Regular Paper

The Systems Engineering Started in the Middle Process: A Consensus of Systems Engineers and Project Managers

A. Terry Bahill[1,*] and Clark Briggs[2,*]

[1]*Raytheon Missile Systems and Systems and Industrial Engineering, University of Arizona, Tucson,* AZ 85721-0020

[2]*Jet Propulsion Laboratory, California Institute of Technology, Pasadena,* CA 91109-8099

Received November 20, 2000; Accepted January 25, 2001

ABSTRACT

The systems engineering process often begins somewhere in the middle of a project. Naturally, the systems engineering process should be different for a project where systems engineering starts in the middle versus a project where it starts at the beginning of the project. This article presents a consensus of senior systems engineers and project managers at the Idaho National Engineering and Environmental Laboratory and Raytheon Missile Systems about how systems engineering is done when it starts in the middle of a project. © 2001 John Wiley & Sons, Inc. Syst Eng 4: 156–167, 2001

1. INTRODUCTION

In the textbooks, the systems engineering process usually starts at the beginning of the project and most of the systems engineering effort occurs during the first two phases of the system life cycle, namely, stating the problem and investigating alternatives [Chapman, Bahill, and Wymore, 1992; Sage, 1992; Wymore, 1993; Blanchard and Fabrycky, 1998; Bahill and Gissing, 1998; Bahill and Dean, 1999; Sage and Rouse, 1999; Buede, 2000]. However, in the real world, the systems engineering process often begins somewhere in the

*All correspondence should be addressed Terry Bahill (e-mail: terry@sie.arizona.edu).

Clark Briggs performed this work while he was on the faculty of the University of Idaho at Idaho Falls.

Contract grant sponsor: U.S. Department of Energy, Assistant Secretary for Environmental Management, under DOE Idaho Operations Office. Contract grant number: DEAC07-94ID13223

Contract grant sponsor: Raytheon Test Systems Design Center, Tucson, Arizona

Systems Engineernig, Vol. 4, No. 2, 2001
© 2001 John Wiley & Sons, Inc.

middle of the project. This paper, which is based on Bahill and Briggs [2000], explains the systems engineering process when systems engineering begins in the middle of an ongoing project. We call this the *systems engineering started in the middle process.*

There are also *many reasons* why it may appear that systems engineering was started in the middle of the project. First, management might have (1) lacked experience, (2) felt that systems engineering cost too much, (3) felt that formal systems engineering took too much time, (4) believed that their process was sufficient, because it had not failed them in the past, (5) thought that they had done the systems engineering, but just did not write anything down, (6) felt that the system was already well enough defined, or (7) done most of the systems engineering tasks, but needed help with some. If the perceived cost created the lacuna, then the systems engineer should not expect to get approval to do everything that needs to be done. In this case, the systems engineer should use euphemisms for systems engineering such as *a systematic, comprehensive approach*.

Systems engineering would also seem to have been started in the middle of a project if there were many COTS components or legacy elements. Similar situations could also arise due to mergers, company reorganizations or just lack of formal systems engineering. Disappearance of important documents such as the SEMP or the requirements documents could also cause systems engineering to start in the middle. Finally, basic process re-engineering is usually a systems engineering started in the middle process.

The consensus of our senior systems engineers and project managers is that a complete systems engineering started in the middle process would cost two to ten times as much as a systems engineering process that started at the beginning of the system life cycle. Therefore, when systems engineering is started in the middle, the systems engineer cannot do a complete job of systems engineering, because it would *cost too much and take too long*. Consequently, he or she must deliberately decide which parts of the systems engineering started in the middle process are essential for each project and tailor the process appropriately.

Problem statement for this paper. Identify the similarities and differences in the systems engineering processes when the process is started at the beginning of the system life cycle versus a project when it is started in the middle. To solve this problem, we interviewed systems engineers and project managers at the Idaho National Engineering and Environmental Laboratory and Raytheon Missile Systems. This paper summarizes their experiences and opinions. It starts with a description of two examples of systems engineering started in the middle processes. Then it presents a general systems engineering process and discusses the differences in each function depending on whether systems engineering is started at the beginning or in the middle.

2. EXAMPLE SYSTEMS ENGINEERING STARTED IN THE MIDDLE PROCESSES

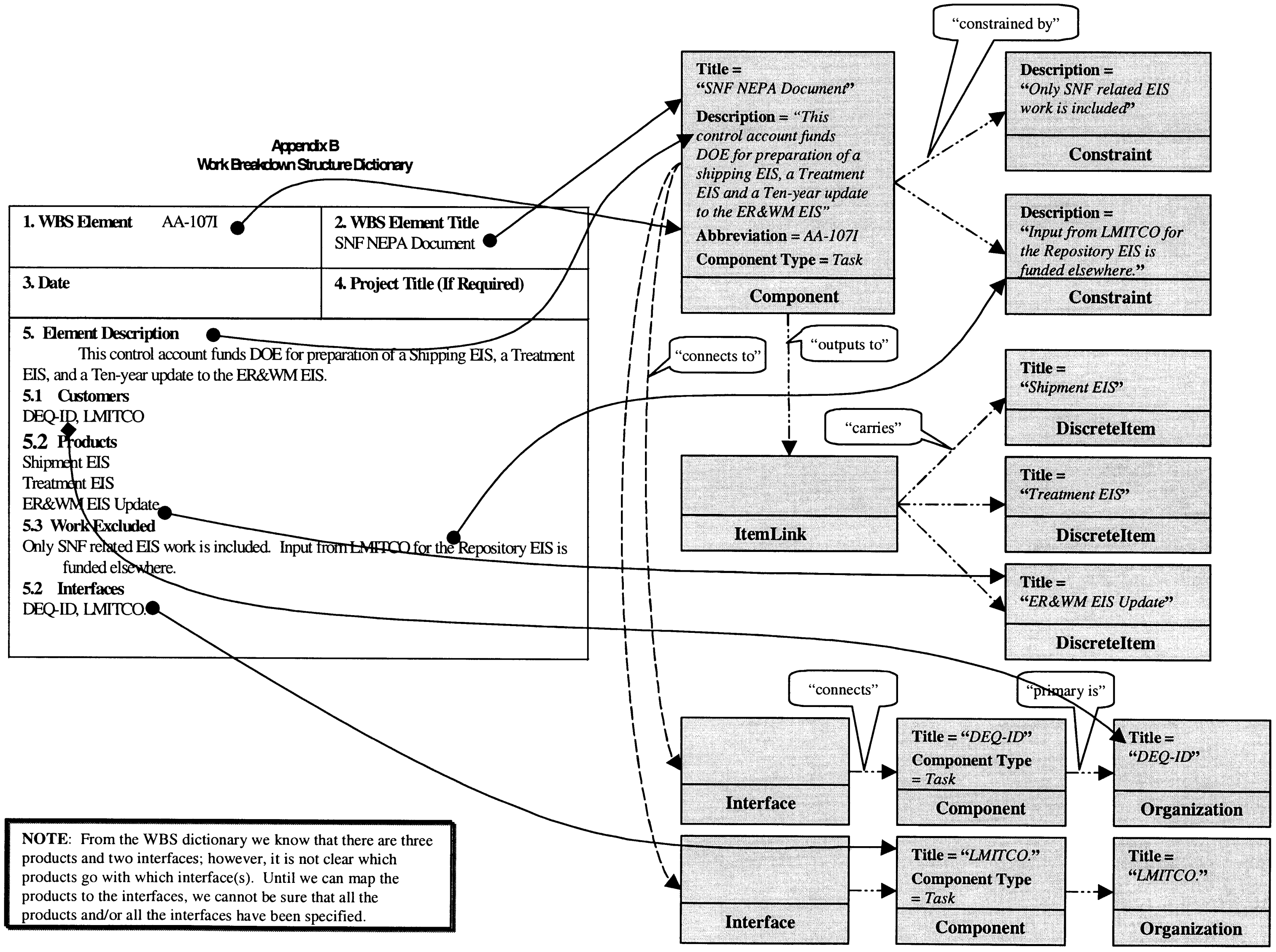
In this section, we will look at two systems engineering started in the middle processes. Only the first one (developed by Kevin Bailey) will be elaborated. Here is an abstract of it. Start with the existing work breakdown structure (WBS) and make sure that it is complete and valid. Then decompose it to the lowest level for which data exist. Then, from the individual WBS blocks, identify the process functions. From these functions, infer system requirements, and then from the requirements write a mission statement, which should then be approved by the customer. Go back to the WBS blocks and find the system deliverables. From these, deduce the system interfaces. Finally, use the WBS blocks to find assumptions.

A WBS is a hierarchical description of work elements that ideally includes start and end dates, a description of the work, customers, inputs, products, interfaces and staffing. The systems engineer will find information about the WBS in various places. There might be an electronic data system where the WBS is maintained. It might also be in the Project Management Plan or even in the overview of a Design Review. The systems engineer should flesh out the WBS as necessary. A typical WBS element is shown in Figure 1.

The text in this WBS element is terse, yet the systems engineer must glean several vital elements from it. He or she can use the WBS Title for the Component Title, capture the WBS Element Description in the Component Description, and use the WBS Element identifier as the Component Abbreviation.

The systems engineer should study the WBS elements for implied functions. Particular system functions will be needed to accomplish the work in the WBS Description. In addition, items delivered across interfaces hint at functions. Create functions in the project model at this time, even if the functional hierarchy is not yet evident. References to compliance requirements and other regulatory documents might also provide hints about needed functions.

The systems engineer should list the requirements in the project model and create source elements for the regulatory document. The hierarchy for the requirements may become clear as the systems engineer discerns the structure of the project through the WBS, or the structure may have to be developed later.


Appendix B
Work Breakdown Structure Dictionary

1. WBS Element AA-1071	2. WBS Element Title SNF NEPA Document
3. Date	4. Project Title (If Required)

5. Element Description
This control account funds DOE for preparation of a Shipping EIS, a Treatment EIS, and a Ten-year update to the ER&WM EIS.
5.1 Customers
DEQ-ID, LMITCO
5.2 Products
Shipment EIS
Treatment EIS
ER&WM EIS Update
5.3 Work Excluded
Only SNF related EIS work is included. Input from LMITCO for the Repository EIS is funded elsewhere.
5.2 Interfaces
DEQ-ID, LMITCO.

NOTE: From the WBS dictionary we know that there are three products and two interfaces; however, it is not clear which products go with which interface(s). Until we can map the products to the interfaces, we cannot be sure that all the products and/or all the interfaces have been specified.

Figure 1. Mapping WBS elements to the project model (complements of Kevin Bailey).

The systems engineer can use the WBS Customers and Interfaces fields to create interface elements and note the Customers as the Primary Organization. The products, the deliverables, are called DiscreteItems in Figure 1. When possible, each DiscreteItem should be linked to an Interface.

WBS elements, as in this example, might not provide sufficient information for the needed connectivity in the project model. For example, the three deliverable products are not associated with either of the two interfaces. Just as this WBS has not listed its inputs, no other WBS element will note accepting these three products as inputs. The project schedule might help, since task interdependencies might indicate that a subsequent WBS element depends on this WBS element for those products. Similarly, products delivered to customers might be represented on the project schedule as delivery milestones.

Some WBS systems have a field for assumptions or issues. In practice, managers tend to document-unverified agreements or planned activities of others in such slots to remind everyone to do it later, or to simply cover their bases. This leaves disconnects in the project planning and provides good areas for the systems engineer to investigate. The expectation is that these issues become negotiated and documented so that no one is surprised later. Each of these assumptions should be considered in the risk analysis.

In *summary*, our first systems engineering started in the middle process creates a model of the process from a WBS. From this model the systems engineer can choose the selling points, translate them into project management terms [Kerzner, 2000] and deliver them to the project manager.

A *second* systems engineering started in the middle process starts with a description of the existing (or proposed) physical architecture. This is decomposed into a hierarchy of components or subsystems. These subsystems become the objects for an object-oriented model and provide a springboard for discovering the functions for a functional model. The interfaces between the subsystems are then defined. The deliverables that are passed between the subsystems help identify these interfaces. Finally, the subsystem owners are queried to help identify the requirements: Experience has shown that this is a hard task.

Bahill and Briggs [2000] presented four systems engineering started in the middle processes, consolidated them, and then compared the consolidation to a general systems engineering started at the beginning process, as is shown in Table I.

3. COMPARISON OF STARTED IN THE MIDDLE AND STARTED AT THE BEGINNING PROCESSES

Humans (individually, on teams, and in organizations) employ simple processes to increase their probability of success and reduce their risk of failure. Many authors, both technical and nontechnical, have described these processes, and their descriptions are very similar. Bahill and Gissing [1998] compared these proc-

Table I. Preliminary Comparison of Systems Engineering Processes

	When systems engineering starts at the *beginning* of the life cycle	When systems engineering starts in the *middle* of the life cycle
Primary customer	Starts with the *customer* and grows to a broader group	Starts with the *project manager* and grows to a broader group
Sources of information	Interviews with customers and other stakeholders	Documents and physical architecture
Emphasis	The product: requirements and functions	The process: systems engineering management
Precedence	Form follows function	Function follows form
Role	Leader	Archaeologist
Products	Complete systems engineering documentation	Limited time and resources force products to be scaled down and not all are produced
Selling points	Complete systems engineering documentation	Description of interfaces, Issues to be resolved, Functions not required, Requirements not verified, Sensitivity analysis
Sequence	Starts with mission statement, then requirements, functions and objects	Varies with the process
Process	Must be tailored for the particular project	Must be tailored for the particular project

esses and extracted the similarities: **S**tate the problem, **I**nvestigate alternatives, **M**odel the system, **I**ntegrate, **L**aunch the system, **A**ssess performance, and **R**eevaluate. These seven functions can be summarized with the acronym SIMILAR: **S**tate, **I**nvestigate, **M**odel, **I**ntegrate, **L**aunch, **A**ssess, and **R**eevaluate. It is important to note that the SIMILAR Process, shown in Figure 2, is not sequential. The functions are performed in a parallel and iterative manner. This SIMILAR Process will now be used as the normal, or the systems engineering started at the beginning process, and the systems engineering started in the middle process will be compared to it.

3.1. State the Problem

SE started at the beginning. The problem statement starts with a description of the top-level function that the system must perform: This might be in the form of a mission statement, a concept of operations or a description of the deficiency that must be ameliorated. Mandatory and preference requirements [Bahill and Dean, 2001] must be traceable to this problem statement. Acceptable systems must satisfy all the mandatory requirements. The preference requirements are traded off to find the preferred alternative. The problem statement should be in terms of *what* must be done, not *how* to do it. It might be composed in words or as a model. Inputs come from end users, operators, suppliers, acquirers, maintainers, owners, regulatory agencies, victims, sponsors, manufacturers, and other stakeholders.

SE started in the middle. When joining in the middle of a project, the first thing the systems engineer should do is *ask questions*. What are you doing? Why are you doing it? Who is the customer? What does the customer need? What products will be delivered to the customer? What is the existing system? Where are we going? How are we going to get there? What is the driving requirement? What might be the next big problem? Is something broken? Answers to these questions will be used to describe the customer and to choose the systems engineering products, the source documents, the selling points, and the particular systems engineering started in the middle process.

The systems engineer's primary *customer* is the project management team, which consists of the project manager, the chief engineer, the program controls representative, and the technical specialists (leads). As the systems engineer expands his or her realm and becomes more valuable to the project, the systems engineer's customer should evolve through three phases. In the beginning, the systems engineer will be struggling to educate and earn the trust of the project manager. Then, the systems engineer provides leadership for the project management team. And in time, the systems engineer interfaces with the external customer, the sponsor, corporate officers, and external oversight boards. The systems engineer often becomes the advocate of the external customer.

Source documents that are usually available with greater or lesser quality in various projects include the contract, the statement of work (SoW), work breakdown structures, task statements (from the contract officers), PERT charts, product descriptions, piles of paper, tech memos, design reviews, requirements documents (if you are lucky), a document hierarchy, decision trees, safety analysis reports, technical specifications, regulatory agreements, project management plans, and monthly management reviews. Of course, a primary source of information is not documents, but conversations with members of the integrated product development team (IPDT).

3.2. Investigate Alternatives

SE started at the beginning. Alternative designs are created and are evaluated based on performance, schedule, cost, and risk figures of merit. No design is likely to be best on all figures of merit, so multicriteria decision-aiding techniques should be used to reveal the preferred alternatives. This analysis should be redone whenever more data are available. For example, figures of merit should be computed initially based on esti-

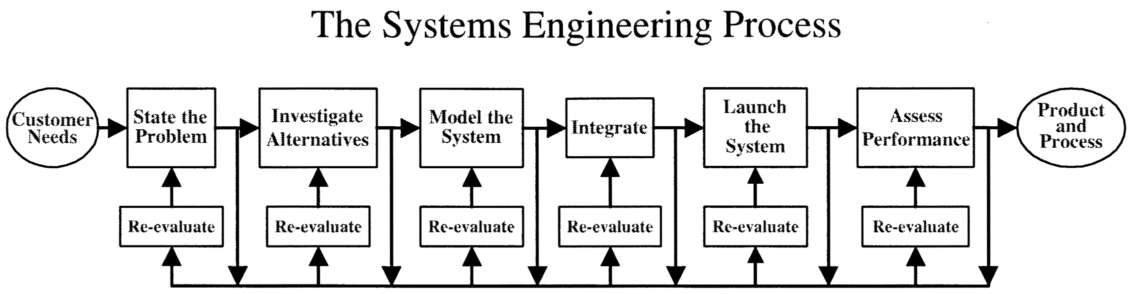


Figure 2. The Similar Process [from Bahill and Gissing, 1998].

mates by the design engineers. Then, concurrently, models should be constructed and evaluated; simulation data should be derived; and prototypes should be measured. Finally, tests should be run on the real system. For the design of complex systems, alternative designs reduce project risk. Investigating innovative alternatives helps clarify the problem statement.

SE started in the middle. Tradeoff studies for the system must be simple. It might not be cost-effective or schedule-compliant to perform extensive *quantitative* tradeoff studies. However, it is important to do at least simple *qualitative* tradeoff studies, because examining unusual alternatives (especially the "do nothing" alternative) is a great way to discover additional system requirements. Decision trees and Ishikawa fishbone diagrams can show graphically the results of alternatives analyses.

3.3. Model the System

SE started at the beginning. Models will be developed for most alternative designs. 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 [Bahill et al., 1998]. Systems engineering is responsible for creating a product and also a process for producing it. So, models should be constructed for both the product and the process. *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.

SE started in the middle. System functions or objects must be identified. For classically trained systems engineers, it is most appropriate to discover the system functions first. This can be done with functional analysis and decomposition [Wixson, 1999] and functional flow block diagrams [Bahill et al., 1998]. For modern software projects, it is more appropriate to highlight the system objects and only provide functions in the detailed breakdowns. For a systems engineering started in the middle project, there are existing (or proposed) subsystems. Therefore, it is easier to identify objects than functions. So, the objects should be the starting point.

Our first systems engineering started in the middle process, which started with work breakdown structures, is primarily a model of the process. Our second systems engineering started in the middle process, which started with the physical architecture, is primarily a model of the product.

For the University of Arizona Student Satellite Program, we are creating two complete sets of systems engineering documents: one for the product (the satellite that will be launched into orbit) and one for the process (the nine teams that are designing the satellite).

Modern quality initiatives say that most inspection and testing should be done on the process, not on the product.

3.4. Integrate

SE started at the beginning. 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 that they work as a whole. Interfaces between subsystems must be designed. Subsystems should be defined along natural boundaries. Subsystems should be defined to minimize the amount of information to be exchanged between the subsystems. Well-designed subsystems send finished products to other subsystems. Feedback loops around individual subsystems are easier to manage than feedback loops around interconnected subsystems. Coevolving systems also need to be integrated so that the final system is built and operated using efficient processes.

SE started in the middle. The existing (or proposed) system might be an integration of many subsystems. The integration plan might provide insight into potential opportunities for improvements in subsystem integration. The interfaces between subsystems might be redefined. Gap analyses and N^2 diagrams help describe interfaces. Our first systems engineering started in the middle process, which started with work breakdown structures, focused on the process interfaces. It used the deliverables to identify the interfaces. It checked the timing of the inputs and outputs to see if any input was required before it was produced as an output. It checked to see if any subsystems were being overwhelmed by inputs.

3.5. Launch the System

SE started at the beginning. Launching the system means running the system and producing outputs. In a manufacturing environment this might mean buying commercial off the shelf hardware or software, or it might mean actually making things. Launching the system means allowing the system do what it was intended to do.

The systems engineer's *products* are a mission statement, a requirements document including verification and validation, a description of functions and objects, 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 may have been made and it should be investigated. There are several valid reasons for violating this advice, for example, when a function is performed by one component in one mode and another component in another mode.

SE started in the middle. Not all of the systems engineering products will be deliverables, only the most important ones, called selling points. *Selling points* are the high value-added systems engineering products or services that should be provided to the project manager early in the process to point out the value of systems engineering. Typically, selling points include a description of interfaces, issues to be resolved, functions not required, requirements not verified, and sensitivity analyses. A sensitivity analysis of the set of requirements can highlight the cost drivers. Showing these to the customer often causes the customer to relax a requirement and thereby reduces cost. The selling points should dramatically improve project performance and reduce cost.

Elaboration of the selling points. (1) There may be issues that need to be resolved. The impact of each issue should be assessed. Those that pose the greatest project risk should be highlighted. TBDs (to be determined) are an example of issues that need to be resolved. (2) By attempting to link system functions to system requirements, we can expose activities or functions that are not linked to requirements. Unlinked functions are often the result of system evolution. Unlinked functions are either non-value-added functions that should be eliminated or they are necessary functions that point out missing requirements. This linking of functions to requirements might also expose requirements that have no function performing them. These would also be cause for concern, necessitating the creation of more functions or perhaps allowing the elimination of a requirement. (3) System documentation with a mission statement, functions, requirements, and validation gives the project manager a defensible position for his or her project and a clearer understanding of it. (4) Providing information and insight, and identifying project omissions should be of value to the project manager, and therefore should help demonstrate the need for using the systems engineeringprocess. (5) The selling points do not include *all* of the traditional systems engineering products, however, just the high value-added ones. Determining which systems engineering products will produce the highest value is done using experience, intuition, knowledge of project management, and customer statements of priority.

3.6. Assess Performance

SE started at the beginning. Figures of merit, technical performance measures, and metrics are all used to assess 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 mitigate risk during design and manufacturing processes. Metrics (including customer satisfaction comments, productivity, number of problem reports, etc.) are used to help manage a company's processes. Measurement is the key. If you cannot measure it, you cannot control it. If you cannot control it, you cannot improve it [Moody et al., 1997]. 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 manger is allocated a reserve. These resource budgets are managed throughout the system life cycle.

SE started in the middle. The three most important care-abouts for a project manager are project performance, cost, and schedule. Project performance should be tracked using metrics, and technical performance measures should be created for the high-risk items [Moody et al., 1997]. We will not be able to manage important resources such as weight, volume, price, etc., but at least we can archive what has been done. Schedule should be managed with the aid of PERT Charts and Gantt Charts. Often three different people do the project specifications (requirements and functions), the project schedule, and the project cost estimates, and they might use three different metrics. The systems engineer can make a valuable contribution by making these three consistent. Then the systems engineer can help the project team make tradeoffs between performance, cost, schedule, and risk.

Table II. Function Impact Matrix

	Adds value	Does not add value
Mandated	Adding value is tested by seeing if the mission or its performance measurements are aided by these functions	Investigate requirements for these functions to see if they can be changed.
Not mandated	Ask the customer if they want to add these functions	Drop these functions immediately

3.7. Reevaluate

SE started at the beginning. Reevaluate is arguably the most important of these functions. For a century, engineers have used feedback to control systems and improve performance. It is one of the most fundamental engineering tools. Reevaluation should be a continual process with many parallel loops. Reevaluate means observing outputs and using this information to modify the system, the inputs, the product or the process. Figure 2 summarizes the SIMILAR Process. This figure clearly shows the distributed nature of the Reevaluate function in the feedback loops. However, all of these loops will not always be used. The particular loops that are used depend on the particular situation.

SE started in the middle. To help reduce cost, the project activities should be analyzed with a function (or activity) impact matrix, as shown in Table II. Systems engineers must continually stand back and question the big picture. They should continually be asking: "What could go wrong? What has not been considered? Do we really need that function?"

Table III is a very brief summary of Section 3. It omits details and overly generalizes the processes.

4. GENERALIZATIONS

4.1. Words of Advice

To map out the best course of action, the systems engineer should discover why he or she was assigned to the project. Some possible reasons are (1) the project needs cosmetic enhancements, (2) the project is under control, but the manager wants to keep the house of cards intact and has some extra money to spend, (3) the project needs some help, a fix or two, or a minor course correction, (4) the project is in deep trouble, well past salvation, and the project manager is grasping at straws,

Table III. Comparison of Two Versions of the Systems Engineering Processes

	When systems engineering starts at the *beginning* of the life cycle	When systems engineering starts in the *middle* of the life cycle
State the problem	Interview stakeholders and discover functions and requirements.	Discover what the system is supposed to do and what information is available about the system.
Investigate alternatives	Do a formal tradeoff study of alternative concepts.	Investigate a few simple alternatives.
Model the system	Develop models for the product and the process that will produce it. Models should be developed for all alternative concepts.	Simple models should be made, but only for the preferred alternative and the existing process.
Integrate	Design subsystems and interfaces.	Study and describe the interfaces.
Launch the system	Produce a mission statement, requirements, functions, objects, test plans, interfaces, models, sensitivity analyses, a tradeoff study, risk analyses, make the hardware, write the software, etc.	Produce the selling points: interfaces, issues to be resolved, functions not required, requirements not verified, and a sensitivity analysis,
Assess performance	Create figures of merit, technical performance measures and metrics.	Track project performance, cost and schedule.
Re-evaluate	This is a continual process with many parallel feedback loops.	Look for what could go wrong and try to identify unneeded requirements and functions.

and (5) the sponsor has said that the project is in trouble and needs systems engineering.

When a systems engineer is assigned to a project in the middle of its life cycle, the project manager may hear the message: "Someone thinks that I have screwed up and they have sent this outsider to bail me out." The systems engineer's job is to help the project manager without seeming threatening. He or she should not try to sell systems engineering. Rather, create the selling points and let them sell systems engineering. These are difficult tasks, and they require a systems engineer with a lot of experience.

When a systems engineer is assigned to a project in the middle of its life cycle, he or she should be sensitive to being nonjudgmental. A lot of water has already gone under the bridge, and decisions were made in a different environment than the current one. One should not presume that the lack of systems engineering effort reflects bad project management, or a good working relationship will not be fostered. It is always best to accept that the project is where it is for whatever reason and take a view to determine what needs to be done to go forward. The systems engineer should define the work required to create the deliverables and meet the objectives within budget and on schedule (or support negotiations to show how an incompatible set of constraints exists from a technical standpoint) and work to build an effective team to produce the deliverables.

The systems engineer needs to learn the political environment so he or she can know where to push for change and where to back off. He or she must know when to walk away and know when to run.

A systems engineer who has been assigned to a project that is in trouble needs to keep lifelines back to corporate sponsors or systems engineering managers who can provide safe extraction. Otherwise their utility on subsequent jobs will suffer, or worse yet, they will think they have failed and will leave the company unhappy.

The systems engineer needs to describe the maximum failing requirement (the best you can do and still fail) and the minimum passing requirement (the worst you can do and still pass). Anything outside of these bounds should be avoided, if one is attempting to fix a project with limited time, money, or personnel.

There is a need for continual communication between the systems engineer and the project manager. If the project was not well thought out at the outset, then even the goals might have changed. Do not tell the project manager this; with your help, he or she will discover it. We cannot overstress the need for continual communication. After all the Tower of Babel did not fail because of technical deficiencies.

4.2. Limitations

Of course, there are limitations to the statements in this paper. There are some projects where nothing applies, and there are others where everything applies. One of the obvious differentiators is the size or cost of the project. For example, we have said that a formal large-scale quantitative tradeoff study might not be cost-effective. This is probably true for small projects, but for a multimillion-dollar project, it might be necessary, as well as cost-effective.

Another differentiator is the innovativeness of the project, which can be characterized by whether or not a similar system has been designed and built before. Precedented projects, where similar systems have been produced previously, such as bridges and cars, are not high-risk projects. There is more risk with unprecedented projects, which can be decomposed into three categories: (1) projects where the individual components have precedent, but their integration into a large system has never been done before, such as the Apollo program to put a man on the moon; (2) projects where similar large-scale integration has been performed before, but the components are of a new technology, such as making a computer with transistors instead of vacuum tubes; and finally (3) projects where both the component technology and the integration is unprecedented, such as the Manhattan Project of World War II. Projects with unprecedented components and integration have the highest risk. As a topic for future research, we would like to determine what type of systems engineering started in the middle process would be best for each of these four types of projects.

4.3. The Increased Cost

Our consensus is that a complete systems engineering started in the middle process would cost two to ten times as much as a systems engineering process that started at the beginning of the system life cycle. One of the reasons for this increased cost is that the systems engineering may create the need for design changes, and design changes made late in the process are expensive. Generally, the cost of design changes is related exponentially to the time in the design process. Duplication of effort is another reason for the increased cost. Some documents might have to be recreated partially or completely. A third reason for increased cost is that important information may no longer be available, e.g., we may not be able to conveniently query the original customer about the rational for requirements, and reasons for incipient design decisions may have been lost. Therefore, a systems engineering started in the middle process must be highly focused. It is too late to impose new requirements that are only slightly better.

If a systems engineering started in the middle process gets in series with the project management, it may cause the schedule to slip, which might be intolerable for the project manager.

4.4. Reasons for Systems Engineering Being Started in the Middle

A possible cause for systems engineering to be started in the middle is poor planning. This would be the case for projects planned by project planners without the technical expertise to understand the technical issues and risks. They are mostly concerned with cost estimation rather than product performance.

Other causes for systems engineering to be started in the middle would be poorly executed or poorly documented initial systems engineering. At the beginning of most projects, the customers' needs are stated vaguely or are based on a concept of operation that was originally used to sell the project, but did not take into account cost or risk. Therefore, the foundation for the systems engineering process is often lacking, and the process seems to be pure chaos. The systems engineer must live with the pressure for tangible results during this period of chaos. Eventually, when the knowledge base grows sufficiently to provide insight to see the answers, chaos diminishes, and a more systematic approach becomes visible. However, the chaos phase consumes most of the available time in these earliest phases. There is insufficient time and opportunity to go back and create the documentation, beyond what is needed to satisfy the current customer. Unfortunately, a new systems engineering team often comes in at this time, and they are lacking in historical knowledge and are poorly equipped to pick up the pieces. They truly feel that they are starting in the middle. This is when a senior systems engineer is really needed, to find clarity and direction, and hold off the naysayers. The new team is most vulnerable to early extinction or distinction during this phase before the first selling points are delivered and the ah-ha's begin.

In some organizations, systems engineering is heavily involved in writing the proposal including a systems engineering management plan (SEMP) and an integrated management plan (IMP). However, after the contract is won, the SEMP and the IMP are not integrated into the project, and therefore, the requirements and the functions have to be discovered all over again, often in the middle of the process. Our experts suggested several possible causes for the disappearance of the SEMP: (1) different teams writing the proposal and doing concept development with an organizational culture that encourages the second team to ignore the work of the first, (2) the concept development team not having the proposal, and (3) the long time delay between proposal writing and concept development.

Personnel retirements, mergers and company reorganizations could also cause the disappearance of systems engineering documents. Or the requirements documents might make the transition, but the history, the how and the why for the requirements, might not. This would also produce the need for a systems engineering started in the middle process.

For some long-term site management projects, the people or even the management company might be changed every five or ten years. When there is a transition from a business management company to a strong systems engineering company, there will be a lot of systems engineering started in the middle projects.

Software engineers are sometimes asked to convert old legacy functional programs into modern object-oriented programs. They often start in the middle and go up to requirements and down to code. However, this example is not limited to software. Any reengineering project is basically a systems engineering started in the middle process.

Some people have said: "That doesn't happen here. We always start systems engineering at the beginning of the project." Well this may depend on the definition of *the beginning of the project.* Some people say the project begins after the contract is issued and the project is formally kicked off. Others push the definition back to submission of the proposal, the request for proposals, the announcement and call for comments, or even the DARPA research white papers.

On the other hand, maybe systems engineering started in the middle is the norm not the exception, because most projects started at the beginning have COTS components or legacy elements (cost and risk constraints) and other predetermined design requirements (political and marketing constraints). Such constraints cannot be systems-engineered. We are not able to question the purpose, the functions, or the requirements.

In some fields of endeavor, for example in the legislative process, there is no beginning: Everything is continually evolving [Gardner, Nipper, and Plowman, 1999].

A six-sigma system is comprised of three subsystems: continual process improvement, process redesign or reengineering, and process management [Pande, Neuman, and Cavanagh, 2000]. All three of these start with an existing system. Therefore, six sigma projects should use a systems engineering started in the middle processes.

4.5. How This Consensus Was Formed

This project started with a grant proposal submitted to the Idaho National Engineering and Environmental Laboratory. After the proposal was funded, we identified and described four projects where systems engineering had been started in the middle. These descriptions were combined with the proposal to create a white paper. Then Bahill took the white paper to a system engineer and sat in front of him or her while he or she read it and marked it up. Bahill discussed the concepts with them and made notes. He discussed the comments with Briggs, edited the paper, and then took it to another systems engineer. This basic process was repeated a dozen times. Then the paper was sent to everyone in the Systems Engineering Department. Comments were received and incorporated. Meanwhile, Briggs documented the four examples of systems engineering stared in the middle that we had found and wrote the final internal research report. Subsequently, the paper was presented at the INCOSE symposium, at a couple of brown-bag lunches, and during two graduate systems engineering classes. Comments were received and incorporated. The interview process was repeated with systems engineers at Raytheon. Finally the interview process was repeated with three project managers at Raytheon.

Hundreds of people have commented on this paper; however, the material has come primarily from three-dozen people. As a result, some sentences were changed three dozen times. For example, the sentence "… a complete systems engineering started in the middle process would cost two to ten times as much as a systems engineering process that started at the beginning…" was revised many times until it was explained that the primary reason for the extra cost was that design changes made late in the process are expensive. Once this explanation was made explicit, it became obvious that whether it was two or ten would depend on how many design changes had to be made and how late in the life cycle they were made. We tried to keep the original words and meanings as they were given to us by the engineers and project managers. This means that at times the paper may seem disjoint. For example, the Introduction cites seven reasons why systems engineering may have been started in the middle. Whereas Section 4.4 gives seven quite different possibilities.

It may be that in most projects systems engineering is started in the middle. This would be one of the reasons why systems engineering is so difficult. You cannot follow a textbook approach. The systems engineering process must be tailored for the particular project.

Work Breakdown Structures. Many web sites give guidance for work breakdown structures. Here are two: (1) This is a URL for the MIL-HDBK-881, which is the DOD document describing how work breakdown structures should be created by and for the military: http://www.acq.osd.mil/pm/newpolicy/wbs/mil_hdbk_881.htm. (2) This is a URL for a presentation that was given by Neil Albert of MCR Federal, Inc at the ninth Annual International Cost Schedule Performance Management Conference. It provides another opinion on the preparation of work breakdown structures: http://www.acq.osd.mil/pm/paperpres/1097conf/albert/index.htm.

ACKNOWLEDGMENTS

We thank Kevin Bailey, George Beitel, Bob Caliva, Jim Close, Bob Harris, A.B. Hubbard, Jerry Jaggers, Jim Johnesee, John Nonte, Cathy Plowman, Mark Powell, Virginia Randall, Don Schilling, Neil Snyder, Rafael Soto, Finis Southworth, Bob Turk, and Larry Zirker of the Idaho National Engineering and Environmental Laboratory in Idaho Falls, and Bill Bracker, John Cameron, Bill Chapman, Paul Diamond, Mike Kaiserman, Steve Larson, Brian O'Cain, Jim Pullen, Art Schneider, and Bob Sklar of Raytheon Missile Systems in Tucson, for the ideas in this paper. This work was supported by the U.S. Department of Energy, Assistant Secretary for Environmental Management, under DOE Idaho Operations Office Contract DE-AC07-94ID13223, and by the Raytheon Test Systems Design Center in Tucson, Arizona.

REFERENCES

A.T. Bahill and C. Briggs, The systems engineering started in the middle process, Proc INCOSE 10th Annu Int Symp, July 16–20, 2000, Minneapolis, MN, pp. 565–571.

A.T. Bahill and F.F. Dean, What is systems engineering? http://www.sie.arizona.edu/sysengr/whatis/, 2001.

A.T. Bahill and F.F. Dean, "Discovering system requirements," in Handbook of Systems Engineering and Management, A.P. Sage and W.B. Rouse (Editors), John Wiley & Sons, 1999, pp. 175–220.

A.T. Bahill and B. Gissing, Re-evaluating systems engineering concepts using systems thinking, IEEE Trans Syst Man Cybernet Part C: Appl Rev 28(4) (1998), 516–527.

A.T. Bahill, M. Alford, K. Bharathan, J. Clymer, D.L. Dean, J. Duke, G. Hill, E. LaBudde, E. Taipale, and A.W. Wymore, The Design-Methods Comparison Project, IEEE Trans Syst Man Cybernet Part C: Appl Rev 28 (1998), 80–103. This paper with additional solutions and discussion is available at http://www.sie.arizona.edu/sysengr/methods2.

B.S. Blanchard, and W.J. Fabrycky, Systems engineering and analysis, Prentice Hall, Englewood Cliffs, NJ, 1998.

D.M. Buede, The engineering design of systems: Models and methods, Wiley, New York, 2000.
W.L. Chapman, A.T. Bahill, and A.W. Wymore, Engineering modeling and design, CRC Press, Boca Raton, FL, 1992.
B. Gardner, D. Nipper, and C.M. Plowman, A systematic approach to environmental legislation, INEL-EXT-99-00096, Idaho National Engineering and Environmental Laboratory, Idaho Falls, 1999.
H. Kerzner, Project management: A systems approach to planning, scheduling, and controlling, Van Nostrand Reinhold, New York, 2000.
J.A. Moody, W.L. Chapman, F.D. Van Voorhees, and A.T. Bahill, Metrics and case studies for evaluating engineering designs, Prentice Hall PTR, Upper Saddle River, NJ, 1997.
P.S. Pande, R.P. Neuman, and R.R. Cavanagh, The six sigma way: How GE, Motorola, and other companies are honing their performance, McGraw-Hill, New York, 2000.
A.P. Sage, Systems engineering, Wiley, New York, 1992.
A.P. Sage and W.B. Rouse, Handbook of systems engineering and management, Wiley, New York, 1999.
J.R. Wixson, Functional analysis and decomposition using function analysis systems technique, Proc Ninth Annu Symp INCOSE, Brighton, UK, June 1999, pp. 963–968.
A.W. Wymore, Model-based systems engineering, CRC Press, Boca Raton, FL, 1993.

A. Terry Bahill is a Professor of Systems Engineering at the University of Arizona and an Engineering Fellow with Raytheon Missile Systems in Tucson. He received his Ph.D. in electrical engineering and computer science from the University of California, Berkeley, in 1975. He holds a U.S. patent for the Bat Chooser™, a system that computes the Ideal Bat Weight™ for individual baseball and softball batters. He is Editor of the CRC Press Series on Systems Engineering. He is a Fellow of the Institute of Electrical and Electronic Engineers (IEEE) and of the INCOSE. He is the chair of the INCOSE Fellows Selection Committee.

Clark Briggs is a Member of Engineering Staff at Jet Propulsion Laboratory in Pasadena, CA. He graduated from the U.S. Air Force Academy in 1972 and received his doctorate from Columbia University in 1978. After 13 years in the Air Force, he worked for 12 years as an Avionics Systems Engineer at the NASA Jet Propulsion Laboratory. He served as an Associate Professor of Systems Engineering at the University of Idaho at Idaho Falls.