Seven systems engineering myths and the corresponding realities

Joseph E. Kasser
Visiting Associate Professor
Temasek Defence Systems Institute
National University of Singapore
Block E1, #05-05, 1 Engineering Drive 2, Singapore 117576
Telephone +65 6516 4604, Fax +65 6778 9656
Cell/mobile/hand phone +65 9776 7464
Email joseph.kasser@incose.org

"It ain't what you don't know that gets you into trouble. It's what you know for sure that just ain't so." - Mark Twain 1835-1910.

Abstract. This paper states that systems engineering is a discipline characterized by debates based on subjective opinions, with participants talking past each other, a lack of listening and a number of myths. The opinions expressed in this paper are based on some of the findings from research into the nature of systems engineering that began in 1994. The paper discusses seven myths of systems engineering and shows the nature of the myth and the reality, and explains how and why each myth arose.

INTRODUCTION

In the second session of the Academic Forum at the 2009 International Symposium in Singapore, the state of systems engineering as a discipline was compared to the state of:

- electrical engineering before Ohm's law was postulated,
- electrical engineering before Maxwell's equations were stated, when engineers built motors by winding coils but had no theory upon which to predict the behaviour of the motor before powering it up for the first time
- chemistry before the periodic table of elements was discovered, and
- medicine in the 1800's before medical science provided a theory of why some medications work and why some don't.

Namely systems engineering is in its early stages. A discipline in these stages is characterized by debates based on subjective opinions, with participants talking past each other, a lack of listening, contradictory and confusing information and a number of myths. This paper addresses some of those myths and the opinions expressed in this paper are based on findings from research into the nature of systems engineering that began in 1994. These partial findings are grouped herein as seven myths of systems engineering. The paper shows the nature of each myth, the reality, and explains how and why each myth arose. The myths discussed are:

- Myth 1: There are Standards for systems engineering.
- Myth 2: The "V" model of the systems engineering process
- Myth 3: Follow the systems engineering process and all will be well
- Myth 4: Complexity needs new tools and techniques
- Myth 5: Systems of systems are a different class of problem and need new tools and techniques
- Myth 6: Changing requirements are a cause of project failure so get the requirements up front.
- Myth 7: The systems engineering process.

Consider each myth and corresponding reality.

MYTH 1: THERE ARE STANDARDS FOR SYSTEMS ENGINEERING.

While (MIL-STD-499, 1969) (EIA 632, 1994) (IEEE 1220, 1998) and ISO/IEC 15288 (Arnold,

Table 1 Chronological focus of the Standards and the CMMI

SE Categories	MIL-STD-499C	ANSI / EIA 632	IEEE-1220	CMMI	ISO-15288
Conceptualizing problem and alternative solutions	No	No	No	No	No
Mission / purpose definition	No	No	✓	✓	✓
Requirements engineering	✓	✓	✓	✓	✓
System architecting	✓	✓	✓	✓	✓
System implementation	No	✓	No	✓	✓
Technical analysis	✓	✓	✓	✓	✓
Technical management / leadership	✓	✓	✓	✓	✓
Verification & validation	✓	✓	✓	✓	✓

2002) are commonly thought of as systems engineering standards, the reality is that the approved Standards used in systems engineering cover systems engineering management and the processes for engineering a system; that is they do not seem to actually apply to systems engineering. Thus:

- Mil-STD-499 covers systems engineering management (MIL-STD-499, 1969).
- Mil-STD-499A covers engineering management (MIL-STD-499A, 1974) dropping the word 'systems' from the title.
- The draft (MIL-STD-499B, 1993) and MIL-STD-499C (Pennell and Knight, 2005) Standards contain the words "systems engineering" in their titles but the Standards were never approved.
- ANSI/EIA-632 covers processes for engineering a system (ANSI/EIA-632, 1999).
- The IEEE 1220 Standard is for the application and management of the systems engineering process (IEEE 1220, 1998).
- The ISO/IEC 15288 Standard lists processes performed by systems engineers (Arnold, 2002) and hence may be considered as being applicable to the role of the systems engineer rather than to the activities known as systems engineering. In addition, many of the activities in ISO/IEC 15288 also overlap those of project management.

The lack of coverage of early stage systems engineering in the standards and the Capability Maturity Model Integration (CMMI) may be seen in Table 1 which contains data extracted from Table 5 in (Honour and Valerdi, 2006) and rearranged in chronological order (based on the issue date of MIL-STD-499, not the draft MIL-STD-499C since the contents of MIL-STD 499A and MIL-STD-499B don't differ from MIL-STD 499C in this respect). The top row in Table 1 has been added in this paper to show that MIL-STD 499 and ANSI EIA 632 do not cover the conceptual activities in the early stages of the system lifecycle. While the CMMI, the draft MIL-STD-499C Standard and ISO 15288 do address the mission/purpose definition activities to some extent they also do not cover the conceptual activities in the early stages of systems engineering process. This recognition also appeared in a survey of [the then] current systems engineering processes in (Bruno and Mar, 1997) and in (Fisher, 1996)'s list of the engineering and systems engineering activities assigned to the systems engineering organization/team based on the (MIL-STD-499B, 1993)/(EIA 632, 1994) Standards.

Studies have shown that the cost of a system is determined in its early stages. A typical example

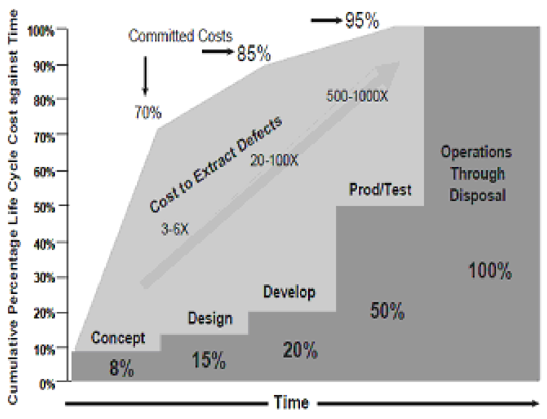

Figure 1 Committed lifecycle costs vs. time

shown in Figure 1 is a Defense Acquisition University study quoted in the INCOSE systems engineering handbook (Haskins, 2006) page 2.6 of 10). The figure shows that 70% of costs of a system are committed by activities in the early stage of systems engineering, yet the Standards ignore those early stages and so seem to be focused on the wrong end of the system lifecycle.

. The United States Department of Defense Architecture Framework (DODAF) (DoDAF, 2004) was designed to be used to "*provide correct and timely information to decision makers involved in future acquisitions of communications equipment*". Volume i contains 83 pages of definitions, guidelines, and background; volume ii contains 249 pages of product descriptions. The Deskbook contains 256 pages of supplementary information to framework users. The underlying data model comes with 696 pages and over 1200 data elements. The degree of micromanagement is phenomenal and expensive. Even a limited subset of the required information took 45,000 man-hours to produce (Davis, 2003). A chart mapping the degree of micromanagement in the standards over time (as measured by the thickness of the document) is shown in Figure 2 which roughly corresponds to the same curve as the cost to fix a defect as a function of the time the defect is discovered. As stated above, the early stages of systems engineering to the left of the vertical axis in Figure 2 is not covered by the standards. While (DOD 5000.2-R, 2002) pages 73-74) does call out some of the early stage activities, those activities are called out as part of the separate seemingly independent Cost as an Independent Variable (CAIV) process which takes place before the DOD 5000.2-R systems engineering process begins. CAIV is to be performed by Integrated Product and Process Development (IPPD) activities which involve organizing the different functions to work concurrently and collectively so that all aspects of the life cycle for the various concepts are examined and a balanced concept emerges (DOD IPPD, 1998). In broad terms, the objectives of the IPPD concept exploration phase are fourfold:

1. to perform concept studies to investigate different solutions,
2. to evaluate these different concepts,
3. to perform tradeoff studies, and
4. to define the requirements for the remainder of the acquisition program.

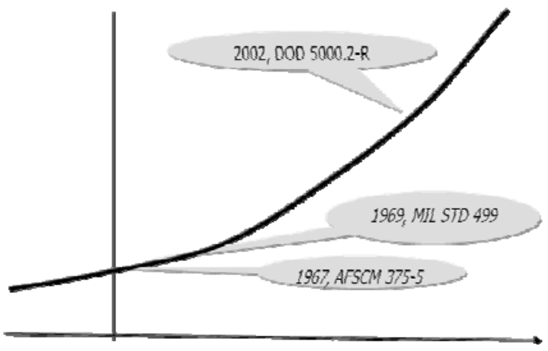

Figure 2 Degree of micromanagement in the US DOD Standards

So, the United States Department of Defense moved early stage systems engineering out of systems engineering into CAIV and the activities were to be performed by IPPD teams rather than by systems engineers. The DOD paradigm resulted in textbooks such as (Martin, 1997) page 95), (Eisner, 1997) page 9), (Wasson, 2006) page 60) which comply with (DOD 5000.2-R, 2002), pages 83-84) and consider requirements as one input to the systems engineering process.

Standards continue to appear yet we need to stop legislating processes, the micromanagement of processes and the production of lists of boxes to be ticked and start educating Type V systems engineers who can solve problems (Kasser, et al., 2009); see Myth 3.

MYTH 2: THE "V" MODEL OF THE SYSTEMS ENGINEERING PROCESS

The V diagram is often described as a depiction of the systems engineering process.

Consider the typical representation of the waterfall model shown in Figure 3 and representation of the V model in Figure 4 (Caltrans, 2007). Note that if the last two boxes in the waterfall are moved up to the corresponding levels as the first two boxes in the waterfall, the result is a V. The V is the waterfall just drawn differently!

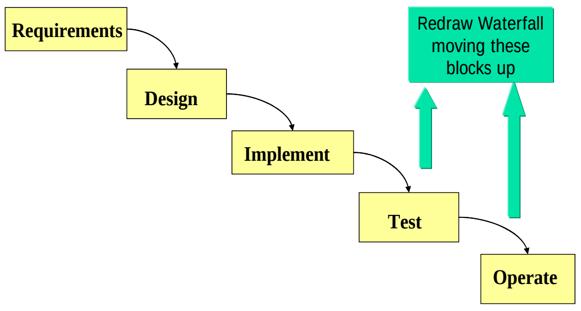

Figure 3 The Waterfall model

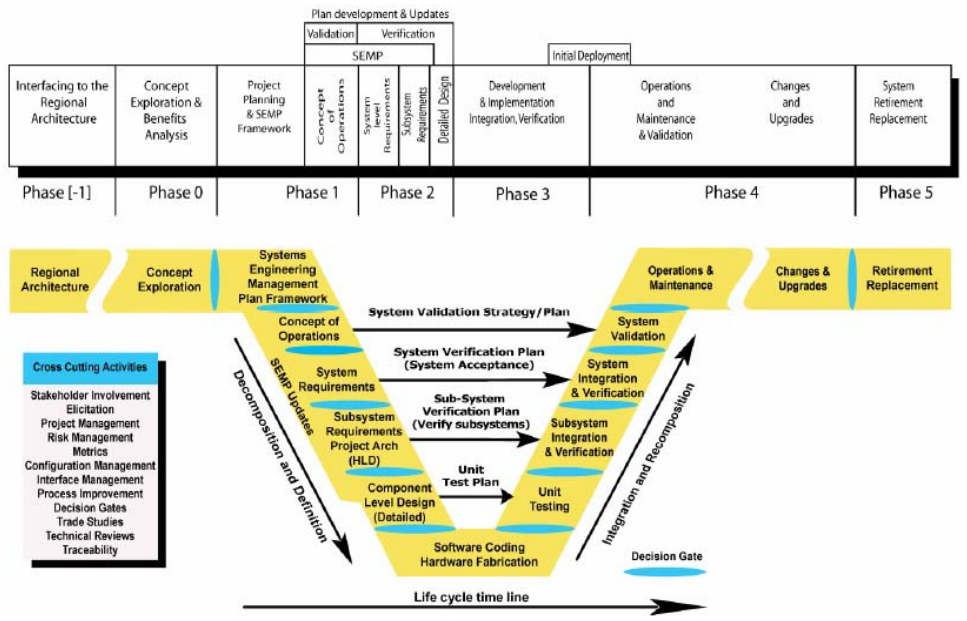
Plan development & Updates
Validation
Verification
SEMP
Initial Deployment
Interfacing to the Regional Architecture
Concept Exploration & Benefits Analysis
Project Planning & SEMP Framework
Concept of Operations
System level Requirements
Subsystem Requirements
Detailed Design
Development & Implementation Integration, Verification
Operations and Maintenance & Validation
Changes and Upgrades
System Retirement Replacement
Phase [-1]
Phase 0
Phase 1
Phase 2
Phase 3
Phase 4
Phase 5
Regional Architecture
Concept Exploration
Systems Engineering Management Plan Framework
Concept of Operations
System Requirements
Subsystem Requirements Project Arch (HLD)
Component Level Design (Detailed)
Software Coding Hardware Fabrication
Unit Testing
Subsystem Integration & Verification
System Integration & Verification
System Validation
Operations & Maintenance
Changes & Upgrades
Retirement Replacement
System Validation Strategy/Plan
System Verification Plan (System Acceptance)
Sub-System Verification Plan (Verify subsystems)
Unit Test Plan
SEMP Updates
Decomposition and Definition
Integration and Recomposition
Decision Gate
Life cycle time line
Cross Cutting Activities
Stakeholder Involvement
Elicitation
Project Management
Risk Management
Metrics
Configuration Management
Interface Management
Process Improvement
Decision Gates
Trade Studies
Technical Reviews
Traceability

Figure 1-2 ITS Project Life cycle Phases and the Life cycle Tasks in this Guidebook

Figure 4 Example of the V Model (Caltrans, 2007)

The reality is that the "V" is a rearranged waterfall (Royce, 1970) view for use as a management tool showing the relationship between design activities and test activities (Forsberg and Mooz, 1991). Practitioners tend to forget or are unaware that the V is a three dimensional view and in its two-dimensional representation it is only an overview of some of the aspects of the project cycle relating development to test and evaluation (T&E) at the various phases of the system lifecycle while abstracting out all other information.

A literature search in systems and software engineering found the first mention of the V diagram in (Rook, 1986) where it was introduced as a software project management tool illustrating the concept of verification of the process-products at established milestones. The original figure shown in Figure 5 was captioned "*the stages in software development confidence*". The figure was drawn to show that the intermediate process products produced at each phase of the software development were to be verified against previous baselines before starting work on the subsequent phase.

The V diagram seems to have been introduced to the systems engineering community by (Forsberg and Mooz, 1991) also as a project management tool not as a systems engineering tool. Both originators state that the simplistic view of the product development cycle is not to be interpreted as a waterfall namely that each phase is to be completed before the next begins. They agree that explanatory work on subsequent phases is often required before a phase is complete and there is a third dimension involved. (Forsberg and Mooz, 1991) include a representation of that third dimension in their paper and one of their figures, extracted from their paper is shown in Figure 6.

The dark side of the V

The use of the V view as a process model also perpetuates the following undesired issues.

- Lack of prevention of defects.
- Failure to consider changes to customer needs during development of the solution system.

Lack of prevention of defects. When the V diagram is used in a simplistic manner to depict the relationship between development and T&E there seems to be no place in the diagram for the prevention of defects. While the development team implements the system, the test team is busy planning the tests. A definition of a successful test is one that finds defects[1] (Myers, 1979). This is because if no defects are found, the result is ambiguous, because either there are no defects or the testing was not good enough to detect any defects. The lack of prevention of defects escalates costs. (Deming, 1986) page 29) wrote

[1] As opposed to the goal of the system development team which is to produce a defect free system.

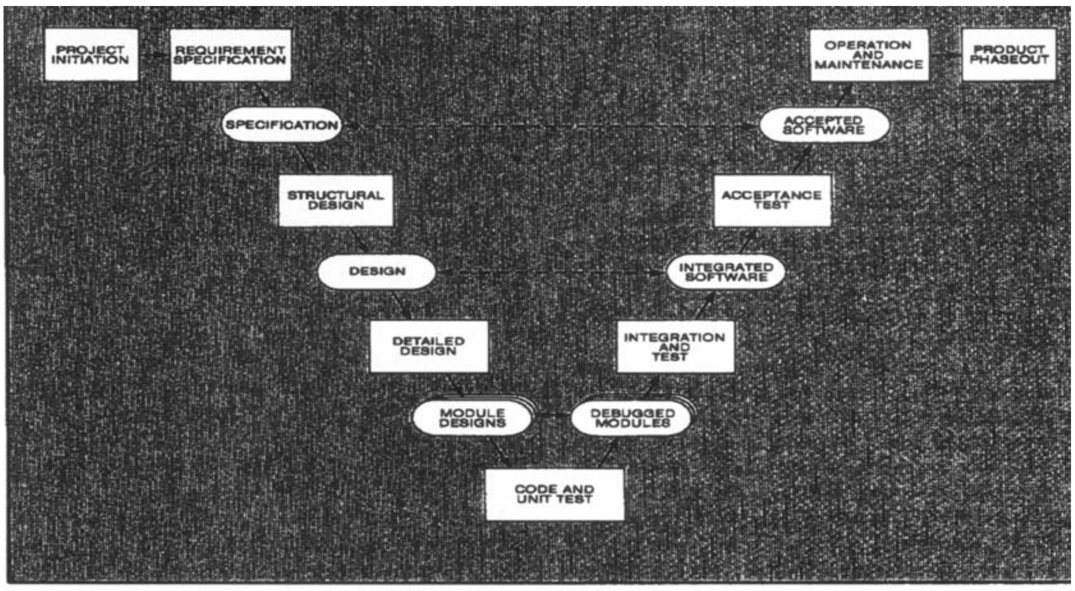

Fig. 3 The stages in software development confidence

Figure 5 The V diagram for Software Development (Rook, 1986)
(Black background in original figure)

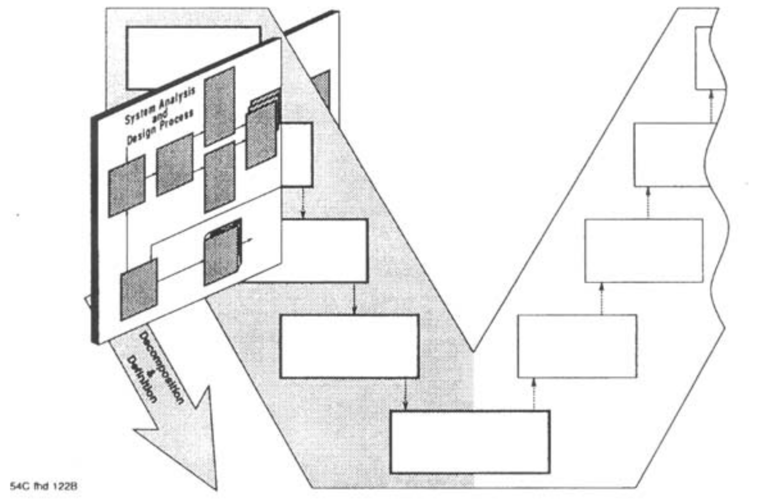
System Analysis and Design Process
Decomposition & Definition
54C fhd 122B

Exhibit 8—Application of the System Analysis and Design Process to the Technical Aspect of the Project Cycle

Figure 6 The three dimensions to the V diagram (Forsberg and Mooz, 1991)

“*Quality comes not from inspection, but from improvement of the production process*”. He also wrote “*Defects are not free. Somebody makes them, and gets paid for making them*” (Deming, 1986) page 11). If the test team can identify defects to test for, why can’t they hold a workshop or other type of meeting to sensitize the development team to those defects and hence prevent them from being built into the system? Such workshops in postgraduate courses at university of Maryland University Collage (1997-1999) and the University of South Australia (2000-2006) have sensitized students to the problems caused by poorly written requirements (Kasser, et al., 2003).

Failure to consider changes to customer needs during development of the solution system. The V is a redrawn waterfall and suffers from the same defect, namely lack of consideration of changes in customer needs. Some attempt however is sometimes made to include the effect of these changes by drawing two V’s in series.

MYTH 3: FOLLOW THE SYSTEMS ENGINEERING PROCESS AND ALL WILL BE WELL

According to the (United States Department of

Defense 5000 Guidebook 4.1.1), "*The successful implementation of proven, disciplined systems engineering processes results in a total system solution that is--*

- *Robust to changing technical, production, and operating environments;*
- *Adaptive to the needs of the user; and*
- *Balanced among the multiple requirements, design considerations, design constraints, and program budgets*".

The reality in the literature is that excellence comes from people not process. The much quoted (CHAOS, 1995) study fails to mention process or lack thereof as a major contribution to project success or failure.

The literature is full of advice as to how to make projects succeed; typical examples are (Peters and Waterman, 1982; Peters and Austin, 1985; Peters, 1987; Rodgers, et al., 1993; Harrington, 1995) which in general tend to ignore process and focus on people. Systems engineers focus on developing processes for organizations – namely the rules for producing products. Companies don't want employees who can follow rules; they want people who can make the rules (Hammer and Champy, 1993) page 70). The contribution of good people in an organization was recognized in the systems engineering literature about 50 years ago, namely "*Management has a design and operation function, as does engineering. The design is usually done under the heading of organization. It should be noted first that the performance of a group of people is a strong function of the capabilities of the individuals and a rather weak function of the way they are organized. That is, good people do a fairly good job under almost any organization and a somewhat better one when the organization is good. Poor talent does a poor job with a bad organization, but it is still a poor job no matter what the organization. Repeated reorganizations are noted in groups of individuals poorly suited to their function, though no amount of good organization will give good performance. The best architectural design fails with poor bricks and mortar. But the payoff from good organization with good people is worthwhile.*"(Goode and Machol, 1959) page 514). Excellence is in the person not the process. Again the focus should be on developing Type V engineer leaders (Kasser, et al., 2009) rather than on developing processes.

MYTH 4: COMPLEXITY NEEDS NEW TOOLS AND TECHNIQUES

Systems engineering has not delivered on its promise to meet the challenge of complexity as documented by Chestnut who wrote "*Characteristic of our times are the concepts of complexity, growth and change*" (Chestnut, 1965) page 1) and "*in a society which is producing more people, more materials, more things, and more information than ever before, systems engineering is indispensable in meeting the challenge of complexity*" (Chestnut, 1965) page vii). There is a growing dichotomy in the literature on the subject of complex systems. On one hand there is literature on the need to develop new tools and techniques to manage them and on the other hand, there is literature on techniques such as aggregation which mask the underlying complexity to ensure that only the pertinent details for the particular situation to deal with the issues are considered. Examples from each side of the dichotomy found in a literature review of complexity in the systems engineering field are:

- (Jenkins, 1969) who defined systems engineering as '*the science of designing complex systems in their totality to ensure that the component subsystems making up the system are designed, fitted together, checked and operated in the most efficient way*".
- (Maier and Rechtin, 2000) who recommend that the way to deal with high levels of complexity is to abstract the system at as high a level as possible and then progressively reduce the level of abstraction.
- (Bar-Yam, 2003) who proposed that "*complex engineering projects should be managed as evolutionary processes that undergo continuous rapid improvement through iterative incremental changes performed in parallel and thus is linked to diverse small subsystems of various sizes and relationships. Constraints and dependencies increase complexity and should be imposed only when necessary. This context must establish necessary security for task performance and for the system that is performing the tasks. In the evolutionary context, people and technology are agents that are involved in design, implementation and function. Management's basic oversight (meta) tasks are to create a context and design the process of innovation, and to shorten the natural feedback loops through extended measures of performance.*" Bar-Yam

- quoted the (CHAOS, 1995) study suggesting that the systemic reason for the challenged project is their inherent complexity. That might be one finding, however, the general finding from the Chaos study that the systemic reason for the challenged projects is poor management!
- Cited own prior work "f*or all practical purposes adequate testing of complex engineered systems is impossible*"
- Suggested evolutionary process for engineering large complex systems.

The reality is that there seem to be two types of complexity as follows:

- **Real world complexity -** in which elements of the real world are related in some fashion, and made up of components. This complexity is not reduced by appropriate abstraction it is only hidden.
- **Artificial complexity** – arising from either poor aggregation as (Maier and Rechtin, 2000) point out, or elements of the real world that, in most instances, should have been abstracted out when drawing the internal and external system boundaries, since they are not relevant to the purpose for which the system was created. It is this artificial complexity that gives rise to complication in the manner of Rube Goldberg or W. Heath Robinson[2]. For example, in today's paradigm, complex drawings are generated that contain lots of information[3] and the observer abstracts information as necessary from the drawings. The natural complexity of the area of interest is included in the drawings. Hence the system is thought to be complex.

Dealing with complexity means using abstraction and elaboration (Hitchins, 2003) pages 93-95) coupled with domain knowledge to develop an understanding of the situation, namely interrelationships among the system components and knowing which are pertinent to the situation and which can be safely ignored. For example, the space transportation system (space shuttle) and the international space station are both complex systems. However, when considering the problem of docking one to the other all aspects of the situation can be abstracted out except for the relative velocities, distance and alignments (yaw and pitch).

Perhaps the existence of the dichotomy is due to the observation that "the classification of a system as complex or simple will depend upon the observer of the system and upon the purpose he has for considering the system" (Jackson and Keys, 1984). Bar-Yam seems to drawing conclusions from poor engineering and management. He is correct in writing "that for all practical purposes adequate testing of complex engineered systems is impossible", However the continuum systems thinking perspective indicates that Bar Yam's statement only applies to the architectures in use today, there should be other architectures that would allow adequate testing. His suggestion for an evolutionary process has been applied to all types for systems since antiquity. The concept of establishing baselines and then using a "build a little, test a little" approach is well established in all areas of activity.

MYTH 5: SYSTEMS OF SYSTEMS ARE A DIFFERENT CLASS OF PROBLEM AND NEED NEW TOOLS AND TECHNIQUES

There is a dichotomy on the issue similar to the dichotomy on complexity. The earliest reference to system of systems was (Jackson and Keys, 1984) who wrote that a problem solver needs a methodology for [selecting the appropriate methodology for] solving a problem which has nothing to do with the use of the term in modern systems engineering.

(Allison and Cook, 1998) defined a system of systems as "*a system made up of elements that are not acquired or designed as a single system but are acquired over time and are in continuous evolution*" they categorized system of systems are permanent, such as airlines and national Defence forces, and temporary, ephemeral or virtual examples of such as multi-national peace keeping forces and project teams. (Cook, 2001) stated that "*the term system of systems in its permanent sense has come to mean a set of interdependent systems evolving at different rates, each at a different phase of their individual system lifecycles*". (Sillitto, 2008) stated that "*physically, a system of system looks just like a (big, spread-out) system with the following characteristics:*

- *Managerial and operational independence of the elements*
- *The elements have purpose and viability*

[2] Cartoonists in the USA and UK who drew cartoons of complicated systems designed to perform simple functions.

[3] DoDAF OV diagrams can be wonderful examples of complexity.

independent of the system of systems

- *procured asynchronously, different budgets*
- *Not necessarily specified to be compatible*
- *May be competing against each other for budget and resources*
- *Emergent properties created by action at a distance through sharing information,*
- *system of systems is continually operating (or ready to operate),*
- *Key attributes are agility and dependability,*
- *System projects must be integrated into the "live" system of systems during operations*."

This description would apply to the Allied convoys in the North Atlantic Ocean in World War II. Optimizing those convoys was a problem that was solved using Operations Research[4].

Other uses of the term "system of systems" describe an exploded view of a system containing several layers in the hierarchy of systems in a single drawing, where one person's subsystem is another person's system is another person's systems of systems depending on the viewpoint.

The reality. On the other side of the dichotomy there is recognition that systems exist within a hierarchy of systems in the context of adjacent systems and one person's system is another person's subsystem. The characteristics of systems of systems described above are the characteristics of systems in Layer 3 of Hitchins' five layers of systems engineering (Hitchins, 2000). The problems being addressed are those that Operations Research was set up to address in the 1940s and the tools and techniques exist and have existed for the last 50 years. Tools for systems engineering in the 1950s and 1960s were (Chestnut, 1965; Au and Stelson, 1969):

- Probability
- Single thread – system logic
- Queuing theory
- Game theory
- Linear programming
- Group dynamics
- Simulation
- Information theory

These tools were mainly used in the early stages of systems engineering. Since these early stages of systems engineering had been ignored in the standards, and the text books followed the standards, over time tools for systems engineering (Eisner, 1988; Jenkins, 2005) devolved to:

- PowerPoint
- Databases (e.g. DOORS and CORE)
- Word processors
- Spreadsheets
- Drawing tools (e.g. Visio)
- Etc.

The myth arose when systems engineers educated and practicing in the Layer 2 United States DOD systems engineering paradigm (DOD 5000.2-R, 2002) lacking the tools of the 1950s and 1960s attempted to tackle Layer 3 problems. Complexity is in the eye of the beholder (Jackson and Keys, 1984); yes, it is a new class of problem to the Layer 2 systems engineers, and no, current operations research tools and techniques that deal with "systems of systems" might need to be modified, but new tools do not need to be developed; such tools do indeed exist and have existed for more than 50 years.

MYTH 6: CHANGING REQUIREMENTS ARE A CAUSE OF PROJECT FAILURE SO GET THE REQUIREMENTS UP FRONT

The myth arose from (1) the failure to capture the entire problem/need and create the full set of matching specifications for the solution system in the early phases of systems engineering, and (2) overlooking the fact that requirements change continuously and failure to manage that change is the cause of project failure. There is thus confusion between the original uncaptured requirements and those requirements that arise due to changes.

The reality is that requirements may be categorized by those that exist at the time the solution system is specified and those that come into existence while the system is being realized. Definitely elicit and elucidate the known requirements in the early stage systems engineering activities. However, plan to use available tools and techniques such as configuration management, stage gates and engineering change processes to manage changes in requirements.

MYTH 7: THE SINGLE SYSTEMS ENGINEERING PROCESS

According to the (United States Department of Defense 5000 Guidebook 4.1.1), "*The successful implementation of proven, disciplined systems*

[4] Operational Analysis in the UK.

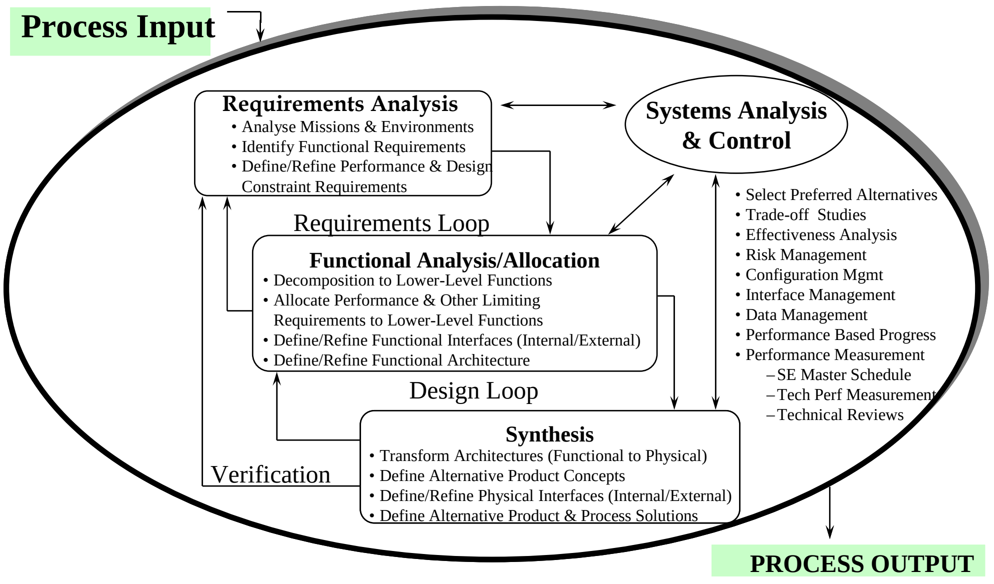
Process Input
Requirements Analysis
• Analyse Missions & Environments
• Identify Functional Requirements
• Define/Refine Performance & Design Constraint Requirements
Systems Analysis & Control
• Select Preferred Alternatives
• Trade-off Studies
• Effectiveness Analysis
• Risk Management
• Configuration Mgmt
• Interface Management
• Data Management
• Performance Based Progress
• Performance Measurement
– SE Master Schedule
– Tech Perf Measurement
– Technical Reviews
Requirements Loop
Functional Analysis/Allocation
• Decomposition to Lower-Level Functions
• Allocate Performance & Other Limiting Requirements to Lower-Level Functions
• Define/Refine Functional Interfaces (Internal/External)
• Define/Refine Functional Architecture
Design Loop
Synthesis
• Transform Architectures (Functional to Physical)
• Define Alternative Product Concepts
• Define/Refine Physical Interfaces (Internal/External)
• Define Alternative Product & Process Solutions
Verification
PROCESS OUTPUT

Figure 9 ANSI/EIA-632 Egg diagram

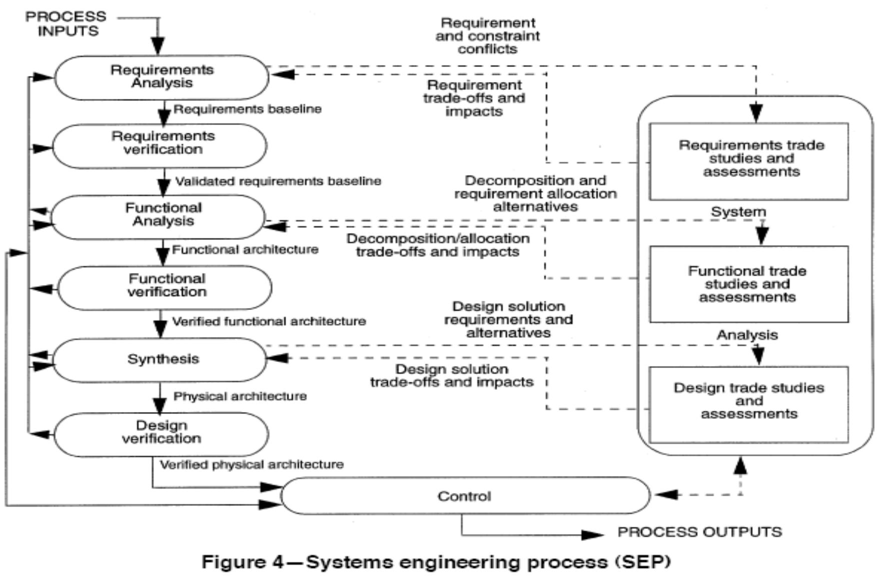
PROCESS INPUTS
Requirement and constraint conflicts
Requirements Analysis
Requirements baseline
Requirement trade-offs and impacts
Requirements verification
Validated requirements baseline
Decomposition and requirement allocation alternatives
Requirements trade studies and assessments
System
Functional Analysis
Functional architecture
Decomposition/allocation trade-offs and impacts
Functional trade studies and assessments
Functional verification
Verified functional architecture
Design solution requirements and alternatives
Analysis
Synthesis
Design solution trade-offs and impacts
Design trade studies and assessments
Physical architecture
Design verification
Verified physical architecture
Control
PROCESS OUTPUTS
Figure 4—Systems engineering process (SEP)

Figure 8 IEEE 1220 Systems Engineering Process

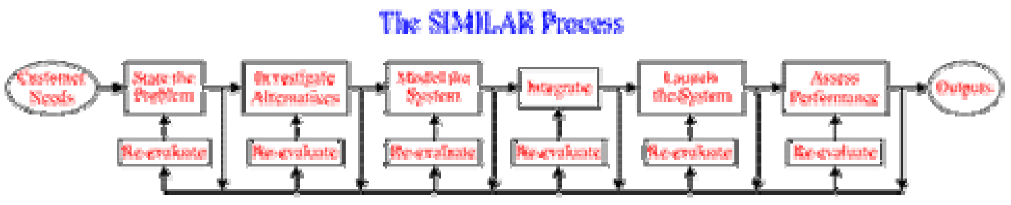

Figure 7 The SIMILAR process (Bahill and Gissing, 1998)

engineering processes results in a total system solution that is--

- *Robust to changing technical, production, and operating environments;*
- *Adaptive to the needs of the user; and*
- *Balanced among the multiple requirements, design considerations, design constraints, and program budgets*".

(Arnold, 2000) quotes (MIL-STD-499B, 1993) and (IEEE 1220, 1998) stating "*a single process,*

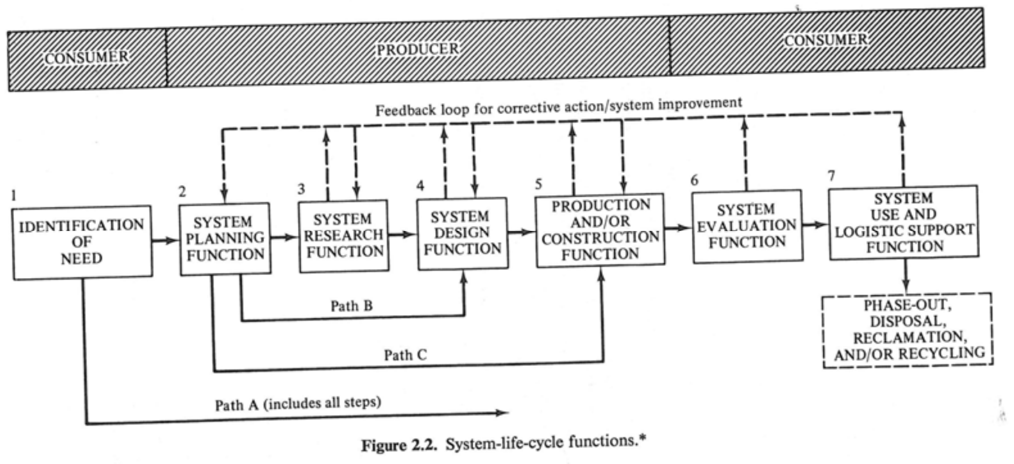
CONSUMER
PRODUCER
CONSUMER
Feedback loop for corrective action/system improvement
1 IDENTIFICATION OF NEED
2 SYSTEM PLANNING FUNCTION
3 SYSTEM RESEARCH FUNCTION
4 SYSTEM DESIGN FUNCTION
5 PRODUCTION AND/OR CONSTRUCTION FUNCTION
6 SYSTEM EVALUATION FUNCTION
7 SYSTEM USE AND LOGISTIC SUPPORT FUNCTION
PHASE-OUT, DISPOSAL, RECLAMATION, AND/OR RECYCLING
Path B
Path C
Path A (includes all steps)
Figure 2.2. System-life-cycle functions.*

Figure 10 System Lifecycle functions (Blanchard and Fabrycky, 1981)

standardizing the scope, purpose and a set of development actions, has been traditionally associated with systems engineering".

The reality is that there is no single widely agreed upon systems engineering process (SEP) since over the years, the SEP has been stated in many different ways, including:

- The (EIA 632, 1994) and (IEEE 1220, 1998) processes shown in Figure 9 and Figure 8;
- Lists of processes in ISO/IEC 15288 (Arnold, 2002);
- The waterfall process (Royce, 1970);
- The V model version of the SEP;
- The spiral process, incremental and evolutionary models;
- State, Investigate, Model, Integrate, Launch, Assess and Re-evaluate (SIMILAR) (Bahill and Gissing, 1998) shown in Figure 7;
- System Lifecycle functions as typically shown by (Blanchard and Fabrycky, 1981) in Figure 10;
- A systems engineering approach to addressing a problem (Hitchins, 2007).

Consequently, given the conflicting and contradictory information in the various versions of the SEP, the SEP concept is difficult to explain and, teaching has focused on using the waterfall and V models since they are simple to explain (Biemer and Sage, 2009) pages 152 and 153).

(Kasser and Hitchins, 2010) identify that from the big picture perspective, there seem to be two interdependent meta-SEPs, one for 'planning' and one for 'doing' or realizing the solution system:

The unique 'doing' SEP is constructed for the realization of a specific system. When designing the unique SEP for the realization of a system in the areas of the Hitchins-Kasser-Massie-Framework (HKMF) for understanding systems engineering (Kasser, 2007) to be inhabited by the unique SEP, systems engineers use knowledge based on experience and the activities functions and processes which can be found in the processes and Standards listed above and in the literature as building blocks. The activities to be performed in the unique SEP will depend on the work that has and has not been done at the point in the system lifecycle in which the process is constructed.

The second meta-SEP is the 'planning' process used by the systems engineer to create the unique SEP. Since this process is a problem solving activity, it ought to, and does, map into the problem solving process.

(Kasser and Hitchins, 2010) explain the conflicting and contradictory information in the various versions of the SEPs by viewing them as different unique subsets of the meta-SEPs appropriate to their situation. Consequently, there is a single SEP, but it is different for every system development project.

SUMMARY

This paper has stated that systems engineering is currently a discipline characterized by debates based on subjective opinions, with participants talking past each other, a lack of listening and a number of myths. The paper discussed seven myths of systems engineering and showed the nature of each myth and the reality, and explained how and why each myth arose.

CONCLUSION

This paper has documented some findings about the current state of systems engineering. These findings are based on research into the history and practice of systems engineering. The findings of the research should provide food for thought and assist educators to improve the teaching of systems engineering.

BIOGRAPHY

Joseph Kasser combines knowledge of systems engineering, technology, management and educational pedagogy. Having been a practicing systems engineer and engineering manager since 1970 in the USA, Israel, and Australia he brought a wealth of experience and a unique perspective to academia in 1997. He has since become internationally recognised as one of the top systems engineering academics in the world. He is an INCOSE Fellow, the author of “A Framework for Understanding Systems Engineering”, “Applying Total Quality Management to Systems Engineering” and many INCOSE symposia papers. He is a recipient of NASA’s Manned Space Flight Awareness Award (Silver Snoopy) for quality and technical excellence for performing and directing systems engineering and the recipient of many other awards, plaques and letters of commendation and appreciation. He holds a Doctor of Science in Engineering Management from The George Washington University, is a Certified Manager and a certified member of the Association for Learning Technology. He gave up his positions as a Deputy Director and DSTO Associate Research Professor at the Systems Engineering and Evaluation Centre at the University of South Australia in early 2007 to move back to the UK to develop the world’s first immersion course in systems engineering as a Leverhulme Visiting Professor at Cranfield University. He is an INCOSE Ambassador and also served as the initial president of INCOSE Australia and as a Region VI Representative to the INCOSE Member Board. He is currently a principal at the Right Requirement Ltd. in the UK and a Visiting Associate Professor at the National University of Singapore.

REFERENCES

Allison, J. S. and Cook, S. C., "*The New Era in Military Systems Thinking and Practice*", proceedings of First Regional Symposium of the Systems Engineering Society of Australia INCOSE Region 6 (SETE 98), Canberra, Australia, 1998.

ANSI/EIA-632, *Processes for Engineering a System*, American National Standards Institute and Electronics Industries Association, Arlington, VA, 1999.

Arnold, S., "*Systems Engineering: From Process towards Profession*", proceedings of The 10th Annual Symposium of the INCOSE, Minneapolis, MN, 2000.

Arnold, S. (Editor), *ISO 15288 Systems engineering — System life cycle processes*, International Standards Organisation, 2002.

Au, T. and Stelson, T. E., *Introduction to Systems Engineering Deterministic Models*, Addison-Wesley Publishing Company, 1969.

Bahill, A. T. and Gissing, B., "*Re-evaluating systems engineering concepts using systems thinking*", IEEE Transaction on Systems, Man and Cybernetics, Part C: Applications and Reviews, Vol. 28 (1998), no. 4, 516-527.

Bar-Yam, Y., "*When Systems Engineering Fails --- Toward Complex Systems Engineering*", proceedings of Systems, Man and Cybernetics, 2003. IEEE International Conference on, 2003.

Biemer, S. M. and Sage, A. P., "Systems Engineering: Basic Concepts and Life Cycle," in *Agent-Directed Simulation and Systems Engineering*, L. Yilmaz and T. Oren (Editors), Wiley-VCH, Weinheim, 2009.

Blanchard, B. B. and Fabrycky, W., *Systems Engineering and Analysis*, Prentice Hall, 1981.

Bruno, M. E. and Mar, B. W., "*Management of the Systems Engineering Discipline*", proceedings of the 7th Annual Symposium of the International Council on Systems Engineering, Los Angeles, CA., 1997.

Caltrans, "System Engineering Guidebook for Intelligent Transportation Systems Version 2.0," California Department of Transportation, Division of Research and Innovation, 2007, p. 313.

CHAOS, Chaos, 1995, http://www.standishgroup.com/chaos.html, last accessed March 19, 1998

Chestnut, H., *Systems Engineering Tools*, John Wiley & Sons, Inc., New York, 1965.

Cook, S. C., "*On the Acquisition of Systems of Systems"*, proceedings of the 11th annual Symposium of the INCOSE, Melbourne, Australia, 2001.

Davis, S., "*FCS System of Systems' Engineering and Integration"*, proceedings of NDIA Systems Engineering Conference, 2003.

Deming, W. E., *Out of the Crisis*, MIT Center for Advanced Engineering Study, 1986.

DOD 5000.2-R, "Mandatory Procedures for Major Defense Acquisition Programs (MDAPS) and Major Automated Information System (MAIS) Acquisition Programs," US Department of Defense, 2002.

DOD IPPD, *DoD Integrated Product and Process Development Handbook*, Office of the Undersecretary of Defense (Acquisition and Technology), Washington, DC., 1998.

DoDAF, *DoD Architecture Framework Version 1.0, 9 February 2004*, 2004.

EIA 632, "EIA 632 Standard: Processes for engineering a system," 1994.

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

Eisner, H., *Essentials of Project and Systems Engineering Management*, John Wiley & Sons, Inc., New York, 1997.

Fisher, J., "*Defining the Roles and Responsibilities of the Systems Engineering Organization/Team"*, proceedings of the 6th Annual Symposium of the International Council on Systems Engineering, Boston, MA, 1996.

Forsberg, K. and Mooz, H., "*The Relationship of System Engineering to the Project Cycle"*, proceedings of Annual Conference of the National Council on Systems Engineering, National Council on Systems Engineering, 1991.

Goode, H. H. and Machol, R. E., *Systems Engineering*, McGraw-Hill, 1959.

Hammer, M. and Champy, J., *Reengineering the Corporation*, HarperCollins, New York, 1993.

Harrington, H. J., *Total Improvement Management the next generation in performance improvement*, McGraw-Hill, 1995.

Haskins, C. (Editor), *Systems Engineering Handbook: A Guide for Life Cycle Processes and Activities, Version 3*, The International Council on Systems Engineering, 2006.

Hitchins, D. K., World Class Systems Engineering - the five layer Model, 2000, http://www.hitchins.net/5layer.html, last accessed 3 November 2006

Hitchins, D. K., *Advanced Systems Thinking, Engineering and Management*, Artech House, 2003.

Hitchins, D. K., *Systems Engineering. A 21st Century Systems Methodology*, John Wiley & Sons Ltd., Chichester, England, 2007.

Honour, E. C. and Valerdi, R., "*Advancing an Ontology for Systems Engineering to Allow Consistent Measurement"*, proceedings of Conference on Systems Engineering Research, Los Angeles, CA., 2006.

IEEE 1220, "Standard 1220 IEEE Standard for Application and Management of the Systems Engineering Process," 1998.

Jackson, M. C. and Keys, P., "*Towards a System of Systems Methodologies"*, Journal of the Operations Research Society, Vol. 35 (1984), no. 6, 473-486.

Jenkins, G. M., "The Systems Approach," in *Systems Behaviour*, J. Beishon and G. Peters (Editors), Harper and Row, London, 1969, p. 82.

Jenkins, S., "*A Future for Systems Engineering Tools"*, proceedings of PDE 2005, The 7th NASA-ESA Workshop on Product Data Exchange (PDE), 2005.

Kasser, J. E., *A Framework for Understanding Systems Engineering*, Booksurge Ltd, 2007.

Kasser, J. E., Hitchins, D. and Huynh, T. V., "*Reengineering Systems Engineering*", proceedings of the 3rd Annual Asia-Pacific Conference on Systems Engineering (APCOSE), Singapore, 2009.

Kasser, J. E. and Hitchins, D. K., "*Unifying the different systems engineering processes*", proceedings of Conference on Systems Engineering Research, Hoboken, NJ., 2010.

Kasser, J. E., Tran, X.-L. and Matisons, S., "*Prototype Educational Tools for Systems and Software (PETS) Engineering*", proceedings of Proceedings of the AAEE Conference, 2003.

Maier, M. K. and Rechtin, E., *The Art of Systems Architecting*, CRC Press, 2000.

Martin, J. N., *Systems Engineering Guidebook: A process for developing systems and products*, CRC Press, 1997.

MIL-STD-499, *Mil-STD-499 Systems Engineering Management*, United States Department of Defense (USAF), 1969.

MIL-STD-499A, *Mil-STD-499A Engineering Management*, United States Department of Defense (USAF), 1974.

MIL-STD-499B, "Draft MIL-STD-499B Systems Engineering," United States Department of Defense, 1993.

Myers, G., *The Art of Software Testing*, Wiley, 1979.

Pennell, L. W. and Knight, F. L., "Draft MIL-STD 499C Systems Engineering," The Aerospace Corporation, 2005.

Peters, T., *Thriving on Chaos: Handbook for a Management Revolution*, Harper and Row, 1987.

Peters, T. and Austin, N., *A Passion for Excellence*, Warner Books, 1985.

Peters, T. J. and Waterman, H. R., *In Search of EXCELLENCE*, Harper and Row, 1982.

Rodgers, T. J., Taylor, W. and Foreman, R., *No-excuses Management*, Doubleday, 1993.

Rook, P., "*Controlling software projects*", Software Engineering Journal, Vol. 1 (1986), no. 1, 7-16.

Royce, W. W., "*Managing the Development of Large Software Systems*", proceedings of IEEE WESCON, 1970.

Sillitto, H., "*Systems and System-of-Systems Architecting*", proceedings of DSTL Systems Skills Symposium, The Defence Science and Technology Laboratory (Dstl), Reading, 2008.

Wasson, C. S., *System Analysis, Design, and Development concepts, principles and practices*, Wiley-Interscience, Hoboken, New Jersey, 2006.