

Appendix A

A Process Review of the Accepted Standards and Best Practices for Developing Systems Engineering Process Models

Systems Engineering Process Review Report

By

Global Intergy Corporation

For

PBS&J

Project Number: 091003.11

Short Title of Main Contract: ITS General Consultant

Short Title of Addendum: Systems Engineering Process Review

Subcontract Addendum No. 3

July 2002

Document Control Panel	
File Name:	W:\ITS Program\ITS GC\TWO11-SEMP\021016 SEMP Apx A (Global).doc
Created By:	Global Intergy Corp.
Date Created:	July 2002
Version Number:	2
Reviewed By:	Traci Matthews, Pamela Hoke
Modified By:	Pamela Hoke
Date Modified:	October 16, 2002

Table of Contents

List of Tables	4
List of Figures	4
List of Acronyms	5
1. Scope of Project	6
2. Introduction	6
3. Systems Engineering Evolution	7
4. Systems Engineering Development Processes and Methods....	10
5. Systems Engineering Process Models	11
5.1 Waterfall Models	11
5.1.1 Single-Pass Waterfall Model	12
5.1.2 Incremental Model (Iterative Waterfall Model).....	12
5.2 The Spiral Model	14
5.3 The Vee Model.....	16
5.4 The EIA-632 and IEEE 1220 Models.....	17
5.5 Summary and Evaluation of Process Models	20
6. Conclusions and Recommendations	24
References	21

List of Tables

Table 1 – Evaluation Summary of Systems Engineering Process Models17

List of Figures

Figure 1 – Evolution of the Systems Engineering Standards3
Figure 2 – The Frameworks Quagmire4
Figure 3 – The Waterfall Process Model6
Figure 4 – The Incremental Process Model8
Figure 5 – The Spiral Process Model10
Figure 6 – The Vee Process Model12
Figure 7 – The EIA-632 Process Model13
Figure 8 – The IEEE-1220 Process Model14
Figure 9 – ITS Deployment Life Cycle20

List of Acronyms

DoD.....	Department of Defense
EIA.....	Electronic Industries Alliance
FDOT.....	Florida Department of Transportation
GIC.....	Global Intergy Corporation
IEEE.....	Institute of Electrical and Electronics Engineers
INCOSE.....	International Council on Systems Engineering
IS.....	Interim Standard
ITS.....	Intelligent Transportation Systems
MIL-STD.....	Military Standard
<i>SEMP</i>	<i>Systems Engineering Management Plan</i>
Std.....	Standard

1. Scope of Project

Global Intergy Corporation (GIC) has been tasked with documenting generalized systems engineering models based on the training GIC provided to the Florida Department of Transportation (FDOT) Intelligent Transportation Systems (ITS) Office personnel and its consultants in October 2001.¹ The following is a summary report of accepted standards and best practices, with recommendations for inclusion in FDOT's *Systems Engineering Management Plan (SEMP)*.

For the purpose of evaluating and comparing different systems engineering models, GIC has identified, acquired, and reviewed various systems engineering standards and guidelines currently in use in government and industry in the United States. This report presents and discusses several systems engineering models contained in the standards and guidelines and provides a listing of the advantages and limitations of each model.

2. Introduction

Various systems engineering models have evolved over the years. This report discusses a number of these systems engineering process models and the standards related to them. The proliferation of different standards and guidelines to accommodate software development and certification processes has created much misunderstanding in terminologies and the applicability of systems engineering standards. As a result, it is necessary to define the following terms so as to avoid confusion while reviewing different systems engineering standards and guidelines.

System Process Model (Life Cycle Process) – This process model is a description of an enterprise's activities as they are related to the total engineering effort to achieve a given outcome (i.e., a product or service). The process model illustrates the sequence and/or interaction among various project activities from creation to disposal of the product/service. The evaluation of various system process models is the focus of this report and it includes discussion of process models such as the Waterfall, Incremental, Spiral, and Vee.

Systems Engineering Standards and Guidelines – The standards and guidelines define the interdisciplinary tasks that are required throughout a system's life cycle to transform customer needs and requirements into a system solution. They usually specify the requirements for the systems engineering process and its application throughout the product life cycle and typically do not attempt to define the implementation of each system life cycle process. The Institute of Electrical and Electronics Engineers' (IEEE) Standard (Std) 1220-1998² (IEEE 1220-1998) and the Electronic Industries Alliance (EIA) Standard 632 (EIA-632) are the most prominent systems engineering standards.

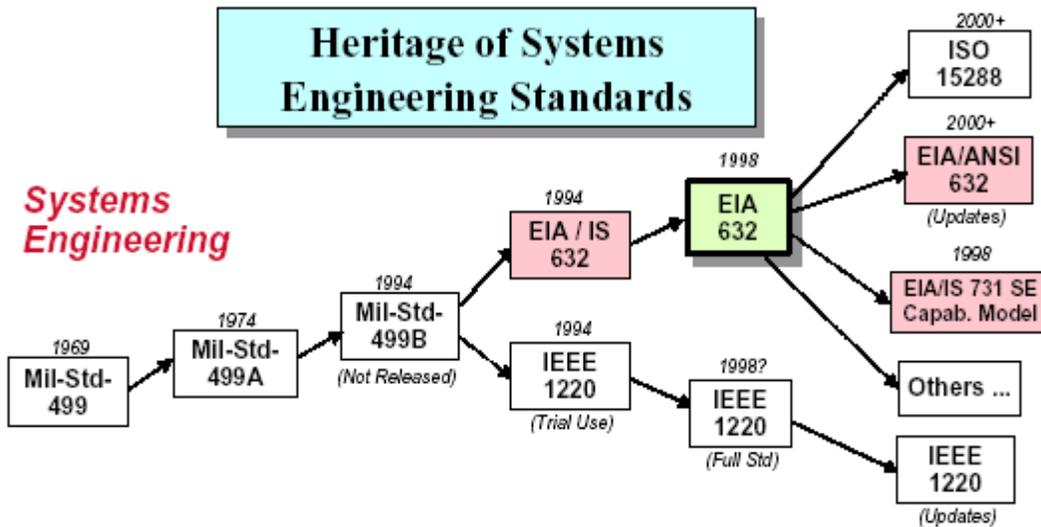
System Capability Maturity Model – Capability models define the characteristics of good processes and do not prescribe how the processes must be implemented. Capability models are not processes. The purpose of capability models is to establish a process improvement roadmap upon which a route can be drawn from “where we are today” to “where we want to be.” In order to determine “where we are today,” an organization performs an assessment, also called an appraisal, sometimes with the aid of an outsider with specific expertise in the model. They intentionally do not address a particular life cycle or sequence of activities. The first capability model, the Software-Capability Maturity Model, addressed software development, or more precisely, the management of software development projects. Later models addressed systems engineering, integrated product development, and other aspects ranging from human resources to security. For example, the EIA-731³, Systems Engineering Capability Model (SECM), has been accepted as the primary capability maturity model standard of the systems and software engineering communities. It should be emphasized that the SECM (EIA-731) is not a process standard but actually a standard for defining and assessing the maturity of the systems engineering discipline⁴.

3. Systems Engineering Evolution

The following is a summary of the evolution of various standards in the systems engineering discipline. A more complete, detailed history can be found in references [1] through [4]. Systems engineering standards have evolved from a federal government contract-centric approach to a commercial, voluntary-compliance approach. The focus has changed from management to process orientation.

Military Standard (MIL-STD) 499, Engineering Management (MIL-STD-499), dated July 17, 1969, and later the A-version dated May 1, 1974, was an early standard on the subject of systems engineering. It was produced by the United States Department of Defense (DoD) for application within the defense industry. MIL-STD-499B, dated in 1992, was distributed to reviewers and was an updated and significantly rewritten MIL-STD-499A. The DoD decreed the end of military standards other than performance specifications in June 1994 before the standard was officially released. This standard, however, has been used extensively by the Air Force. An industry working group was formed composed of representatives from the Aircraft Industry Association, DoD, the National Security Industries Association, the EIA, IEEE, and the International Council on Systems Engineering (INCOSE). This working group released a "commercialized" version of MIL-STD-499B in December 1994 known as EIA Interim Standard (IS) 632.⁵ This was done with the understanding that considerably more industry input would go into a replacement version, to be called EIA-632. In parallel, the IEEE also released in December 1998 a commercialized standard IEEE 1220-1998, *Standard for Application and Management of the Systems Engineering Process*. Figure 1 illustrates the evolution of various systems engineering standards and guidelines that dominate the systems engineering process models and capability models in practice today.

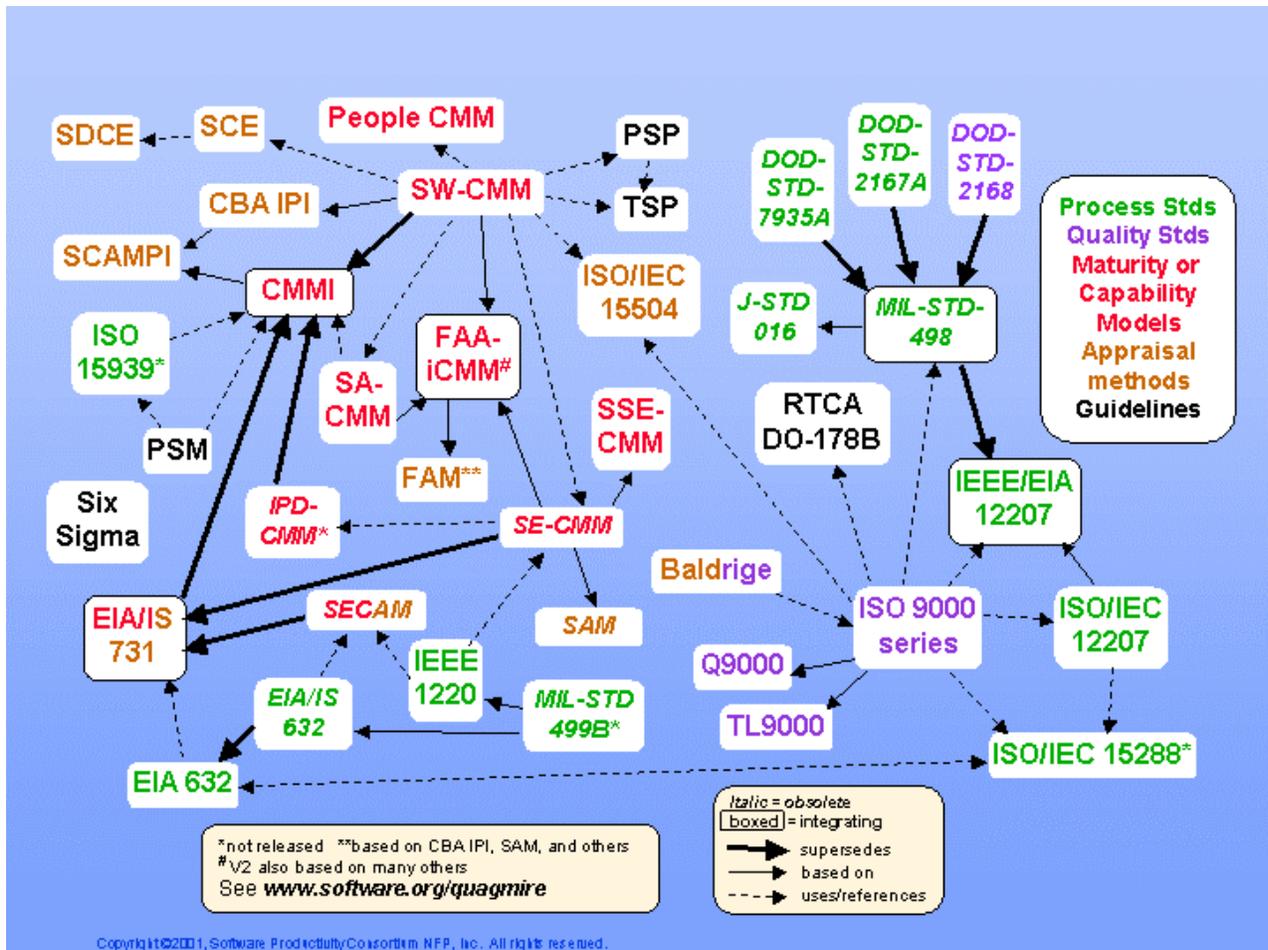
Figure 1 – Evolution of the Systems Engineering Standards^{6/7}



In 1992, INCOSE sponsored a working group that began to address the assessment of systems engineering capability. This group has evolved the Systems Engineering Capability Assessment Model (SECAM), which was released in July of 1996. Also, in December 1993, the Enterprise Process Improvement Collaboration (EPIC) group spun off from the INCOSE SECAM working group. This group developed a Systems Engineering Capability Maturity Model (SE-CMM) that was released in December 1994. This standard evolved from a software legacy.

The development of these groups meant there were now two (2) systems engineering models in the market. INCOSE and the Director for Systems Engineering in the Office of the Secretary for Defense agreed that the two (2) models had to come together. EPIC and INCOSE agreed to work towards a merged model, eventually called the Systems Engineering Capability Model (SECM), EIA/IS-731. The Frameworks Quagmire^{4/8}, shown in Figure 2, provides a detailed map of the standards evolution and adds to the potential confusion over current systems engineering standards and models.

Figure 2 – The Frameworks Quagmire^{4/8}



4. Systems Engineering Development Processes and Methods

An extensive body of experience collected from the industry indicates that nearly two-thirds of software development projects in the United States fail, due to either cancellation, overrunning of budgets, or delivery of software that is never put into production.⁹ The reasons for these failures in descending order are as follows:

1. Lack of user (i.e., customer and stakeholder) involvement;
2. No clear statement of requirements;
3. No project ownership;
4. No clear vision and objectives; and
5. Lack of planning.

A good systems engineering development model should impose discipline on developers to produce a consistent set of requirements, functional arrangements, and design solutions. The end goal is to improve productivity and at the same time provide deliverables that satisfy the product's end purpose. Almost every systems development effort varies in its specifics, so project managers and engineers need to have structured freedom to customize their work processes.

Processes – A process is a sequence of activities executed by a human or machine, often with the goal of transforming a set of inputs into outputs. A complete description of a process includes naming of the steps within the process and using models of the system in various textual/graphical abstractions. Systems engineering processes are concerned with the step-by-step development of complex engineering systems, from the identification of user needs through the specification of all components and subsystems to be designed.

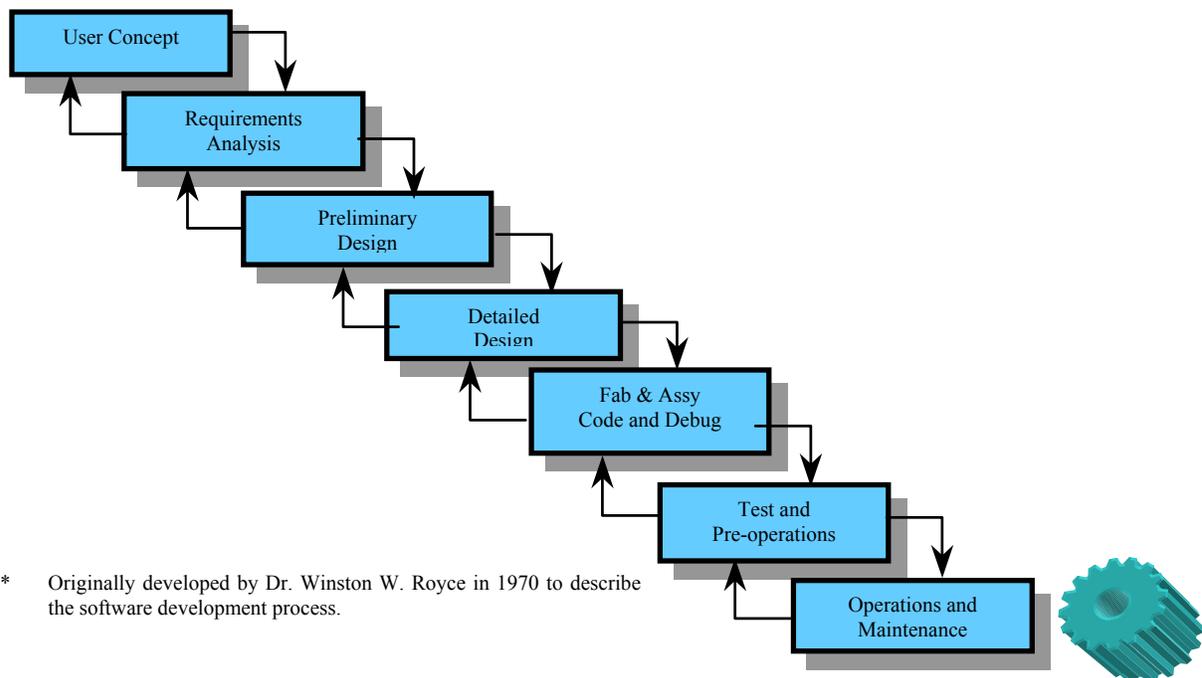
Methods – A methodology is simply the implementation of a specific process. Methodologies for systems engineering development should contain an underlying model. The underlying model refers to the ensemble of objects (i.e., data types or data structures) represented, manipulated, and analyzed by the method. Modeling is a key element of systems engineering that helps to close the gap between "what is needed" and "how the system will work."

5. Systems Engineering Process Models

5.1 Waterfall Models

Figure 3 illustrates a classical Waterfall Model. In a simplified view, system life cycle development begins with the gathering of requirements and domain knowledge and ends with system deployment, maintenance, and, eventually, retirement.

Figure 3 – The Waterfall Process Model



The Waterfall Model was accepted as the primary systems engineering process model through most of the 1980s. Each phase of sequential development is completed, via formal review, before the next phase begins. The Waterfall Model is most useful when the problem and solution method are well understood.

Variations on the Waterfall Model include those listed below.

5.1.1 Single-Pass Waterfall Model

The single-pass is the simplest view of the Waterfall Models where there is no iteration in the process activities. The three (3) main phases of development are analysis, design, and build (i.e., construction or implementation). The key components of each phase are:

- *Analysis Phase* – The analysis phase begins with the project's inception and continues through the requirements definition. The latter may include a user needs study (i.e., what does the customer really want?) and a feasibility study (i.e., from a technical standpoint, is the project feasible?).
- *Design Phase* – The design phase covers all aspects of system design, logical design, physical design, and so forth. Often, this phase is loosely defined, with design evolving as a series of progressive decompositions towards increased technical detail.
- *Build Phase* – Now, the system is actually built, integrated into neighboring systems, and tested for proper functionality. In other words, does the system do what it is meant to?

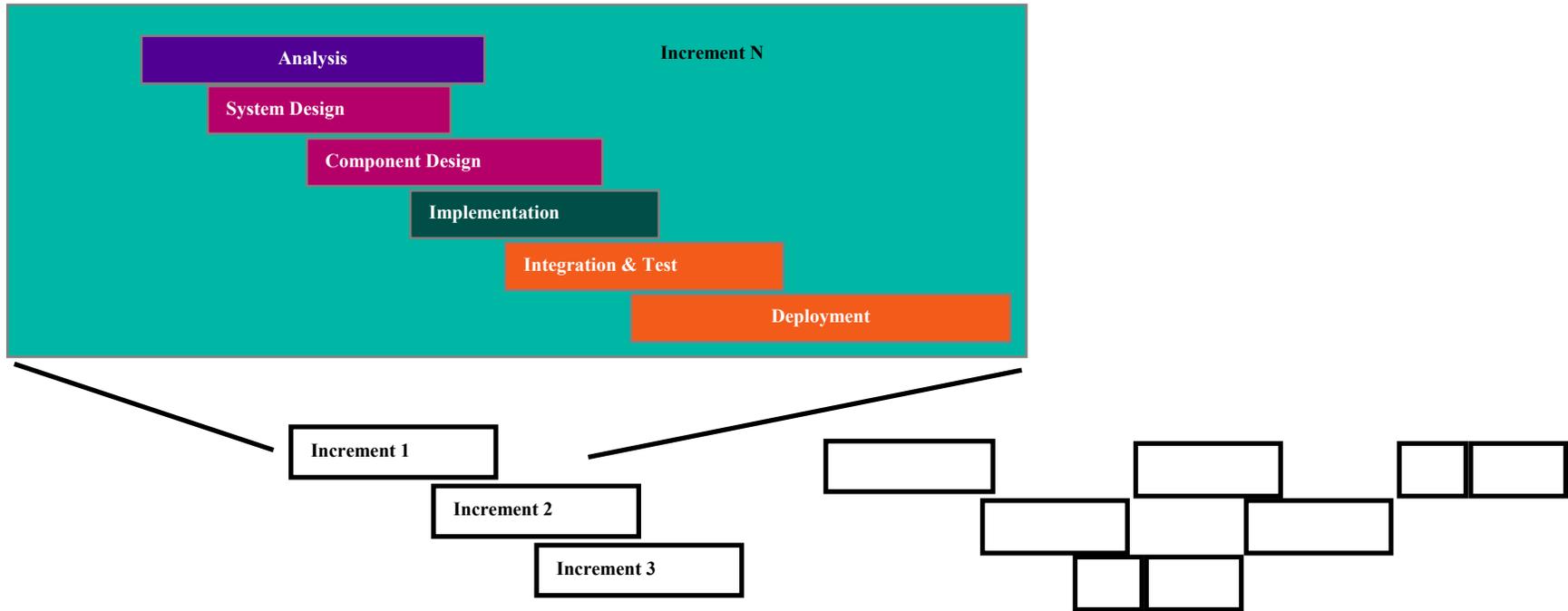
5.1.2 Incremental Model (Iterative Waterfall Model)

The Incremental Model, sometimes known as the Iterative Waterfall Model, is illustrated in Figure 4.

Incremental development is defined as the development of a system in a series of versions or increments. At each increment, a subset of functionality is selected, designed, developed, and implemented. Additional increments extend the system functionality.

Often, the rework of system functionality can proceed with a reasonable amount of certainty that the desired result will be achieved (i.e., enhancements to software functionality). In these cases, this process can be modeled with a series of Waterfall Models.

Figure 4 – The Incremental Process Model



5.2 The Spiral Model

The Spiral Model of systems development corresponds to a sequence of Waterfall Models and is displayed in Figure 5. This model corresponds to risk-oriented, iterative enhancement and it recognizes that implementation options are not always clear at the beginning of a project. An implementation option may be uncertain, for example, because it is critically dependent on a technology still under development.

The radial direction of Figure 5 corresponds to cumulative cost incurred and the angular direction corresponds to progress made in completing each cycle of the spiral. Each cycle of development has the following phases:

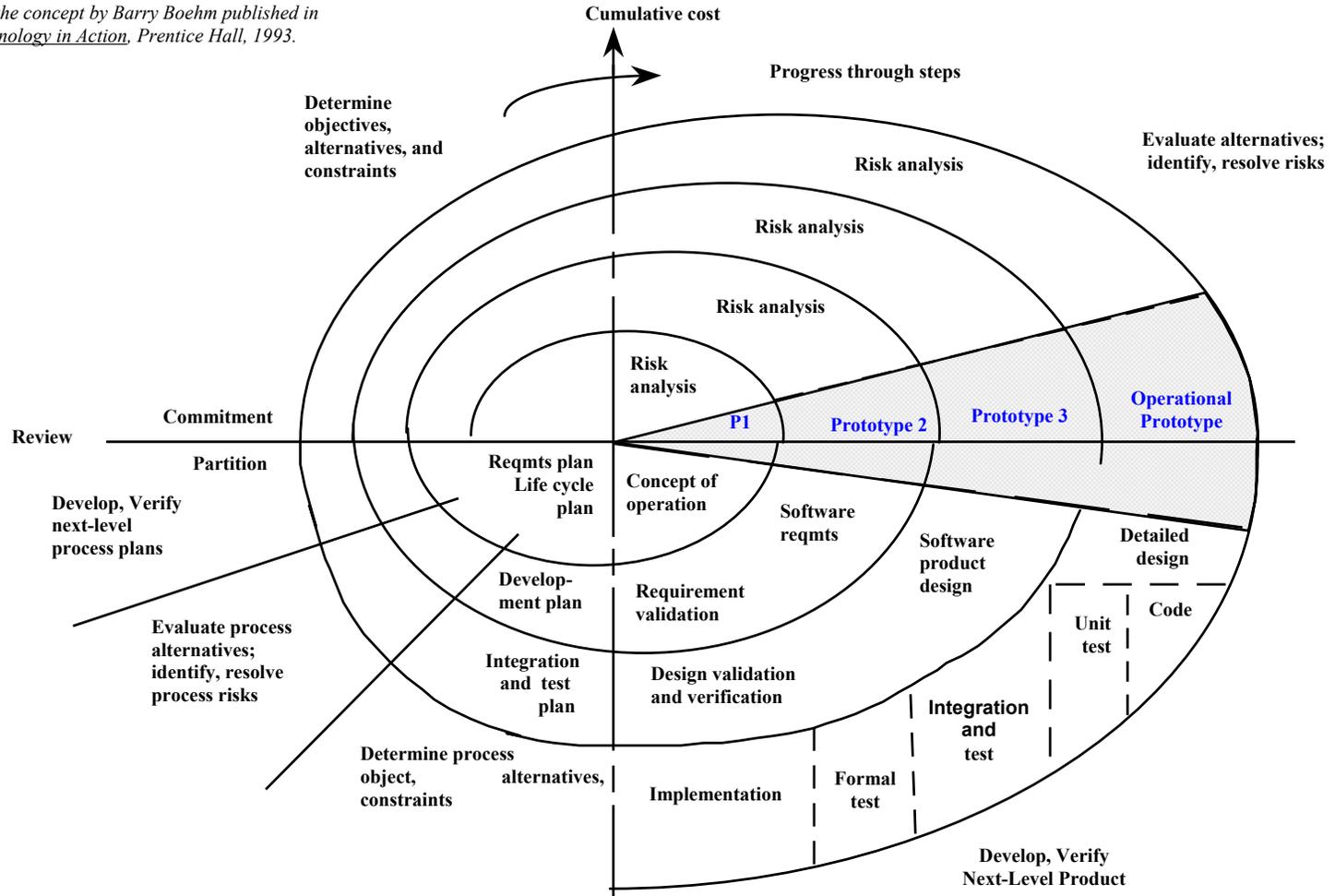
- Identify the design and development objectives for the cycle and the alternatives that are possible to achieve the goals;
- Evaluate different alternatives based on objectives and constraints and, where appropriate, identify uncertainties and risks;
- Develop strategies such as simulation, prototyping, and benchmarking for resolving uncertainties and risks; and
- Plan the next stage, allowing for any of the possible life cycle models to be used.

The initial Spiral Model release is a small subset of the anticipated system. Subsequent releases add capability to previous releases and each release is developed using the Waterfall Model. In some application domains, early releases have been called rapid prototypes.

A key characteristic of the Spiral Model is the assessment of management risks at regular stages in the project and the initiation of actions to counter these risks. Before each cycle, risk analysis is initiated, and at the end of each cycle, a review procedure assesses whether or not to proceed to the next loop in the spiral.

Figure 5 – The Spiral Model Process

Figure based on the concept by Barry Boehm published in *Information Technology in Action*, Prentice Hall, 1993.

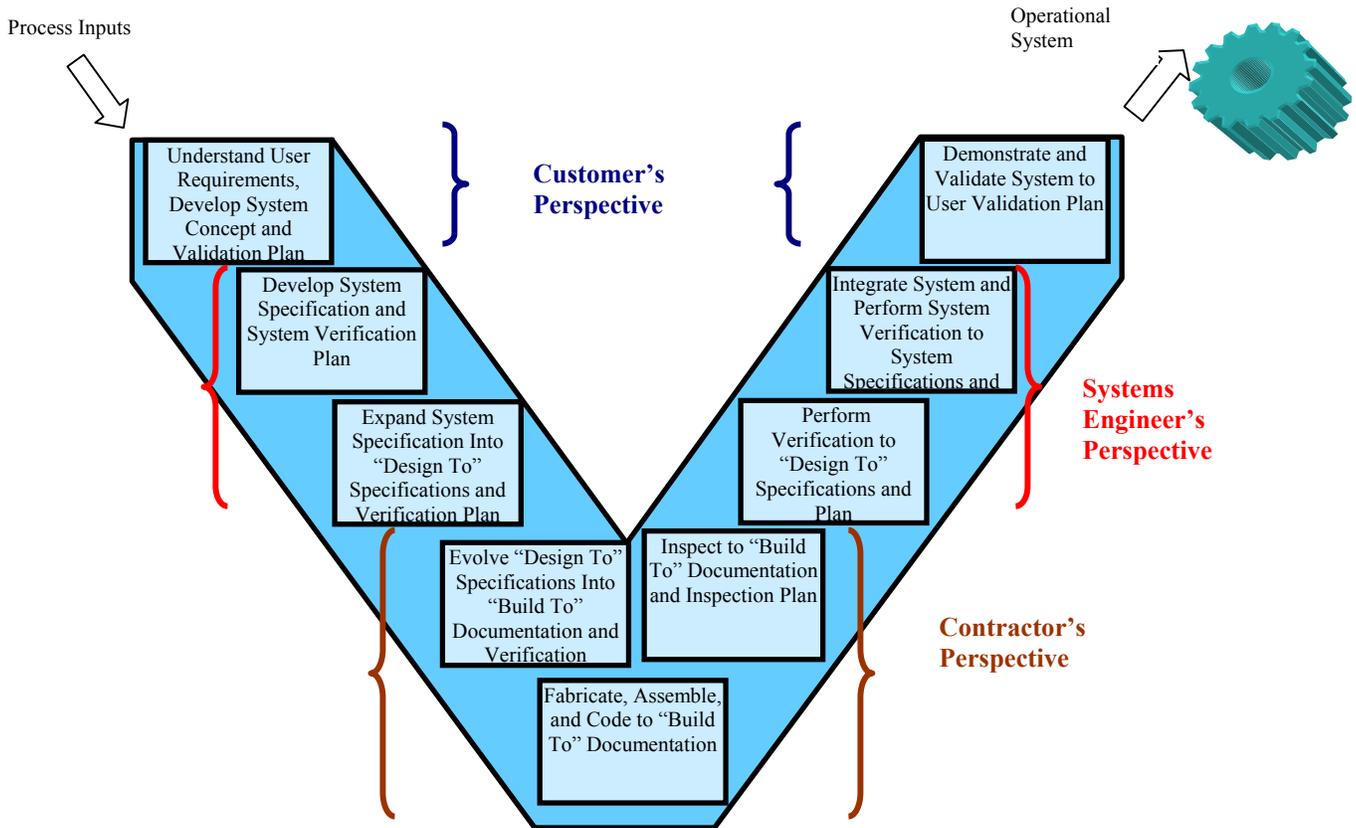


5.3 The Vee Model

The Vee Model depicts a “top-down” development and “bottom-up” implementation approach. On the left side of Figure 6, decomposition and definition descends as in a traditional Waterfall Model. On the right side of Figure 6, integration and verification ascends as successfully higher levels of units, assemblies, and subsystems are integrated and verified, culminating at the system level. The Vee Model is a composition of three (3) layers or perspectives of the system in increasing engineering detail:

- *User’s Perspective* – This is the view of the customer or stakeholder who is interested in presenting a list of requirements and receiving a finished product that meets the requirements.
- *Systems Engineer’s Perspective* – This perspective encompasses the architectural details that address the decomposition of the system-level specification into system design and the subsystems’ specifications and designs. It is paired with built and tested subsystems and, finally, the tested system.
- *Contractor’s Perspective* – This perspective covers the implementation process that is normally performed by contractors and/or subcontractors. In practice, the contractor’s perspective is associated with component specifications and designs with fully tested components.

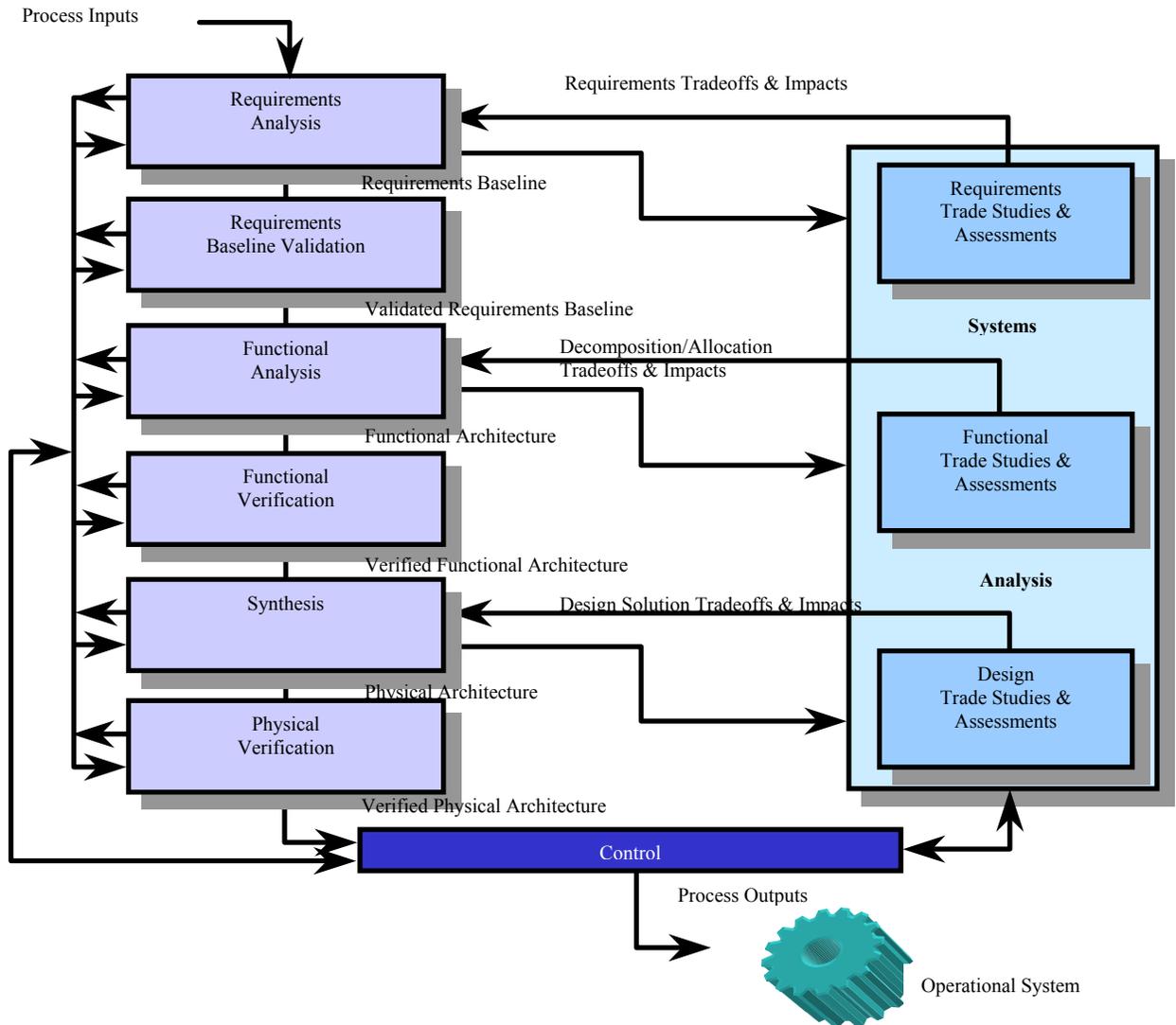
Figure 6 – The Vee Process Model



5.4 The EIA-632 and IEEE 1220 Models

The EIA-632 Model (Figure 7) and IEEE 1220 Model (Figure 8) information that follows describes generic problem-solving systems engineering processes that produce the specifications, baselines, and related products. The processes provide the mechanism for identifying and evolving the product and process definitions of a system and they apply throughout the system life cycle to all process activities. *Both EIA-632 and IEEE 1220 models resemble the Iterative Waterfall Model discussed earlier in this report.* In general, these models include not only the “operations product,” which is delivered to the customer and used by a user, but also the enabling products associated with that operations product. The operations product consists of one (1) or more end products (so called since these are the elements of the system that “end up” in the hands of the ultimate user). The associated processes are performed using products that enable the end products to be put into service, kept in service, and retired from service.

Figure 8 – The IEEE 1220 Process Model



5.5 Summary and Evaluation of Process Models

Six (6) models were presented above – the Waterfall, Incremental, Spiral, Vee, EIA-632, and IEEE 1220 models. It is obvious in the discussion that, for the most part, all of the models are derivatives of the Waterfall Model with different variations in how the process flows and iterates. Each of these models has advantages and disadvantages associated with their use. Table 1 compares the models that have distinctive characteristics, namely, the Waterfall, Incremental, Spiral, and Vee.

The Waterfall and Spiral Models of development lend themselves to a functional decomposition approach to design that follow a top-down systems design. The top-down approach has the following advantages:

It is an orderly, systematic model for managing the size and complexity of system development.

It can be customized to a specific system. If a portion of the system is not well understood, prototyping can be used in the analysis phase. If the project is large, risks can be reduced by scheduling incremental deliveries.

The limitations of top-down systems design include:

- Top-down design does not take into account evolutionary changes.
- In top-down design, the system is characterized by a single function. This is a questionable concept.
- Top-down design is based on a functional mind-set. The underlying data types (or data structures) are often ignored.
- Top-down design by itself does not encourage reusability. Reusability is handled by bottom-up synthesis of previously developed components and concepts.

Item 1 is the most serious shortcoming of the Waterfall and Spiral Models. With engineering and computing applications rapidly becoming more complex and businesses being forced to reorganize in order to remain competitive in global markets, the ability of an engineering process to adapt to change has recently become of paramount importance. For the Single-Pass Waterfall Model, changing requirements are the biggest cause of cost overruns and schedule slips. Users have been unable to define the requirements of a complex system without having had hands-on previous experience with the system. Applying an iteration loop to each of the process activities can alleviate this disadvantage. On the other hand, the Spiral Model, which is a Waterfall Model with a great number of iterations, can easily become corrupted unless each release is developed with discipline and standards.

The Vee Model incorporates both the top-down and bottom-up approach in the process, where each phase is distinctively allocated to appropriate personnel resources according to their role in the system development. The Vee Model leverages the advantage of the Waterfall Model in illustrating the evolution of user requirements into preliminary and detailed designs in the top-down manner. It also accommodates integration and verification of system components through system and subsystem testing using a bottom-up path that tends to allow design reuse. The Vee Model further emphasizes the hierarchy of the decomposition process and its culmination in the build-up process that facilitates typical development of software subsystems.

Table 1 – Evaluation Summary of Systems Engineering Process Models

Models	Advantages	Limitations
The Waterfall Model	<ul style="list-style-type: none"> • This model is the oldest and most widely used life cycle. • It is well documented and supported. • It is accepted and well understood by customers. • The model is a logical sequence of processes and includes: <ul style="list-style-type: none"> o Direct mapping to phase specific processes; and o Clear boundaries of tasks. 	<ul style="list-style-type: none"> • Real projects rarely follow a sequential flow of processes. • The model requires the customer to state all requirements up-front with high fidelity. • It is inflexible to changes in the program scope. • The customer must have patience because the product is only delivered once, at the end of the process. • Early major problems may be undetected until later stages with disastrous results.
The Incremental Model	<ul style="list-style-type: none"> • The model contains multiple deliveries – one (1) for each increment. • Development phases are executed in each increment. • Planning is performed on an incremental basis. • The focus is on building the system in increments because: <ul style="list-style-type: none"> o Increments are cohesive system elements; and o System functionality is provided by “horizontal” slices of the system. • Requirements in an increment should be “frozen.” • Standard development phases are executed in each increment. • Strong emphasis is placed on the early production of an initial capability. • Parallel development efforts are supported. • Deliveries are made to the customer from each increment. • The Incremental Model is nicely compatible with integrated product team concepts and advantages include: <ul style="list-style-type: none"> o Early program functionality, decreased risks, and increased customer satisfaction; o Adjustment to scope and requirements changes; o Suppression of detail on future increments; o Development cycles within an increment are efficient due to the size of the increment; and o Incremental integration is facilitated. 	<ul style="list-style-type: none"> • The comprehensive planning and management of increment sequence can satisfy stable requirements at first but future deliveries are undefined. • Some rework is usually required of early increments. • Baseline management can be awkward. • The model requires a new mind set for integrated product teams and program management. • It requires a “fluid” relationship with the customer through: <ul style="list-style-type: none"> o Increment definition; and o Frozen requirements for an increment.

Table 1 (Continued)

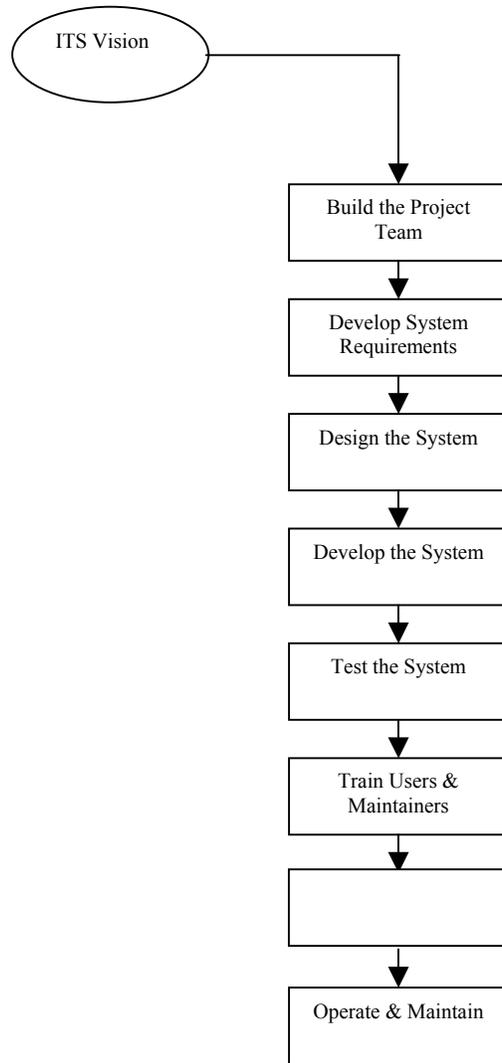
Models	Advantages	Limitations
The Spiral Model	<ul style="list-style-type: none"> • This model has the potential for multiple deliveries. • It has the potential for multiple executions of phase specific processes. • It is better for software intensive systems. • Planning is conducted on a spiral basis. • The model is focused on reducing risks. • It allows for an evaluation of risks before proceeding to a subsequent phase. • High-risk requirements are identified, implemented, and evaluated using prototyping. • The model provides a formal opportunity to determine completion or redirection of the development effort. • It may be applied to individual system components independently. • Each spiral requires customer approval to proceed. • Standard life cycle phases are executed only in the development phase; precisely what is done is a program decision. 	<ul style="list-style-type: none"> • The rework of prototypes is usually required. • At its core, a Spiral may still be a Waterfall development. • The completion opportunity is not likely to be exercised on multi-year contracts. • The model may encounter some difficulty in overall planning and costing. • The deliverables may not be well defined. • Prototypes are used primarily to assess risk, not as a basis for product development. • There is a perception of low value added quadrants. • Each spiral requires customer approval to proceed. • Standard life cycle phases are executed only in the development phase; precisely what is done is a program decision.
The Vee Model	<ul style="list-style-type: none"> • The model reflects both the top-down and bottom-up approach with: <ul style="list-style-type: none"> o Evolution of user requirements into preliminary and detailed designs as shown on the left side; and o Integration and verification of system components through subsystem and system testing. • The model emphasizes the hierarchy of the decomposition process and its culmination in the build up process. • Software subsystems can be easily included in the Vee Model. 	<ul style="list-style-type: none"> • One must ensure that feedback is included in the process. • There should be integration planning during the design requirements phase. • There should be requirements verification during the integration phase.

6. Conclusions and Recommendations

While the systems engineering approach is critical to the success of the development of ITS projects, no single development approach may be best in all situations. It is not as important which methodology is used, as it is extremely important that a methodology be used. If one looks at the typical ITS deployment life cycle as shown in Figure 9, it can be seen that the Vee Model probably provides the best fit. That is, the ITS proceeds from vision (user requirements) to system requirements (specifications) to system to testing to implementation. The Vee Model also allows the customer's perspective, the systems engineer's perspective, and the contractor's perspective to influence the process, which fits well with the transportation project development approach used by FDOT wherein user inputs are solicited, FDOT and/or its prime contractor develops the specification(s), and subcontractors provide implementation.

However, as larger and more complex projects are undertaken, especially those involving the insertion of new technologies and the development or use of substantial amounts of software, the Spiral Model should be considered. Since large, complex projects are quite often not undertaken all at once, the Spiral Model provides a good fit for the process. The Spiral Model can be envisioned as a series of Vee Models implementing an evolutionary development process. It allows the developer of large, complex systems to provide an initial capability that may be less than the full (or ultimate) requirement. This initial capability will provide an earlier delivery, will be more affordable, and will provide risk reduction. The initial capability is then enhanced as more capability is added through successive iterations of the Spiral Model. It should be noted that DoD has recently adopted the Spiral approach in their new "evolutionary acquisition" strategy.¹⁰ This approach allows for deployment and subsequent enhancement(s) while providing a substantial risk reduction orientation.

Figure 9 – ITS Deployment Life Cycle¹¹



References

- ¹ Global Intergy Corporation, *Systems Engineering in Intelligent Transportation Systems*, a course developed for FDOT, October 2001.
- ² IEEE 1220-1998, *IEEE Standard for Application and Management of the Systems Engineering Process*, IEEE Computer Society, IEEE, December 8, 1998.
- ³ EIA-731-1, *Systems Engineering Capability Model (SECM)*. EIA.
- ⁴ Sheard, Sarah A., *Help! How Do I Make My Organization Comply With Yet Another New Model?* Proceedings of INCOSE, 2001.
- ⁵ EIA/IS-632, *Interim Standard: Systems Engineering*. 1994. EIA, December 1994.
- ⁶ Martin, James N., *Evolution of EIA-632 from an Interim Standard to a Full Standard*. Proceedings of INCOSE, 1998.
- ⁷ Martin, James N., *Overview of EIA-632: Processes for Engineering a System*. Proceedings of INCOSE, 1998.
- ⁸ Sheard, Sarah A., *The Frameworks Quagmire, A Brief Look*. Proceedings of INCOSE, 1997b.
- ⁹ Grapham, I., *Object-Oriented Methods: Principles and Practice*, Third Edition, Addison-Wesley, 2001.
- ¹⁰ Memorandum, Under Secretary of Defense; Evolutionary Acquisition and Spiral Development, April 12, 2002.
- ¹¹ Intelligent Transportation Primer, *Deploying ITS*.
- ¹² Sage, Andrew P., Rouse, William B., *Handbook of Systems Engineering and Management*, John Wiley and Sons, 1999.