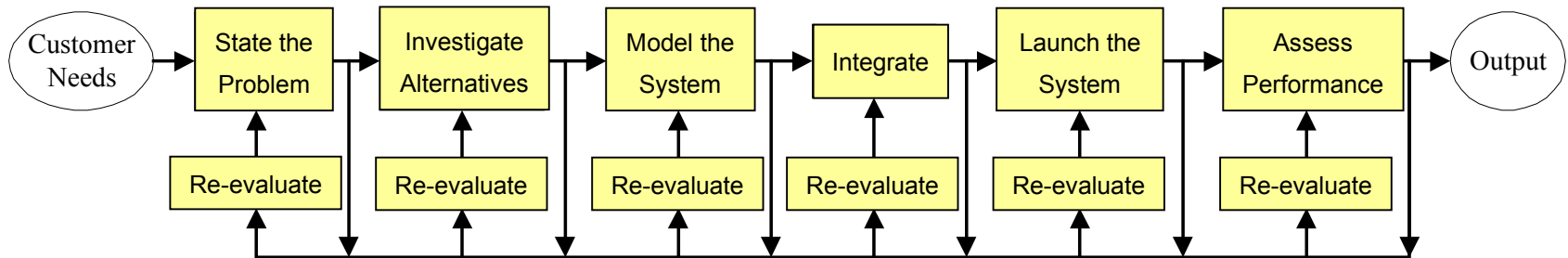
## Variations

## Key References

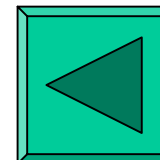
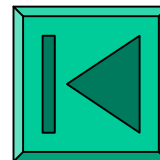


# State the Problem

The problem statement starts with a description of the top-level functions that the system must perform: this might be in the form of a vision, mission statement and mission requirements, a set of scenarios for how the system will be used and how it interacts with other systems or a description of the deficiency that must be ameliorated. Most mandatory and preference requirements should be traceable to this problem statement. Acceptable systems must satisfy all mandatory requirements. The preference requirements are traded-off to find the preferred alternatives. The problem statement should be in terms of *what* must be done, not *how* to do it. Inputs come from end users, operators, maintainers, suppliers, acquirers, owners, regulatory agencies, victims, sponsors, manufacturers and other stakeholders. The output problem statement should express customer requirements as characteristics (quality, quantity and timing) of these outputs. There are also system-wide requirements such as availability. These requirements can be composed in words or as a model.

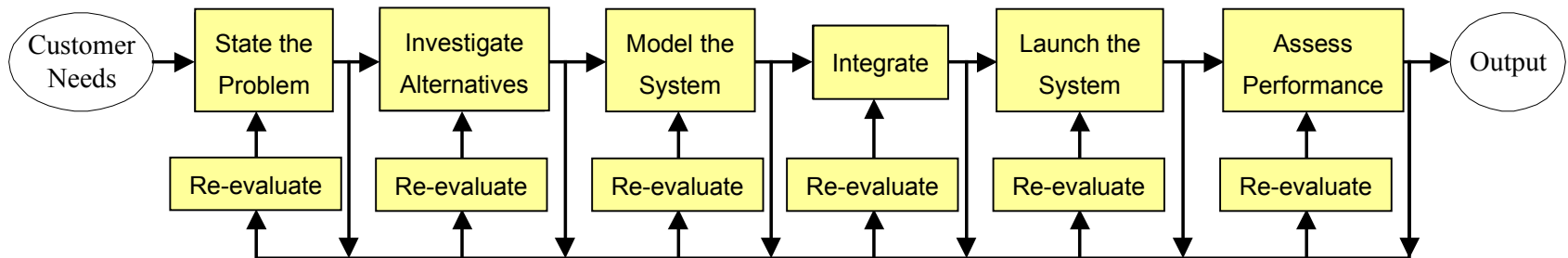


Key References

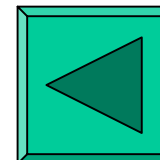
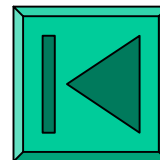


# Investigate Alternatives

Alternative solutions are created and are evaluated based on performance, schedule, and cost figures of merit. No single solution is likely to be the best across all the figures of merit. Multi-criteria decision-aiding techniques are available to help discover the preferred alternatives. This analysis should be repeated, as better data becomes available. For example, figures of merit should be computed initially based on estimates by the engineering design team. As the project evolves, models should be constructed and evaluated; simulation data should be collected and analyzed, and prototypes should be built and measured. Finally, tests should be run on the real system. Alternatives should be judged for compliance of capability against requirements. For the design of complex systems, alternative designs reduce project risk. Investigating innovative alternatives helps clarify the problem statement.



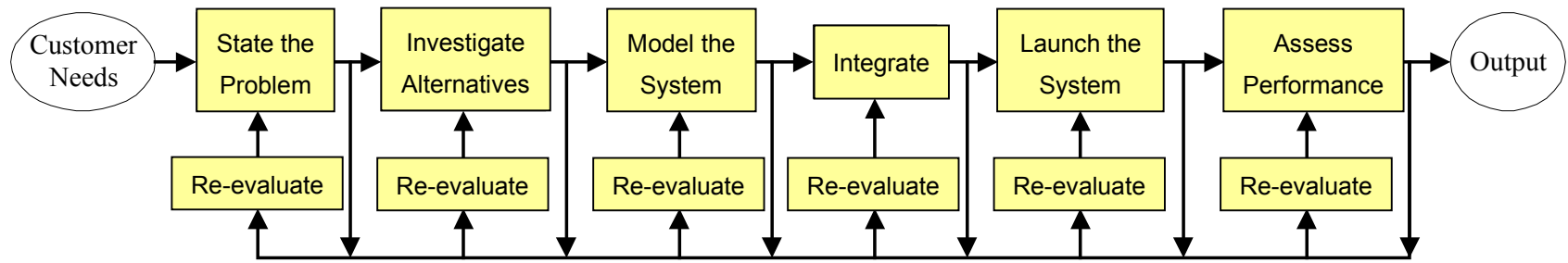
**Key References**



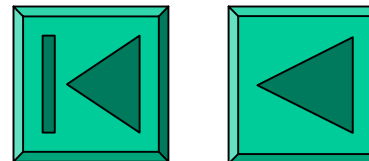
# Model the Systems

Models will be developed for most alternate designs of the product system. The model for the preferred alternative will be expanded and used to help manage the system throughout its entire life cycle. Many types of system models are used, such as physical analogs, analytic equations, state machines, block diagrams, functional flow diagrams, object-oriented models, computer simulations and mental models. Systems Engineering is responsible for creating (developing) a product and also processes for producing, deploying, training, refining, and retiring it. So, models should also be constructed for each of these processes. Development *process* models allow us, for example, to study scheduling changes, create dynamic PERT charts and perform sensitivity analyses to show the effects of delaying or accelerating certain subprojects. Running the process models reveals bottlenecks and fragmented activities, reduces cost and exposes duplication of effort. *Product* models help explain the system. These models are also used in tradeoff studies and risk management.

As previously stated, the Systems Engineering Process is not sequential. It is parallel and iterative. The complex interrelationship between creating and improving models throughout the process of developing and selecting alternatives is a good example of the dynamic nature of the systems engineering process.



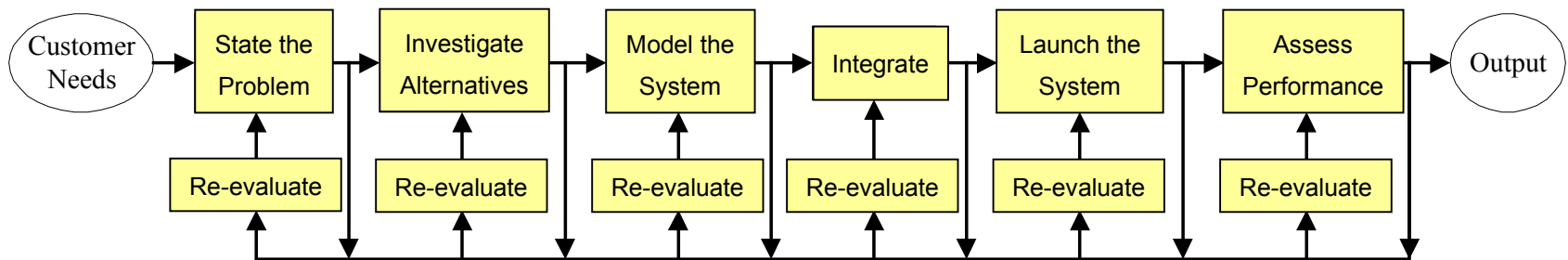
**Key References**



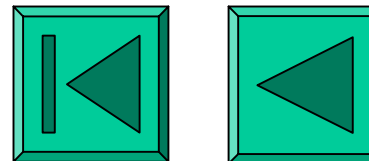
# Integrate

No man is an island. Systems, businesses and people must be integrated so that they interact with one another. Integration means bringing things together so they work as a whole. Interfaces between subsystems must be designed to facilitate the movement of physical items, information and energy. Subsystems should be defined along natural boundaries. Subsystems should be defined to minimize the amount of information, physical items and energy to be exchanged between the subsystems via interfaces. Well-designed subsystems send finished products to other subsystems. Feedback loops around individual subsystems are easier to manage than feedback loops around interconnected subsystems. Processes of co-evolving systems also need to be integrated. The consequence of integration is a system that is built and operated using efficient processes.

Typical products produced by the systems engineering process are a mission statement, a requirements document including verification and validation methods, a description of functions and objects, figures or merit, a test plan, a drawing of system boundaries, an interface control document, a listing of deliverables, models, a sensitivity analysis, a tradeoff study, a risk analysis, a life cycle analysis and a description of the physical architecture. The requirements should be validated (Are we building the right system?) and verified (Are we building the system right?). The system functions should be mapped to the physical components. The mapping of functions to physical components can be one to one or many to one. But if one function is assigned to two or more physical components, then a mistake might have been made and it should be investigated. One valid reason for assigning a function to more than one component would be that the function is preformed by one component in a certain mode and by another component in another mode. Another would be deliberate redundancy to enhance reliability, allowing one portion of the system to take on a function if another portion fails to do so.



**Key References**

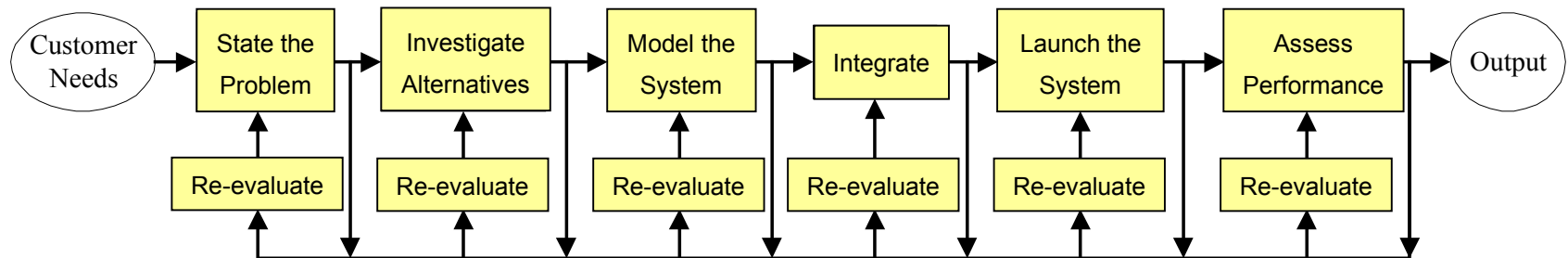


# Launch the System

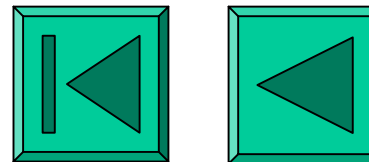
Launching the system means using the system to obtain results. This is where the design team learns if the system does what it was intended to do. This phase includes the system engineering of deploying multi-site, multi-cultural systems.

This phase is where the preferred alternative is designed in detail; components are built or bought (Commercial Off-The Shelf - COTS), components are integrated and sub-system elements tested at various levels leading to the certified product. In parallel, the processes necessary for assembling and deploying the system are developed – where necessary - and applied so that the product can be produced and deployed. In addition, training systems (if necessary) must be designed and produced for operators or maintainers.

In designing and producing the product, due consideration is given to its interfaces with operators (humans, who will need to be trained) and other systems with which the product will interface. In some instances, this will cause interfaced systems to co-evolve. The process of designing, developing and producing the system is iterative to accommodate new knowledge developed along the way to be incorporated in an improved system.

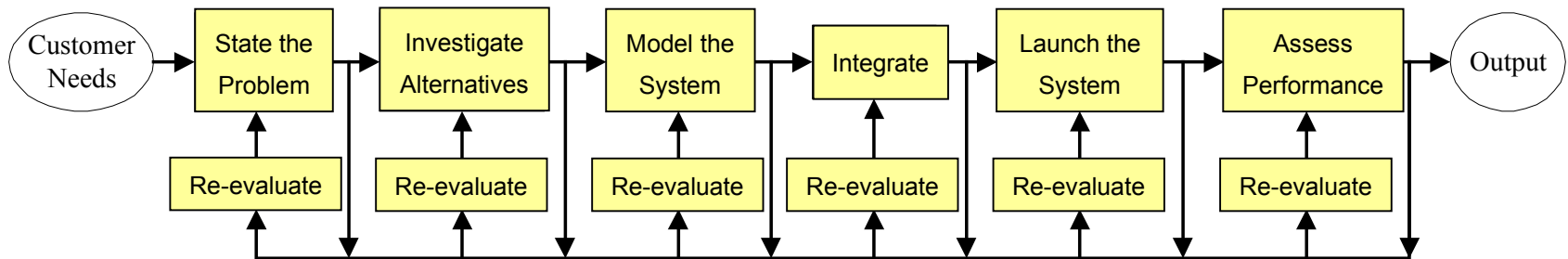


**Key References**

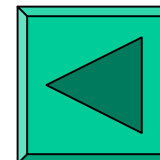
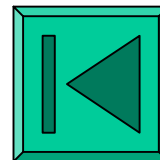


# Assess Performance

Figures of merit, technical performance measures and metrics are all used to measure performance. Figures of merit are used to quantify requirements in the tradeoff studies. They usually focus on the product. Technical performance measures are used to monitor system performance during the design, development and manufacturing. Metrics (including customer satisfaction comments, productivity, number of problem reports, or whatever one believes is critical to a venture's success) are used to help manage organization processes. Measurement is the key. If you cannot measure it, you cannot tell success from failure and therefore cannot control it. If you cannot control it, you cannot improve it. Important resources such as weight, volume, price, communications bandwidth and power consumption should be managed. Each subsystem is allocated a portion of the total budget and the project manager is allocated a reserve. These resource budgets are managed throughout the system life cycle.

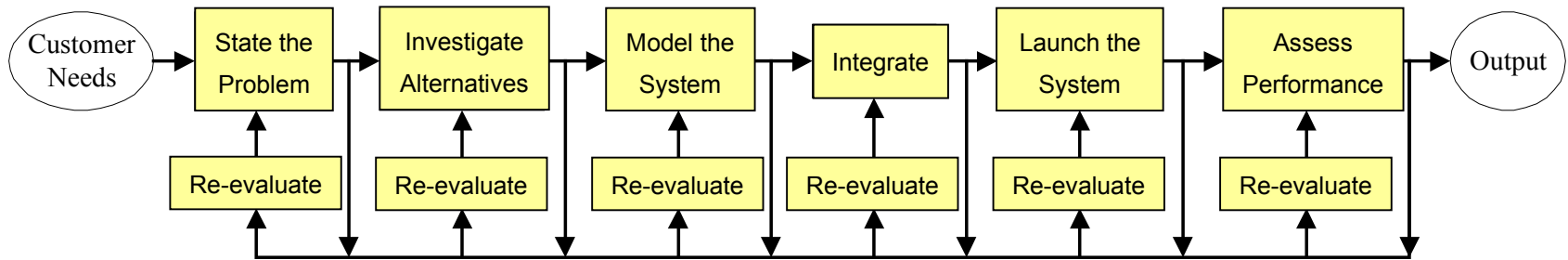


**Key References**

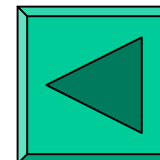
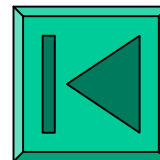


# Re-evaluate

Re-evaluate is arguably the most important of these functions. Since the first systems developed by the ancient engineers in Egypt, Greece and Rome, improved designs have been the result of feeding back what was learned in testing and using previously developed systems. Feedback is one of the most fundamental engineering tools. Re-evaluation should be a continual process with many parallel loops. Re-evaluate means observing outputs and using this information to modify the system, the inputs, the product or the process. The Systems Engineering process illustrated in Figure 1 clearly shows the distributed nature of the re-evaluate function in the feedback loops. It is not necessary for all these loops be used. The actual loops used are unique to the particular problem being solved.



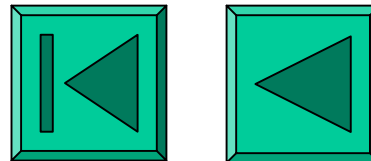
**Key References**



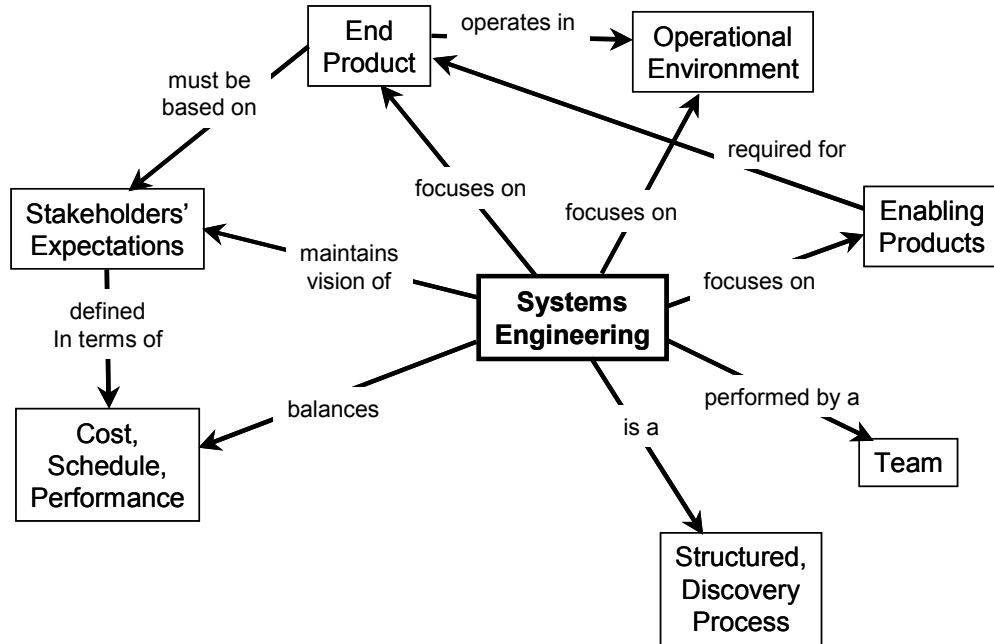
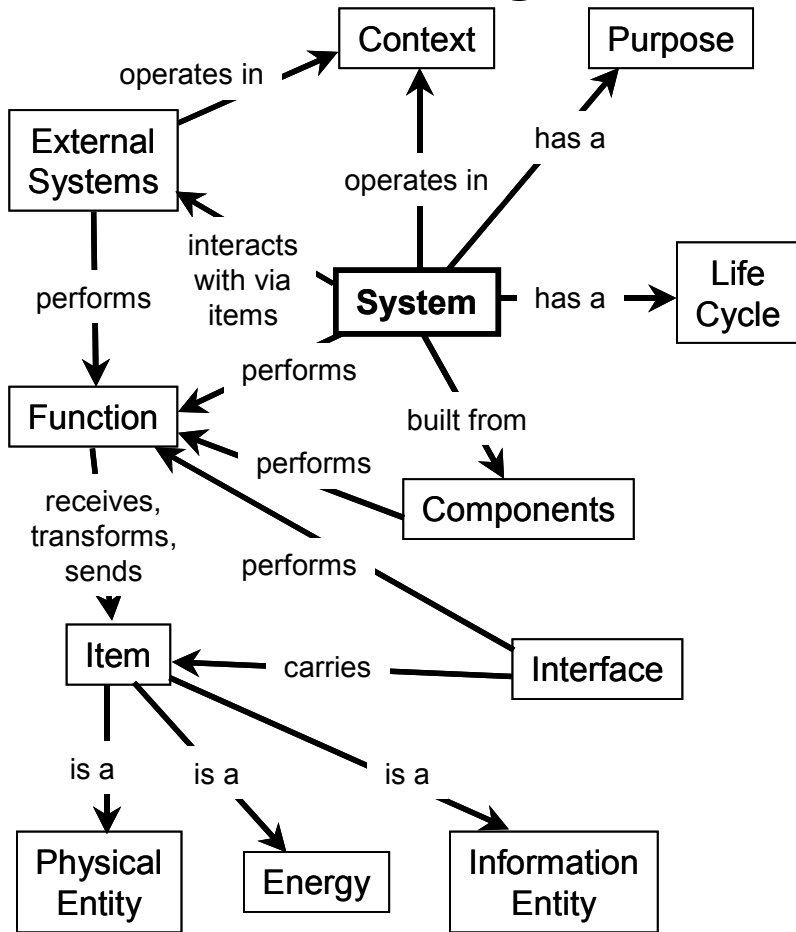
# Variations

Like all processes, good engineering practice dictates that the Systems Engineering processes implemented by an organization should be documented, measurable, stable, of low variability, used consistently throughout an organization, adaptive, and tailorable! This may seem like a contradiction. And perhaps it is. But one size does not fit all. The above description of the Systems Engineering process is one of many that have been formulated. Some are more complex and some are simpler. But all contain elements of the process discussed above.

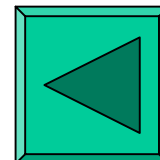
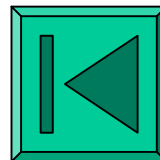
**Key References**



# Important Concepts of Systems Engineering



Key References



# **Key References: SE Fundamentals**

# Systems Engineering Processes

Traditional System Development Life Cycle Model of the U.S. DoD

Highway Design Life Cycle Process Model

Vee Model of Systems Engineering Design and Integration

Tufts' Systems Engineering Process Model

Plowman's Model of the Systems Engineering Process

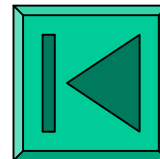
INCOSE Handbook SE Process Model

Systems Engineering is a set of processes performed by people (assisted by automated support tools). This section provides the reader with various descriptions of these systems engineering processes and their temporal relationships. The reader should understand that there is no cook-book recipe for performing systems engineering because there is such great diversity in the kinds of systems that are produced by the systems engineering activities, in the characteristics of customers and associated users/operators/maintainers/manufacturers/etc. As a result experienced systems engineering practitioners tailor their favorite processes to meet the needs of each specific project.

The purpose of this section is to describe several generic systems engineering process models, each of which highlight different aspects of an overall, complex systems engineering process. These process models shown on the buttons to the right.

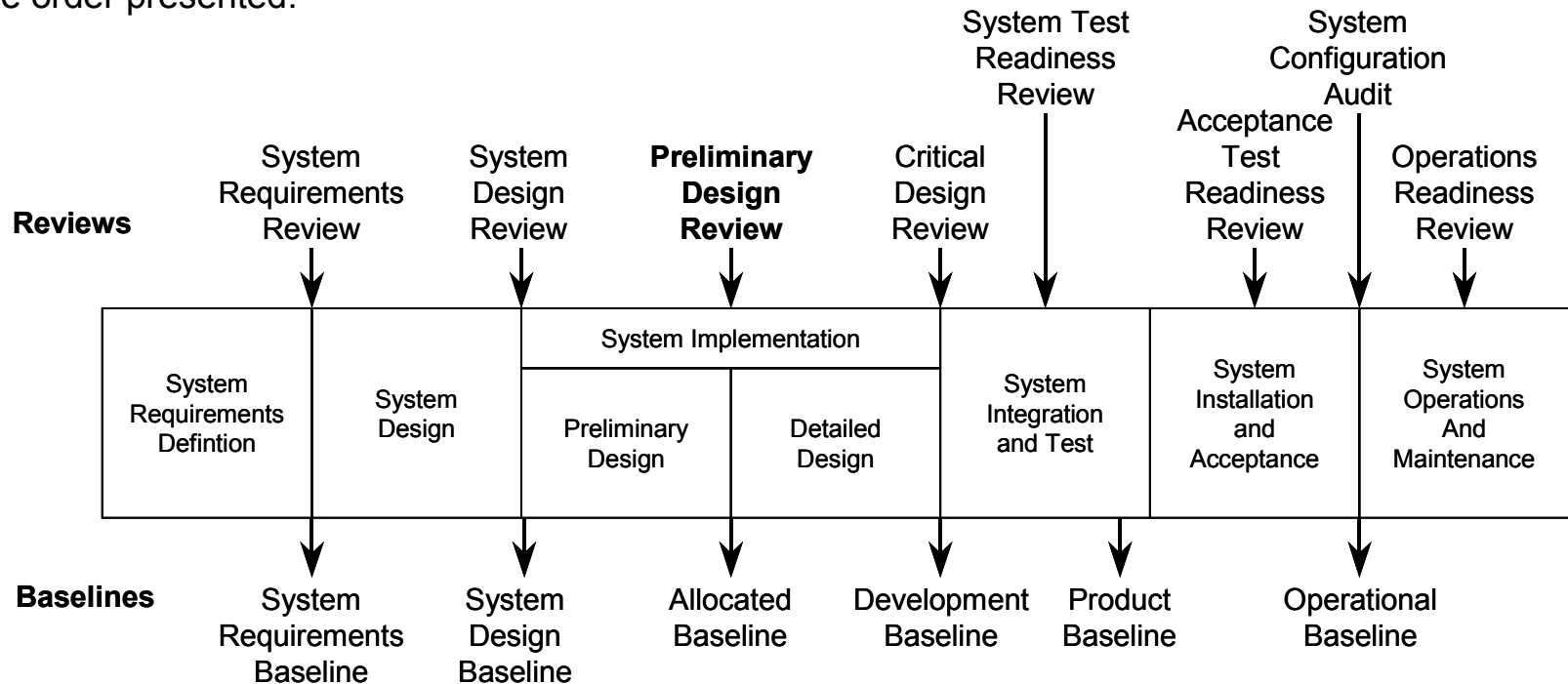
## Key References

G2SEBoK Navigation Page

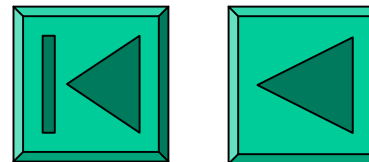


# Traditional System Development Life Cycle Model of the U.S. DoD

The traditional system development life cycle model (see Figure II-1) of the U.S. Department of Defense provides a serial representation of major activities as well as well-defined reviews and document baselines. Time progresses from left to right in this model; most people are willing to agree that the activities in the boxes should not be serial. Generally though these process start and stop in the order presented.



Key References



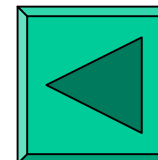
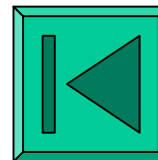
# Highway Design Life Cycle Process Model

Members of the Systems Engineering Applications Working Group of INCOSE modified the traditional system development life cycle model of the U.S. Department of Defense to represent highway design. The reviews and baselines were removed and defining issues were added.

Topographic Coordination and Requirements	Corridor Study and Design	Preliminary Design: Horizontal Design Vertical Design Cross Section Design	Detailed Design: Right of Way Plans Final Construction Plans	Highway Construction and Testing	Highway Inspection and Acceptance	Highway Operations and Maintenance
--	---------------------------------	---	--	--	--	---

- Surveying
  - Aerial Mapping
  - Photogrammetry
  - Terrain Modeling
- Design Analyses
    - Environment
    - Cultural Resource
    - Economic
    - Drainage
    - Traffic
    - Community Impact
    - Displacement
- Horizontal
    - Interchanges
    - At-Grade Connects
    - Traffic Flow
    - Restrictions
  - Vertical
    - Maximum Grades
    - Sight Distances
    - Truck Lines
  - Cross-Section
    - Roadway Width
    - Shoulder Width
    - Slope Limits
    - Pavement Template
- Right-of-Way Plans
    - Property ID
    - Easements
    - Access
    - Acquisition
    - Hearings
    - Taking
  - Final Construction Plans
    - Maintenance
    - Drainage
    - Structures
    - Grading
    - Earthwork
    - Pavement
    - Estimates

**Key References**

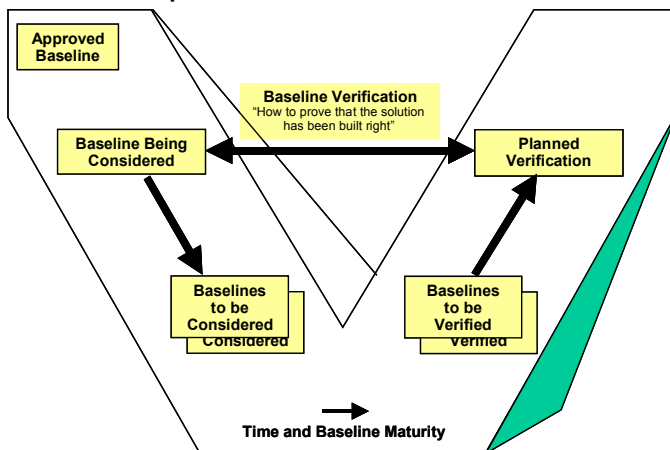


# Vee Model of Systems Engineering Design and Integration

The Vee Model is a system development model designed to simplify the understanding of the complexity associated with developing systems. Predecessor models of the Waterfall and Spiral, while making important contributions to the understanding of the process, fall short in conveying several important concepts necessary to doing development right the first time and every time.

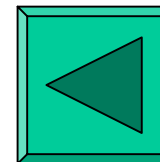
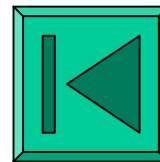
The Vee model is rooted in the project cycle, which is displayed from left to right to represent project time and maturity. Coupled with this depiction is the recognition of levels of decomposition, which are illustrated, in the vertical dimension from top to bottom. The User is at the highest level and parts and lines of code the lowest. The plane orthogonal to the plane of the Vee illustrates the number of entities at each level of decomposition, which relates to the complexity of the system.

The concept of an evolving baseline, progressively increasing in depth and under change control, creates the Vee shape, which is called the core of the Vee. The left leg of the Vee represents Decomposition and Definition and the right leg represents Integration and Verification



Decomposition and Definition

Integration and Verification



Key References

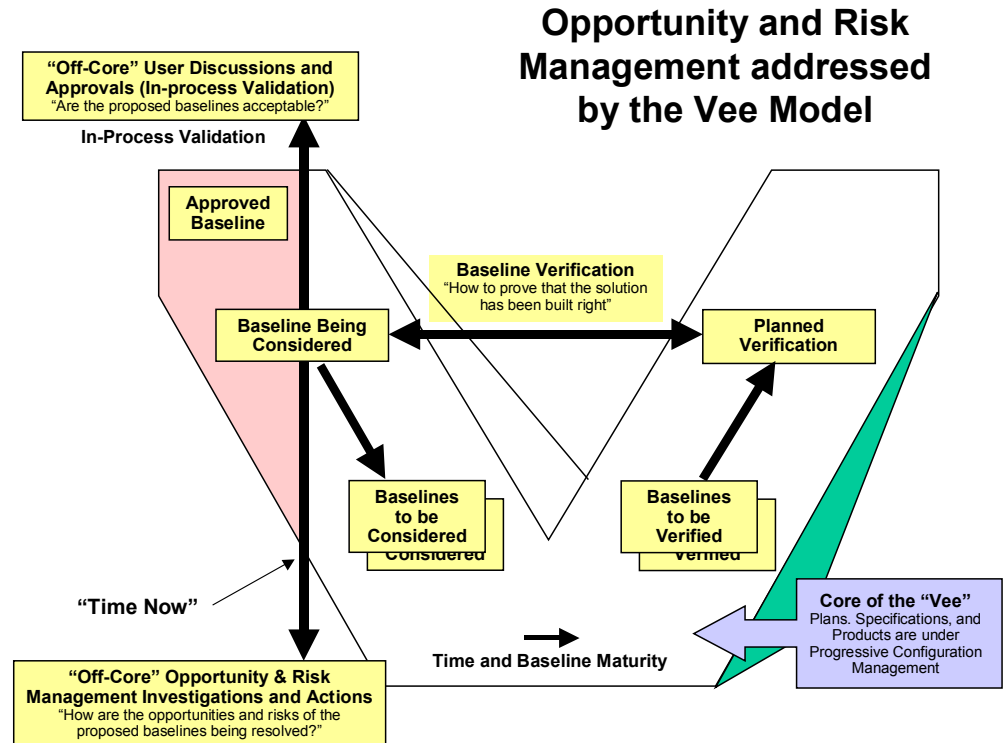
# Decomposition and Definition

An important attribute of the Vee diagram is the “Time Now” line, which explicitly highlights the iterative nature of the development process. Iterations between levels of decomposition are an important part of system development and definition. Since one can never go back in time, all such iterations are vertical on the “Time Now” line.

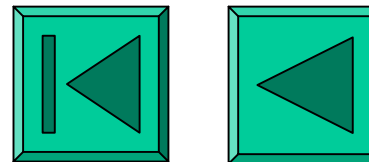
The most significant contribution of the Vee is recognition of the off-core activities. Downward off-core activities on the left leg allow investigation of opportunities and risks to justify baseline decisions on the Vee core. The off-core results are not baselined but are used to justify on-core decisions as being valid. Unlike the spiral model investigations may be conducted in series with or in parallel with normal development activities.

Upward off-core activities encourage in-process validation with the user. When the user approves progressive baselining decisions then it is highly probable that the end product will be satisfactory.

The Decomposition Analysis and Resolution Process orthogonal to the left Vee leg provides for the trade studies leading to the baselining decisions.



## Key References

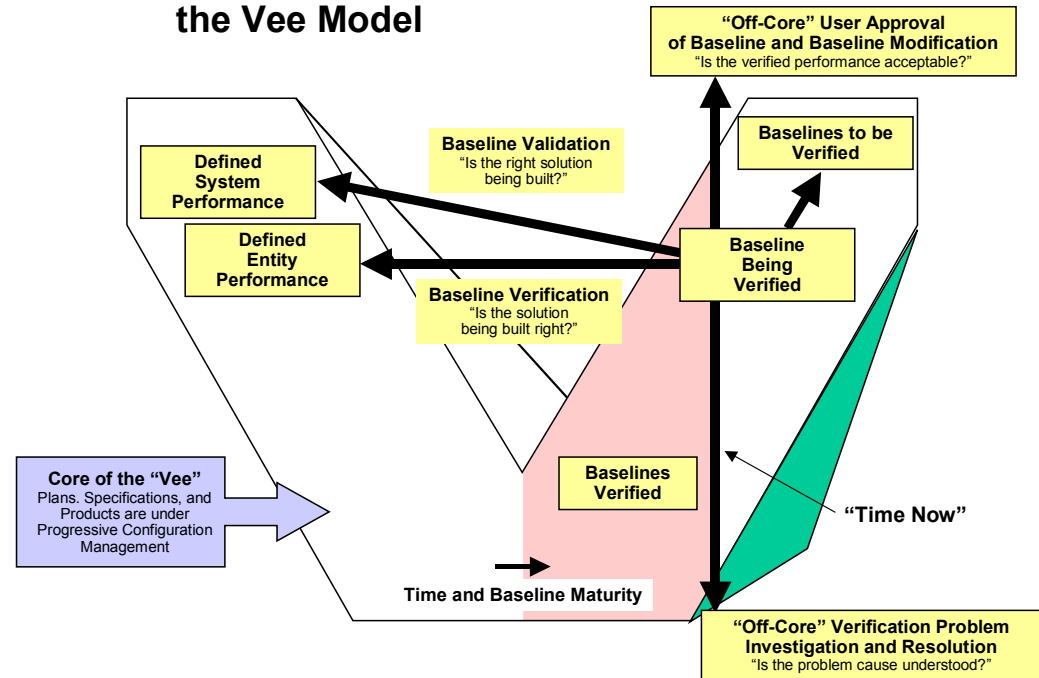


# Integration and Verification

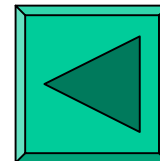
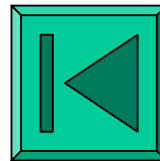
Downward off-core activities on the right side of the Vee are used to investigate and resolve anomalies discovered during integration and verification, which may lead to adjustment of the as verified baseline. Again upward off-core activities provide in-process validation with the user to achieve continuous buy-in to the solution as it evolves.

The Verification Analysis and Resolution process orthogonal to the right Vee leg provides for anomaly correction and/or baseline modification.

## Integration and Verification Management addressed by the Vee Model



Key References



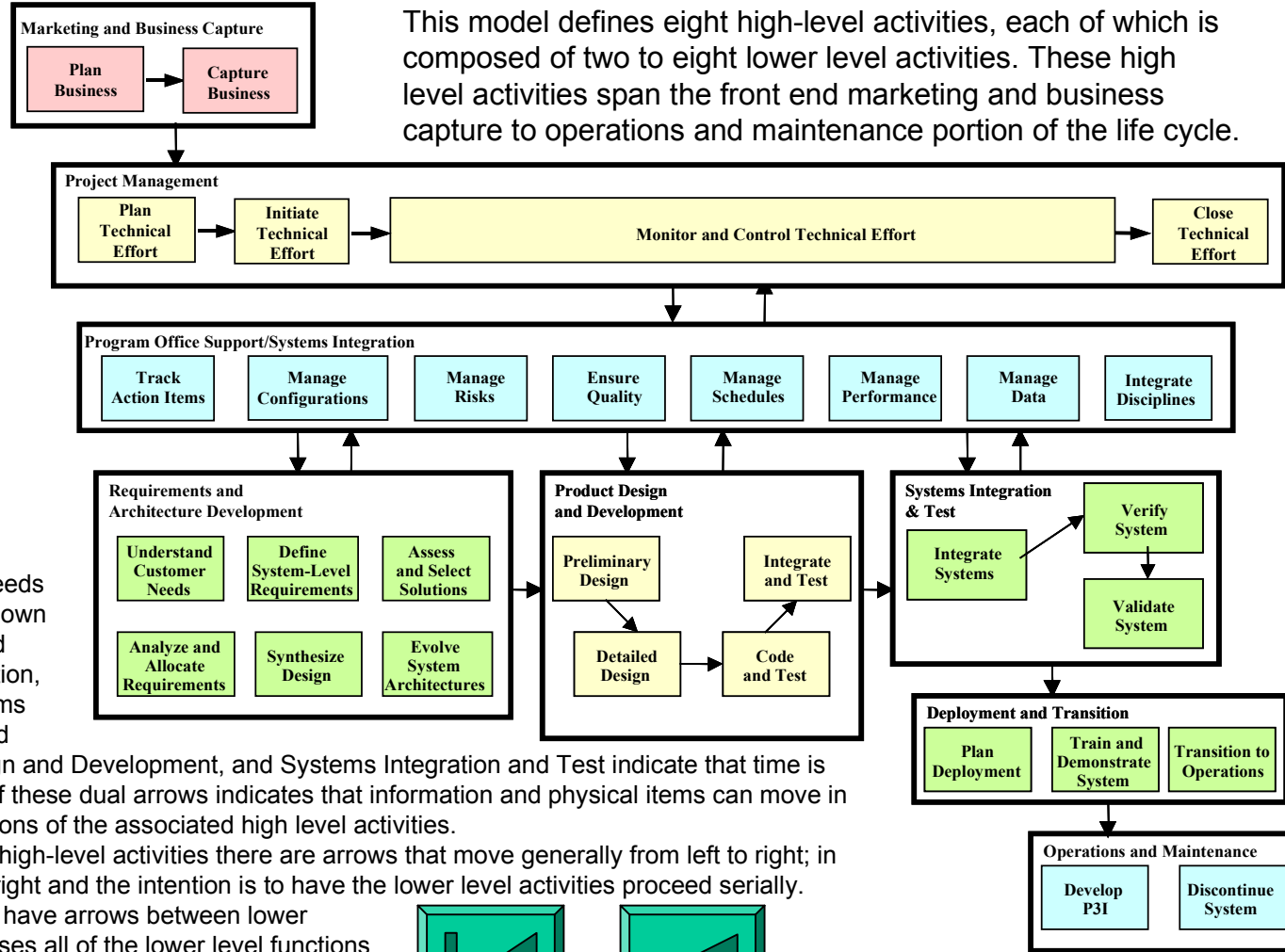
# Tufts' Systems Engineering Process Model

The processes selected for this model are those associated with EIA632 and the Integrated Capability Maturity Model (CMMI). One purpose for selecting these systems engineering processes is to use those most recognized as the core processes for most systems engineering projects. A second purpose is to identify other systems engineering processes used on many projects, but are not recognized as common enough to be included in formal standards.

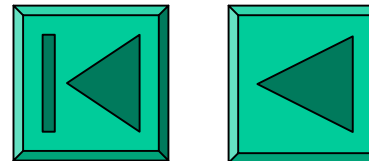
Time generally starts at the top and proceeds to the bottom. However the dual up and down arrows between Project Management and Program Office Support/Systems Integration, as well as Program Office Support/Systems Integration and each of Requirements and Architecture Development, Product Design and Development, and Systems Integration and Test indicate that time is not strictly moving down. The presence of these dual arrows indicates that information and physical items can move in both directions between lower level functions of the associated high level activities.

Within some of the boxes associated the high-level activities there are arrows that move generally from left to right; in these cases time does move from left to right and the intention is to have the lower level activities proceed serially.

The remaining high-level activities do not have arrows between lower level functions inside the box. In these cases all of the lower level functions can proceed concurrently due to the complex and iterative nature of the systems engineering process.



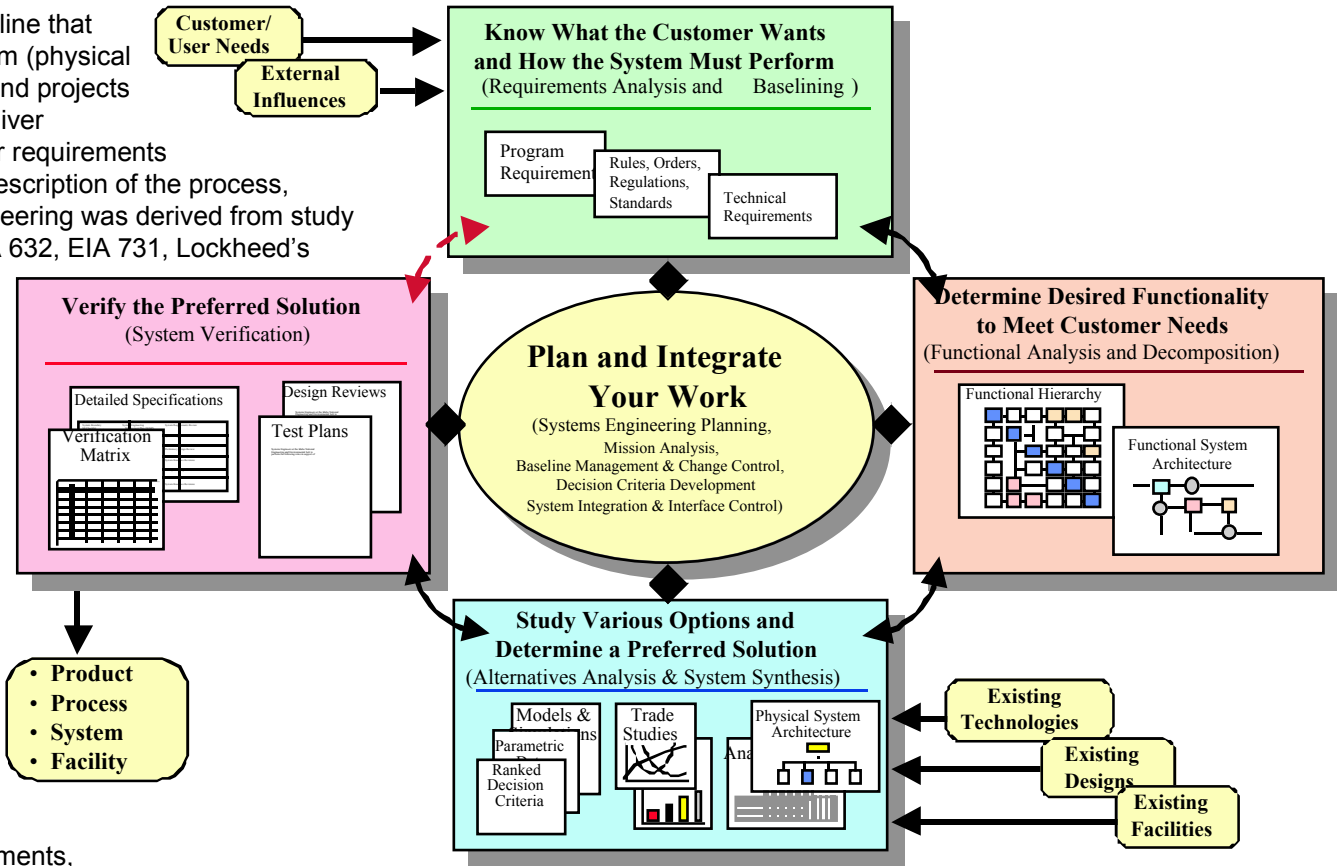
This model defines eight high-level activities, each of which is composed of two to eight lower level activities. These high level activities span the front end marketing and business capture to operations and maintenance portion of the life cycle.



**Key References**

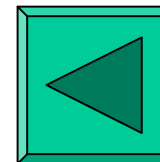
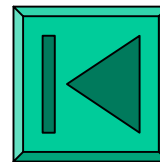
# Plowman's Model of the Systems Engineering Process

System Engineering is a structured discipline that considers all aspects of a particular system (physical and organizational) in helping programs and projects conceive, develop, integrate, test, and deliver products and services that meet customer requirements within cost and schedule. The following description of the process, functions, and products of Systems Engineering was derived from study of: MIL-STD 499B, ISO 15288 (draft), EIA 632, EIA 731, Lockheed's Code 66, Lockheed-Martin's EPIC SE Process Description, INCOSE SE Handbook (draft), TRW's SE Handbook, Blanchard's *Systems Engineering Management* text, and other related text books. These references all point to the following high-level functions in common:



Systems Engineering supports programs and projects by performing the following activities):

- defining and managing system requirements,
- identifying and minimizing risk,
- integrating system components (physical and organizational),
- managing system complexity,
- enhancing communication and system understanding,
- conducting trade studies as a basis for informed decision making,
- verifying that products and services meet customer needs.



**More Detail**

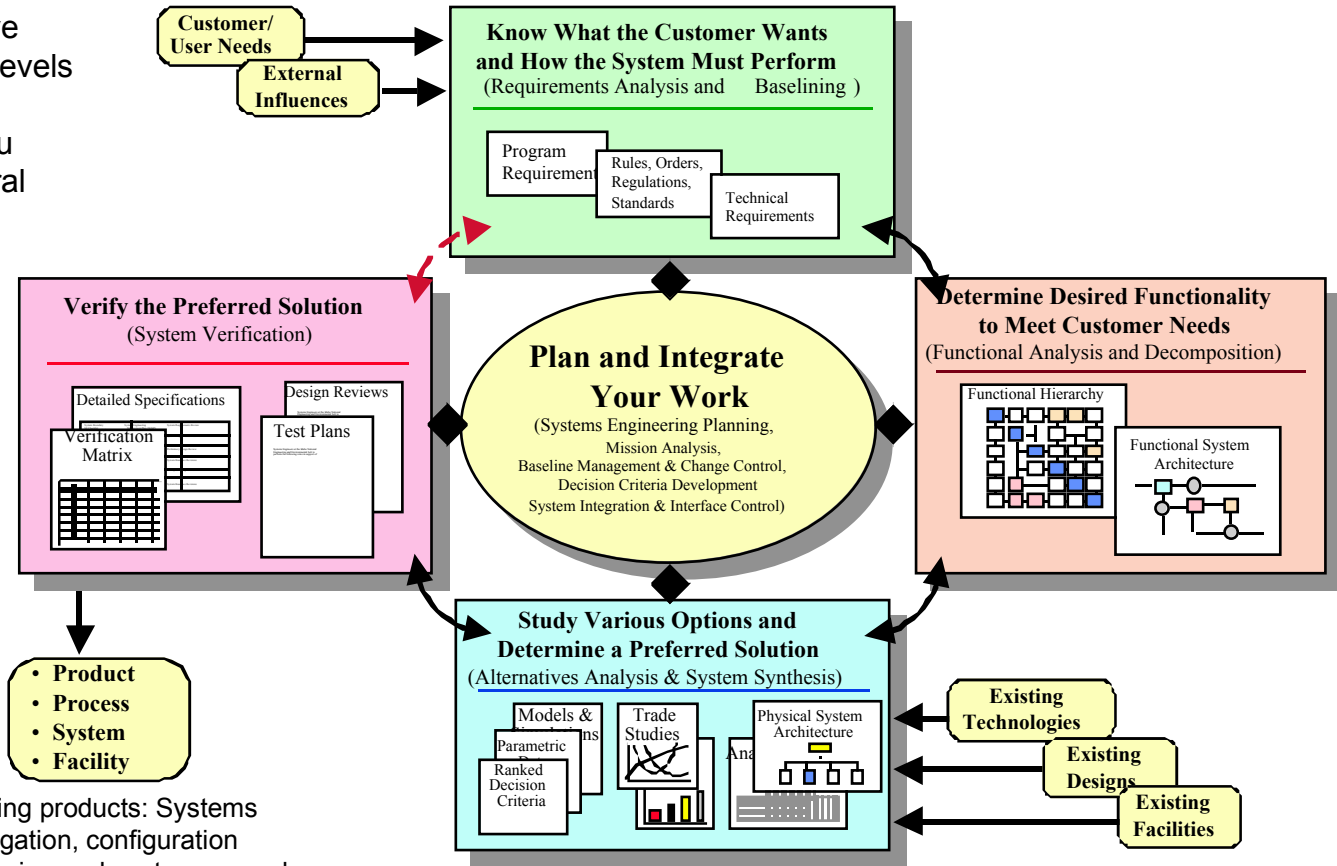
**Key References**

# Plowman's Model of the Systems Engineering Process (2)

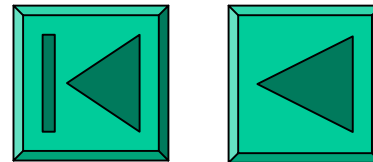
The result is a cyclic and recursive model that is applied at different levels of rigor depending on the type of program or project involved. If you “unroll” the SE Process, the central “Plan and Integrate Your Work” element breaks into five distinct activities that bridge the efforts from the four main blocks.

These five planning and integrating efforts are different from the four main functions in that they are typically done by systems engineering practitioners acting on the team-based input from the previous step. While many cycles of the SE process typically occur during the course of a project, each cycle covers the same basic steps only with different levels of emphasis and rigor.

This process typically creates the following products: Systems Engineering Management Plan, risk mitigation, configuration management, and verification plans; mission and customer needs statements; system requirement, description, and specification documents; Technical Performance Metrics and progress towards them; Functional Flow Block Diagrams, N2 charts, functional hierarchies, system physical architecture(s); system alternatives, trade study results, and preferred solution(s); cost, schedule, and technical baselines; interface/integration control agreements and Interface Control Documents; and verification matrices/verification discrepancies.



- Product
- Process
- System
- Facility

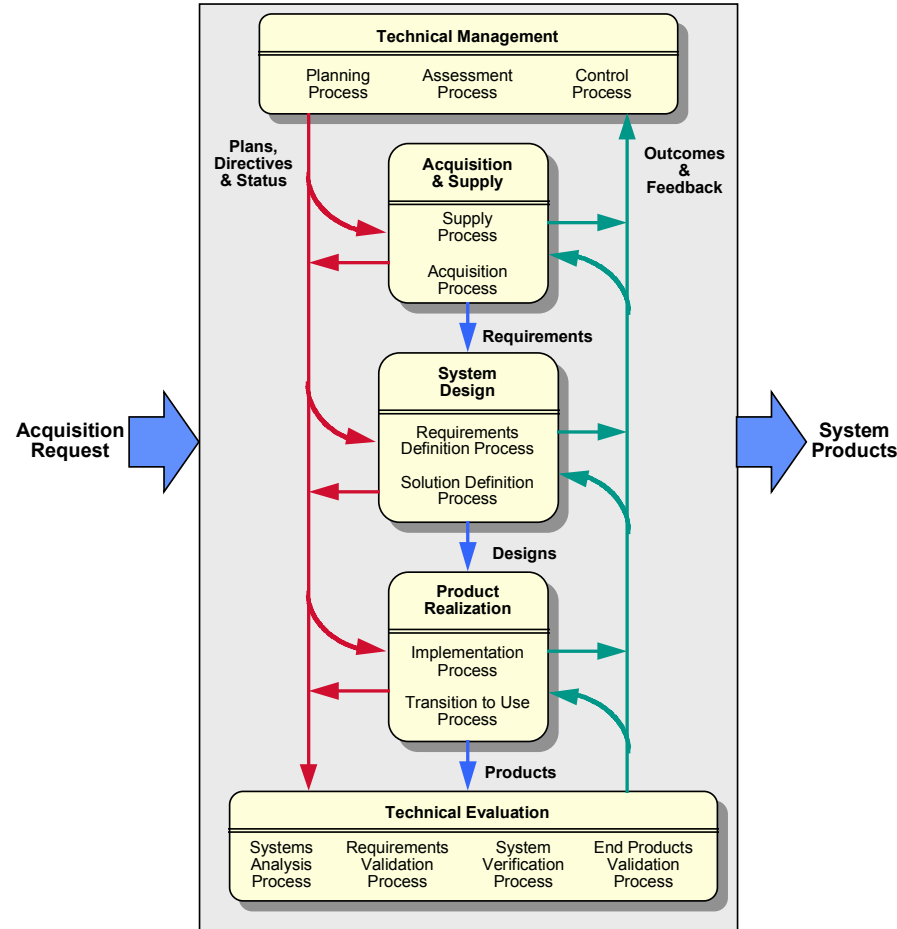


**Key References**

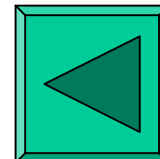
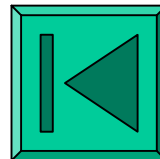
# INCOSE Handbook SE Process Model

What is known as the *systems engineering process* is basically an iterative process of technical management, acquisition and supply, system design, product realization, and technical evaluation at each level of the system, beginning at the top (the system level) and propagating those processes through a series of steps which eventually lead to a preferred system solution. At each successive level there are supporting, lower-level design iterations which are necessary to gain confidence for the decisions taken.

During each iteration, many concept alternatives are postulated, analyzed, and evaluated in trade-off studies. There are many cross-coupling factors, where decisions on one subsystem effect other subsystems. These factors must also be evaluated. An overview of the steps in the systems engineering process is shown to the right. Systems engineering is involved in all steps and *leads* during System Design down into the subsystem level, and *integrates* many other activities including design, design changes and upgrades, customer feedback, and operational support.



**Key References**



**More Detail**

# INCOSE Handbook

## SE Process Model (2)

The basic engine for systems engineering is an iterative process that expands on the common sense strategy of (1) understanding a problem before you attempt to solve it, (2) examining alternative solutions (do not jump to a point design), and (3) verify that the selected solution is correct before continuing the definition activities or proceeding to the next problem.

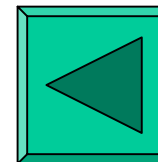
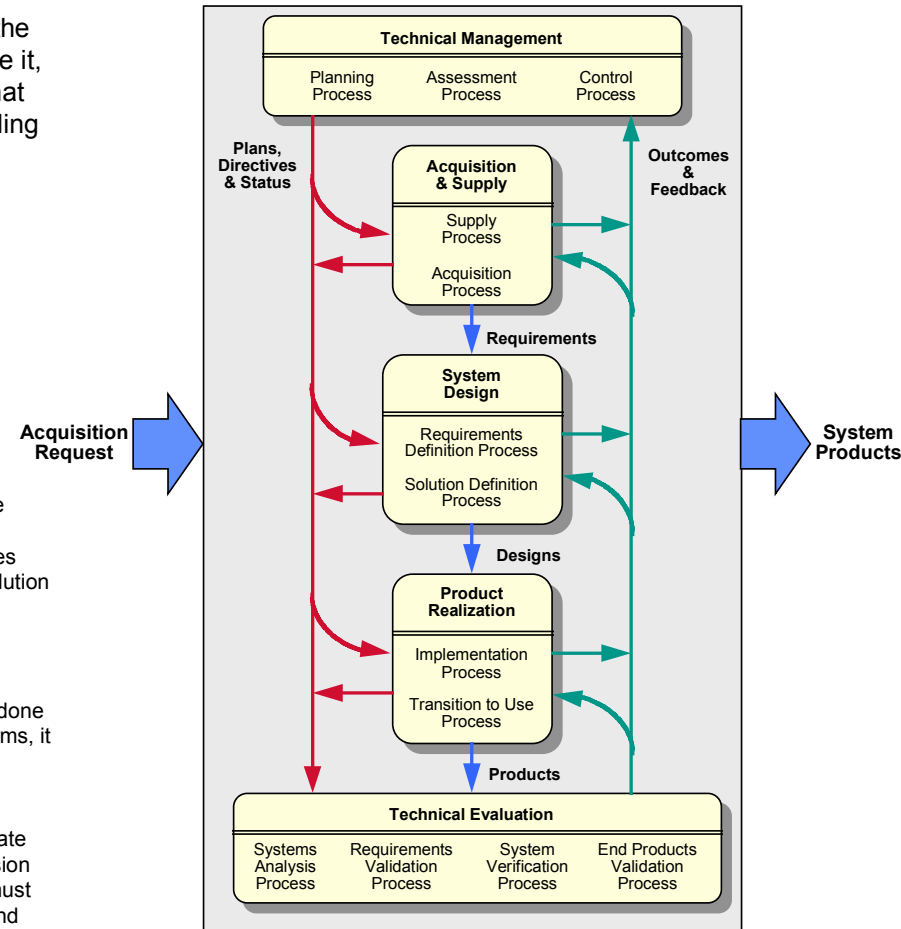
The basic steps in the systems engineering process are:

- (1) Define the System Objectives (User's Needs)
- (2) Establish the Functionality (Functional Analysis)
- (3) Establish Performance Requirements (Requirements Analysis)
- (4) Evolve Design and Operations Concepts (Architecture Synthesis)
- (5) Select a Baseline (Through Cost/Benefit Trades)
- (6) Verify the Baseline Meets Requirements (User's Needs)
- (7) Iterate the Process Through Lower Level Trades (Decomposition)

This process shown to the right is for the system level. The basic steps listed above are focused in the System Design process block. It is iterated at lower levels to produce hardware, software, and service implementation requirements. The process involves a Requirements Definition Process that establishes both performance (quantified) requirements and functional area (architecture) requirements, and a Solution Definition Process that translates those requirements into design and operations concepts solutions. Overarching Technical Management Processes and Technical Evaluation Processes are performed continuously throughout the development processes.

A clear understanding of the difference between defining what must be done and how well it must be done is mandatory for effective systems engineering. Unless requirements are expressed in measurable terms, it is difficult to determine when a job is done or when a product is acceptable. Also, a requirement is not effective unless it can be verified.

The systems engineering process is used iteratively at each phase of the development cycle to generate more detailed descriptions of the system under development. These descriptions constitute the "Decision Database". The database includes what needs to be achieved (functional requirements), how well it must be achieved (performance requirements), how it is to be achieved (design and operation concepts), and analysis or test results of the latter's capability to actually satisfy requirements (verification). The hardware engineers traditionally developed physical views of the systems, while software engineers traditionally provided functional views of their code. Systems engineers must be able to assist hardware engineers to appreciate the power of functional views, and assist software engineers to link physical descriptions to their functional processing. The keys to effective systems engineering are to effectively communicate a "shared vision" of the systems being developed and avoidance of omissions or confusion that often result from a lack of integration.



**Key References**

# **Key References:**

## **SE Processes**

# Competency Development of SE Practitioners

## SE Competencies

To be defined as a competency, a set of unique tools, techniques and methodologies must enable the practitioner to be effective in a wide range of system development and system management roles. They must make a significant contribution to a perceived customer benefit. Finally, although they may not be unique to SE, an SE practitioner must be able to perform them better than those from other disciplines.

The set of SE competencies include the following sets of tools, techniques and methodologies:

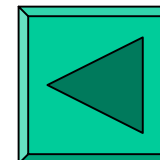
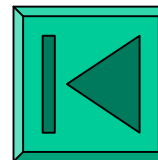
1. **Requirements Analysis and Management**
2. **Decision Analysis**
3. **Systems Engineering**
4. **Risk Management**
5. **Modeling**
6. **Human Factors**
7. **Engineering Management**
8. **Discipline Oriented Skills**
9. **Service**

## Purpose

Competency development in a System Engineer (SE) serves two purposes. First, it ensures that an individual will reach their full potential as an SE. This competency enhances the value of the individual, so they can excel in the practice of developing and managing systems. Second, an individual's competency enhances the value the discipline of SE, which is then seen as a value by product development teams, by the organizations who sponsor those teams, and by the customer to whom the teams are ultimately responsible.

## SE Competency Levels

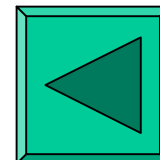
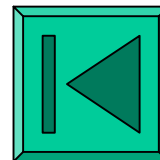
## Key References



# SE Competency Areas

1. **Requirements Analysis and Management** – How to generate, deploy and manage mission, originating, system, and subsystem level requirements.
2. **Decision Analysis** – Simulation, optimization and decision-making techniques to support the evaluation and selection of alternative system architectures.
3. **Systems Engineering** – Understanding and application of basic phases of system development and how those phases interface with the project management process and the system life-cycle.
4. **Risk Management** – Methodologies to identify, assess, and mitigate system and process risks.
5. **Modeling** – Tools that support the development of the functional, operational, physical, and interface architectures of systems.
6. **Human Factors** – Understanding requirements for human-system and human-human interfaces.
7. **Engineering Management** – Understanding the technical, administrative, budget and scheduling aspects of effective project management in conjunction with the development of the people aspect of system development.
8. **Discipline Oriented Skills** – Development of domain specific skill sets in related technical and managerial disciplines.
9. **Service** – Skills to provide mentoring, research, teaching, and service to the discipline.

[1](#) List was developed from “Training Technology’s Maestros,” by Beth Panitz, in the November 1997 issue of ASEE Prism, pp. 18-24.



# SE Competency Levels

**Knowledge:** Knowledge consists of the facts, conventions, definitions, methodologies, procedures and technical terms that define the field of systems engineering. An SE must be fluent in systems engineering knowledge, although knowledge by itself is not sufficient for problem solving. An example of SE knowledge is being able to list the stages of a system's life cycle.

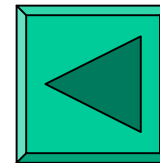
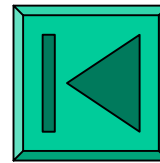
**Comprehension:** Comprehension is the ability to understand the meaning of knowledge. An SE must be able to take knowledge, understand it, and communicate it to others. An example of SE comprehension is the ability to compare and contrast the product development processes between to different organizations.

**Application:** Application is the ability to apply knowledge and comprehension to solve specific problems. An SE must be able to solve real world engineering problems. An example of systems engineering knowledge is determining schedule, cost and risk variances of an actual project's performance versus planned performance.

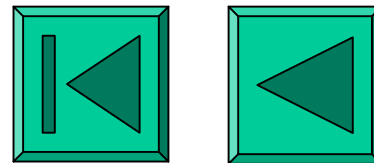
**Analysis:** Analysis is the ability to decompose a complex problem into its component parts, and then solve each of the parts. A key SE skill is the ability to support the analysis of specific subsystems, components and parts. An example of a SE analysis would be to evaluate the energy usage for the components of a specific subsystem to determine if the overall subsystem meets total power consumption requirements.

**Synthesis:** Synthesis is the ability to take individual pieces and combine them into a new solution. This is a key component of engineering in general and SE specifically. An example of SE synthesis is the ability to conduct system-level trade studies to evaluate the insertion of alternative components into an existing system architecture.

**Evaluation:** The highest level of SE cognitive development is the ability to evaluate and provide a judgment about a proposed solution, process, design, or analysis. The evaluation is comprehensive. At a minimum it looks at the logic, accuracy, presentation, documentation and logic of the proposal. An example of an SE evaluation is to serve as a member of an independent review team assessing a proposed system design.



# Key SE Competency References



# SE Process Capability Assessment

## Scope of capability

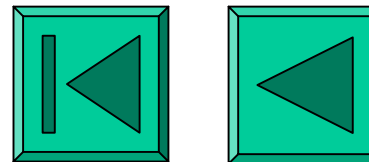
The capability of an organization to do systems engineering is a resultant of three capability vectors: 1) the processes used by the organization; 2) the defined context in which the processes are applied; and 3) the workforce that applies the processes within the defined context.

## Notions of Capability Assessment

Capability Assessment deals with the determination of the capability of a particular organization to do Systems Engineering. The Assessment process transforms a set of requirements into a quotation, which aims to indicate the level of Capability of an organization. Since the Assessment provides a clear snapshot of the activities of an organization, it is a valuable tool to understand organizational behavior and correct it. As these models are essential measuring tools when evaluating the capability of a given organization, it usually does not fit the needs of procurement agencies or services, since the result of a few past projects is typically difficult to extrapolate to new ones.

## Notions of Capability Improvement

The sponsor of the assessment is given a profile of the assessed organization, showing its major strengths and weaknesses. From this material, the organization can start thinking about addressing the flaws that prevent it from reaching some of their objectives. Some models embed an improvement path that can be directly used to determine what actions are to be performed to gain a better capability. Those models are said to be “Staged”, since they define a number of classes of organizations, that the improving organization must cross one after another.



# Scope of Capability

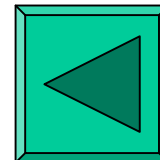
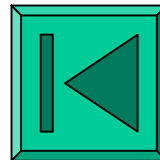
**Processes:** The processes are necessary in that they define what has to be done to convert customer requirements into system products that satisfy the customer and the other interested parties. These processes are selected as applicable to the organization's business. By themselves processes are not sufficient, however. Processes are applied within a given organizational context to include the concepts of systems engineering adopted by the organization and the common methods and tools adopted to perform process activities and tasks.

**Context:** A uniform understanding of basic systems engineering concepts facilitate enhanced organization-wide communications through use of a set of common terminology. Important concepts include:

- How the organization looks at the system to be engineered.
- Also the products that enable the operational products to perform within the various stages of the system life cycle.
- Life cycle service functions -- enabling systems.
- How the organization looks at the role of systems engineering within the life cycle model used by the organization to guide a system through its service life.

Thus systems engineering capability includes not only engineering the right system but also creating the information required to make appropriate life cycle management decisions with respect to the system.

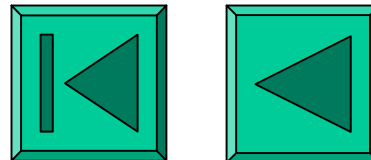
**Workforce:** Two attributes of the workforce help determine the systems engineering capability of an organization-Knowledge and skills. The workforce requires the domain knowledge essential to the system being engineered and also knowledge of the processes to be used for systems engineering, understanding of the concepts adopted by the organization, and knowledge of project management. The individuals within the workforce also need the special skills required to perform systems engineering processes; use the common methods and tools applicable to their assigned work; and work in an integrated, multi-disciplinary team environment.



# Context

**Context:** A uniform understanding of basic systems engineering concepts facilitate enhanced organization-wide communications through use of a set of common terminology. One important concept that needs to be defined is how the organization looks at the system to be engineered. One system definition has a focus on the products that will perform the operational functions. This system concept is called the system-of-interest in ISO/IEC 15288. Another system definition defines the system to consist of not only the products that perform the operational functions but also the products that enable the operational products to perform within the various stages of the system life cycle. This system concept is defined in ANSI/EIA 632 calls the products that perform operational functions end products and those that perform life cycle service functions as enabling products. ISO/IEC 15288 includes consideration of the life cycle service functions and calls the products as enabling systems. ISO/IEC 15288 has only a loose connection between systems-of-interest and enabling systems whereas ANSI/EIA 632 has a strong system project link between end products and enabling products. The organization's concept of the system will influence the scope of a project's concern with respect to engineering a solution to a customer's problem that provides life cycle satisfaction to the customer and also considers the interest of other parties involved in that life cycle.

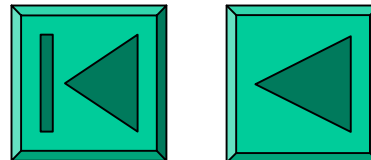
This leads to the second important concept that needs to be defined by an organization. This concept is how the organization looks at the role of systems engineering within the life cycle model used by the organization to guide a system through its service life. Such a model has early exploratory phases such as pre-concept definition and/or concept definition to study technologies and feasible concepts and later an execution phase in which the system is designed; pre-production models created and proofed; products produced, utilized and supported; and then retired. Typically organizations have gates within a life cycle model that are used to determine whether a system is continued to the next service life phase, terminated or retired, or redirected (through appropriate improvements). To provide management with the information to make decisions at each gate the system has to be appropriately engineered to create the needed information to meet decision gate criteria.



# Notions of Capability Assessment

Unlike audits where an auditor seeks what is missing and tracks what is incorrect, the assessment is a constructive approach that aims to understand why the organization is at a given level and how it can improve its Capability. Assessments are mostly based on cooperative interviews and work product shared examination. Starting with very open questions, the assessment team gathers from the answers and from the discussions that the engineers may start together, the evidences that activities are performed. The different existing processes to assess organizations can be tailored to get a proper process that will be used to create the organization's assessment process.

Although most of the processes are closely related to a particular model, some defined assessment processes may be used with different models.

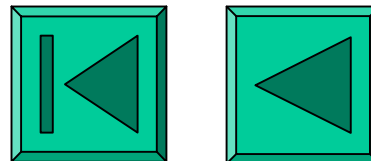


# Notions of Capability Improvement

Some models do not provide this improvement path, which then must be built by the user of the model, based on the different requirements expressed by the sponsor. These models are called “continuous”, since there is a path to follow step by step. As those two approaches are complementary and often used together, some models have been designed to fit both. Those models provide two views of the same information: one as a profile for each elementary process of the organization, and one as a measurement on a defined improvement path. The architecture of the improvement path, regardless of the model to be used, usually follows simple rules: technical matters must be done first, then management and support at a project level, then management and support at an institution level.

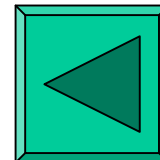
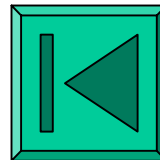
The greatest challenge in Capability improvement is usually not in the techniques of capability evaluations and improvement definition. The most problematic achievement is to conduct organizational changes where all the people are modifying their uses. This goal implies often widely spread training efforts, acquisition of new tools, introduction of new procedures, collection of a series of indicators, etc. Success in such a project is expensive and runs, by definition, over several years. The price for improvement is high, but the returns on investment have been found, across several enterprises and institutions, highly attractive. Many resulting organizations have been able to drastically reduce their inefficient use of money and their sensitivity to major risks, while becoming more cost effective.

The major pitfall to overcome is the strong feeling that improvement is a non-regressive act. The Capability of an organization can easily sink if no attention is paid to sustain the effort and the willingness for improvement.



# Key References for Capability

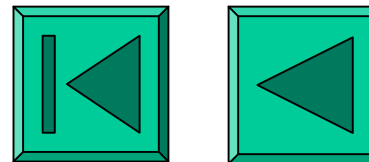
- SE-CMM (Software Engineering Institute)
  - SECAM (International Council on Systems Engineering)
  - EIA-731 (Electronic Industries Alliance)
  - CMMI (National Defense Industries Association)
- ISO/IEC 15504 (International Organization of Standards)
- CBI-IPI
  - EIA 731-2
- ISO 15504 – Part 3 & 4



# What is/isn't the G2SEBoK Guide?

The SEBoK Guide is INCOSE's attempt to provide a service for its members as well as service to non-members. You will find this guide helpful in understanding not only systems engineering, but also the roles that INCOSE and its members play in promoting the discipline of systems engineering.

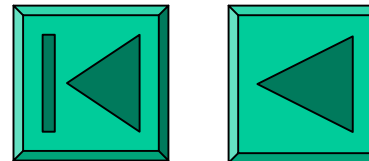
We have organized the guide to allow both discrete fact-finding on an aspect of systems engineering and broad inquiry. This guide is a holistic approach to the study and application of systems engineering and is not a "cookbook" that prescribes how systems engineering must be accomplished.



# What is unique about the G2SEBoK?

This guide seeks to define systems engineering broadly, in terms of its concepts, processes, skills required, and capability. In addition, this guide provides substantial linkage to related works. Through the use of this guide you can:

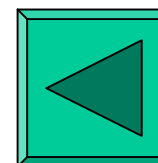
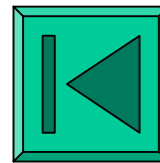
- Gain an understanding of the overall and specific aspects of systems engineering
- Investigate the discipline from a functional, product or process perspective
- Comprehend the integrative nature of systems engineering through linkage to related disciplines and where applicable their bodies of knowledge
- Access available articles, papers, member authored texts and distance learning products
- Study existing standards and developing standards
- Interact with authors, members and leadership through discussion groups and forums



# When is the G2SEBoK useful?

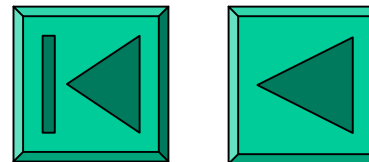
This is highly dependent on the user impetus. The following categories of users/uses are provided for illustration.

Possible User Types	Possible Uses
<b>Novice</b>	Explore the discipline Determine applicability
<b>Licensing or Certification Services Acquirer</b>	Determination of professional standards Source selection Evaluation of sources and techniques Services evaluation
<b>Practitioner</b>	Expansion of knowledge Specific problem solving Identification and evaluation of sources and techniques
<b>Academician</b>	Referencing Curriculum/course development Expansion of knowledge
<b>Learner</b>	Expansion of knowledge Accomplishment of corporate or certification requirements Verification of derived knowledge



# Doesn't this knowledge already exist?

The supporting body of work referenced in the guide is referred to as the “systems engineering body of knowledge”. The processes, lessons learned, products and standards have been developed since the 1930s. The guide does not create extensive new work, however it does provide organization and context for the information.



# Communities of Practice



G2SEBoK Navigation Page

# Reference Works



G2SEBoK Navigation Page

# INCOSE Membership



G2SEBoK Navigation Page

# Related Resources



G2SEBoK Navigation Page