Gaining the Competitive Edge through Effective Systems Engineering

By:
Joe Kasser
POB 3419
Silver Spring, MD 20918
Phone - (301) 593-6136

and

Robin Schermerhorn
Computer Sciences Corporation
4600 Powder Mill Rd, Beltsville, MD 20705
Work phone - (301) 572-8634

Abstract Gaining the competitive edge through effective Systems Engineering means reducing costs by eliminating ineffective Systems Engineering. This paper first defines Systems Engineering and then identifies the role of the Systems Engineer. The consequences of ineffective Systems Engineering are then discussed at a high level in terms of the cost of, and delays to, both the process and the product. The paper concludes with a strategy for proactive Systems Engineering which can produce the desired system without incurring the cost escalations and schedule delays associated with ineffective Systems Engineering.

INTRODUCTION

The Defense and Aerospace industries are shrinking. These days, more and more contractors are bidding on fewer and fewer government contracts. Contractors, who several years ago would not have considered bidding on anything less than $100 million, are now responding to Requests for Proposals for less than $20 million. Those contractors are also attempting to reduce costs to be able to lower their bids.

The total cost of a system development project is the sum of the costs of each of the work activities performed on that project. As time passes during the implementation phase of a project, the actual amount of dollars spent diverges from the pre-award budgeted or estimated amounts. The difference between estimated and actual costs is due to several parameters one of which is ineffective Systems Engineering. This paper focuses on that parameter.

THE SYSTEMS ENGINEERING PROCESS

Systems Engineering has been recognized as the *process* by which the orderly evolution of man-made systems can be achieved (Blanchard et al. 1981). As a process, it can be defined, measured and improved. One definition of Systems Engineering is (Eisner 1988):

> *"An iterative process of top down synthesis, development and operation of a real world system that satisfies in a near optimal manner, the full range of requirements for the system."*

This process takes place over a period of time while the project producing the system passes through the concept definition, requirements analysis, design, implementation, test, delivery and operational support phases. Systems Engineers have a role to play in each phase of the Systems Development Life Cycle (SDLC) or project, and that role is different in each phase. While each phase has its own specific priorities and needs, there are several Systems Engineering factors common to each phase. Each, when performed effectively, can minimize cost escalation during the latter phases of the project.

IMPROVING THE SYSTEMS ENGINEERING PROCESS

Anyone interested in improving Systems Engineering can also learn from other disciplines. Tom DeMarco conducted an annual survey of real world development projects between 1977 and 1987 (DeMarco et al. 1987).

He accumulated over 500 project histories and reported that 15% of all projects studied came to naught; they were canceled, aborted, "postponed" or delivered products that were never used. Fully 25% of projects that lasted 25 person-years or longer failed to complete. In the majority of projects *there was not a single technological issue to explain the failure*. The cause of failure most frequently cited were people related, including:

- staffing problems
- disenchantment with management or the client
- lack of motivation and high turnover
- communications (inter-personal) problems.

Studies in the software production industry have shown that failure to fully identify and baseline system requirements early in the SDLC leads to overruns. Other studies in the manufacturing industries have shown that the design stage is frequently the point at which cost of a product is determined. The cost of fixing defects escalates as a function of the stage of the life cycle of the product in which defects are identified. Concurrent Engineering techniques are now being employed to identify both product and process defects as early in the product life cycle as possible. This principle also applies to Systems Engineering.

A survey of quality implementation in 100 different companies (Bureau of Business Practices, 1990, 1991) showed that the single most important factor for success was a *common vision* of:

- what the system was supposed to do
- who the customer was
- the implementation plan for achieving the goals.

The role of the Systems Engineer is to make the system work in an effective manner within the constraints of schedule, budget and resources. The Systems Engineering Group, by virtue of its contacts across the specialty disciplines, is charged with **communicating the vision** or concept of the system. The Systems Engineering Group has the responsibility for the global technical design, optimal implementation and proper verification of the system over the whole SDLC life-cycle. Systems Engineers have to understand and communicate:

- what the system does
- how the system works
- how and why the system was designed
- the implications of changes
- the boundary conditions
- the benefit of the system
- the risks inherent in the system.

This means *the Systems Engineer is a generalist*. Think of the Systems Engineer as the project expert who ensures that the process is optimally planned and implemented during the course of the project life cycle. Systems Engineering is a learned skill based on (Todaro 1988.):

- Design, development, and test **experience** gained at the unit, module and subsystem levels.
- Analysis and modeling **experience**.
- Reliability Maintainability & Availability (RMA) and other specialty activity **experience**.
- **Knowledge** of past, current, and future technology and techniques.
- Cost and time estimating **skills**.
- Pragmatic decision making **skills**.
- Correct and timely implementation and control **skills**.

Systems Engineers are the primary communications interface between the developers and the customers in the initial formative phase of the SDLC. During the remaining phases of the SDLC, they take part in the activities shown in Table 1, produce documentation and reviews, *communicate the vision* and coordinate the process. This broad spectrum of activities means that Systems Engineers need a working knowledge of the different disciplines to communicate with the specialists who perform them. This sensitization is gained by both education and experience. To remain effective, the Systems Engineer must continually and constantly improve his or her skills through education and experience. This is an example of *continual process improvement*.

INEFFECTIVE SYSTEMS ENGINEERING

If effective Systems Engineering consists of optimizing the process and product, then ineffective Systems Engineering results from:

- Failing to understand the customer's requirements
- Failing to perform Systems Engineering with skilled Systems Engineers
- Failing to communicate and maintain the vision
- Failing to perform adequate specialty engineering
- Failing to apply lessons learned from previous projects
- Failing to plan ahead to ensure resources are available when needed
- Failing to document the reasons for decisions
- Failing to use a Systems Engineering methodology that seamlessly interfaces to the software development methodology
- Failing to control changes.

Table 1 - System Engineering Activities

Definition of needs/goals/objectives	Organizing
Operational scenarios analyses	Directing junior level personnel
System Requirements analysis	Program and decision analysis
System Requirements allocation	Risk analysis
Functional Analysis	Integrated logistics support
Functional Allocation	Transition planning
Specification development	Reliability Maintainability & Availability
System and subsystem design	Integration
Trade off/alternatives evaluation	Test and evaluation
Development of benchmark tests	Configuration management
Software Requirements analysis	Quality assurance
Hardware analysis and recommendations	Training
Interface definition and control	Technical Writing
Schedule development	Installation
Life cycle costing	Operations support
Technical Performance Measurement	System evaluation and modification
Planning	

CONSEQUENCES OF INEFFECTIVE SYSTEMS ENGINEERING

Consider the consequences of each of the failure listed above.

The major consequence of failing to understand the customer's requirements is an unhappy customer.

The major consequences of failing to communicate and maintain the vision are complete project failure and cancellation, or initial failure then recovery. Without a clear vision, different sections of the project proceed in different directions at different rates.

The major consequence of failing to perform System Engineering with skilled Systems Engineers is that the problem tends to be posed in terms of a solution optimized for the expertise responsible for the system design. For example, when hardware expertise is used, it tends to purchase some brand new hardware that looks like a neat thing on which to base a project, but is later found to be unsuitable. When software expertise is used, it is so easy to make changes in software that there is a propensity for the following to occur:

- failure to recognize the need to plan ahead, and thus
- failure to set up objectives, and consequently,
- to end up in a continual crisis management situation.

The major consequences of failing to perform adequate specialty engineering are unexpected problems that cause schedule delays and potential costly redesigns.

The major consequence of failing to apply lessons learned from previous projects is that mistakes are repeated and paid for on the current project, with accompanying cost escalation and schedule delays.

The major consequences of failing to plan ahead to ensure resources are available when needed are schedule delays, high pressure and crisis management which leads to high corporate visibility.

The major consequence of failing to document the reasons for decisions is that decisions are questioned throughout the design, implementation and test phases.

The major consequences of failing to use a Systems Engineering methodology that seamlessly interfaces to the software development methodology are delays and errors in translation from the system requirements to the design phase.

The major consequences of failing to control changes are moving baselines and confusion leading to cost escalation and schedule delays.

The consequences of ineffective Systems Engineering at any stage of a project are cost escalations and schedule delays throughout the remainder of the project. Visualize a typical GANTT Chart for a project. The total cost of the project is the sum of the costs of each line item. Each line item is the sum of its elements in a work breakdown structure. As each item is completed, the cost account is closed out accordingly. However, if a task is performed in an ineffective manner, time is wasted in subsequent tasks with corresponding unplanned cost escalations. Eliminating ineffective Systems Engineering is akin to fixing defects in a manufacturing process.

The major by-products produced by Systems Engineers on their way to delivering the system are presentations and documentation. Ineffective Systems Engineers produce flawed or defective documentation. As an example of the cost of defective documentation, consider just the formal and informal meetings on a typical large project resulting from trying to interpret a single defective document. If you multiply the time spent in these meetings, by the number of meetings and the numbers of attendees, *the unplanned labor cost of these meetings can very quickly reach $500,000 or so* over the course of the project. Now multiply that by the number of

defective documents in a large, and think about the effect on the project budget and schedule.

PROACTIVE SYSTEMS ENGINEERING

The proactive Systems Engineering process is the means to gain the competitive edge. The system concept has to be developed, adequately documented and communicated to everyone working on the team as well as representatives from the customers, operators and users. In this way, people will know where their piece fits, and will have a sense of contributing to the system. Optimizing the initial design is the first characteristic of this process. This is akin to Value Engineering during a product design. Once an optimal design is achieved, it must be implemented. The major mitigating technique for ineffective Systems Engineering at this second phase of the process is called "*communicating the vision*".

The documents used for communicating the vision to all people who interact with the system are:

- the Systems and Operations Concept Document (SOCD), and
- the Systems Engineering Management Plan (SEMP).

These documents provide the answers to the generic questions, who, what, where, when, why and how; *are critical to the success of a project* and must be tailored as applicable to the complexity of the project. Both documents are initially written early in the system planning phase to clarify and organize the systems' objective and constraints. Subsequently as the system evolves and requirements and constraints change, each document also has to be updated for each major milestone to keep the vision current. All subsequent project documents are driven from the SOCD and the SEMP. If the vision of the SOCD and SEMP is defective, the subsequent documentation will be defective and time will be wasted:

- determining what should have been in the SOCD/SEMP while producing the subsequent documentation, and then
- incorporating the information in lower level documents.

THE SYSTEMS AND OPERATIONS CONCEPT DOCUMENT

The SOCD describes the system in each phase of its life cycle. The SOCD:

- states the objectives of the system
- contains a general discussion of how those objectives will be met
- identifies the users of the system
- states how the users interact with the system
- describes the end-to-end functional flows through the system
- defines the interfaces with external entities and describes how they interact
- describes how the system operates by means of normal and contingency operations scenarios.

During the planning and design phases, the SOCD:

- drives the system requirements
- guides the design
- drives the design
- describes the design.

During the development phase, the SOCD:

- keeps the design focussed on the operational needs
- assists in the resolution of design tradeoffs
- provides the basis for developing test plans and procedures
- provides the basis for operator training.

THE SYSTEMS ENGINEERING MANAGEMENT PLAN

The SEMP describes the resources and technology needed to implement the system and when they will be needed (schedule). The SEMP is a comprehensive work plan and describes how the fully integrated program engineering effort will be managed and conducted. The SEMP consists of:

- Technical Program Planning and Control
- Systems Engineering Process
- Engineering Specialty Integration

The Technical Program Planning and Control portion of the plan identifies organizational responsibilities and authority for Systems Engineering management. It describes the method of controlling the program, subcontracted engineering, and schedules. It contains the contract work breakdown structure (WBS), the specification tree that relates to the WBS, program risk analysis, system test planning, the decision and control process, and Technical Performance Measurement (TPM).

TPM compares actual performance with planned performance. It is supposed to detect or predict problems which require management attention, and support the assessment of the impact of changes on the program. A basic assumption of this approach is that the design is adequate to meet the technical requirements. Designs are complete when they meet the specifications. Further changes in performance are to be negotiated (contractually) as improvements, and not incorporated

into the design since any change has some impact on the schedule. Designs must not be changed due to cost and schedule reviews, the only reason for a design change is to solve a technical problem. *Design changes must always be documented* and the documentation must be complete but sparing.

The Systems Engineering Process section of the plan contains details of the process to be used during the development program. It describes the general process was tailored to the specifics of the project. It also contains information regarding procedures, document-ation, methodology of trade-off studies, details about the models to be used for system cost effectiveness evaluation, and the generation of specifications.

The Engineering Specialty section of the plan shows how the engineering specialties involved apart from hardware and software design and production, are integrated into the overall effort. Where these disciplines overlap, the SEMP defines the responsibilities and authorities of each. *The SEMP also contains guidance for the trade off to be made in the event of a conflict between the different engineering disciplines*.

SUMMARY

This paper has described the consequences of ineffective Systems Engineering and suggested a way to mitigate those consequences. By avoiding those consequences, Systems Engineering can be performed in a cost effective manner. In tomorrow's marketplace, those companies that provide systems on-time and within budget will be the ones to survive.

REFERENCES

Blanchard, Benjamin S. and Fabrycky, Wolter J., *Systems Engineering and Analysis*, Prentice Hall, 1981.

Eisner, Howard, *Computer Aided Systems Engineering*, Prentice Hall, 1988.

DeMarco, Tom and Lister, Timothy, *Peopleware*, Dorset House Publishing Company, 1987.

Excellence Achieved, Customer Service, 1990, Blueprints for Action from 50 Leading Companies, Bureau of Business Practices.

Quality Excellence Achieved, Quality Assurance, 1991, Blueprints for Action from 50 Leading Companies, Bureau of Business Practices.

Todaro, R.C., Lecture Handout, ENEE 648R, University of Maryland, 1988.

AUTHOR'S BIOGRAPHIES

Joe Kasser has spent the last 20 years applying TQM to Systems Engineering resulting in the achievement of the cost effective implementation of international and domestic aerospace, communications and solar power systems. He is a recipient of NASA's Manned Space Flight Awareness (Silver Snoopy) Award for quality and technical excellence. He is also an ICPM Certified Manager and a recipient of ICPM's 1993 Distinguished Service Award. Parts of this paper were developed for his forthcoming book on "*Applying TQM to Systems Engineering*", and his proposed Doctoral dissertation on the same topic at The George Washington University.

Robin Schermerhorn has spent the last 12 years applying Software and Systems Engineering concepts on large communications software systems. She is a recipient of NASA's Manned Space Flight Awareness Award (Launch Honoree) for verification of NASA Communications systems. Parts of this paper were developed for her Master of Science in Computer Systems Management project at the University of Maryland.