SAND95-1776C

What is Systems Engineering?

A consensus of opinions
of senior Systems Engineers
compiled by A. Terry Bahill
Systems and Industrial Engineering
University of Arizona

Systems Engineering is an interdisciplinary process that ensures that the customers' needs are satisfied throughout a system's entire life cycle. This process includes

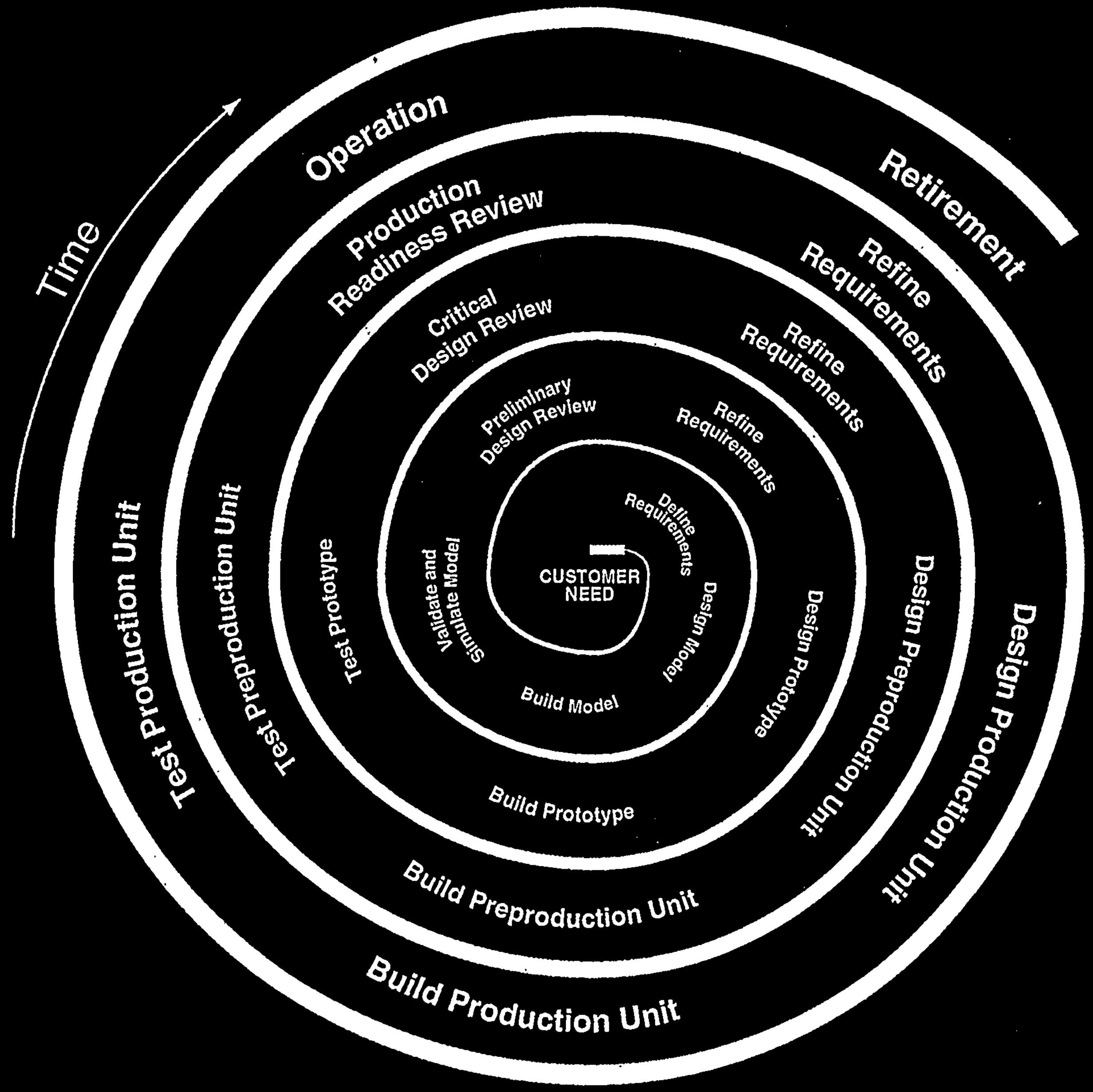
understanding customer needs,
stating the problem,
specifying requirements,
defining performance and cost measures,
prescribing tests,
validating requirements,
conducting design reviews,
exploring alternative concepts,
sensitivity analyses,
functional decomposition,
system design,
designing and managing interfaces,
system integration,
total system test,
configuration management,
risk management,
reliability analysis,
total quality management,
project management,
and documentation.

Material for this paper was gathered from senior Systems Engineeris at Sandia National Laboratories and also the following references: Blanchard and Fabrycky (1990); Sage (1992); Chapman, Bahill and Wymore (1992); Wymore (1993); IEEE P1220 (1994); Grady (1994); Hughes Aircraft Company (1994); Martin-Marietta (1994); Proceedings of IEEE SMC International Conference (1994); Proceeding of NCOSE Conference (1994); Grady (1995).

An early version of this paper was published in the *IEEE Systems, Man, and Cybernetics Newsletter* December 1994, pp. 11-12. The most recent version is available via anonymous ftp on tuson.sie.arizona.edu at pub/WhatIsSystemsEngineering. Comments are welcome at terry@sie.arizona.edu.

This work was supported by the United States Department of Energy under Contract DE-AC04-94AL85000.

The System Design Process



The system life cycle.
The system life cycle has seven phases: (1) requirements development, (2) concept development, (3) full-scale engineering design and development, (4) manufacturing and deployment, (5) system integration and test, (6) operation, maintenance and modification, and (7) retirement, disposal and replacement. However, the system life cycle is different for different industries, products and customers. Chapman, Bahill and Wymore (1992); Wymore (1993); Kerzner (1995).

Understanding customer needs.
Customers may not be aware of the details of what they need. Systems Engineers must enter the customers' environment, discover the details and explain them. Flexible designs and rapid prototyping help identify details that might have been overlooked. Talking to your customer's customer and your supplier's supplier can be very useful.

Stating the problem.
This is one of the System Engineer's most important tasks. An elegant solution to the wrong problem is less than worthless.

The word optimal should not appear in the statement of the problem, because Systems Engineers do not try to design and build optimal systems. Most system designs have several performance and cost criteria. Systems engineering creates a set of alternative designs that satisfy these performance and cost criteria to varying degrees. Moving from one alternative to another will improve at least one criteria and worsen at least one criteria, i.e. there will be trade-offs. None of the feasible alternatives is likely to optimize all the criteria (Szidarovszky, Gershon and Duckstein, 1986). Therefore, we must settle for less than optimality.

It might be possible to optimize some subsystems, but when they are interconnected, the overall system will not be optimal. The best possible system is not that made up of optimal subsystems. An all star team might have the optimal people at all positions, but is it likely that such an all star team could beat the world champion team in basketball or football?

If the system requirements demanded an optimal system, tests could not be designed to prove that any resulting system was indeed optimal. In general, it can be proven that a system is at a local optimum, but it cannot be proven that it is at a global optimum.

Specifying requirements.
There are two types of system requirements: mandatory and preference. Mandatory requirements insure that the system satisfies the customer's operational need. Mandatory requirements (1) must specify the necessary and sufficient conditions that a minimal system must have in order to be acceptable (They are usually written with the words *shall* and *must*.), (2) must be passed or failed, there is no middle ground (i.e. They must not use scoring functions.), and (3) must not be susceptible to trade-offs between requirements. Typical mandatory requirements might be of the following form: "The system shall not violate federal, state or local laws." Mandatory requirements state the minimal requirements necessary to satisfy the customer's need.

After understanding the mandatory requirements, Systems Engineers propose alternative candidate designs, all of which satisfy the mandatory requirements. Then the preference requirements are evaluated to determine the "best" designs. The preference requirements (1) should state conditions that would make the customer happier (They are often written with the words *should* and *want.*), (2) should use scoring functions (Chapman, Bahill and Wymore, 1992) to produce figures of merit, and (3) should be evaluated with a multicriteria decision aiding technique (Szidarovszky, Gershon and Duckstein, 1986), because none of the feasible alternatives is likely to optimize all the criteria and there may be trade-offs between these requirements. Typical preference requirements might be of the following form: "The system should have high performance and low cost: the performance and cost figures of merit will be weighted equally." Sometimes there is a relationship between mandatory and preference requirements, e.g. the mandatory requirement could be a lower threshold value for a preference requirement.

The words optimize, maximize and minimize should not be used in stating requirements, because we could never prove that we were there. Consider the following criteria: (1) we should minimize human suffering, and (2) we should maximize the quality and quantity of human life. A starving child should be fed, even if the child continues to live in misery. However, the criteria of minimal suffering could lead to the conclusion that the child should die.

Quality function deployment (QFD) can help identify system requirements. Grady (1992); Bahill and Chapman (1993); Bicknell and Bicknell (1994).

Defining performance and cost measures.
A technical performance measurement, often called a performance figure of merit, describes the result of a test, e.g., "In this test that car accelerated from 0 to 60 in 6.5 seconds." Such measurements are made throughout the evolution of the system: based first on estimates by the design engineers, then on models, simulations, prototypes and finally on the real system.

Prescribing tests.
Early in the system life cycle Systems Engineering should describe the tests that will be used to prove compliance of the final system with its requirements.

Validating requirements.
Validating requirements means ensuring that the requirements are consistent and that a real-world solution can be built and tested to prove that it satisfies the requirements. If Systems Engineering discovers that the customer has requested a perpetual-motion machine, the project should be stopped.

Conducting design reviews.
After the system model has been simulated and validated the requiremnents are reanalyzed and reformulated. This is called a preliminary design review. After the prototype has been validated the requirements are again reformulated. This is called the critical design review. After the preproduction unit has been validated the requirements are again revised in the preproduction design review.

Exploring alternative concepts.
Alternative designs should be proposed. Multicriteria decision aiding techniques should be used to reveal the *best* alternatives based on performance and cost figures of merit. This analysis should be redone whenever more data are available. For example, figures of merit should be computed initially based on estimates by the design engineers, Then models should be

constructed and evaluated. Next simulation data should be derived. Subsequently prototypes should be measured and finally tests should be run on the final system. For the design of complex systems, alternative designs reduce project risk.

Sensitivity analyses.
A sensitivity analysis can be used to point out the requirements that have the largest effects on determining the best alternatives. They are used to help allocate resources. Karnavas, Sanchez and Bahill (1993).

Functional decomposition.
Systems engineers do functional decomposition on new systems (1) to map functions to physical components, thereby ensuring that each function has an acknowledged owner, (2) to map functions to system requirements, and (3) to ensure that all necessary tasks are listed and that no unnecessary tasks are requested. This list becomes the basis for the work breakdown structure.

When analyzing an existing system, or re-engineering an existing system, Systems Engineers do functional analysis to see what the system does in order to improve its performance (often called value engineering), and they also do functional decomposition to see what the system is supposed to do. In this manner they can describe the present state of the system and the desired (or goal) state of the system. They can then suggest how the system design can be changed. Making radical dramatic changes in the system is called re-engineering. Making small incremental changes is called total quality management.

Icarus, and many flight wanna-bes after him, tried to understand how to fly by analyzing the physical components that birds used to fly:

Legs,
Eyes,
Brain, and
Wings.

Using this paradigm man was not able to fly. The Wright brothers, in contrast, identified the following functions for the flight problem:

Takeoff and land,
Sense position and velocity,
Navigate,
Produce horizontal thrust, and
Produce vertical lift.

Once it was understood that thrust and lift were two functions, two physical components could be assigned to them. By using a propeller to produce thrust and wings to produce lift, manned flight was possible. The following table shows a mapping of functions to physical components.

Function	Airplane Physical Component	Bird Physical Component
Takeoff and land	Wheels, skis or pontoons	Legs
Sense position and velocity	Vision or radar	Eyes
Navigate	Brain or computer	Brain
Produce horizontal thrust	Propeller or jet	Wings
Produce vertical lift	Wings	Wings

Birds use one physical component for two functions: thrust and lift. Man had to use two physical components for these two functions.

Recently object-oriented analysis has been replacing function decomposition for re-engineering existing systems (Jacobson, Ericsson and Jacobson, 1995).

System design.
The overall system should be divided into subsystems of similar complexity. Reusability should be considered in creating subsystems. For new designs, subsystems should be created so that they can be reused in future products. For redesign, subsystems should be created to maximize the use of existing, particularly commercially available, products. Systems engineers must also decide whether to make or buy the subsystems, first trying to use commercially available subsystems. If nothing satisfies all the requirements, then modification of an existing subsystem should be considered. If this proves unsatisfactory, then some subsystems will have to be designed. Engineers designing one subsystem must understand the other subsystems that their system will interact with. Flexibility is more important than optimality. Hardware, software and bioware must be considered. Bioware applies to humans and other biological organisms that are a part of the system. For example, in designing a race track the horses or dogs are a part of the bioware. Facilities for their care and handling must be considered, as should provisions for education, human factors, and safety. These activities are called System Design for new systems and Systems Analysis for existing systems.

Designing and managing interfaces.
Interfaces between subsystems and interfaces between the main system and the external world must be designed. Subsystems should be defined along natural organizational units. When the same information travels back and forth among different subsystems a natural activity may have been fragmented. 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.

System integration.
System integration means bringing subsystems together to produce the desired result and ensure that the subsystems will interact to satisfy the customers' needs. End users and design engineers need to be taught to use the system with courses, manuals and training on the prototypes. Grady (1994).

Total system test.
The system that is finally built must be tested to see (1) if it is acceptable, and (2) how well it satisfies the preference requirements. A total system test was never done on the Hubble Telescope, in order to save money. As a result we paid $850,000,000 to fix the undetected system error.

Configuration management.
Configuration management (also called modification management) ensures that any changes in requirements, design or implementation are controlled, carefully identified, and accurately recorded. All stakeholders should have an opportunity to comment on proposed changes. Decisions to adopt a change must be captured in a database and reflected in system documentation. (This documentation is a time frozen design for costing, building, testing, etc.). All concerned parties must be notified of changes to ensure that they are all working on the same design. The phrase requirements tracking is now being used for an important subset of configuration management.

Risk management.
There are two types or risk: risk of project failure (due to cost overruns, time overruns or failure to meet performance specifications) and risk of harm (usually called personnel safety). A failure modes and effects analysis and risk mitigation must be performed. Project risk can be reduced by supervising quality and timely delivery of purchased items.

Reliability analysis.
Major failure modes must be analyzed for probability of occurrence and severity of occurrence.

Total quality management.
Everyone must continually look for ways to improve the quality of the system. Major tools used in this process include basic concurrent engineering, quality function deployment (QFD) and Taguchi's quality engineering techniques. Bicknell and Bicknell (1994).

Project management.
Project management is the planning, organizing, directing, and controlling of company resources to meet specific goals and objectives within time, within cost and at the desired performance level. Kerzner (1995).

Documentation.
All of these Systems Engineering activities must be documented in a common repository. The stored information should be location, platform, and display independent: which means any person on any computer using any tool should be able to operate on the fundamental data. Results of trade-off analyses should be included. The reasons for making critical decisions should be stated. Chapman, Bahill and Wymore (1992); Wymore (1993).

Creating systems engineers.
The traditional method of creating Systems Engineers was to select well-organized engineers with lots of common sense and let them acquire 30 years of diverse engineering experience. But recently these traditional Systems Engineers have written books and standards that explain what they do and how they do it. So now that the tools, concepts and procedures have been formalized, in four years of undergraduate education we can teach Systems Engineers who will have performance levels 80% that of the traditional Systems Engineers. The other 20% increment in performance will come with 20 years of experience.

References

ANSI/ASQC Q9000, Q9001, Q9002, Q9003, Q9004, American Society for Quality Control, Milwaukee, WI, 1994.

Bahill A.T. and Chapman, W.L., "A tutorial on quality function deployment," *Engr Management J,* 5(3):24-35, 1993.

1994 Malcolm Baldrige National Quality Award Criteria Pack, National Institute of Standards and Technology.

Bicknell, K.D. and Bicknell, B.A., *The Road Map to Repeatable Success: Using QFD to Implement Changes,* CRC Press, Boca Raton, 1994.

Blanchard, B.S. and Fabrycky, W.J., Systems Engineering and Analysis, Prentice-Hall, 1990.

Chapman, W.L., Bahill, A.T. and Wymore, W., *Engineering Modeling and Design,* CRC Press, Boca Raton, 1992.

Grady, J.O., *System Requirements Analysis,* McGraw Hill Inc., 1993.

Grady, J.O., *System Integration,* CRC Press, Boca Raton, 1994.

Grady, J.O., *System Engineering Planning and Enterprise Identity,* CRC Press, Boca Raton, 1995.

Hughes Aircraft Co., *Systems Engineering Handbook,* 1994.

Humans, Information and Technology, *Proceedings 1994 IEEE International Conference on Systems, Man, and Cybernetics,* San Antonio, October 2-5, 1994.

IEEE P1220 Standard for Systems Engineering, IEEE Standards Dept., NY, 1994.

Jacobson, I., Ericsson, M. and Jacobson, A., *The Object Advantage: Business Process Reengineering with Object Technology,* Addison-Wesley, New York, 1995.

Karnavas, W.J., Sanchez, P. and Bahill A.T., "Sensitivity analyses of continuous and discrete systems in the time and frequency domains," *IEEE Trans Syst Man Cybernetics,* **SMC-23:** 488-501, 1993.

Kerzner, H., *Project Management: a Systems Approach to Planning, Scheduling, and Controlling,* Van Nostrand Reinhold, New York, 1995.

Lamprecht, J.L., *Implementing the ISO-9000 Series,* Marcel Dekker, NY, 1993.

Martin-Marietta, *Systems Engineering Methodology Handbook* EPI 270-01, 1994.

MIL-STD-499B, Draft Military Standard for Systems Engineering, AFSC/EN, 1992. This standard was not signed by the Department of Defense. They said that government should not be in the business of writing standards. They said they will adopt standards written by professional societies.

NCOSE, *Systems Engineering Process Activities, A "How To" Guide,* Draft of a handbook by The National Council on Systems Engineering, 1994.

Sage, A.P., Systems Engineering, John Wiley & Sons Inc., 1992.

Systems Engineering: A Competitive Edge in a Changing World, *Proceedings of the Fourth Annual Symposium of the National Council on* Systems Engineering (NCOSE), August 10-12, 1994, San Jose.

Szidarovszky, F., Gershon, M. and Duckstein, L., *Techniques for Multiobjective Decision Making in Systems Management,* Elsevier Science Publishers, Amsterdam, 1986.

Wymore, W., *Model-Based Systems Engineering,* CRC Press, Boca Raton, 1993.