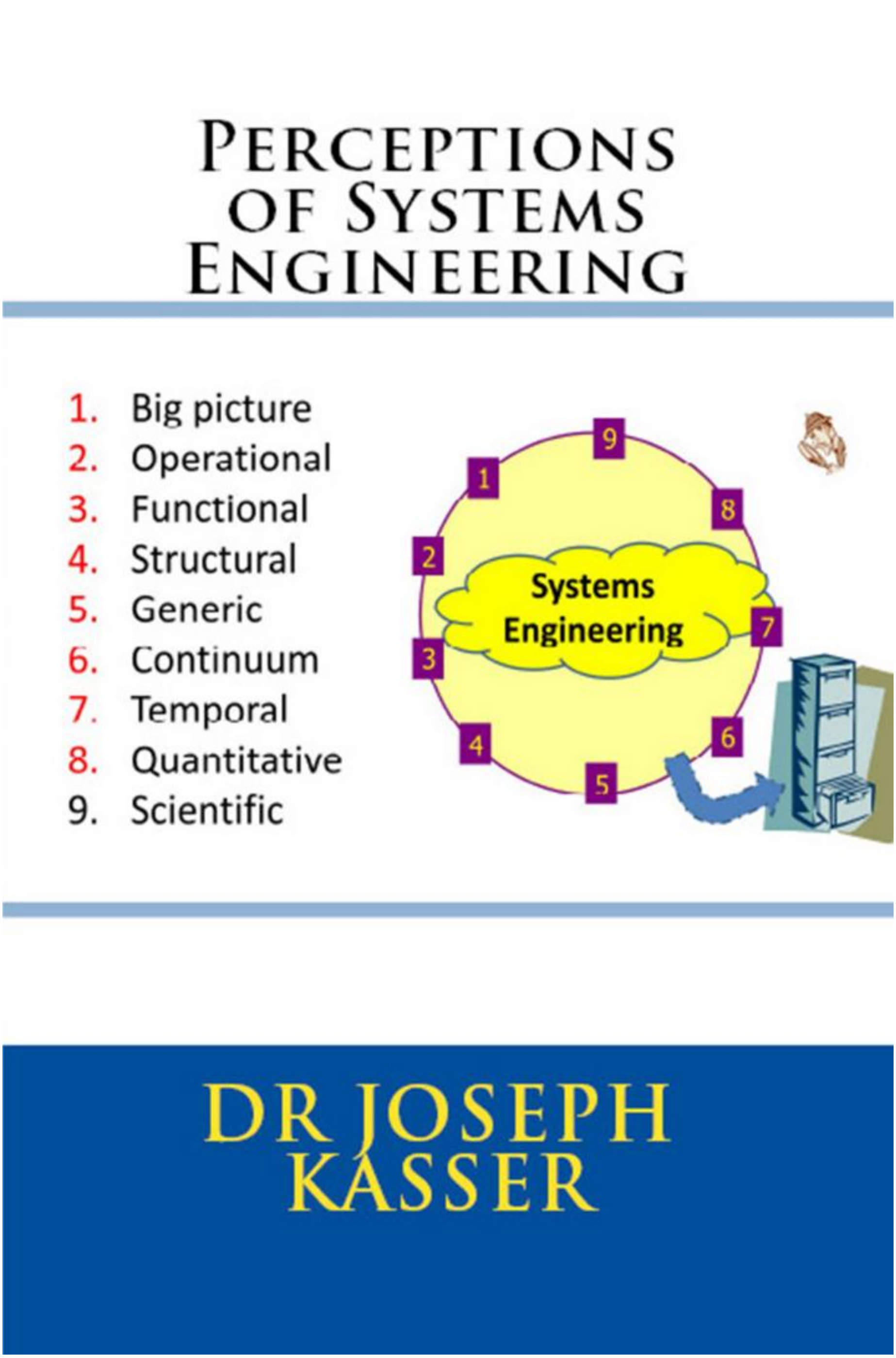
Desk copy


Perceptions of Systems Engineering
1. Big picture
2. Operational
3. Functional
4. Structural
5. Generic
6. Continuum
7. Temporal
8. Quantitative
9. Scientific
Systems Engineering
Dr Joseph Kasser

Perceptions of Systems Engineering

Dr Joseph E Kasser, CEng, CM,
FIET, FIES, CMALT, G3ZCZ

Published by The Right Requirement via Createspace
50 Crane Way, Cranfield, Bedfordshire, MK43 0HH, England
Visit the web site at http://therightrequirement.com
First Edition, Second Printing with revisions

ISBN-13: 978-1512257762
ISBN-10: 1512257761

Dedication

To my wife Lily, always caring, loving and supportive

Acknowledgments

This book would not have been possible without the co-authors of the papers upon which some of these chapters are based, and colleagues and friends who helped review the manuscript:

Eileen Arnold
Stephen Cook
Moti Frank
Derek Hitchins
Eric Honour
Horng Leong Lim
Chandru Mirchandani
Kent Palmer
Sharon Shoshani-Tavory
Xuan-Linh Tran
Willem A N van den Worm
Yang-Yang Zhao
Avigdor Zonnenshain

Other books by Joseph E. Kasser

Systems engineering

- A Framework for Understanding Systems Engineering, The Right Requirement Ltd. via Createspace, 2nd edition, 2013.
- Holistic Thinking: creating innovative solutions to complex problems, The Right Requirement Ltd. via Createspace, 2013.
- Applying Total Quality Management to Systems Engineering, Artech House, 1996.

Amateur radio

- Basic Packet Radio, Software for Amateur Radio, First and Second Editions, 1993, 1994.
- Software for Amateur Radio, TAB Books, December 1984.
- Microcomputers in Amateur Radio, TAB Books, November 1981.

Judaism and Jewish

- Conceptual Laws and Customs of Christmas, Createspace, 2015.
- The 87th Company. The Pioneer Corps. A Mobile Military Jewish Community, Createspace, 2015 (Editor).

About the author

Joseph Kasser has been a practicing systems engineer for 45 years and an academic for about 15 years. He is a Fellow of the Institution of Engineering and Technology (IET), an International Council on Systems Engineering (INCOSE) Fellow, the author of "*Holistic Thinking: creating innovative solutions to complex problems*", "*A Framework for Understanding Systems Engineering*" and "*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 other awards. He holds a Doctor of Science in Engineering Management from The George Washington University. He is a Certified Manager and holds a Certified Membership of the Association for Learning Technology. He has performed and directed systems engineering in the USA, Israel and Australia. 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 to the UK to develop the world's first immersion course in systems engineering as a Leverhulme Visiting Professor at Cranfield University. He is currently a Visiting Associate Professor at the National University of Singapore.

Contents

Figures

Tables

--oo--

PART I

1. Introduction

Twenty-first-century civilization increasingly depends on complex socio-technical systems. It is difficult to acquire and maintain these systems in a cost-effective and timely manner. There is a growing perception around the world that systems engineering is the best way to tackle this problem and this situation has caused an unprecedented demand for systems engineers. However, the literature on systems engineering seems to be confusing and contradictory raising a number of questions including:

1. What is systems engineering?
2. Why are there different opinions on the nature of systems engineering?
3. Why does systems engineering succeed at times?
4. Why does systems engineering fail at other times?
5. Why does systems engineering seem to overlap project management and problem-solving?
6. Why do the textbooks about systems engineering cover such different topics?
7. What do systems engineers actually do in the workplace?
8. Is systems engineering an undergraduate course or a postgraduate course?
9. Which come first, functions or requirements?
10. Why is there no standard definition of a system?

It took twenty years of research to achieve satisfactory answers to these questions. The research which delved into systems engineering, systems engineering tools, Operations Research, process improvement, project management, innovation and systems engineering's attempts to manage complexity produced a mass of semi-organised perceptions of, and insights about, systems engineering, that was published in a number of peer-reviewed publications from 1995 to 2015; the 1995 to 207 papers were updated and published as an anthology in a "*A Framework for Understanding Systems Engineering*", in 2007 and the second edition published in 2103 (Kasser, 2013b) was an anthology of the updated papers published between 1995 to 2013. Trying to find common threads in the research findings was a complex well-structured problem (Section 7.6) which was formulated using the problem formulation template discussed in Section 14.5 as follows:

- ***The undesirable situation*** was a mass of semi-organised perceptions of, and insights about, systems engineering documented in notes and published in journals and conference papers between 1994 and 2015.
- ***The Feasible Conceptual Future Desirable Situation (FCFDS)*** was:
  - The documented perceptions of systems engineering organised in a systemic and systematic manner (Section 2.1) that facilitates understanding and explaining the nature of systems engineering.
  - Being able to locate specific information in a speedy manner.
- ***The solution*** was this book in which the documented perceptions of systems engineering are stored in the Holistic Thinking Perspectives (HTP) (Section 2.2.2) with each perspective presented in a different chapter.
- ***The problem*** was to create the FCFDS.

When faced with a problem it is always useful to find out if anyone has faced the same or a similar problem and understand their approach to remedying their problem. Some research found that Mendeleev, when faced with the problem of making sense of the relationships between chemical elements and their properties, sorted the elements into a table. His contribution was to create a framework, the Periodic Table of Elements, and populate it with the known elements, leaving gaps which represented unknown elements. Using a similar approach to Mendeleev, perceptions of systems engineering were extracted from the publications, sorted and grouped into the HTPs (Section 2.2.2) using the rules discussed in Section 3.1.

This book is divided into the following parts:

- ***Part I*** contains this Chapter and Chapter 2 which:
  - Provides a brief overview of systems thinking and holistic thinking, summarizing the contents of "*Holistic Thinking: creating innovative solutions to complex problems*"; Volume 1 of the series (Kasser, 2013c).
  - Contains the definitions of the HTPs, which provide the framework template for the sorted stored information about systems engineering.
- ***Part II:***

  - Begins with Chapter 3 which describes the methodology used to store the perceptions of systems engineering in a systemic and systematic manner.
  - Contains the many perceptions and observations of systems engineering sorted into the descriptive perspectives in Chapters 4 to 11. The application of systems engineering is wide and the literature is broad, so the information in Chapters 4 to 11 should only be considered as being a representative sample, and should not be considered as complete.

This approach has advantages including:

  - Using the HTPs as a template so that the reader can readily identify specific types of information stored according to the rules in Section 3.1.
  - Separating the facts from the opinions. The facts are in Part II and the opinions begin in Part III.

  The approach does however suffer from the following disadvantages:

    - Information about a specific topic may be located in different chapters because the information is perceived from different perspectives.
    - There will be some forward references to information in a later section of the book.

  However an index is provided in Chapter 25[1] to facilitate locating information about specific topics.

- ***Part III*** is where the first set of ways to improve systems engineering are stored. These inferences and insights from the *Scientific* perspective begin in Part III and continue through Parts IV and V.
  - Chapter 12 contributes to the improvement of systems engineering by containing the insights, inferences and explanations from analysing the information in Chapters 4 to 11. This Chapter and the following chapters should invoke discussions and debates between systems engineers with different perspectives from single viewpoints of systems engineering.
  - Chapter 13 contributes to the improvement of systems engineering by perceiving the System Lifecycle (SLC) as a State Machine producing some innovative insights.

[1] Not in the eBook versions

  - Effective workmen sharpen their tools. Effective systems engineers not only sharpen their tools they are also always on the lookout of new tools that they can adopt or modify for their own use. Chapter 14 contributes to the improvement of systems engineering by containing a selection of tools and frameworks for improving the practice of systems engineering.
  - Chapter 15 contributes to the improvement of systems engineering by suggesting seven principles for the solution system.

- ***Part IV*** contributes to the improvement of systems engineering by using the insights, inferences and conclusions from Part III to suggest more tools and frameworks for improving systems engineering. Where the insight leads to a complex tool or concept, that complex tool or concept is presented in a separate chapter. As such:
  - Chapter 16 improves systems engineering by introducing the Nine-System Model. Note the Nine-System Model is not a model of systems engineering, it is a framework and tool.
  - Chapter 17 improves systems engineering by describing how to manage stakeholder expectations using a combination of the HTPs to identify the stakeholders, and the Nine-System Model to identify the stakeholders' areas of concern in the context of a Case Study.
  - Chapter 18 improves systems engineering by filling a gap in the systems engineering literature by suggesting a process for creating systems to be used in the early states of the System Development Process (SDP) to help to manage complexity at the time the system is created by optimizing the interfaces.
  - Chapter 19 improves systems engineering by providing a way to measure technical progress and identify potential problems in near real-time.

- ***Part V*** focuses on systems engineering education to produce better Case Studies for systems engineering education as a way to improve the practice of systems engineering.
  - Chapter 20 contributes to the improvement of systems engineering education by suggesting a template to improve the quality of Experiential Case Studies by providing a way for practitioners to link their experiences into the literature to provide information in a systemic and systematic manner to assist students studying systems engineering to locate infor-

mation and to help researchers improving the practice of systems engineering. Examples of the use of the template are included.

    - Chapter 21 contributes to the improvement of systems engineering education by introducing a multi-purpose Role-Playing Case Study for classes on systems engineering and engineering (project) management written in such a manner so as to provide additional examples of the tools, templates and frameworks described in Parts III and IV.

  These Chapters provide classroom exercises in postgraduate settings by asking the students to examine each inference or insight and then develop an argument to support or refute the inference or insight. The students will have to research the topic and apply critical thinking to develop their argument. It might also be useful and interesting in some situations to set up the exercise as a debate in which the students present the supporting and refuting arguments in the same session.

- ***Part VI*** concludes the book wherein:
    - Chapter 22 contains the summary and answers to the questions posed above.
    - Chapter 23 contains the glossary of acronyms used in the book.
    - Chapter 24 contains the list of references.
    - Chapter 25 contains the alphabetical index to assist locating topics.

--oo--

2. Systems thinking and beyond

This Chapter summarises thinking, systems thinking and holistic thinking to provide the context to the Holistic Thinking Perspectives (HTP) used to store the perceptions of systems engineering. Readers are advised to refer to Volume 1 of the series for more information (Kasser, 2013c).

2.1. Systems thinking

"*You think only when you have questions*". "*Asking the right questions speeds up the process of learning* " (Paul and Elder, 2006). Systems thinking helps you think effectively. The literature on systems thinking can be grouped or aggregated into the following two types of systems thinking:

1. ***Systemic:*** thinking about a system as a whole.
2. ***Systematic***: employing a methodical step-by-step manner.

Since both types of systems thinking are needed (Gharajedaghi, 1999), consider each of them.

2.1.1. Systemic thinking

One example of this type of systems thinking is, "*When people know a number of things, and one of them understands how the things are systematically categorized and related, that person has an advantage over the others who don't have the same understanding*" (Luzatto, circa 1735). This type of systems thinking has three steps (Ackoff, 1991):

1. A thing to be understood is conceptualized as a part of one or more larger wholes, not as a whole to be taken apart. As Senge wrote, "*Systems thinking is a discipline for seeing wholes*" (Senge, 1990).
2. An understanding of the larger system is sought.
3. The system to be understood is explained in terms of its role or function in the containing system.

Proponents and followers of this type of systems thinking tend to:

- Equate causal loops or feedback loops with systems thinking because they are thinking about relationships within a system, e.g. (Senge, 1990; Sherwood, 2002).
- Define systems thinking as looking at relationships rather than unrelated objects, connectedness, process (rather than structure),

the whole (rather than its parts), the patterns rather than the contents of a system and context (Ackoff, et al., 2010: page 6).

2.1.2. Systematic thinking

Systematic thinking is discussed in the literature on problem-solving, systems thinking, critical thinking and systems engineering usually in the context of a process or processes.

2.1.3. Systemic and systematic thinking

The one practical approach to thinking in a systemic and systematic manner discovered in the literature on systems thinking was Richmond's seven skills of system thinking (Richmond, 1993). Richmond applied a reductionist approach to thinking and identified seven different complementary cognitive processes or thinking skills.

2.2. Beyond systems thinking

However to identify problems and conceptualize and provide solutions we need to go beyond systems thinking and use the holistic thinking approach discussed in Volume 1 (Kasser, 2013c) or equivalent. The holistic thinking approach:

1. Is an iterative process of inquiry (Gharajedaghi, 1999).
2. Goes beyond Gharajedaghi's four perspectives (Gharajedaghi, 1999).
3. Is based on an adaptation of Richmond's seven streams of systems thinking (Richmond, 1993).
4. Perceives an undesirable situation from nine different external, internal, progressive and other Holistic Thinking Perspectives (HTP).
5. Recognizes that each descriptive perspective provides a partial view of the situation.
6. Couples the perspectives with Active Brainstorming (Kasser, 2013c) to allow the problem-solver to think in a systemic and systematic manner about a system (ideation).
7. Incorporates critical thinking (ideation and idea evaluation)..

Holistic thinking goes beyond systems thinking by not only thinking about a system as a whole but also by doing the thinking in a systemic and systematic manner embodying both types of systems thinking. It does this by perceiving issues from a set of standard points (the HTPs) on the perspectives perimeter coupled with Active Brainstorming to think in a systemic and systematic manner about a system (ideation), coupled with critical thinking (ideation and idea evaluation) The elements of holistic thinking include:

- Perceiving issues from the perspectives perimeter using the perspectives discussed in Section 2.2.2.
- Documenting perceptions in the perspectives as discussed in Section 3.1.
- Triggering ideas about remedying undesirable situations, and identifying problems using Active Brainstorming as discussed in Volume 1 (Kasser, 2013c) Chapter 6).
- Using critical thinking to test the ideas and insights.

2.2.1. The need for multiple perspectives

The concept that a single perspective may lead to errors in understanding what is being viewed has been known for centuries if not longer and is best illustrated by the parable of the blind men perceiving a different part of an elephant and inferring what animal they are perceiving (Yen, 2008). Since each man perceives a different part of the elephant, they each infer that they perceive a different animal. It takes a combination of the perceptions to understand the true nature of the animal being felt[1].

The concept of using multiple views and models of a system has long been known in systems and software engineering, and several approaches have been introduced, including:

- The models used in Ward and Mellor's version of structured systems analysis (Ward and Mellor, 1985).
- The models used by Hately and Pirbhai in specifying, respectively, the requirements for and the design structure of software-based systems which grew up around real-time embedded systems (Hately and Pirbhai, 1987).
- The views in the United States Department of Defense (DoD) Architecture Framework (DoDAF, 2004).
- The HTPs (Kasser, 2013c: pages 90-110).

One holistic approach to implementing the process for examining the situation from several perspectives is to use Active Brainstorming which poses the questions 'who', 'what', 'where', 'when', 'why' and 'how' (Kipling, 1912) from the HTPs (Kasser, 2013c: page 150).

But how can you learn to perceive things from different points of view? Well, consider the act of thinking about different aspects of a situation while perceiving the situation from different perspectives, and doing some internal (analysis) and some external (systems thinking). Consider

---

[1] Is this true? Because without the sense of sight, would someone be able to combine the individual perceptions and infer that the animal was an elephant? Perhaps, but probably only if prior experience had shown that the elephant manifested itself as different animals under different conditions.

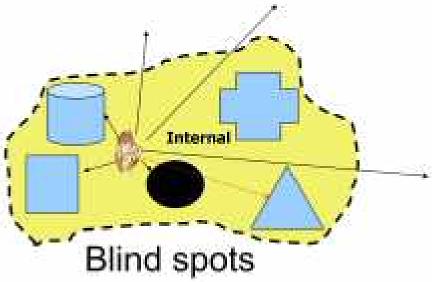


Figure 2.1 Internal views from a single perspective

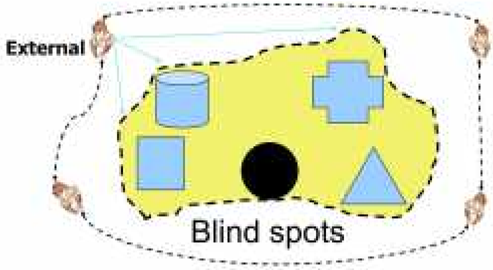


Figure 2.2 Internal from a different single perspective

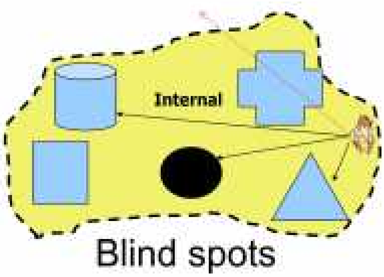


Figure 2.3 External perspectives

the example shown Figure 2.1 where the view of the observer is blocked by the round object. This situation produces blind spots or locations that cannot be seen from that viewpoint. A smart thinker then changes the perspective and views the situation once more from the different perspective as shown in Figure 2.2. A well-chosen second perspective reveals information located in the first perspective's blind spots. Sometimes a third or fourth internal perspective is needed to fully understand a situation. Figure 2.1 and Figure 2.2 show that a single external perspective also has blind spots so a number of external perspectives are needed such as those shown in Figure 2.3.

Using this principle, draw a circle about the situation. Consider the internal perspectives in Figure 2.1 and the external perspectives in Figure 2.3 as chords (areas) on the perimeter of a circle. Call the perimeter of the circle the *perspectives perimeter*. Now when thinking about a situation, problem or issue, it has been observed that some minds:

- Seem to range over the entire perimeter and perceive the issues in a systemic and systematic manner.
- Seem to range over the entire perimeter and perceive the issues from the set of perspectives:
  - Seem to do so in a systemic and systematic manner.
  - Seem to not do so in a systemic and systematic manner.
- Seem to be fixed at one point on the perimeter and observe the issues from a single fixed perspective.
- Can't seem to stop moving round the perimeter.

Since there are no standard stopping points along the perspectives perimeter, each time communications between two parties takes place, time is spent ensuring that both parties to the communication are viewing the issue from the same perspective (stopping point on the perspectives perimeter). This situation can be observed by the use of phrases such as, "*are we on the same page?*" and, "*are we on the same wavelength?*" etc. A standard set of perspectives or "anchor points" is needed to facilitate communications.

2.2.2. The holistic thinking perspectives

This Section now introduces a set of standard viewpoints on the perspective perimeter called the HTPs which can be used to provide anchor points for thinking and communicating in a systemic and systematic manner. These viewpoints go beyond combining analysis (internal views) and systems thinking (external views) by adding quantitative and progressive (temporal, generic and continuum) viewpoints.

Research[2] produced a set of nine viewpoints called System Thinking Perspectives (STP) (Kasser and Mackley, 2008) based on Richmond's work (Richmond, 1993). The STPs soon evolved into the HTPs upon the realization that the HTPs went beyond systems thinking. The STP/HTP viewpoints have produced good results in teaching in postgraduate classes and workshops in Israel, Japan, Singapore, Taiwan and the UK.

The observations and perceptions of a situation are stored in the eight descriptive perspectives and the ideas and inferences from the analysis of the perceptions are stored in the *Scientific* perspective. This approach:

- Separates facts from opinion.
- Provides a standard format or template for storing information

[2] Funded by a grant from the Leverhulme trust to Cranfield University in 2007.

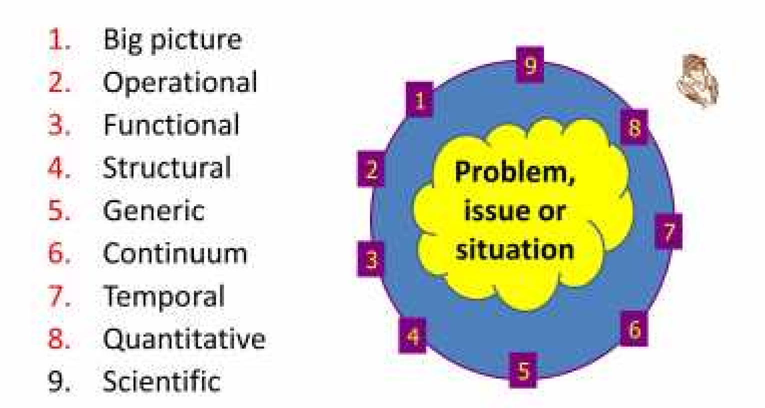


Figure 2.4 The perspectives perimeter

about situations that facilitates storage and retrieval of information about situations such as those documented in Case Studies.

The nine HTP external, internal, progressive and other anchor points shown in Figure 2.4 are as follows:

2.2.2.1. External perspectives

The external perspectives are:

1. ***Big Picture:*** includes the context for the system, the environment and assumptions.
2. ***Operational:*** what the system does as described in scenarios; a black box perspective.

2.2.2.2. Internal perspectives

The internal perspectives are:

3. ***Functional:*** what the system does and how it does it; a white box perspective.

Table 2.1 Function to scenario mapping

	Scenario A	Scenario B	Scenario C	Scenario D	Scenario E
Function 1	X		X		
Function 2	X	X		X	
Function 3		X			X
Function 4	X		X		
Function 5		X	X	X	
Function 6					X

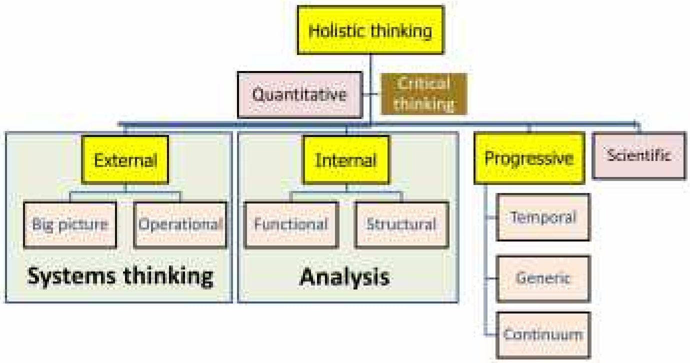


Figure 2.5 The HTPs (*Structural* perspective)

4. ***Structural:*** how the system is constructed and organised. For example the perceptions of the HTPs from the *Structural* perspective are shown in Figure 2.5.

Table 2.1 provides an example of mapping the functions performed in a system to the scenarios in which the functions are used. For example, the table shows that Functions 1, 2 and 4 are used in Scenario A.

2.2.2.3. Progressive perspectives

The progressive perspectives, where holistic thinking begins to go beyond analysis and systems thinking, are the:

5. ***Generic*:** perceptions of the system as an instance of a class of similar systems; perceptions of similarity.
6. ***Continuum:*** perceptions of the system as but one of many alternatives; perceptions of differences. For example, when hearing the phrase, "*she's not just a pretty face*"[3], the thought may pop up from the *Continuum* perspective changing the phrase to, "she's not even a pretty face"[4] which means the reverse.
7. ***Temporal:*** perceptions of the past, present and future of the system.

2.2.2.4. Other perspectives

The other perspectives are:

8. ***Quantitative:*** perceptions of the numeric and other quantitative information associated with the other descriptive perspectives.

[3] Which acknowledges that she is smart

[4] Which means that not only is she not smart, she is also not pretty.

9. ***Scientific:*** insights and inferences from the perceptions from the descriptive perspectives leading to the hypothesis or guess about the issue after using critical thinking.

The first eight perspectives are descriptive, while the ninth (*Scientific*) perspective is prescriptive. While the HTPs provide a standard set of perspective, perceptions from the *Continuum* perspective point out that there are other perspectives including emotional, cultural, personal, the other party's (in a negotiation), etc. These other perspectives should be used as and when appropriate

2.2.3. Linking the perspectives

Each perspective provides a partial view as shown in Figure 2.1, Figure 2.2 and Figure 2.3. Accordingly, perceptions from each perspective provide information about part of the situation. For example, consider a car as the system in the context of home family life. When the car is perceived from the HTPs, the perceptions might include:

1. ***Big picture:*** road network, cars drive the economy, etc.
2. ***Operational:*** going shopping, taking children to school, etc.
3. ***Functional:*** traveling from place to place.
4. ***Structural:*** car with doors, chassis, wheels and boot[5].
5. ***Generic:*** (4-wheeled land vehicle) trucks, vans, etc.
6. ***Continuum:*** different types of engines and vehicles (land and non-land), etc.
7. ***Temporal:*** Stanley steamer, Ford Model T, internal combustion, Ford Edsel, hybrid cars, future electric cars, etc.
8. ***Quantitative:*** miles per hour (mph), engine power, number of passengers, four doors, six wheels, cost, price, etc.
9. ***Scientific:*** depends on problem/issue.

2.3. Summary

This Chapter summarised some aspects of systems thinking discussed in Volume 1 of this series. The key points being:

- Thinking systemically and systematically.
- Systems thinking provides understanding.
- Holistic thinking goes beyond systems thinking to provide insight as to causes of undesirability and solutions that may remedy the undesirable situations.
- The blind spots when perceiving a situation from a single perspective.

[5] Known as a trunk in the US.

- The need for perceiving situations from multiple perspectives.
- The perspectives perimeter.
- One set of standard perspectives, the HTPs, on the perspectives perimeter.
- There are other perspectives on the perspectives perimeter not discussed in this Chapter.

--oo--

PART II

Part II:

- Begins with Chapter 3 which describes the methodology used to store the perceptions of systems engineering in a systemic and systematic manner.
- Contains the perceptions of systems engineering sorted into the descriptive HTPs in Chapters 4 to 11 where:
  - Perceptions from the *Big Picture* perspective are stored in Chapter 4.
  - Perceptions from the *Operational* perspective are stored in Chapter 5.
  - Perceptions from the *Functional* perspective are stored in Chapter 6.
  - Perceptions from the *Structural* perspective are stored in Chapter 7.
  - Perceptions from the *Generic* perspective are stored in Chapter 8.
  - Perceptions from the *Continuum* perspective are stored in Chapter 9.
  - Perceptions from the *Quantitative* perspective are stored in Chapter 10.
  - Perceptions from the *Temporal* perspective are stored in Chapter 11.

This approach has advantages including:

- Using the HTPs as a template so that the reader can readily identify specific types of information stored according to the rules in Section 3.1.
- Separating the facts from the opinions. The facts are in Part II and the opinions begin in Part III.

The approach does however suffer from the following disadvantages:

- Information about a specific topic may be located in different chapters because the information is perceived from different perspectives.
- There will be some forward references to information in a later section of the book. Please ignore them on the first reading, and then use the references as pointers when consulting the information in the book.

--oo--

3. Perceptions of systems engineering

The 20 years of research which delved into systems engineering, systems engineering tools, Operations Research, process improvement, project management, innovation and systems engineering's attempts to manage complexity produced a mass of semi-organised perceptions of, and insights about, systems engineering, that was published in a number of peer-reviewed and a few non-peer reviewed publications and conferences between 1995 and 2015. The 1995 to 2007 publications were updated and published in "*A Framework for Understanding Systems Engineering*", an anthology, in 2007. The anthology was later updated and revised to contain additional papers published between 2007 and 2013 into a second edition (Kasser, 2013b).

This Chapter describes the methodology used to sort and store the perceptions of systems engineering in a systemic and systematic manner in the HTPs into Chapters 4 to 11. This approach has advantages including:

- Using the HTPs as a template so that the reader can readily identify specific types of information stored according to the rules in Section 3.1.
- Separating the facts from the opinions. The facts are in the descriptive perspectives in Chapters 4 to 11; the opinions are in the *Scientific* perspective starting with Chapter 12 in Part III. Accordingly, information may be split between the perspectives. For example, when perceptions from the *Continuum* perspective indicate differences, perceptions from the *Quantitative* perspective identify the number of items that are different. As such, the descriptions of the differences may be stored in either the *Continuum* or *Quantitative* perspectives.

  The approach does however suffer from the following disadvantages:

  - Information about a specific topic may be located in different chapters because the information is perceived from different perspectives.

- There will be some forward references to information in a later section of the book.

However an index is provided in Chapter 25 to facilitate locating information about specific topics.

In each chapter, to chunk the information into manageable blocks, in presenting the perceptions:

- Where the information is short, it is presented in the introductory text in each chapter.
- Where there is a significant amount of information, it is presented in a linked subparagraph or even in a separate chapter.

3.1. Documenting perceptions in the HTPs

The methodology for storing information in the perspectives is that in general, with respect to the system or situation, perceptions of:

- ***"Who"*** belong in the:
  - *Operational* perspective if pertinent to who is performing in a scenario, vignette or Use Case.
  - *Big Picture* perspective if pertinent to an adjacent system or systems.
- ***"What"*** belong in the:
  - *Big Picture* perspective if it is pertinent to the purpose of the system.
  - *Operational* perspective if pertinent to a scenario, vignette or Use Case.
  - *Structural* perspective if pertinent to technology, a physical or information element of the situation.
- ***"Where"*** belong in the *Big Picture* perspective or the *Structural* perspective.
- ***"When"*** belong in the:
  - *Operational* perspective if pertinent to a scenario, vignette or Use Case.
  - *Temporal* perspective if pertinent to the timeline in the story leading up to the situation.
- ***"Why"*** belong in the *Big Picture* perspective.
- ***"How"*** belong in the:
  - *Functional* perspective or the *Structural* perspective (how it works).
  - *Operational* perspective (how it is used).

In addition:

- If the system went through different states and there were major differences in its attributes as time passed, then there should be a different set of HTPs for each state.
- Numeric information is stored in the *Quantitative* perspective.
- The cause or reason for the situation is then inferred and stored in the *Scientific* perspective.
- Perceptions stored in the *Operational* and *Functional* perspectives should be written as verbs in the present tense using words ending in 'ing', such as reading, writing, and designing.

3.2. Documenting real-world situations

If you are dealing with a real world situation rather than a Case Study and writing a situational analysis, perceive your situation from the *Generic* perspective and think of yourself as living a Case Study. The documentation process becomes:

1. Understand the purpose of what you are doing (why you are doing it, and what outcome you hope to achieve).
2. Try to look at the big picture often called a bird's eye or helicopter view.
3. Think about the 'who', 'what', 'where', 'when', 'why' and 'how'.
4. Collect pertinent material.
5. Stop and think about the relationships between items in the material.
6. Make notes, sorting and storing the information in the appropriate HTP using the rules provided in Section 3.1.

3.3. Summary

This Chapter:

- Described the methodology used to store the perceptions of systems engineering accumulated from 20 years of research in a systemic and systematic manner.
- Provided a set of rules for storing the perceptions of situations for use in the workplace and in Case Studies.

The key points were:

- Using the HTPs as a document template to facilitate storing and locating information.
- Separation of facts from opinions, insights, inferences and conclusions.
- How to store information in the HTPs.
- Details on how and why specific information is located in specif-

ic chapters in this book.

--oo--

4. The Big Picture perspective

The *Big Picture* perspective incorporates Richmond's forest thinking (Richmond, 1993) and:

- Is an external perspective.
- Perceives the purpose of the system, often called the system of interest (SOI).

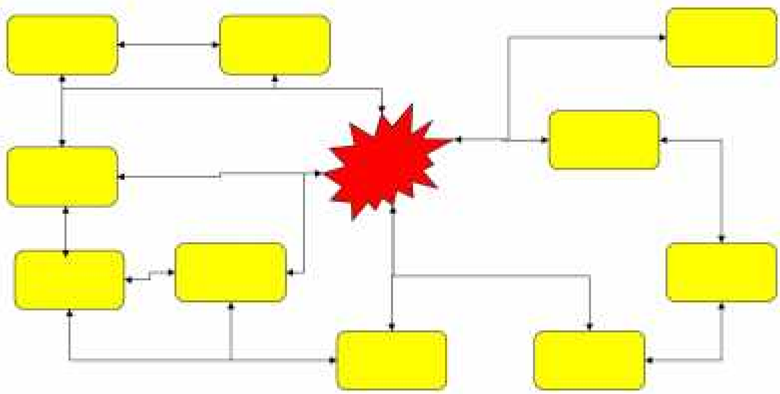


Figure 4.1 The SOI in its context

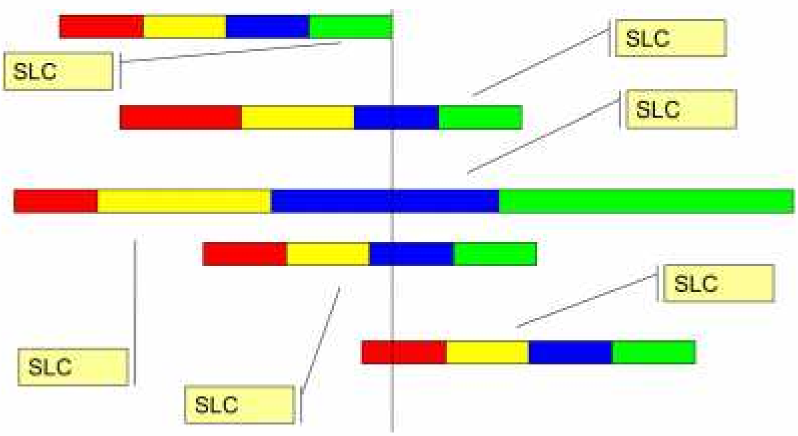


Figure 4.2 The SOI and adjacent systems are evolving

- Perceives the SOI from a bird's eye or helicopter view showing the context of the system hence providing a view of the forest rather than the trees.
- Looks down from the meta-level in the hierarchy of systems perceiving the SOI (the blob in the centre) within the context of its containing system (meta-system) - its environment, the closely coupled adjacent systems with which it interacts and any pertinent loosely coupled more distant systems with which it may indirectly interact as shown in Figure 4.1.
- Shows that the adjacent systems are evolving as shown in Figure 4.2. The colours in the system timelines represent different states in the generic extended system lifecycle (SLC) (Section 13.3).
- Shows the external boundary of the system.
- Contains the assumptions about the system.

Perceptions of systems engineering from the *Big Picture* perspective included:

- Systems engineering:
    - Covers a broad spectrum of activities from people-based systems and organizations to technology-based systems.
    - Takes place in the context of projects.
    - Is performed in the context of three streams of work between milestones as discussed in Section 9.8[1].
    - Is practiced in many domains.
    - Is practiced in different states of the SLC (Section 9.12).
    - Interfaces to other disciplines.
    - Overlaps with other disciplines such as project management and Operations Research as discussed in Section 9.20.
    - Is performed in other disciplines but not necessarily using the name systems engineering.
    - Is iterative and recursive.
- The context for the generic sequence of activities known as systems engineering is shown as in Figure 4.3 and begins with the existence of a problematic or undesirable situation. Many systems engineers are used to starting the cycle with a mission and vision for a new system. In that situation, the cycle can be restated as the undesirable situation is the need for the mission and the new system to make the mission possible.

[1] This is a forward reference because the perceptions are stored in the HTPs in parallel and the information belongs in that section of the book.

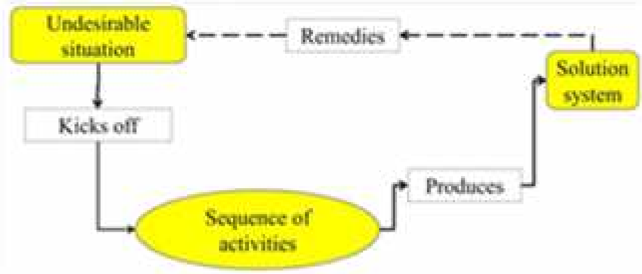


Figure 4.3.The context for systems engineering

A sequence of activities known as systems engineering then takes place which produces a solution system. The solution system is then tested or validated to confirm that the system, operating in its context, remedies the undesirable situation.

- Systems engineering has produced successes and failures as discussed in Section 4.1.

4.1. Successes and failures of systems engineering

Systems engineering successes include:

- Landing men on the moon and returning them safely to earth in the 1960's and 1970's.
- The transcontinental (US) television microwave relay system (Hall, 1962).
- The Semiautomatic Ground Environment (SAGE) project, a computer and radar-based air defence systems created in the United States of America in the 1950s (Hughes, 1998: page 15). SAGE was a massive networked system of radars, anti-aircraft guns, and computers.
- The Public Housing System, Industrial development and the Air Defence System (ADS) in Singapore (Lui, 2007).
- The Atlas Inter-Continental Ballistic Missile (ICBM) development of the 1950's where, "*systems engineering was the methodology used to manage the problem of scheduling and coordinating hundreds of contractors developing hundreds - even thousands - of subsystems that eventually would be meshed into a total system*" (Hughes, 1998: page 118).
- The Standard Central Air Data Computer (SCADC) project (Howard, 2001) which was one of about a dozen standardization programs initiated in the late 1970's by the US Department of Defense (DoD) in the desire to obtain substantial reductions in equipment lifecycle costs through the wide use of digital common modules in aircraft. It was thought that SCADC, because of

the complexity and accuracy requirements of air data computation, would be a difficult concept to bring to fruition. In addition, the SCADC program required the delivery of up to 150 units per month shared between two winning suppliers, in a continuously competitive leader-follower arrangement.

Two of the three largest US suppliers of airborne air-data systems, Honeywell and Sperry declared the concept impossible and declined to bid, despite the potential of $500 million of business. GEC Avionics in the UK were interested in the business and designed and built a SCADC in a replacement form fit-function for the then existing analogue components. The system was a modular core set of Standard Air Data Computer modules made extendable by the use of the 1553 data bus.

The ability to replace "old for new" in around 30 minutes on thousands of the older inventory aircraft, raising the Mean Time Between Failure (MTBF) rates from about 100 hours to greater than the aircraft operational life-times and at the same time equipping them for plug-in new attack systems (via the 1553 data bus) was a significant technical innovation. However, it also had the effect of putting many logistics people out of work overnight. When the first prototypes were demonstrated, a huge effort was launched in Washington by the Logistics fraternity to have the project cancelled. This was supported by much of the US industry who could see an outcome that depleted a large portion of their diverse business with the danger of much of it going overseas. Although the SCADC production programs continued, the implementation in service was delayed for up to two years. Another casualty was that all the other standardization programs fell by the wayside.

By 1998, 6000 units in various configurations had been sold including a version modified into a digital flight control system adopted by the US Navy for the F14. It was the most widely used digital system and most reliable in all aircraft in the Gulf War. The program was arguably the most successful of any airborne equipment supply program in the history of world aerospace, and 100% of the production units came from the UK source, the leader-follower concept being abandoned. It was estimated that by the time GEC Avionics received the third or fourth order for production units, the direct Defence savings exceeded $US 500 million.

Systems engineering failures include:

- Failed projects particularly in the National Aeronautical and

Space Administration (NASA) and the US DoD, where inadequate systems engineering is repeatedly cited as a major contributor to the failure, e.g. (Evans, 1989; Leveson, 2004; Welby, 2010; Wynne and Schaeffer, 2005).

- Failure of systems engineering in the early stages of large projects (Hiremath, 2008) and other examples of poor systems engineering implementation (GAO, 2006).

These perceptions of success and failure lead to the question why does systems engineering succeed sometimes and fail at other times?

4.2. Summary

This Chapter contains perceptions of systems engineering from the *Big Picture* perspective. The key points were:

- Systems engineering covers a broad spectrum of activities.
- Systems engineering is performed in the context of three streams of activities between reporting milestones.
- Systems engineering succeeds and fails in the real world.

--oo--

5. The Operational perspective

The *Operational* perspective incorporates Richmond's operational thinking (Richmond, 1993) and:

- Is an external perspective.
- Corresponds to the traditional black box 'closed system' view.
- Provides a view of the normal and contingency mission and support functions performed by a system.
- Tends to be documented in the form of Use Cases, Concepts of Operations (CONOPS), Operations Concept Documents (OCD), the Business Process Reengineering (BPR) 'to-be' and 'as-is' views and other appropriate formats.

Perceptions of systems engineering from the *Operational* perspective included:

1. Systems engineers performing systems engineering to provide value (Weiss, 2013).
2. Systems engineers ensuring that the constructed system remedies the operational need.
3. Systems engineers are not producing systems, they transform an operational need into a description of system performance parameters and a system configuration (FM_770-78, 1979), but other personnel actually construct the systems.
4. Systems engineering transforming an undesirable situation into a situation without the undesirable characteristics, called the desirable situation, using resources and constrained by rules and regulations as shown in Figure 5.1.
5. Systems engineers performing different activities in the different

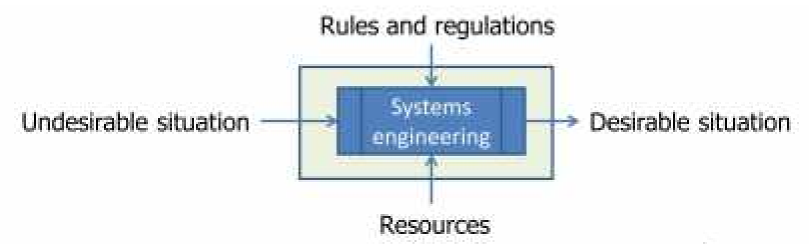


Figure 5.1 System Engineering - Operational perspective

states of the project as discussed in Section 5.1.

5.1. The different activities of the systems engineer

Perceptions from the *Operational* perspective indicate that systems engineers are:

- Performing activities in various scenarios in the work place in projects in different domains; activities which include:
  - Conceptual design.
  - Requirements management.
  - Architecting.
  - Interface management.
  - Testing.
  - Integrating.
  - Verification and validation.
  - Engineering management.
  - Human factors engineering.
- Continuing their education and sharing their knowledge by reading and contributing to technical journals and participating and contributing to annual practitioner-oriented conferences such as the:
  - International Council on Systems Engineering (INCOSE) international symposia and regional conferences.
  - National Defense Industrial Association (NDIA) systems engineering conference.
  - Institute of Electrical and Electronics Engineers (IEEE) Engineering Management conference.
  - Australian Systems Engineering Test and Evaluation (SETE) conference.
  - Asia-Pacific Council on Systems Engineering Conference (APCOSEC).
  - Conference on Systems Engineering Research (CSER).

  The papers presented in these conferences can be sorted into a number of categories including:

  - Theoretical papers about various aspects of the profession.
  - Personal experiences.
  - Case Studies.
  - Tutorials on an aspect of the profession.

  The theoretical papers provide ideas about improving systems engineering. Practitioner written conference papers in these latter three categories provide anecdotal descriptions of situations, problem solving approaches and lessons learned from real-world projects.

Students also attend these conferences to learn about systems engineering and sometimes to present part of their research towards a Master of Science or a Doctoral degree.

- Making use of the many opportunities for education and training. Perceptions of the:
  - Number of such opportunities is discussed in Section 10.2.
  - Content of the courses which indicate that they seem to line up with the different camps in systems engineering is discussed in Section 9.17.
- Creating and using models of systems and conceptual systems.
- Taking part in Model Based System Engineering (MBSE) meetings and workshops which are characterized by people talking past each other and not communicating[1].
- Interpreting the word 'model' in different ways as can be seen in the following examples:
  - To some people a model is a way of expressing knowledge in an abstract way, yet exact, without showing unnecessary details. Others such as software engineers model as a way of communicating knowledge (Kasser and Shoshany, 2000; Kasser, 2013b: pages 115-131).
  - To some people, models and blueprints are those that have been used by hardware engineers in the form of schematic diagrams and sketches since the early days of engineering.
  - To some people, the word model can be used to mean a reference model such as, "*We propose a [reference] model for reusability based on …*" (Prieto-Díaz, 1987).

5.2. Scenarios in the activities

Hitchins states, *"systems engineering … is a philosophy and a way of life"* (Hitchins, 1998). Thus interpreting Hitchins, a systems engineer is a person who intuitively lives and acts according to the philosophy of systems engineering demonstrating the following types of behaviour in performing the activities discussed in Section 5.1.

- Scheduling meetings so that they do not conflict with other meetings. Any time this person arranges meetings, training courses, etc., they ensure that the event meshes with and does not conflict with existing events.

---

[1] Which is a typical characteristic of the early years of a discipline.

- Setting schedules so things are done in a logical order in which early activities do not negatively affect later ones, and so that things are not done in a hurry at the last minute. This applies to all types of schedules, not just systems integration.
- Coordinating meetings with external events. For example, research organizations hold periodic meetings. This person tries to set up the program for such meetings so that in the meeting before a conference paper deadline, authors get a chance to present their paper to both rehearse the presentation and get comments on it, before the submission deadline. The author-presenters can then incorporate the often-insightful comments made by their peers into the manuscript.
- Viewing situations both 'as they are', and 'as they could be' at the same time. This person can walk into a situation and point out improvements using this ability. For example, consider an undesirable situation in a conference setting. The presenters were having a problem pacing their presentations to keep time. This person would examine the undesirable situation and then ask the conference organizers to put a clock on the wall at the back of the room to enable the presenters to invisibly pace themselves (provided that the clock was pointed out to them before they began their talk).
- Being able to examine problems from more than one perspective (most of the time). These perspectives have been called cognitive filter's in the behavioural science literature, e.g. (Wu and Yoshikawa, 1998), and decision frames (Russo and Schoemaker, 1989) in the management literature. Whatever we call them, they are the internal filters through which we view the world. They include political, organizational, cultural, and metaphorical filters, and each of them highlights relevant parts of the system and hides (abstract out) the parts not relevant to the filter.
- Being able to act as a catalyst (Demarco, 1997) to invisibly resolve system and process related issues speedily and peacefully.
- Being able to challenge assumptions by asking disconfirming questions (Russo and Schoemaker, 1989: page 103) or good questions (Frank, 2006) to identify and then state the real cause of undesirability (Wymore, 1993).
- Making use of lessons learned by others before starting a new project. This person may make mistakes, but at least they will be new ones.
- Being able to quickly determine the aspects of a situation which are relevant to the problem, or, in electrical engineering lan-

guage, separate signals from noise.

Other systems engineers have difficulty in these scenarios and things do not go as smoothly.

5.3. Summary

This Chapter contained perceptions of systems engineering from the *Operational* perspective. The key points were systems engineers:

- Performing systems engineering to provide value.
- Performing a wide range of different activities in projects in the workplace.
- Continuing their education and training and also mentoring, educating and training junior personnel via journals, books and conferences.
- Using holistic thinking as a way of life or at least some of them.

--oo--

6. The Functional perspective

The *Functional* perspective incorporates Richmond's system-as-a-cause and closed-loop thinking (Richmond, 1993) and:

- Is an internal perspective.
- Corresponds to the traditional white box 'open system' view.
- Provides a view of the functions or activities (and the relationships between them) performed within the system without reference to which of the physical elements in the system performs those functions.
- Can be a view of what is being done or how it is being done depending on the level of system elaboration.

Perceptions of systems engineering from the *Functional* perspective identified pure systems engineering functions (Section 12.2) performed in the operational scenarios. These pure systems engineering functions include:

1. Systems thinking and beyond discussed in Chapter 2.
2. Problem-solving discussed in Section 6.1.
3. Analysis discussed in Section 6.2.
4. Synthesis discussed in Section 6.3.
5. Decision-making discussed in Section 6.4.
6. Communicating discussed in Section 6.5.
7. Innovating discussed in Section 6.6.

Note all functions may not be used in all scenarios as shown in Table 2.1.

6.1. Problem solving

The core pure systems engineering function is thinking. Thinking:

- Is the major sub-function in problem-solving
- Goes hand in hand with asking and answering questions (Paul and Elder, 2006).

Problem-solving is used in all the applications of systems engineering in all domains. The first step in problem-solving is examining the situation to determine the nature of the problem. When examining a situation, the systems engineer makes observations, performs research to answer

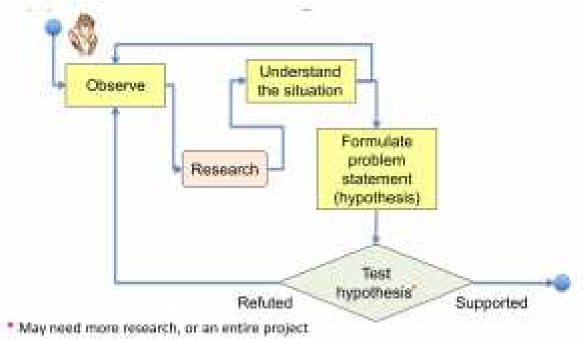


Figure 6.1 Approach to dealing with situations

questions that cannot be answered immediately and develops an understanding of the situation as shown in Figure 6.1. See Section 18.7.1 for an example of using this process to optimize your sex life. The output of this thinking process in systems engineering is:

- A statement of the cause of the undesirability.
- A conceptual solution that remedies the undesirability.
- An approach to realize the conceptual solution.

All three of which are hypotheses until they have been tested.

When thinking about a situation, in general:

- ***Why questions*** can be used to develop an understanding of the situation. See the "five Why's" (Serrat, 2009).
- ***What questions*** can help define the root cause of an undesirable symptom and also what needs to be done to remove the

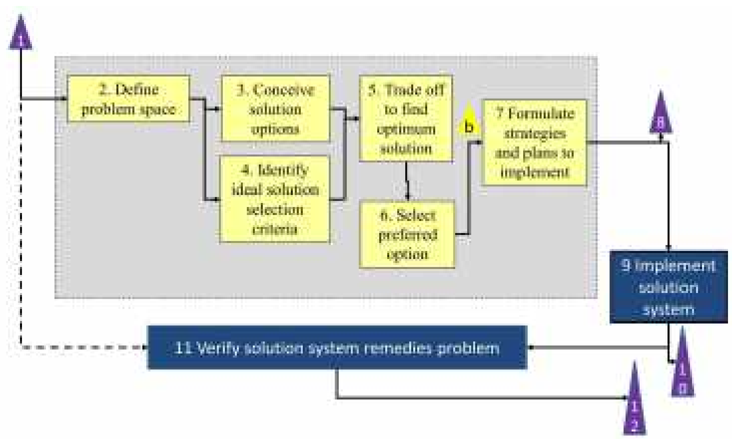


Figure 6.2 Functional perspective of the decision-making process with the implementation states added

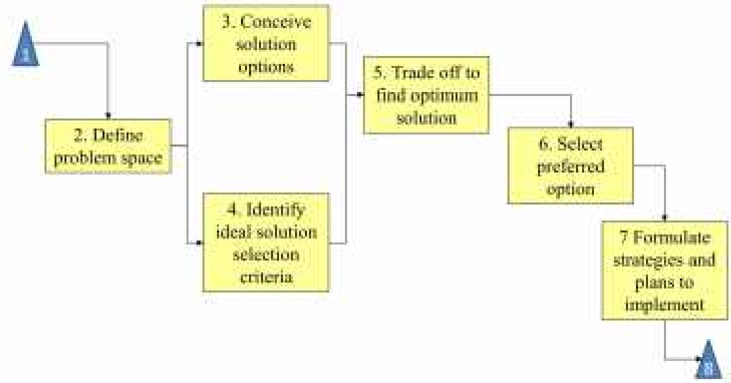


Figure 6.3 Modified Hitchins' view of the problem-solving decision making process

cause.

- ***How questions*** tend to provide solutions.

The *Functional* perspective of the decision-making/problem-solving process based on (Hitchins, 2007: page 173) shown in Figure 6.3[1] depicts the series of activities which are performed in series and parallel that transform the undesirable situation into the strategies and plans to realize the solution system operating in its context. The process contains the following major milestones and tasks:

1. The milestone to provide authorization to proceed.
2. The process to define the problem.
3. The process to conceive several solution options.
4. The process to identify ideal solution selection criteria.
5. The process to perform trade-offs to find the optimum solution.
6. The process to select the preferred option.
7. The process to formulate strategies and plans to implement the preferred option.
8. The milestone to confirm consensus to proceed with implementation. This milestone is also Milestone 1 for the subsequent iteration of the process.

Once the stakeholder consensus is confirmed at Milestone 8 at the end of Figure 6.2, the project can move on to the Implementation states shown from the *Functional* perspective in Block 9 of Figure 6.3 where the additional following major milestones and tasks are:

[1] Hitchins' version of the process has been modified to add milestones at the beginning and end of the process.

Figure 6.4 The single correct solution

9. The process to implement the solution system often using the System Development Process (SDP).
10. The milestone review to document consensus that the solution system has been realized and is ready for validation.
11. The process to validate the solution system remedies the evolved need in its operational context.
12. The milestone to document consensus that the solution system remedies the evolved need in its operational context.

6.1.1. The short and extended holistic problem-solving processes

The traditional systems engineering approach to problem-solving uses the shortened problem-solving process beginning with a problem and ending with a single correct solution as shown in Figure 6.4. However, *"Problems do not present themselves as givens; they must be constructed by someone from problematic*[2] *situations which are puzzling, troubling and uncertain"* (Schön, 1991). The extended holistic problem-solving process begins with an undesirable situation which has to be converted to a Feasible Conceptual Future Desirable Situation (FCFDS) and ends when the undesirable situation no longer exists as shown Figure 6.5 (Kasser, 2013c). In this extended holistic problem-solving process, an entity becomes aware of an undesirable situation. A project is authorized to do something about the undesirable situation[3]; the problem. The problem solver:

1. Collects and analyses the information.
2. Tries to understand the situation.
3. Determines what makes the situation undesirable.
4. Determines if someone has faced a similar problem, what they did about it, and the similarities and differences between the other situation and the current undesirable situation and how those affect the problem and solution in this instance.
5. Conceptualises and then creates a vision of a FCFDS; the solution system operating in its context.

---

[2] or undesirable

[3] Often made up of a number of related factors

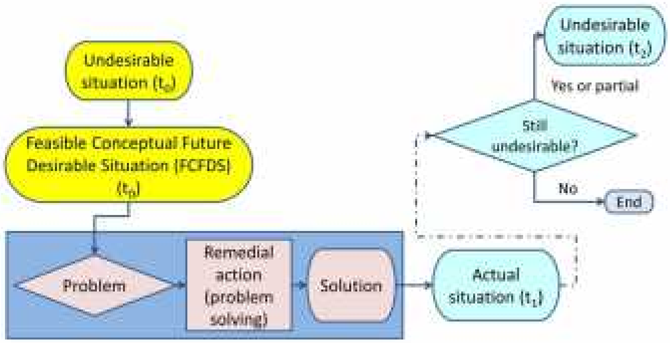


Figure 6.5 The extended holistic problem-solving process

After the stakeholders concur on the FCFDS, the problem becomes how to transition from the undesirable situation existing at time t_0 to the FCFDS that will exist in the future t_1 often known as "the solution" by making some sort of transformation. This transformation or remedial action creates the solution system which will operate in the context of the FCFDS. If the situation is complex, the remedial action transformation process often takes the form of a SDP for the solution system that will be operational in the context of the FCFDS.

Figure 6.5 also includes the time dimension, because the remedial action or problem-solving process takes time[4], and during that time the original undesirable situation which existed at time t_0 may have changed, which means that the solution system operating in the context of the actual situation at time t_1 may not have remedied the changed undesirable situation as it exists at time t_1 because of one or more of the following:

- The solution system operating in its context does not remedy the entire original undesirable situation.
- New undesirable aspects have shown up in the situation during the time taken to develop the solution system.
- Unanticipated undesired emergent properties of the solution system and its interactions with its adjacent systems may produce new undesirable outcomes.

Errors made in each part of SDP can produce undesirable outcomes. For example if the wrong cause of the undesirable situation is identified, the wrong problem will be stated, and not only will the solution not remedy the undesirable situation, the solution may make the situation even more undesirable. Similarly, if the correct problem is identified but the wrong solution conceptualized and realized then again the undesirable

[4] For large scale systems the development process can take years.

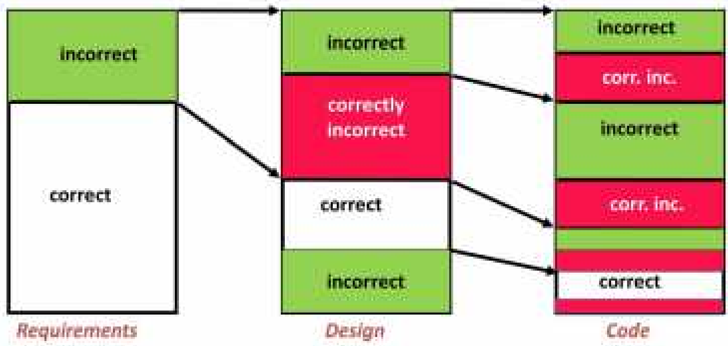


Figure 6.6 Propagation of errors

situation will not be remedied and may become even more undesirable. Even if the correct solution is conceptualized, errors in the realization process may produce an incorrect solution. This concept is shown in Figure 6.6.

Once realized, the solution system is tested in operation in the actual situation existing at time t_1 to determine if it remedies the undesirable situation. If the undesirable situation is remedied, then the process ends; if not, the process iterates from the new undesirable situation at t_2.

Figure 6.5 also contains and masks an important assumption; namely, the domain knowledge to understand what is being observed and infer a correct conclusion is present in the personnel performing the analysis of the observations. If this assumption is wrong, then the conclusions may be incorrect.

The descriptions of the extended holistic problem-solving process in this chapter are notional. That means the description is the way things should be done. Perceptions indicate that not all projects perform all the activities described in the process.

6.1.2. The time delays in realizing solutions

Figure 6.7 provides another view of problem solving as a causal loop. The effect manifests itself as an undesirable situation, and the decision maker is faced with the problem of deciding what actions to take to identify and mitigate the cause of the undesirable situation. Note the dotted feedback loop between 'effect' and 'cause' is there to indicate that there may or may not be an interaction. Similarly the other dotted links indicate that there may or may not be a link.

The figure also shows that there are time delays in the causal loop, so that when an action is taken, the response or responses may only be noted after some time delay. The interaction between the cause, effect and remedial action also suffers from time delays. The time delays in the

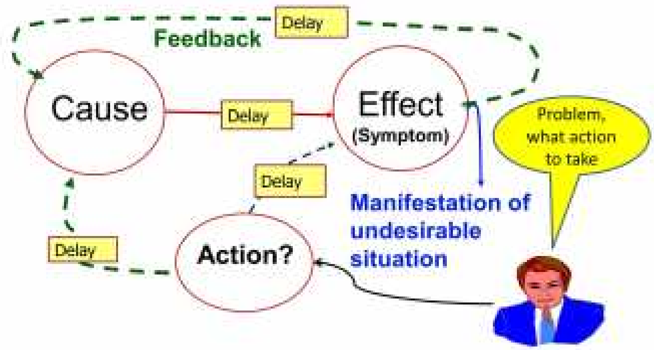


Figure 6.7 Problem-solving as a causal loop

loops, well known to engineers and systems engineers working on control systems, were grouped as (Kasser, 2002c):

- ***First order:*** a noticeable effect within a second or less.
- ***Second order:*** a noticeable effect within a minute or less.
- ***Third order:*** a noticeable effect within an hour or less.
- ***Fourth order:*** a noticeable effect within a day or less.
- ***Fifth order:*** a noticeable effect within a week or less.
- ***Sixth order:*** a noticeable effect within a month or less.
- ***Seventh order:*** a noticeable effect within a year or less.
- ***Eighth order:*** a noticeable effect within a decade or less.
- ***Ninth order:*** a noticeable effect within a century or less.
- ***Tenth order:*** a noticeable effect after a century or more

6.2. Analysis

Systems engineers perform analysis particularly in the Needs Identification State of the SLC (Section 9.12.1). Analysis:

- Can be considered as a top-down approach to thinking about something and is associated with René Descartes (Descartes, 1637, 1965).
- Has been termed reductionism because it is often used to reduce a complex topic to a number of smaller and simpler topics.
- May use the tools of the 1960's (Section 7.3).

6.3. Synthesis

Systems engineers perform synthesis which is combining two or more entities to form a more complex entity. Synthesis can be considered as a bottom-up approach to thinking about or integrating something.

6.4. Decision-making

Systems engineers make decisions using quantitative and qualitative decision-making tools (Kasser, 2013c: pages 215-257).

6.5. Communicating

Systems engineers communicate using verbal and written methods, each of which can be formal and informal. Communications uses words which are symbols for the ideas used to communicate ideas, not the ideas themselves. Verbal communications may be informal in meetings or formal to announce something. A presentation is a speech reinforced with text and graphics used in the presentation slides. Presentations can be formal and informal. Formal written communications tend to be in the form of documents, notes and emails (Kasser, 2013c: pages 49-84).

Communication scenarios include:

- Formal and informal milestones.
- Peer reviews.
- Meetings with stakeholders.

6.6. Innovating

Good systems engineers innovate, formulate and solve problems. Poor systems engineers can cause problems.

6.7. Summary

This Chapter contained perceptions of systems engineering from the *Functional* perspective. The key points were:

- Some systems engineers think; most follow the problem-solving process thinking through the problem, conceiving solutions and selecting the most acceptable solutions; some systems engineers just follow processes without thinking.
- Systems engineers remedying problems.
- The difference between the short problem-solving process and the extended holistic problem-solving process.
- The time delays in realizing solutions.

--oo--

7. The Structural perspective

The *Structural* perspective is an internal perspective incorporating the traditional physical, technical and architectural framework views of a system. Perceptions from this perspective include:

- The discipline of systems engineering.
- Hierarchies.
- Structural elaboration.
- Architectures.
- Internal subsystem boundaries.
- Physical and virtual components including the tool used by systems engineers.
- Effects on the system due to its internal structure.
- The interconnections between physical elements and subsystems.
- The structure of the information in the system.

Perceptions of systems engineering from the *Structural* perspective included:

1. Systems engineering is a discipline as discussed in Section 7.1.
2. The principle of hierarchies discussed in Section 7.2.
3. The tools paradox discussed in Section 7.3.
4. The Standards for systems engineering discussed in Section 7.4.
5. The systems engineer discussed in Section 7.5.
6. The structure of the problem discussed in Section 7.6.

7.1. Systems engineering is a discipline

The discipline of systems engineering provides the structure of systems engineering. One view of the elements of a discipline was provided by (Kline, 1995: page 3) who states, "*a discipline possesses a specific area of study, a literature, and a working community of paid scholars and/or paid practitioners*". Since systems engineering already has an area of study, a literature and a working community of paid scholars and/or paid practitioners it meets Kline's definition of a discipline.

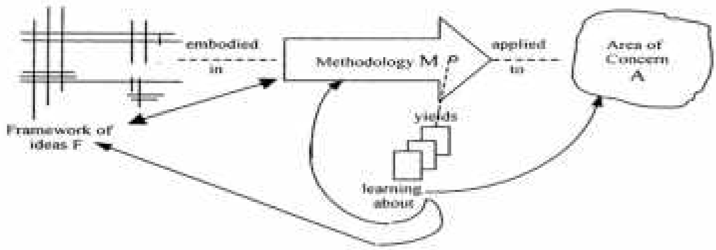


Figure 7.1 Elements relevant to any piece of research (Checkland and Holwell, 1998: p 13)

7.1.1. Elements relevant to research in a discipline

There is ongoing research in systems engineering. For example, between 2000 and 2008, the Systems Engineering and Evaluation Centre (SEEC) at the University of South Australia (UniSA) was the leading world-wide institution in systems engineering research. SEEC members performed research into the nature of systems engineering and the tools used by systems engineers and their findings were published in the literature, e.g. (Tran, et al., 2008; Kasser, et al., 2006; Kasser and Cook, 2004; Kasser, 2004; Kasser, et al., 2002; Kasser, et al., 2003). In addition, some of the research into the value of systems engineering discussed in Section 10.3 was performed in the PhD program at UniSA (Honour, 2013).

Perceptions from the *Structural* perspective indicated that research into a discipline needs the following three items (Checkland and Holwell, 1998):

- ***An Area of Concern (A)*** which might be a particular problem in a discipline (area of study), a real-world problem situation, or a system of interest.
- ***A particular linked Framework of Ideas (F)*** in which the knowledge about the area of concern is expressed. It includes current theories, bodies of knowledge, heuristics, etc. as documented in the literature as well as in tacit knowledge.
- ***The Methodology (M)*** in which the framework is embodied. The methodology incorporates methods, tools, and techniques in a manner appropriate to the discipline that uses them to investigate the area of concern.

Figure 7.1 (Checkland and Holwell, 1998: page 23) illustrates the relationship between these elements. Given that there is a working community of paid scholars and/or practitioners, these same three elements can also be used to characterize a discipline because they expand Kline's specification and encompass the key aspects of a discipline (Cook, et al., 2003). Consider each of these elements in turn, as they apply to systems engineering.

7.1.2. An area of concern (A)

The Area of Concern (A) covers what systems engineers do, where they do it, and the overlapping of, and differences in, the roles of systems engineering, systems architecting, and project management. There have been many diverse opinions on these topics over the years, and the opinions are summarized in the different camps of systems engineering discussed in Section 9.17.

The (A) of specific systems engineering research depends on the paradigm. See the Systems Engineering - the Activity (SETA) and Systems engineering - the role (SETR) paradigms discussed in Section 9.18.

- In the SETR paradigm, the Area of Concern (A) of systems engineering is very broad since it needs to span the activities performed in all the roles of the systems engineer, Operations Researcher and project manager in all organisations.
- In the SETA paradigm, the Area of Concern (A) of systems engineering is much narrower and only needs to cover the activities performed in pure and applied systems engineering as discussed in Section 12.2.

7.1.3. The framework of ideas (F)

Checkland and Holwell discuss the importance of a "declared-in-advance" epistemological framework (F) when undertaking interpretive research (Checkland and Holwell, 1998: pages 23-25). Even though the (F) depends on the viewpoint or camp of systems engineering (Section 9.17), the (F) for systems engineering can be considered as being documented in the literature on the pure, applied and domain systems engineering (Section 12.2) activities that take place in the (A).

7.1.4. The methodology (M)

The methodology depends on the system engineering paradigm (Section 9.21).

- ***In the SETR paradigm,*** systems engineering may be considered as a meta-methodology incorporating the methodologies, tools and techniques used in the (A) by both systems engineers and practitioners of the other organizational activities[1].
- ***In the SETA paradigm,*** systems engineering may be considered as an enabling discipline used to tackle complex problems

[1] This is the perspective from the meta-discipline camp, see Section 9.17.4.

in other disciplines.

In both paradigms, this puts a considerable number of tools into the toolbox of the systems engineer, including:

- Total Systems Intervention (TSI) (Flood and Jackson, 1991).
- Soft Systems Methodology (SSM) (Checkland and Holwell, 1998).
- A process-oriented, blended, object-oriented, rapid development, people oriented, and organisational-oriented methodology (Avison and Fitzgerald, 2003).
- A whole suite of problem solving tools for use in requirements elicitation and elucidation (Hari, et al., 2007); space precludes summarisation of the tools in this section, however, information can easily be found in many other sources. These tools include:
  - Interviews (Alexander and Stevens, 2002).
  - Joint Applications Development (JAD) (Wood and Silver, 1995).
  - Analytical Hierarchical Process (AHP) (Saaty, 1990).
  - Nominal Group Technique (NGT) (Memory Jogger, 1985).
  - Scenario building.
  - User/customer interviews.
  - Questionnaires.
  - Customer visits.
  - Observation.
  - Customer value analysis.
  - Use Cases.
  - Contextual inquiry.
  - Focus groups.
  - Viewpoint modelling (Darke and Shanks, 1997).
  - Quality Function Deployment (QFD) (Hauser and Clausing, 1988; Clausing and Cohen, 1994).
  - Quality Requirements Definition (QRD) (Hari, et al., 2007).
- The more commonly used hard systems methodologies (Blanchard and Fabrycky, 1981; Buede, 2000) and other treatments of the SEP.

Both the SETA and SETR paradigms of systems engineering meet (Kline, 1995)'s view of a discipline namely they have *"a specific area of study, a literature, and a working community of paid scholars and/or paid practitioners"* and having an (A), (M) and (F), systems engineering contains the elements relevant to research in a discipline (Checkland and Holwell, 1998).

7.2. The principle of hierarchies

The principle of hierarchies in systems (Spencer, 1862) cited by (Wilson, 2002) is one of the ways humanity has managed complexity for most of its recorded history and is defined in the following three quotations.

1. *"All complex structures and processes of a relatively stable character display hierarchical organisation regardless of whether we consider galactic systems, living organisms and their activities or social organisations"* (Koestler, 1978: page 31).
2. "*Once we adopt the general picture of the universe as a series of levels of organisation and complexity, each level having unique properties of structure and behaviour, which, though depending on the properties of the constituent elements, appear only when those are combined into the higher whole*[2]*, we see that there are qualitatively different laws holding good at each level"* (Needham, 1945) cited by (Koestler, 1978: page 32).
3. Wilson wrote "*The English philosopher Herbert Spencer appears to be the first to set out the general idea of increasing complexity in systems (Spencer, 1862). The term itself was first used by the English biochemist (and scholar of Chinese science) Joseph Needham (Needham, 1937). The following quotation from a Web source provides an insight into the fundamentals of the theory (ULA, 2002):*

    a) *The structure of integrative levels rests on a physical foundation. The lowest level of scientific observation would appear to be the mechanics of particles.*
    b) *Each level organizes the level below it plus one or more emergent qualities (or unpredictable novelties). The levels are therefore cumulative upwards, and the emergence of qualities marks the degree of complexity of the conditions prevailing at a given level, as well as giving to that level its relative autonomy.*
    c) *The mechanism of an organization is found at the level below, its purpose at the level above.*
    d) *Knowledge of the lower level infers an understanding of matters on the higher level; however, qualities emerging on the higher level have no direct reference to the lower-level organization.*
    e) *The higher the level, the greater its variety of characteristics, but the smaller its population.*
    f) *The higher level cannot be reduced to the lower, since each level has its own characteristic structure and emergent qualities.*
    g) *An organization at any level is a distortion of the level below, the higher-level organization representing the figure which emerges from the previously organized ground.*

[2] Namely emergent properties (author)

h) *A disturbance introduced into an organization at any one level reverberates at all the levels it covers. The extent and severity of such disturbances are likely to be proportional to the degree of integration of that organization.*

i) *Every organization, at whatever level it exists, has some sensitivity and responds in kind"* (Wilson, 2002).

7.3. The tools paradox

Systems engineers use tools hence the perceptions about tools has been stored in the *Structural* perspective.

There seems to be a paradox when looking at the tools used by systems engineers. The tools have changed over time. The tools of systems engineering were different in 1960's and in 2005. The tools of the 1960's (Alexander and Bailey, 1962; Wilson, 1965; Chestnut, 1965; Goode and Machol, 1959) some of which were also used in Operations Research were:

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

Yet by, 2005 systems engineering tools (Jenkins, 2005) were:

- Databases.
- DOORS.
- CORE™.
- PowerPoint.
- Visio.
- Drawing tools.
- Word processors.
- Spreadsheets.
- Etc.

7.4. The Standards for systems engineering

The Standards commonly used in systems engineering cover systems engineering management and the processes for engineering a system. However they do not seem to actually apply to systems engineering since:

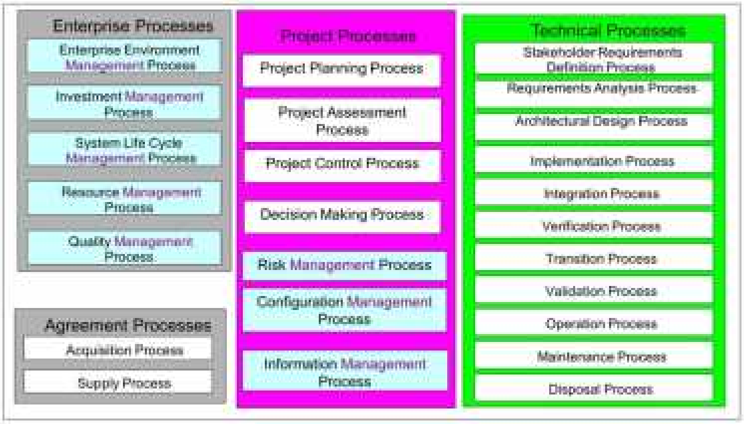


Figure 7.2 ISO 52888 systems engineering processes (Arnold, 2002)

- 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 formally approved.
- American National Standards Institute (ANSI)/Electronic Industries Alliance (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 (SEP) (IEEE 1220, 1998).
- The International Standards Organisation (ISO)/International Electrotechnical Commission (IEC) 15288 Standard (Arnold, 2002) lists processes performed by systems engineers as shown in Figure 7.2 (Arnold, 2002: page 61). In addition, many of the activities in ISO/IEC 15288 also overlap those of project management.

While the Standards cited are out of date, their replacements (if any) follow the theme of the original Standard.

7.5. The systems engineer

Systems engineers are a combination of hardware, software and bioware and accordingly perceptions of the systems engineer are stored in the *Structural* perspective of systems engineering.

What is a systems engineer? The simple answer is a person who does systems engineering. However, Hitchins states, *"systems engineering … is a philosophy and a way of life"* (Hitchins, 1998). Thus interpreting Hitchins, a systems engineer should be a person who intuitively lives and acts according to the philosophy of systems engineering demonstrating the behaviour discussed in Section 5.2. While these abilities link back to Hitchins' top-down definition they do not seem to implicitly link to the commonly accepted domain knowledge of systems engineering taught in academic institutions. This linkage may however be seen from a bottom-up perspective. From a bottom-up perspective, the literature contains several publications on the characteristics of systems engineers. For example, Hall provided the following specifications or traits for an "Ideal Systems Engineer" grouped in the following areas (Hall, 1962: pages 16-18):

- An ability to see the big picture.
- Objectivity.
- Creativity.
- Human Relations.
- A Broker of Information.
- Education - graduate training in the relevant field of interest (application), as well as courses in probability and statistics, philosophy, economics, psychology, and language.
- Experience in research, development, systems engineering and operations.

Hall concluded by stating that since the ideal is not available because the scope of the task is beyond the capabilities of a single individual, mixed teams of specialists and generalists are used. Later bottom-up studies include those by Frank who consolidated and classified the characteristics of successful systems engineers as ten cognitive characteristics, eleven abilities, ten behavioural competences and fifteen dealing with knowledge (Frank, 2002; 2006).

Other systems engineers, mostly from the meta-discipline camp (Section 9.17.4) describe systems engineers as being 'T' shaped with some knowledge of all engineering disciplines and in-depth knowledge of one (Zonnenshain, 2015). However, in the talent-seeking field, the definition of 'T' shaped is slightly different. The vertical stem of the 'T' is the foundation: an in-depth specialized knowledge in one or two fields. The horizontal crossbar refers to the complementary skills of communication (including negotiation), creativity, the ability to apply knowledge across disciplines, empathy (including the ability to see from other perspectives), and an understanding of fields outside one's area of expertise (Brooks, 2012).

The contribution of good people in an organised organisation was recognised in the systems engineering literature 65 years ago, namely, *"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).

The activities performed by the systems engineer are discussed in Section 5.1.

7.6. The structure of the problem

The structure of the problem is an objective measurement and applies to the full range of non-complex through complex problems where:

- ***Well-structured problems*** are problems where the existing undesirable situation and the solution are clearly identified. These problems may have a single solution or sometimes more than one acceptable solution. Well-structured problems with single solutions tend to be posed as closed questions, while well-structured problems with multiple solutions tend to be posed as open questions. Well-structured complex problems consist of a set of interconnected well-structured non-complex problems and since the remedy to one may affect another, these problems cannot be solved in one pass thorough the problem solving process. In general:
    - ***Easy*** well-structured problems are simple problems and require little if any research before creating the solution.
    - ***Medium*** well-structured problems are less simple and require some research before creating the solution.
    - ***Ugly*** well-structured problems are complicated yet require little if any research before creating the solution.
    - ***Hard*** well-structured problems are complicated and require a significant amount of research before creating the solution.
- ***Ill-structured problems, sometimes*** called 'ill-defined' problems or 'messy' problems when complex, are problems where either or both the existing undesirable situation and the FCFDS are unclear (Jonassen, 1997). Ill-structured problems cannot be remedied. They must be converted to well-structured problems first. However, different people convert ill-structured problems

into different and sometimes contradictory well-structured problems and which would generate different and sometimes contradictory solutions.

- ***Wicked problems*** are extremely ill-structured problems first stated in the context of social policy planning (Rittel and Webber, 1973). The fundamental paradox with respect to wicked problems is that there are no such problems; since while the stakeholders may agree that the situation is undesirable, they cannot agree on "the problem".

7.7. Summary

This Chapter contained perceptions of systems engineering from the *Structural* perspective. The key points were:

- Systems engineering meets one set of requirements for being a discipline.
- The principle of hierarchies.
- The tools paradox.
- The Standards for systems engineering are not standards for systems engineering.
- Some characteristics of a systems engineer.
- The structure of the problem.

--oo--

8. The Generic perspective

The *Generic* perspective:

- Is a progressive perspective.
- Looks at and for similarities.
- Provides information about the class or type of system.
- Perceives a system as an instance of a class of systems which leads to the realization that a system inherits desired and undesired functions and properties from the generic class of system.
- Perceives similarities between the system and other systems in the same or other domains.
- Leads to the:
    - Type of thinking known as "out of the box".
    - Ability to perceive connections where others don't.
    - Understanding of analogies/parallelism between systems.
    - Adoption of lessons learned from other projects and determination if those lessons are applicable to the current project.
    - Adoption of innovative design approaches using approaches from other domains.
    - Use of pattern matching.
    - Use of benchmarking.

Perceptions of the similarities between systems engineering and other disciplines from the *Generic* perspective included:

1. The similarity between systems engineering and mathematics; both disciplines provide tools used to solve problems in other disciplines.
2. The focus on process is not unique to systems engineering For example, Drucker wrote, *"Throughout management science - in the literature as well as in the work in progress - the emphasis is on techniques rather than principles, on mechanics rather than decisions, on tools rather than on results, and, above all, on efficiency of the part rather than on performance of the whole"* (Drucker, 1973: page 509).

Table 8.1 Varieties of the problem solving process

<table>
<tr><th>Reference</th><th>SSM</th><th>GDRC, 2009</th><th>OVAE, 2005</th><th>Scientific Method</th></tr>
<tr><td>1. Planning</td><td colspan="4">Assumed</td></tr>
<tr><td rowspan="2">2. Situation analysis</td><td>Step 1</td><td>1. Problem definition</td><td>1. Identify and select the problem</td><td>1. Observe</td></tr>
<tr><td>Step 2</td><td>2. Problem analysis</td><td>2. Analyse the problem</td><td rowspan="3">2. Research</td></tr>
<tr><td rowspan="2">3. Conceptual solution design</td><td>Step 3</td><td>3. Generating possible solutions</td><td rowspan="2">3. Generate potential solutions</td></tr>
<tr><td>Step 4</td><td>4. Analysing the solutions</td></tr>
<tr><td>4. Solution selection</td><td>Step 5</td><td>5. Selecting the best solution(s)</td><td rowspan="2">4. Select and plan the solution</td><td>3. Formulate the hypothesis</td></tr>
<tr><td>5. Solution realization planning</td><td>Step 6</td><td>6. Planning the next course of action (next steps)</td><td>Plan the experiment</td></tr>
<tr><td>6. Solution realization</td><td>Step 7</td><td>-</td><td>5. Implement the solution</td><td>Perform the experiment</td></tr>
<tr><td>7. Test and evaluation</td><td>-</td><td>-</td><td>6. Evaluate the solution</td><td>4. Analyse the experimental results to test the hypothesis</td></tr>
<tr><td>8. In-service</td><td>-</td><td>-</td><td>-</td><td>-</td></tr>
<tr><td>9. Disposal</td><td>-</td><td>-</td><td>-</td><td>-</td></tr>
</table>

3. The identification of system engineering styles (Goleman, 2000; Mooz, et al., 2007; Kemp and Elphick, 2012) similar to the styles of management (Goleman, 2000; Mooz, et al., 2007; Kemp and Elphick, 2012).
4. The similarity between the SEP, the decision-making process and the problem-solving process discussed in Section 8.1.
5. The use of models discussed in Section 8.2.

8.1. The similarity between the SEP, the decision-making process and the problem-solving -process

The literature contains many versions of the problem-solving and decision-making process. Each description tends to depict parts of the same linear sequential linear series of activities as the SEP. Three examples are:

1. Hitchins' version of systems engineering which covers the early states of the SDP ending when the solution and strategies and plans to realise the solution system have as been conceptualised as shown in Figure 6.3.
2. The Global Development Research Center (GDRC) version which covers the problem identification-solution identification steps (GDRC, 2009).
3. The Office of Vocational and Adult Education (OVAE) version which goes beyond the GDRC version and contains steps that not only realize the solution but evaluate the solution to determine if the solution remedied the problem (OVAE, 2005).

Checkland's Soft Systems Methodology (SSM), the GDRC, OVAE and Scientific Method variations on the problem solving process are compared in Table 8.1 to show the similarities and differences in the grouping of tasks. For example, Steps 1, 2 and 3 of the GDRC and OVAE processes seem to align. Steps 4, 5 and 6 of the GDRC version are bundled into Step 4 of the OVAE version. Steps 5 and 6 in OVAE's version are absent in the GDRC version. Thus the GDRC version ends with the last box in Figure 6.2 while the OVAE version ends with realizing the solution and maps into the whole of Figure 6.3.

Notice that:

- The planning stage is generally left out of the various descriptions.
- The activities in the in-service and disposal states of the SLC generally do not show up in the various descriptions of the problem-solving process and in many versions of the SDP/SLC.

In addition, with reference to Figure 6.2 the problem-solving and decision-making processes are identical, where:

- Problem-solving is the name of the process from a helicopter or bird's eye external view of the entire process.
- Decision-making is the name of the same process from a viewpoint anchored to the decision-making blocks in Steps 5 and 6.

8.2. The use of models

The use of models is not limited to systems engineering. Models have been use in engineering for thousands of years[1], and latterly in the 20th century in software engineering in the form of the Unified Modeling Language™ (UML) (UML, 1999; 2005). The UML:

- Being a language has no inherent limitation on the number and types of objects, and is extendable. Holt applies UML to systems design (Holt, 2001) making modifications for systems engineering. Holt provides practical examples that show how the UML can be applied to non-software-based systems.
- Perspective on complex systems can be summarized as:
  - Best approached through a small set of nearly independent views.
  - No single view is sufficient.
  - Every model may be expressed at different levels of fidelity.
  - The best models are connected to reality.
- Is a modelling language for specifying, constructing, visualizing, and documenting the artefacts of a software-intensive system.
- Is not a process nor is it a methodology. This fact does not seem to be appreciated in the systems engineering community, as for example, Gibbons writes, "*The UML has provided a methodology that encompasses many of the up-front systems engineering activities in an otherwise object-oriented-based program*" (Gibbons, 2001).
- Can be used to document systems engineering products within conventional systems engineering methodologies since UML diagrams for models cover:
  - Use Cases.
  - Classes.
  - Behaviour in terms of state, activity, and interaction.
  - Charts - showing sequence and collaboration.
  - Implementation, aspects of components and deployment.
- Has a four-layered architecture, which can result in systems being constructed from the centre outward. The focus is on Use Cases, which drive the design. This is similar to the systems engineering approach in which the OCD sometimes also known as a CONOPS drive the requirements and hence the rest of the project work in the 'A' paradigm of systems engineering (Section 9.21).

[1] The engineers building the pyramids in ancient Egypt must have used models.

Gabb:

- Summarizes the purpose of the OCD as describing the operation of a system in the terminology of its users stating that it may include identification and discussion of the following (Gabb, 2001):
    - Why the system is needed and an overview of the system itself.
    - The full SLC from deployment through disposal.
    - Different aspects of system use including operations, maintenance, support and disposal.
    - The different classes of user, including operators, maintainers, supporters, and their skills and limitations.
    - Other important stakeholders in the system.
    - The environments in which the system is used and supported.
    - The boundaries of the system and its interfaces and relationships with other systems.
    - When the system will be used, and under what circumstances.
    - How and how well the needed capability is currently being met (typically by existing systems).
    - How the system will be used, including operations, maintenance and support.
- Provides the traditional systems engineering perspective when he writes that, "*An OCD is not a specification or a statement of requirement - it is an expression of how the proposed system will or might be used, and factors which affect that use. As such it is not obliged to follow the 'rules' of specification writing and can be relatively free in its language and format. Generally it will contain no 'shalls'*"

Lagakos et al. state that a Use Case is simply a set of system scenarios tied together by a common user goal (i.e., aspect of system functionality), and describes a way in which a real-world actor would interact with the system (Lagakos, et al., 2001). According to Lagakos et al., a Use Case specification contains:

- A list of actors (actors are anything that interfaces with the system externally).
- A boundary separating the system from its external environment.
- A description of information flows between the actors and individual Use Cases.
- A description of normal flow of events for the Use Case.
- A description of alternative and/or exceptional flows.

Gabbar et al. state that UML has been proven to be an efficient and comprehensive approach that can describe all three dimensions of the physical aspects of a production plant (static, behaviour and function) (Gabbar, et al., 2001).

8.3. Summary

This Chapter contained perceptions of the similarities between systems engineering and other disciplines from the *Generic* perspective. The key points were:

- The similarity between the SEP, the decision-making process and the problem-solving process.
- The use of models is not unique to systems engineering.
- There are different styles of system engineering in the same way as there are different styles of management.

--oo--

9. The Continuum Perspective

The *Continuum* perspective is a progressive perspective which looks at, and for, differences and recognizes that:

- Alternatives exist.
- Any solution or issue is located on at least one continuum of some kind.
- Things are not necessarily 'either-or'; there may be states in between.
- Changing conditions may cause movement along a continuum.
- There may be more than one objective for a system.
- There may be more than one way to achieve an objective.
- Systems sometimes fail partially as well as completely.
- Things can and must be seen from different viewpoints as discussed in Section 2.2.1.
- Changes are not necessarily improvements.
- Different people see things differently.
- There still may be other unknown variables that may or may not affect the situation. This situation is known as Simpson's paradox (Savage, 2009).

Perceptions of differences within systems engineering and between systems engineering and other disciplines from the *Continuum* perspective included:

1. The word "system" means different things to different people. For example, Webster's dictionary contains 51 different entries for the word "system" (Webster, 2004).
2. At least 40 different definitions of the term "systems engineering" discussed in Section 9.1.
3. The different definitions of the term "requirement" discussed in Section 9.2.
4. The different meanings of the word "problem" discussed in Section 9.3.
5. There may be more than one acceptable solution to a problem discussed in Section 9.4.
6. The different types of knowledge discussed in Section 9.5.

7. The different definitions of MBSE discussed in Section 9.6.
8. The differences in the content of postgraduate academic programs teaching systems engineering discussed in Section 9.7.
9. The different streams of activities in the SDP discussed in Section 9.8.
10. The difference between problem formulators and problem solvers discussed in Section 9.9.
11. The different layers of systems engineering discussed in Section 9.10.
12. The differences between the problem, solution and implementation domains discussed in Section 9.11.
13. The different states in the SLC discussed in Section 9.12.
14. The recursive nature of systems engineering discussed in Section 9.13.
15. The different milestones in the SDP discussed in Section 9.14.
16. The different lifecycle models discussed in Section 9.15.
17. The different levels of technological uncertainty discussed in Section 9.16.
18. The different camps in systems engineering discussed in Section 9.17.
19. The difference between Systems Engineering - the Role (SETR) and Systems Engineering - the Activity (SETA) discussed in Section 9.18.
20. The Roles Rectangle discussed in Section 9.19.
21. The overlap between systems engineering and project management discussed in Section 9.20.
22. The 'A' and the 'B' paradigms in systems engineering discussed in Section 9.21.
23. The difference in the contents of the publications discussed in Section 9.22.
24. The different processes for creating a system discussed in Section 9.23.
25. The paradoxes and dichotomies discussed in Section 9.24.
26. The different definitions of complexity discussed in section 9.25.
27. The difference between subjective and objective complexity discussed in Section 9.26.
28. The different types of objective complexity discussed in Section 9.27.
29. The previous proposed approaches to manage complexity in the INCOSE literature discussed in Section 9.28.
30. The different ways of measuring competency discussed in Section 9.29.

9.1. The many different definitions of "systems engineering"

The research identified at least 40 different definitions of systems engineering, some of which are shown below in chronological order.

- "*The methodology used to manage the problem of scheduling and coordinating hundreds of contractors developing hundreds - even thousands - of subsystems that eventually would be meshed into a total system*" (Hughes, 1998: page 118) discussing the management methodology of the ATLAS ICBM project of the 1950's.
- "*The combination of advanced chemical engineering science with the tool of electronic computers and the viewpoint of considering the process as an entity*" (Williams, 1961).
- "*Considers the content of the reservoir of new knowledge, then plans and participates in the action of projects and whole programs of projects leading to applications. It considers the needs of its customers and determines how these can best be met in the light of all knowledge both old and new. Thus systems engineering operates in the space between research and business, and assumes the attitudes of both. For those projects which it finds most worthwhile for development, it formulates the operational, performance and economic objectives, and the broad technical plan to be followed*" (Hall, 1962: page 4).
- "*The design of the whole as distinct from the design of the parts. Systems engineering is inherently interdisciplinary because its function is to integrate the specialized separate pieces of a complex of apparatus and people - the system - into a harmonious ensemble that optimally achieves the desired end*" (Ramo cited by (Hughes, 1998: page 69).
- "*The science of designing complex systems in their totality to ensure that the component sub-systems making up the system are designed, fitted together, checked and operated in the most efficient way*" (Jenkins, 1969).
- "*Covers the comprehensive aspects of engineering practice, and the application of the modern rational approach to the formulation and solution of technical problems*" (Au and Stelson, 1969: page 1).
- "*The application of scientific and engineering efforts to (a) transform an operational need into a description of system performance parameters and a system configuration through the use of an iterative process of definition, synthesis, analysis, design, test, and evaluation; (b) integrate related technical parameters and ensures compatibility of all physical, functional, and program interfaces in a manner that optimises the total system definition and design; (c) integrate reliability, maintainability, safety, survivability, human engineering, and other such factors into the total engineering effort to meet cost, schedule, supportability, and technical performance objectives*" (MIL-STD-499, 1969) Section 3.3).
- "*The transforming of an operational need into a description of system per-*

formance parameters and a system configuration" (FM_770-78, 1979).

- "*A hybrid methodology that combines policy analysis, design and management. It aims to ensure that a complex man-made system, selected from the range of options on offer, is the one most likely to satisfy the owner's objectives in the context of long-term future operational or market environments*" (M'Pherson, 1986: pages 330-331).
- "*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*" (Eisner, 1988: page 17).
- "*The management function which controls the total system development effort for the purpose of achieving an optimum balance of all system elements. It is a process which transforms an operational need into a description of system parameters and integrates those parameters to optimise the overall system effectiveness*" (DSMC, 1996: pages 1-2).
- "*A robust approach to the design and creation of systems to accomplish desired ends*" (Chamberlain and Shishko, 1991: page 23).
- "*An interdisciplinary approach to evolve and verify an integrated and lifecycle balanced set of system product and process solutions that satisfy customer needs. Systems engineering*:
  a) encompasses the scientific and engineering efforts related to the development, manufacturing, verification, deployment, operations, support, and disposal of system products and processes,
  b) develops needed user training equipments, procedures, and data,
  c) establishes and maintains Configuration Management of the system,
  d) develops work breakdown structures and statements of work, and
  e) *provides information for management decision making*" (MIL-STD-499B, 1992).

- "*A management technology*" (Sage, 1992: page 1).
- "*The design, production, and maintenance of trustworthy systems within cost and time constraints*" (Sage, 1992: page 10).
- "*Integrates all the disciplines and specialty groups into a team effort forming a structured development process that proceeds from concept to production to operation. Systems engineering considers both the business and the technical needs of all customers with the goal of providing a quality product that meets the user needs*" (Sage, 1992).
- "*The intellectual, academic and professional discipline the principal concern of which is the responsibility to ensure that all requirements for a bioware/hardware/software system are satisfied throughout the life of the system*" (Wymore, 1993: page 5).
- "*Comprises systems analysis, systems integration and human factors including human-computer interaction*" (Anderson and Dibb, 1996).

- "*A set of activities which control the overall design, development, implementation and integration of a complex set of interacting components or systems to meet the needs of all the users*" (DERA, 1997).
- "*The activity of specifying, designing, implementing, validating, installing and maintaining systems as a whole*" (Sommerville, 1998).
- "*An interdisciplinary approach and means to enable the realisation of successful systems. It focuses on defining customer needs and required functionality early in the development cycle, documenting requirements, then proceeding with design synthesis and system validation while considering the complete problem*" (INCOSE, 2000).
- "*The design and analysis process which decomposes an application into software and hardware*" (Brodie, 2001: page 249).
- "*The process that identifies the technical characteristics and operating rules of that system that best achieves the objectives in question*" (Westerman, 2001: page 6).
- "*The art and science of creating systems*" (Hitchins, 2003).
- "*Deals with the planning, development, and administration of complex systems, particularly of computing systems*" (Endres and Rombach, 2003: page 1).
- "*Provides a framework, within which complex systems can be adequately defined, analysed, specified, manufactured, operated, and supported*" (Faulconbridge and Ryan, 2003).
- "*Guides the engineering of complex systems"* (Kossiakoff and Sweet, 2003).

9.2. The different definitions of the term "requirement"

There does not seem to be a widely accepted baseline definition of a requirement. For example the IEEE definition of a requirement (IEEE 610, 1990) is:

> *"(1) A condition or capability needed by a user to solve a problem or achieve an objective.*
>
> *(2) A condition or capability that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed documents.*
>
> *(3) A documented representation of a condition or capability as in (1) or (2)."*

Yet variations of the definition continue to appear in the literature, including:

- "*Something that is wanted or needed, called for or demanded as being essential*" (Mason, et al., 1999).
- "*A statement which translates (or expresses) a need or constraints (technical,*

costs, times...)" (Fanmuy, 2004).

- "*Something obligatory or capabilities the system must satisfy*" (Powell and Buede, 2006).

Kossmann et al. cite a number of additional definitions in the literature and also provide a useful overview of the state-of-the-art of requirements engineering based on a wide collection of publications from previous years (Kossmann, et al., 2007).

9.3. The different meanings of the word "problem"

The word 'problem' has different meanings since the word 'problem' has been defined or used to mean:

1. A question proposed for solution or discussion (dictionary.com, 2013).
2. Any question or matter involving doubt, uncertainty, or difficulty (dictionary.com, 2013). For example, this type of problem might be:
   - ***An undesirable situation.*** You might hear someone end a sentence with, "*… and that's the problem*" when they mean, "*… and that's the undesirable situation*".
   - ***The underlying cause of an undesirable situation,*** usually a failure of some kind. For example, one may hear someone say, "*my phone stopped working; the problem was a discharged battery*". In reality, they mean that the cause of the phone stopping working was a discharged battery; the symptom or effect was that the phone stopped working.
3. The need to determine the necessary sequence of activities to convert an initial undesirable situation into a desirable situation[1].

9.4. There may be more than one acceptable solution to a problem

The relationship between problems and solutions seems to be based on the assumption that there is a well-defined problem and a single well-defined correct solution as shown Figure 6.4 which starts with a problem, shows that there are a number of solutions, one of which is the single correct solution and all of the other solutions are incorrect. This focus on a single correct solution is adopted from mathematics.

Perceptions from the *Continuum* perspective indicate that systems engineering deals with problems that generally have a range of equally ac-

[1] Once the necessary sequence of activities is determined, the subsequent problem is to plan the process to perform the necessary sequence of activities. Once the plan is created, the subsequent problem is to realize the desirable situation by carrying out the plan.

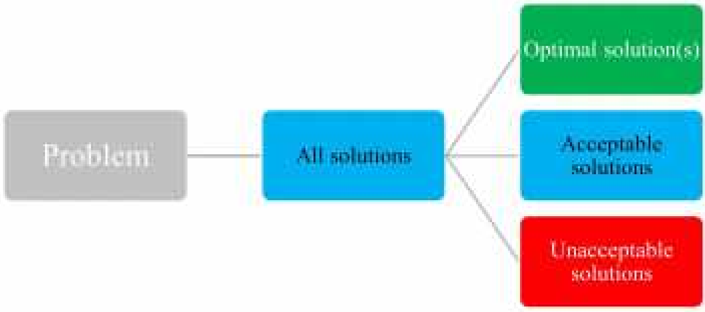


Figure 9.1 The range of solutions

ceptable solutions. For example, you are hungry, which is generally an undesirable situation. Your problem is to figure out a way to remedy that undesirable situation by consuming some food to satisfy the hunger. There are a number of solutions to this problem including cooking something at home, going out to a restaurant, collecting some takeaway food, and telephoning for home delivery. Then there is the choice of what type of food; Italian, French, Chinese, pizza, lamb, chicken, beef, fish, vegetarian, etc. Now consider the vegetables, sauces and drinks. There are many solutions because there are many combinations of types of food, meat, vegetables and method of getting the food to the table. Which solution is "the right one"? The answer is "it depends". In nearly every situation, an acceptable solution is one that satisfies your hunger in a timely and affordable manner, meets any other dietary requirements you may have and does not cause any gastric problems. If several of the solution options can perform this function and you have no preference between them, then each of them are just as correct or acceptable as any of the other ones that satisfy your hunger. The words 'right solution' or 'correct solution' should be thought of as meaning 'one or more acceptable solutions' as shown in Figure 9.1.

In addition, conventional systems engineering and project management wisdom suggests that when a decision cannot be made because two choices score almost the same in the decision-making process, the decision maker should perform a sensitivity analysis at this point varying the parameters and/or the weighting to see if the decision changes. By recognizing that there may be more than one acceptable solution, the situation may eliminate the need for the sensitivity analysis.

Perceptions from the *Continuum* perspective indicate that an alternative relationship between problems and solutions may be represented as shown in Figure 9.1 which leads to the concept of acceptable solutions instead of using the relationship shown in Figure 6.4 that aims at a single correct solution. Figure 9.1 can also be used to show the relationship between 'satisfy' and 'satisfice' where:

- ***Satisfy*** means provide solutions that are optimal.
- ***Satisfice*** means provide solutions that are acceptable.

9.5. The three different types of knowledge

Knowledge has been categorised in several different ways usually by content. One content-free classification was by Woolfolk who described the following three types of knowledge (Woolfolk, 1998):

1. ***Declarative knowledge:*** knowledge that can be declared in some manner. It is knowing that something is the case. Describing a process is declarative knowledge.
2. ***Procedural knowledge:*** knowing how to do something. It must be demonstrated; performing the process demonstrates procedural knowledge.
3. ***Conditional knowledge:*** knowing when and why to tailor and apply the declarative and procedural knowledge.

This perception of different types of knowledge facilitates assessing the competencies of systems engineers (Section 14.2).

9.6. The different definitions of MBSE

Perceptions from the *Structural* perspective provide a number of different definitions of MBSE, including:

- "*MBSE is the formalized application of modeling to support system requirements, design, analysis, verification and validation activities beginning in the conceptual design phase and continuing throughout development and later life cycle phases*" (INCOSE, 2007: page 15).
- "*MBSE is the formalized application of modeling to support system requirements, design, analysis, verification and validation, beginning in the conceptual design phase and continuing throughout development and later life cycle phases*" (Friedenthal, et al., 2007).
- "*MBSE is fundamentally a thought process. It provides the framework to allow the systems engineering team to be effective and consistent right from the start of any project*" (Long and Scott, 2011: page 65).
- "*MBSE is a Systems Engineering paradigm that emphasizes the application of rigorous visual modeling principles and best practices to Systems Engineering activities throughout the [System Development Life Cycle] SDLC*" (MBSE, 2011).

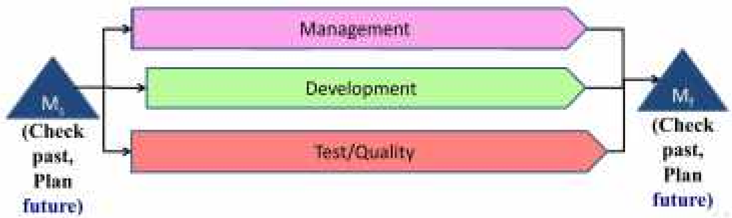


Figure 9.2 The three streams of activities

9.7. The differences in the content of postgraduate academic programs teaching systems engineering

A benchmarking study of the content of postgraduate Masters degrees in systems engineering was performed in 2014 (Kasser and Arnold, 2014). The conclusions from the study showed that in general:

- Different degrees teach different things.
- One can get a Master of Industrial and Systems Engineering (MISE) without a single required course on systems engineering if one picks the right institution.
- Knowledge topics are bundled into courses in various ways.
- There are lots of differences in knowledge content in various degrees, which may be due to local sponsor's requirements or the lack of any requirements for bundling the knowledge into a master's degree.
- The focus of the coursework seemed to be on:
  - Cookbook solutions rather than on reasoning namely Type II systems engineering rather than Type V (Section 10.9).
  - Processes (take one and apply it) instead of creating one to fit the specific situation.

9.8. The different streams of activities in the SDP

The SDP may be mapped into three streams of activities that take place in a sequence of states where each state contains different serial and parallel streams of activities which come together at the major milestones as shown in Figure 9.2 where:

- ***Management:*** the set of activities which include:
  - Monitoring and controlling the development and test stream activities to ensure performance in the state in accordance with the Project Plan (PP).
  - Updating the PP to elaborate the Work Packages (WP) for the three streams of activities in the subsequent state in more detail.

  - Endeavouring to ensure that needed resources in the subsequent state will be available on schedule.
  - Providing periodic reports on the condition of the project to the customer and other stakeholders.
  - Being the contractual interface with the customer.
  - Performing the appropriate risk management activities on the process.
- ***Development:*** the set of activities which produce the products appropriate to the state by performing the design and construction tasks.
- ***Test:*** the set of activities which include:
  - Identifying defects in products.
  - Verifying the degree of conformance to specifications of the products produced by the development stream in the State by performing appropriate tests or analyses.

  This stream of activities is also often known as Quality Control (QC) or Quality Assurance (QA), Test and Evaluation (T&E) and Independent Verification and Validation (IV&V).

9.9. The difference between problem formulators and problem solvers

Gordon et al. provided a way to identify the difference in cognitive skills between innovators, problem formulators, problem solvers and imitators (Gordon G. et al., 1974). The difference is based on the:

- Ability to find ***differences*** among objects which seem to be ***similar.***
- Ability to find ***similarities*** among objects which seem to be ***different.***

The differences in the '***ability to find*** …' leads to the different type of personalities shown in Table 9.1 (Gordon G. et al., 1974). For example:

- ***Problem formulators*** score high in ability to find differences among objects which seem to be similar, namely they are good at using the *Continuum* perspective.
- ***Problem solvers*** score high in ability to find similarities among objects which seem to be different, namely they are good at using the *Generic* perspective.

From a slightly different perspective, Gharajedaghi discussed four personality types based on the same abilities in the context of separating the problem from the solution (Gharajedaghi, 1999: pages 116-117) where:

Table 9.1 Factors conducive to innovation (Gordon G. et al., 1974).

Ability to find ***similarities*** among objects which seem to be ***different***	HIGH	Problem solvers	Innovators
	LOW	Imitators/Doers	Problem Formulators
		LOW	HIGH
		Ability to find ***differences*** among objects which seem to be ***similar***	

- ***Leaders and pathfinders*** (innovators in Table 9.1) have a holistic orientation to seeing the bigger picture and putting issues in the proper perspective.
- ***Problem solvers*** are scientifically oriented with a tendency to find similarities in things that are different. They are concerned with immediate results.
- ***Problem formulators*** are artistically oriented having a tendency to find differences in things that are similar. They are concerned with the consequences.
- ***Doers*** are practitioners producing tangible results.

Both Gordon et al. and Gharajedaghi discuss the same abilities in the context of separating the problem from the solution, however they do not provide a way to evaluate a person's skills in those areas which overlap with the *Generic* and *Continuum* perspectives.

9.10. The different layers of systems engineering

There are differences between systems engineering as performed on products, systems and large-scale systems. Hitchins proposed the following five-layer model for systems engineering (Hitchins, 2000) where:

- "***Layer 5 -*** *Socioeconomic, the stuff of regulation and government control.*
- ***Layer 4 -*** *Industrial Systems Engineering or engineering of complete supply chains/circles. Many industries make a socio-economic system. A global wealth creation philosophy. Japan seems to operate most effectively at this level.*
- ***Layer 3 -*** *Business Systems Engineering - many businesses make an industry. At this level, systems engineering seeks to optimize performance somewhat independent of other businesses*
- ***Layer 2 -*** *Project or System Level. Many projects make a Business. Western engineer-managers operate at this level, principally making complex artefacts.*

- ***Layer 1 -*** *Product Level. Many products (subsystems) make a system. The tangible artefact level. Many [systems] engineers and their institutions consider this to be the only "real" systems engineering".*

This model can be extended downwards to add:

- ***Layer 0 -*** the component layer where many components make a product.

9.11. The differences between the problem, solution and implementation domains

There are three domains relevant to systems engineering, namely:

1. The problem domain.
2. The solution domain.
3. The implementation domain.

It is tempting to assume that the problem domain and the solution domain are the same, but they are not necessarily so. For example, the problem domain may be urban social congestion, while the solution domain may be a form of underground transportation system to relieve that congestion. Lack of problem domain competency may lead to the identification of the wrong problem and lack of solution domain competency may lead to selection of a less than optimal, or even an unachievable, solution system. Risk management is an activity (process) that requires competency in the problem, solution and implementation domains.

9.12. The different states in the System Lifecycle

The SLC may be perceived as taking place in a series of sequential states as shown in the waterfall view in Figure 9.3 where each state commences and terminates at a formal milestone review (Section 9.14).

The waterfall view (Royce, 1970) was among the first attempts to document the software production process. It represented the process as a serial sequence of states as shown in the typical representation of Figure 9.3. Each state ideally starts and ends at a milestone review to confirm that the work allocated to a specific state at the previous milestone is complete and the process is ready to advance to the next state. The name of the view was adopted because the pictorial representation shows each state seeming to flow naturally into the next state like water flowing over a series of falls.

The waterfall view of the SLC is:

- A planning perspective looking forward in time from the point before the Needs Identification State of the SDP/SLC (Section 9.12.1).
- An ideal model that does not take into account the effect of the

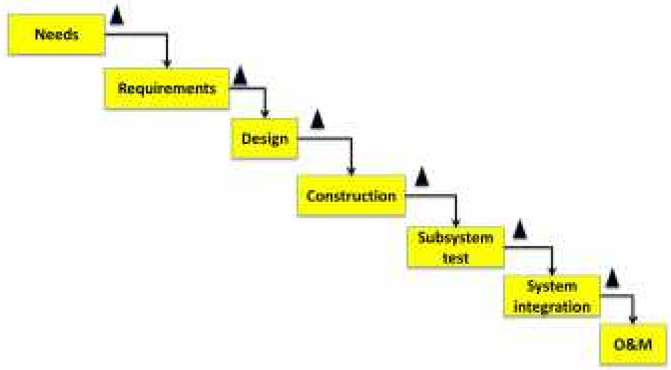


Figure 9.3 The waterfall view of most of the SLC

change in the customer's needs during the time between the start of the Requirements State and the start of the Operations and Maintenance (O&M) State of the SLC (Section 9.12.7).

- Not representative of the real world. However, since it is simple to explain, teaching of systems engineering has focused on using the waterfall model and the V view of the waterfall model (Biemer and Sage, 2009: pages 152 and 153).

Perceptions from the *Continuum* and *Quantitative* perspectives identified nine different states defined in generic terms as:

A. The Needs Identification State.
B. The Requirements State.
C. The System Design State.
D. The Subsystem Construction State.
E. The Subsystem Test State.
F. The System Integration and System Test States.
G. The Operations, Maintenance (O&M) and Upgrade States.
H. The Disposal State.

These states have been stated in various ways in various Standards, conference papers and books. Consider each of them.

9.12.1. The Needs Identification State

The Needs Identification State:

- Is based on Hall, Gelbwaks and Hitchins (Hall, 1962; Gelbwaks, 1967; Hitchins, 1992) and the summary in Brill (Brill, 1998).
- Is where the bulk of the set of activities known as systems engineering is performed. These activities transform an operational need into a description of system performance parameters and a system configuration (FM_770-78, 1979).

- Activities address the problem and determine the conceptual solution.
- Tends to be glossed over in courses and books following the 'B' paradigm (9.21.2).
- Ends at the Operations Concept Review (OCR).

9.12.2. The Requirements State

The Requirements State addresses the physical solution and its implementation and contains the set of activities that:

- Specify the conceptual solution system as a full set of specifications for the whole and for the parts and their infrastructure, including the environment/Weltanschauung or paradigm that justifies them. If the specifications are in the form of text mode requirements, the output of this State tends to be at the 'A' specification level (MIL-STD-490A, 1985).
- Plan (create) the process that will be followed to realize the solution system and produce the plans at the appropriate level of detail for the project.

The actual activities in this state depend on whether the project is following the 'A' or 'B' paradigms discussed in Section 9.21.

- In the 'A' paradigm, the CONOPS and the conceptual solution system designs produced in the Needs Identification State are converted to a matched set of specifications for the system and subsystems and their infrastructure.
- In the 'B' paradigm, the SDP commences here, so since there was no Needs Identification State, a CONOPS does not exist. Accordingly, this State is where the system engineer elicits and elucidates the requirements to create the System Requirements Document (SRD). Depending on the variation of the 'B' paradigm, the conceptual design and CONOPS are then created from the requirements (Denzler and Mackey, 1994; Guo, 2010) in either this State or the subsequent System Design State.

The Requirements State ends at the System Requirements Review (SRR).

9.12.3. The System Design State

The System Design State activities:

- Begin at the conclusion of the SRR.
- Are different in the 'A' and 'B' paradigms:
  - ***In the 'A' paradigm,*** since the conceptual design was performed in the Needs Identification State, the system design

activities focus on converting the functions in the conceptual design and CONOPS into physical designs.

- ***In the 'B' paradigm,*** since the CONOPS and conceptual design were not created in the Needs Identification State, if they were not created in the Requirements State, they are created from the CONOPS (Denzler and Mackey, 1994; Guo, 2010) in the System Design State and then the subsystem design activities focus on converting the functions in the conceptual design into physical designs.

- Convert the functions in the conceptual design into physical designs by performing the set of activities that create a more detailed design of the whole solution system through a combination of people, doctrine, parts, subsystems, interactions, etc., including configuration, architecture and implementation criteria.
- Are split into two sub-states:
  a) The Preliminary System Design sub-state which ends at the Preliminary Design Review (PDR).
  b) The Detailed Design System sub-state which ends at the Critical Design Review (CDR).

9.12.4. The Subsystem Construction State

The Subsystem Construction State:

- Begins at the conclusion of the CDR.
- Contains the set of activities that create the individual parts, subsystems, interactions, etc. *in isolation.* Consequently the set of activities are mainly engineering, training, etc., not systems engineering[2]. Systems engineering does monitor the subsystem development to ensure conformance to system level specifications.
- Ends at the Test Readiness Review (TRR).

9.12.5. The Subsystem Testing State

The Subsystem Testing State:

- Begins at the conclusion of the TRR.
- Contains the last set of Development Test and Evaluation (DT&E) activities that validate the performance of the individual parts, subsystems, interactions, etc. in isolation against their requirements. Consequently the set of activities are mainly engineering, not systems engineering. Systems engineering does monitor the subsystem testing to ensure conformance to specifi-

---

[2] In this layer, however, since a subsystem may also be a system according to the Principle of hierarchies, there may very well be systems engineering on the subsystem.

cations.

- Ends at the Integration Readiness Review (IRR).

9.12.6. The System Integration and System Test States

These States begin following the conclusion of the IRR and contain the following sub-states:

- (F1) The System Integration sub-state activities which combine the parts, subsystems, interactions, etc., to constitute the solution system that remedies the evolved undesirable situation.
- (F2) The System Test sub-state activities which perform Operational Test and Evaluation (OT&E) to establish, under test conditions, the performance of the whole solution system, with optimum effectiveness, in its operational context.
- (F3) The Handover or Deployment sub-state activities which deploys the system into its operational environment and hands it over to the customer.

These States:

- Are where systems engineering picks up again as the system is integrated and tested as shown in Figure 10.7[3].
- End when the system is deployed into service.

9.12.7. The Operations, Maintenance (O&M) and Upgrade State

The O&M State:

- Begins when the system is deployed into service.
- Is also known as the In-Service State.
- Contains the set of systems engineering and non-systems engineering activities that actively provide a solution/remedy to the problem for which the whole system was created. These being the activities involved in operating the system, support to maintain operations, improvements to the whole to enhance effectiveness, and to accommodate changes in the nature of the problematic or undesirable situation over time, ideally without rendering the operating solution system materially inoperative for an unacceptable period of time.
- Ends when:
    - The need for the system no longer exits.
    - The system can no longer perform its desired functions in an economic manner.

---

[3] The figure contains quantitative information so it is located and discussed in the *Quantitative* perspective.

9.12.8. The Disposal State

The Disposal State contains the set of activities that dispose of the system. This State is rendered necessary when one of the following occurs:

- The problem no longer exists.
- The solution system is no longer capable of remedying the problem effectively or economically.

9.13. The recursive nature of systems engineering

The first part of each state of the SDP may also be mapped into the problem-solving process shown in Figure 6.3 where each state:

- Contains the set of problem-solving activities shown in which may be mapped into three streams of activities discussed in Section 9.8.
- Begins with an undesirable situation driving the first set of activities which explore the problem space to develop an understanding of the causes of the undesirability and produce a definitive statement of the problem in context which is the need to transition to a situation that does not exhibit the undesirable characteristics of the undesirable situation.
- Conceives a number of solutions that could remedy the undesirable situation.
- Identifies the solution selection criteria for selection which of the options would be the optimal (in context) solution.
- Performs the trade-off to make find the optimal solution.
- Selects the preferred option.
- Formulates the strategies and plans to implement the selected option which is the desirable situation.

Once the stakeholders have approved the strategies and plans and a project is approved, the plans are carried out to realise the solution in the second half of the problem-solving process or the solution creation process. Since each state begins with exploring the problem and ends with a solution, the solution output of any state becomes the problem input to the subsequent state. This situation, shown in Figure 9.4 is often referred to as the:

- ***"What's"*** – which refer to what needs to be done, or the problem.
- ***"How's"*** – which refer to how it is done, or the solution.

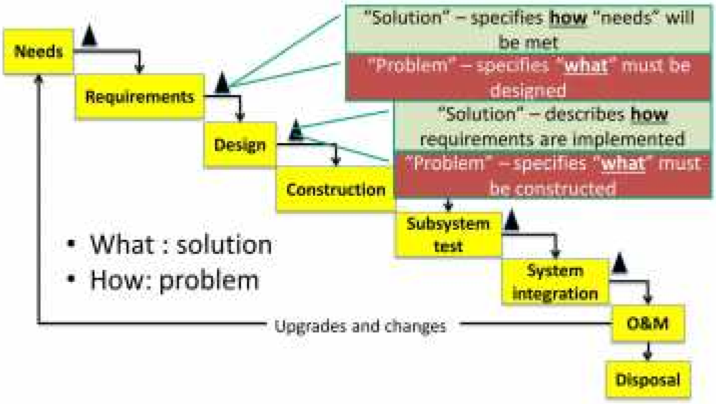


Figure 9.4 The waterfall view – problem-solving perspective (partial)

9.14. The different milestones in the SDP

Each state commences and terminates at a major formal milestone. Different projects use different names for the formal and informal milestones, but a milestone by any name is still a milestone. Typical formal milestones as used in this book are:

- ***Start of project:*** formally starts the project and commences the Needs Identification State.
- ***Operations Concept Review (OCR):*** terminates the Needs Identification State and commences the Requirements State.
- ***Systems Requirements Review (SRR):*** terminates the Requirements State and commences the Preliminary Design sub-state of the System Design State.
- ***Preliminary Design Review (PDR):*** terminates the Preliminary Design sub-state of the System Design State and commences the Critical Design sub-state of the System Design State.
- ***Critical Design Review (CDR):*** terminates the System Design State and commences the Subsystem Construction State.
- ***Test Readiness Review (TRR):*** terminates the Subsystem Construction State and commences the Subsystem-Testing State.
- ***Integration Readiness Review (IRR):*** terminates the Subsystem-Testing State and commences the System Integration State.
- ***Delivery Readiness Review (DRR):*** terminates the System Integration and System Test States and commences the activities that deliver the system and lead to terminating the project.
- ***End of project:*** formally terminates the project.

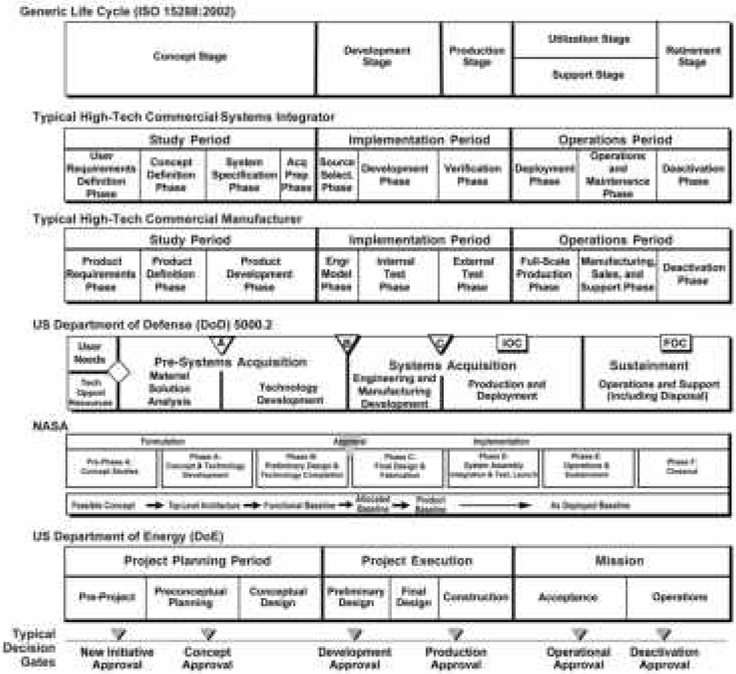


Figure 9.5 Comparison of lifecycle models (INCOSE SE Handbook Version 3.2.1)

9.15. The different lifecycle models

The literature contains a number of different lifecycle models all shown as a high-level linear sequence of activities. Each model may use a different name for an activity or group a set of activities differently to the other models. Figure 9.5 (Haskins, 2011) compares the generic life-cycle stages in ISO-15288:2002 to other life-cycle viewpoints showing that the concept stage is aligned with the commercial project's study period and with the US DoD and Department of Energy's pre-systems acquisition

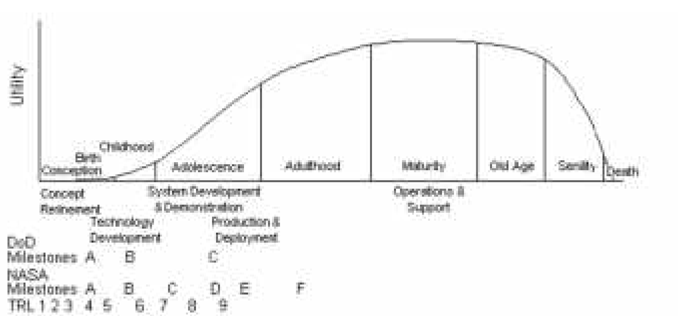


Figure 9.6 The technology lifecycle (Nolte, 2005)

and the project planning period. Typical decision gates are presented in the bottom line. The focus of the models in the figure is on the front-end early states of the lifecycle; the system acquisition states. This focus can be seen for DoD and NASA in the whale diagram shown in Figure 9.6 (Nolte, 2005) which covers the technology lifecycle where the acquisition milestones, A, B and C, all show up in the childhood and adolescence states of the technology lifecycle.

9.16. The different levels of technological uncertainty

Shenhar and Bonen characterised projects in a four-level scale of technological uncertainty (Shenhar and Bonen, 1997) which comprises:

- ***Type A: Low Technological Uncertainty.*** Typical projects in this category are construction, road building, and other utility works that are common in the construction industry that require one design cycle or pass through the Waterfall development methodology discussed in Section 7.4.
- ***Type B: Medium Technological Uncertainty.*** Typical projects of this kind tend to be incremental improvements and modifications of existing products and systems.
- ***Type C: High Technological Uncertainty.*** Typical projects of this kind tend to be high-tech product development and Defence state-of-the-art weapons systems.
- ***Type D: Super High Technological Uncertainty.*** These projects push the state-of-art and are few and far between in each generation. A typical example from the 20th century is the NASA Apollo program which placed men on the moon.

In addition, as perceived from the *Continuum* perspective, *"Systems engineering is a wide-range activity, and it should not be handled in the same form for all kinds of systems"* (Shenhar and Bonen, 1997). The differences between the four types of projects including the different approaches to systems engineering in each project are summarized in Table 9.2.

9.17. The different camps of systems engineering

Perceptions from the *Continuum* HTP identified eight different somewhat overlapping camps of systems engineering (Kasser and Hitchins, 2012) based on sorting the different views of/opinions on/worldviews of systems engineering. Each opinion seems to represent a viewpoint based on the experience of the writer[4]. The somewhat overlapping camps are:

1. Lifecycle.
2. Process.

[4] At least in my case (Kasser, 1995)

Table 9.2 Shenhar and Bonen's project classification by Technology Uncertainty

	Type A	**Type B**	**Type C**	**Type D**
	Low - Tech	**Medium - Tech**	**High - Tech**	**Super – High - Tech**
Technology	All exist	Integrates some new with mostly existing	Integrates mostly new with some existing	Key technologies do not exist at project's initiation
Development	None	Some	Considerable	Extensive
Testing	None	Some	Considerable	Extensive
Prototyping	None	Some	Considerable	Extensive
Requirements	Known prior to project start	Joint development effort between customer and contractor	Strong involvement of contractor	Extensive contractor involvement many changes and iterations
Design cycles	1	1 or 2	At least 2	2 to 4
Design freeze	Prior to project start	1st Quarter	1st or 2nd Quarter	2nd or 3rd Quarter
Changes	None	Some	Many	Continuous
Management and systems engineering style	Firm and formal	Moderately firm	Moderately flexible	Highly flexible

3. Problem.
4. Discipline and Meta-Discipline.
5. Systems thinking.
6. Non-systems thinking.
7. Domain.
8. Enabler.

Consider some perceptions of each of these camps.

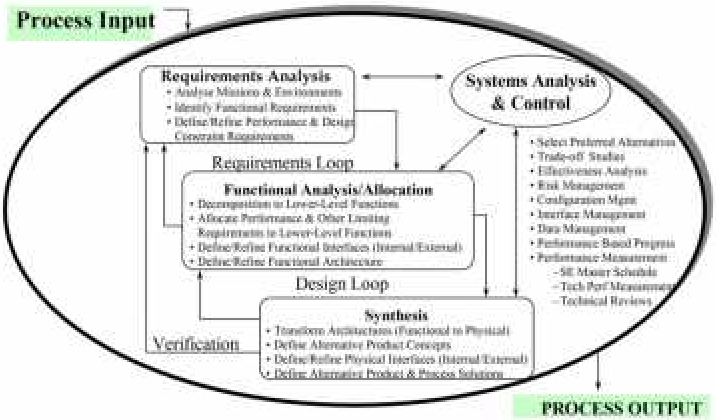


Figure 9.7 ANSI/EIA-632 egg diagram

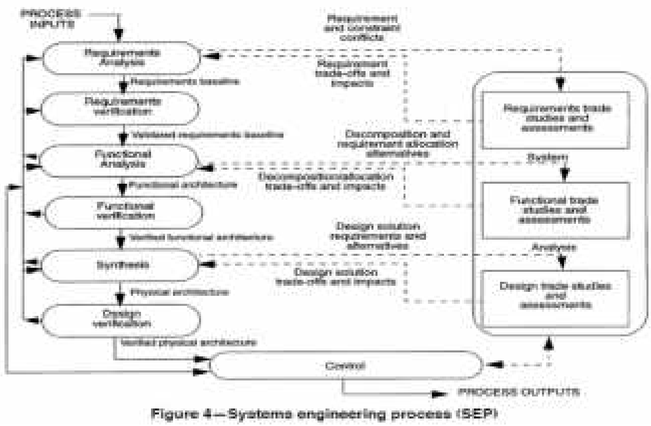


Figure 9.8 IEEE 1220 Systems Engineering Process

9.17.1. The Lifecycle camp

This camp is one of the earliest camps, articulated when Chapanis wrote, *"Despite the difficulties of finding a universally accepted definition of systems engineering, it is fair to say that the systems engineer is the man who is generally responsible for the over-all planning, design, testing, and production of today's automatic and semi-automatic systems"* (Chapanis, 1960: page 357).

This is the camp of the systems engineers who seem to have an understanding of the activities performed the early states of the SDP particularly in the Needs Identification State (Section 9.12) and also conform to the 'A' paradigm (Section 9.21). The early state campers tend to be the old timers; while the others tend to be those systems engineers educated in the last 20-30 years in the 'B' paradigm.

9.17.2. The Process camp

Some systems engineers, particularly in INCOSE and the US DoD, are process-focused (Lake, 1994) seemingly in accordance with US DoD 5000 Guidebook 4.1.1, which states, *"The successful implementation of proven, disciplined system 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 focus is on conforming to the process and not on providing an understanding of the context. Even though the ability to tailor the process was called out in MIL-STD-499 (MIL-STD-499, 1969) that aspect tends to be ignored. These campers are often graduates from 'B' paradigm systems engineering courses which focus on the process.

Some of these campers tend to insist that organisations must modify themselves to follow a particular process standard. However, these campers can't seem to see the big picture and don't seem to realise there is also currently no single widely agreed upon SEP since the SEP has been stated in many ways including:

- The V view of the waterfall process.
- The spiral, incremental and evolutionary models.
- The lists of processes in ISO/IEC 15288 (Arnold, 2002) shown in Figure 7.2.
- The waterfall process (Royce, 1970) shown Figure 9.3.
- The EIA processes shown in Figure 9.7 (EIA 632, 1994).
- The IEEE processes shown Figure 9.8 (IEEE 1220, 1998).
- The State, Investigate, Model, Integrate, Launch, Assess and Re-evaluate (SIMILAR) (Bahill and Gissing, 1998) view shown in Figure 9.9.
- The system lifecycle functions (Blanchard and Fabrycky, 1981) shown in Figure 9.10.
- A systems engineering approach to addressing a problem (Hitchins, 2007) discussed in Section 6.1.

These campers also ignore:

- The literature on "excellence" which focuses on the need for good people (Rodgers, et al., 1993; Peters and Waterman, 1982; Peters and Austin, 1985), i.e. Type V systems engineers (Section 10.9).
- The axiom "Garbage-In-Garbage-Out" (GIGO) which although originally was applied to computer data, holds true for all types of processes.

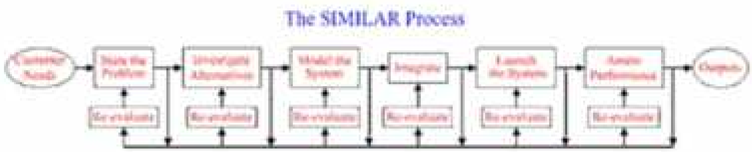


Figure 9.9 The SIMILAR process (Bahill and Gissing, 1998)

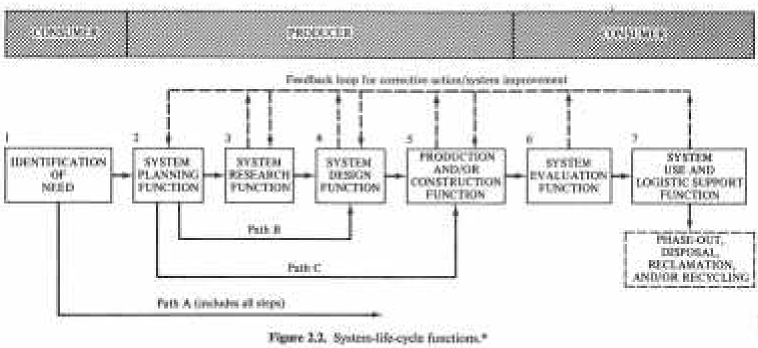


Figure 9.10 System Lifecycle functions (Blanchard and Fabrycky, 1981)

- Attempts to warn against "overemphasis on the institutionalization of processes rather than the value or effectiveness of the effort" (Armstrong, 1998; Drucker, 1973: page 509).
- Processes seen to work in one culture or organization have been copied verbatim by other organizations, with dismal results. Examples can typically be found in the lessons learned outside the system engineering literature, e.g., (O'Toole, 2004) and some of the reasons for a claimed Six Sigma initiative 60% failure rate (Angel and Froelich, 2008).
- The overlap between the SEP and the problem-solving process discussed in Section 12.6.

9.17.3. The Problem-Solving camp

The problem solving camp can be traced back at least as far as 1980 (Gooding, 1980). Examples in the literature include:

- Wymore summing up the philosophy of the principal functions of systems engineering as, *"to develop statements of system problems comprehensively, without disastrous oversimplification, precisely without confusing ambiguities, without confusing ends and means, without eliminating the ideal in favour of the merely practical, without confounding the abstract and the concrete, without reference to any particular solutions or methods, to resolve top-level system problems into simpler problems that are solvable by* *technology: hardware, software, and bioware, to integrate the solutions to the simpler problems into systems to solve the top-level problem"* (Wymore, 1993: page 2).

- IEEE 1220 which stated that, *"the systems engineering process is a generic problem-solving process"* (IEEE 1220, 1998) Section 4.1).

These campers maintain focus on the problem and identifying the best solution available given the constraints at the time (Hitchins, 2007). Some of these campers also address carrying out that process to realize the solution system (Bahill and Gissing, 1998).

9.17.4. The Discipline and Meta-Discipline camp

Wymore defined systems engineering as a discipline (Wymore, 1994). Systems engineering meets the requirements for a discipline as discussed in Section 12.17. However, all the elements of the current mainstream DoD and INCOSE SETR approach to systems engineering overlap those of project management and other disciplines which make it difficult to identify systems engineering as a distinct discipline for tackling complex problems.

The discipline camp tends to account for the overlap by viewing systems engineering as a meta-discipline incorporating the other disciplines and hold that systems engineering needs to widen its span to take over the other disciplines.

9.17.5. The Systems Thinking camp

The systems thinking camp tends to be systems engineers who can view an issue from several perspectives (Evans, 1996; McConnell, 2002; Rhodes, 2002; Martin, 2005; Selby, 2006; Beasley and Partridge, 2011).

9.17.6. The Non-Systems Thinking camp

The non-systems thinkers tend to have a single viewpoint of systems engineering and generally exhibit the 'biased jumper' level of critical thinking (Section 14.2.1.2.2).

9.17.7. The Domain Systems Engineering camp

There is a domain systems view of the role of systems engineers/engineering based on SETR. Thus an engineer working on a widget system is a widget system engineer. Examples are network systems engineers/engineering, control system engineers/engineering, communications systems engineers/engineering, hydraulic systems engineers/engineering, transportation systems engineers/systems engineering, etc. However, the name of the widget system is dropped from the role. For example, my first job as a systems engineer was as an Apollo Lunar Surface Experiment Package (ALSEP) Control System systems engineer. Each experiment in the ASLEP had its own systems engineer and there was a meta-systems engineer for the ALSEP itself.

9.17.8. The Enabler camp

The enabler camp evolved from the problems-solving camp (Section 9.17.3). In the enabler camp, systems engineering is the application of holistic thinking to problem-solving. Moreover, it can be, and is, used in all disciplines for tackling certain types of complex and non-complex problems; see *"[systems engineering] is a philosophy and a way of life"* (Hitchins, 1998).

9.18. The difference between Systems Engineering - the Role (SETR) and Systems Engineering - the Activity (SETA)

This perception perceives differences between roles and activities and separates Systems Engineering - the Role (SETR) and Systems Engineering - the Activity (SETA) paradigms (Kasser and Hitchins, 2009; Kasser, et al., 2009). While the role of the systems engineer and the role of the project manager may overlap when viewing the roles of systems engineers and project managers in different organizations, the activities known as systems engineering and project management do not overlap as discussed in Section 9.19. This perception differentiates between SETR - the role or job description of the systems engineer and SETA - a set of activities known as systems engineering where:

- SETR is a subjective definition from the *Operational* perspective of systems engineering. It can be a:
  - Job title such as network systems engineering, control system engineering, communications systems engineering, etc. In many instances the type of system is dropped from the title (the Domain Systems Engineering camp discussed in Section 9.17.7). The on-the-job activities performed in such a role include, systems engineering, design, engineering, project management, testing, etc. SETR is performed in many domains, generally associated with technology and is often process-centric.
  - "*Philosophy and a way of life*" (Hitchins, 1998) which Kasser interpreted as the application of holistic thinking to problem-solving (Kasser, 2013c).
- SETA is:
  - A return to Hall's definition of *"systems engineering as a function not what a group does"* (Hall, 1962: page 11).
  - An objective definition of an activity based on the following criterion (Kasser and Hitchins, 2009; Kasser, et al., 2009):

- *If the activity deals with parts and their interactions as a whole,* then it is an activity within the set of activities to be known as SETA.
- *If the activity deals with a part in isolation*, then the activity is not an activity within the set of activities to be known as SETA but is part of another set of activities ('something else'), e.g., engineering, project management, software engineering, etc.

The ISO/IEC 15288 Standard lists processes performed by systems engineers (Arnold, 2002) and hence may be considered as being applicable to SETR rather than SETA.

The people who do SETA do it as a way of life (Hitchins, 1998) whether they are, or are not, known as systems engineers (SETR). For example, SETA is used when:

- ***Cooking a meal:*** the meal emerges from the process and the combination of, and the interaction between, the ingredients. The best ingredients will not save a meal that was over-cooked or under-cooked.
- ***Diagnosing an illness:*** good physicians consider the symptoms holistically in the context of the physiology of the patients and their environments.
- ***Organising a conference:*** the conference emerges from the combination of, and interaction between, the location, speakers, reviewers, delegates, and other entities.
- ***Solving a crime:*** detectives, upon investigation, find a variety of clues which (should) lead to the perpetrator.

Other examples of SETA are:

- Crosby's "completeness" (Crosby, 1979).
- Deming's "system of profound knowledge" (Deming, 1993).
- Senge's "fifth discipline" (Senge, 1990).

Crosby, Deming and Senge all state the need for systems thinking, and the benefits to be gained therefrom[5].

9.19. The Roles Rectangle

Perceptions from the *Continuum* perspective indicate that the task of developing systems is currently split between the four interdependent roles of systems architecting, systems engineering, project management and process architecting shown in the Roles Rectangle in Figure 9.11. Consider each quadrant in the Roles Rectangle.

[5] Which also puts them in the systems thinking camp.

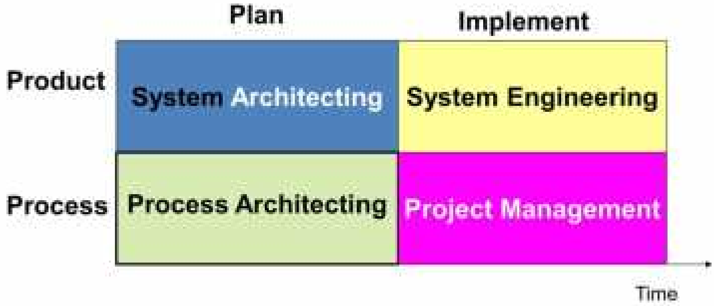


Figure 9.11 The Roles Rectangle

9.19.1. Systems architecting

Systems architecting is defined as, "*the art and science of creating and building complex systems. That part of the systems development most concerned with scoping, structuring, and certification*" (Maier and Rechtin, 2000). The role of the systems architect is to apply architectural methods analogous to those used in civil works. The systems architecting role is concerned with:

- Meeting the overall client needs.
- Directing the high-level design.
- Focusing on keeping the interfaces between contractors manageable.
- Working for the client to ensure that the resulting system satisfies the client's expectations, even if the expectations are not clearly articulated.

9.19.2. Systems engineering

The role of the systems engineer is to perform the activities discussed in the *Operational* perspective (Chapter 5) using the activities discussed in the *Functional* perspective (Chapter 6).

A number of definitions of systems engineering are listed in Section 9.1.

9.19.3. Project management

The role of the project manager is embodied in one of the many definitions of project management, "*the planning, organizing, directing, and controlling of company resources (i.e. money, materials, time and people) for a relatively short-term objective. It is established to accomplish a set of specific goals and objectives by utilizing a fluid, systems approach to management by having functional personnel (the traditional line-staff hierarchy) assigned to a specific project (the horizontal hierarchy)*" (Kezsbom, et al., 1989).

9.19.4. Process architecting

The role of the process architect is to perform the activities that design, set up, and continuously optimise (improve), the process for the development of the specific system being produced by the specific organisation over the specific time period of the SDP to optimise productivity (Biemer and Sage, 2009: page 153). The choices faced by the process architect include:

- ***Choice of lifecycle*** such as the traditional requirement driven lifecycle or a capability driven lifecycle.
- ***Choice of methodology*** such as (which) soft-systems, functional, object-oriented, waterfall, agile, rapid, spiral, cataract, etc.
- ***Choice of process for implementing the methodology*** as well as the milestone process-products and the checkpoints within the process. The process is scaled to the size of the project. Sometimes this may require combining activities or products, e.g. combining the operations concept with the systems requirements documents for small projects, or even choosing to produce milestone documents in the form of PowerPoint presentations instead of text mode documents.
- ***Build–buy decisions***. The decision to build or buy components of the product affects the development process as well as the product architecture. This decision is made after considering its implications on both the product system and development process.

9.20. The overlap between systems engineering, project management and other disciplines

The overlap between systems engineering, project management and other disciplines in the SDP is often represented in the manner shown in Figure 9.12 and can be seen in:

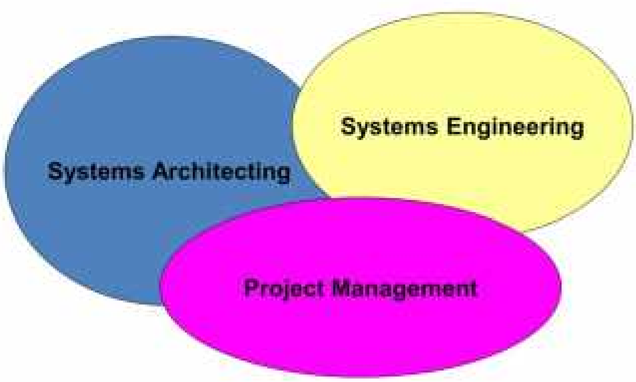


Figure 9.12 Overlapping organizational roles in the development of systems

- Examples of the different overlaps between systems engineering and project management (Jenkins, 1969; Brecka, 1994; Roe, 1995; DSMC, 1996; Sheard, 1996; Johnson, 1997; Watts and Mar, 1997; Bottomly, et al., 1998; Kasser, 1996).
- Emes at al. who discussed overlaps between systems engineering and other disciplines (Emes, et al., 2005).
- Eisner who listed a general set of 28 tasks and activities that were normally performed within the overall context of large-scale systems engineering (Eisner, 1988). Eisner calls the range of activities 'specialty skills' because some people spend their careers working in these specialties. Thus according to Eisner in 1988 systems engineering[6] overlapped at least 28 engineering specialties.
- Eisner who expanded his earlier list and discussed 30 tasks that form the central core of systems engineering (Eisner, 1997: page 156). The whole area of systems engineering management is covered in just one of the tasks. Eisner states that, *"not only must a Chief Systems Engineer understand all 30 tasks; he or she must also understand the relationships between them, which is an enormously challenging undertaking that requires both a broad and deep commitment to this discipline as well as the supporting knowledge base"*.
- The then INCOSE President John Thomas expanded on this role in his presentations on the need for systems engineers with moxie (Thomas, 2011).
- The overlap between Operations Research and systems engineering was noted as early as 1954 when Johnson wrote, "*Operations Research is concerned with the heart of this control problem – how to make sure that the whole systems works with maximum effectiveness and least cost*" (Johnson, 1954: page xi) a goal that many modern systems engineers would apply to systems engineering. Goode and Machol wrote that the steps of the Operations Research and systems engineering processes have much in common however there is a fundamental difference in approach namely, "*the operations analyst is primarily interested in making procedural changes, while the systems engineer is primarily interested in making equipment changes*". A lasting difference was noted by Roy as, "*Operations Research is more likely to be concerned with systems in being than with operations in prospect*" (Roy, 1960: page 22).
- Goode and Machol make no distinction between 'systems engineering' and 'engineering design' or even 'design' and use the

[6] Author's interpretation as the role of systems engineering

terms interchangeably (Hall, 1962: page 20 citing (Goode and Machol, 1959).

- Archer defined design as, *"a goal-directed problem solving activity"* (Archer, 1965).
- Fielden defined 'engineering design' as, *"the use of scientific principles, technical information and imagination in the definition of a mechanical structure, machine or system to perform prespecified functions with the maximum economy and efficiency"* (Fielden, 1963).
- Matchett and Briggs defined 'design' as, *"the optimum solution to the sum of the true needs of a particular set of circumstances"* (Matchett and Briggs, 1966).
- Bahill and Dean, in discussing the requirements in the SEP call it the 'system design process' and use the terms 'design' and 'solution' interchangeably (Bahill and Dean, 1997).
- Hari et al. provided an example of the various activities performed in new product design that overlap those of systems engineering (Hari, et al., 2004).
- The United Kingdom (UK) Defence Evaluation and Research Agency (DERA) definition of systems engineering is, *"a set of activities which control the overall design, development, implementation and integration of a complex set of interacting components or systems to meet the needs of all the users"* (DERA, 1998). Controlling activities are project management activities, development and testing activities are engineering activities.
- Project management is defined as, *"the planning, organizing, directing, and controlling of company resources (i.e. money, materials, time and people) for a relatively short-term objective. It is established to accomplish a set of specific goals and objectives by utilizing a fluid, systems approach to management by having functional personnel (the traditional line-staff hierarchy) assigned to a specific project (the horizontal hierarchy)"* (Kezsbom, et al., 1989). Kezsbom's systematic approach to project management requires the break down and identification of each logical subsystems component into its own assemblage of people, things, information or organization required to achieve the subobjective (Kezsbom, et al., 1989: page 7).
- The DoD defined Integrated Product and Process Development (IPPD) as, *"a management process that integrates all activities from product concept through production/field support, using a multifunctional team, to simultaneously optimize the product and its manufacturing and sustainment processes to meet cost and performance objectives"* (DOD, 1996).

In industry today, Hall's mixed systems engineering teams (Hall, 1962) seem to be called Integrated Product Teams (IPT) and are working in the context of "concurrent engineering" which has existed as a recognizable topic since the mid 1980's.

- The aim of both concurrent engineering and systems engineering is, *"to provide a good product at the right time … suitably free of defects and ready when the customer wants it"* (Gardiner, 1996)
- Configuration Management (CM) is defined as, *"a field of management that focuses on establishing and maintaining consistency of a system's or product's performance and its functional and physical attributes with its requirements, design, and operational information throughout its life"* (MIL-HDBK-61A, 2001). There are two types of configuration audits within Configuration Management which overlap systems engineering activities. These configuration audits are:
  - ***Functional configuration audits:*** ensure that functional and performance attributes of a configuration item are achieved,
  - ***Physical configuration audits:*** ensure that a configuration item is installed in accordance with the requirements of its detailed design documentation.

  These configuration audits can occur either at delivery or at the moment of effecting a change. Note 'doing' is commonly known as verification and validation or testing and 'ensuring' is verifying that it happens or exists, which is also a part of the quality control/assurance activity in systems engineering.

9.21. The two different process paradigms in systems engineering

The SEP has evolved into two process paradigms, the 'A' paradigm and the 'B' paradigm (Kasser, 2012b). Consider the two paradigms.

9.21.1. The' A' paradigm

The 'A' paradigm begins with the systems engineering activities performed in the Needs Identification State of the SLC discussed in Section 9.12. Research into the systems engineering literature found that successful projects such as the NASA Apollo program were characterised by a common vision of the purpose and performance of the solution systems among the customers, users and developers; namely a paradigm that began in the Needs Identification State of the SLC. Moreover, the common vision related to both the mission and support functions performed by the solution system program (Hitchins, 2007). Perceptions from the *Generic* perspective (outside the systems engineering literature) support the research with similar findings in the process improvement and Quality literature, e.g., (Deming, 1993; Dolan, 2003). In addition, BPR creates and disseminates/communicates a 'to-be' model of the operation of the

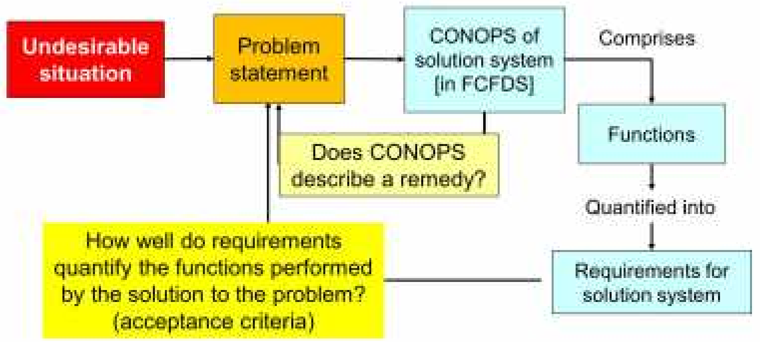


Figure 9.13 Relationships between CONOPS, functions and requirements ('A' paradigm)

conceptual reengineered organisation (i.e. a FCFDS) before embarking on the change process.

Since the 'A' paradigm is characterized by a common vision of the purpose of the mission and support functions of solution systems among the customers, users and developers, the quality of the requirements tends to have little if any impact on the functionality of the solution system.

The relationship between the undesirable situation, the CONOPS, functions and requirements in the 'A' paradigm (Section 9.21.1) can be expressed as shown in Figure 9.13 (Kasser, 2012b).

9.21.2. The 'B' paradigm

The 'B' paradigm begins with the systems engineering activities performed in the Requirements State of the SDP discussed in Section 9.12. Many systems and software engineers have been educated to consider the systems engineering activities in the Requirements State of the SDP as the first state of the SEP. For example:

- In one postgraduate class at University of Maryland University College (UMUC) the instructor stated that systems engineering began for him when he received a requirements specification (Todaro, 1988).
- Requirements are one of the inputs to the SEP (Martin, 1997: page 95), (Eisner, 1997: page 9), (Wasson, 2006: page 60) and (DOD 5000.2-R, 2002), pages 83-84).
- The CONOPS is derived from the requirements (Denzler and Mackey, 1994; Guo, 2010)

While DoD 5000 does call out the 'analysis of possible alternatives' subset of activities, those activities:

- Take place *before* the DoD 5000.2-R SEP begins.
- Are called out as part of the separate independent Cost as an Independent Variable (CAIV) process. CAIV is just a part of the concept of designing budget tolerant systems (Denzler and Kasser, 1995).

Recognition of the situation (addressing mission/purpose definition activities to some extent while failing to cover other early state activities in the SDP also appeared in a survey of then current SEPs (Bruno and Mar, 1997) and in the list of the engineering and systems engineering activities assigned to the systems engineering organization/team based on the MIL-STD-499B (MIL-STD-499B, 1993) and EIA 632 Standards (EIA 632, 1994; Fisher, 1996).

9.22. The difference in the contents of publications

Textbooks on project management, in general, covered the same material in different levels of detail. However, while there was some commonality in the textbooks on systems thinking and systems engineering, each book had a different focus and contained a lot of unique material. For example:

- Books and papers on systems thinking, in general:
  - Covered a wide variety of topics, e.g. (Allen, 2004; Boardman and Sauser, 2008; Checkland, 1991; Jackson, 2003; Paul and Elder, 2006; Senge, 1990).
  - Contained almost as many different definitions of 'systems thinking' as there were different definitions of 'systems engineering'.
  - Drew different boundaries between the activities included in systems thinking and the activities included in critical thinking.
  - Did not provide any good ways of teaching systems thinking. They focused on the need for, and history of, systems thinking. In some books systems thinking constituted the application of specific methodologies such as Checkland's SSM (Checkland, 1991) and the use of causal loops and feedback techniques in examining relationships between the parts of a system (Senge, 1990).
- Textbooks and papers on systems engineering, in general:
  - Focused on one of the different camps or views of systems engineering discussed in Section 9.17.
  - Did not contain a critical thinking element commenting on the efficiency and efficacy of what was being described.

- Seemed to document the author's perspective and experience at some point of time in a specific place. For example[7]:
    - Goode and Machol focus on systems design and the mathematical tools listed in Section 7.3 (Goode and Machol, 1959).
    - Hall focuses on the activities performed in the early states of realizing a system and the need to do research into the problem and solution domains (Hall, 1962).
    - Martin's systems engineering handbook focuses on a process for developing systems and products (Martin, 1997).
    - Khisty and Mohammadi focus on economics, probability and statistics (Khisty and Mohammadi, 2001).
    - Wasson seems to focus on the DoD version of systems engineering (Wasson, 2006) and much of the material seems to be extracted from the US Military Standards (MIL-STD).
    - Eisner contains a mixture of topics on systems engineering and project management (Eisner, 2008).
    - Weiss focuses on the process for developing new products (Weiss, 2013).

9.23. The different processes for creating a system

A literature search only found the following two approaches for creating systems.

1. Athey's systemic systems approach (Athey, 1982: page 13).
2. O'Connor and McDermott's set of guidelines (O'Connor and McDermott, 1997).

9.23.1. Athey's systemic systems approach

Athey drew the boundary of a system such that:

- The set of components which can be directly influenced or controlled in a system design are included in the system.
- The factors which have an influence on the effectiveness of the system, but which are not controllable, are part of the environment, namely are outside the system.

[7] This is a random sample and is not meant to imply that books not mentioned are less important.

9.23.2. O'Connor and McDermott's set of guidelines

O'Connor and McDermott introduced the following set of guidelines for drawing systems (O'Connor and McDermott, 1997: page 166):

1. Draw with a goal in mind.
2. Start wherever you want.
3. Include events.
4. Define system boundaries.
5. Include time span and people involved.
6. Only include elements that can change when influenced by another element.

9.24. The paradoxes and dichotomies

The literature review produced the following paradoxes and dichotomies.

9.24.1. The paradoxes

The paradoxes are:

1. ***The process paradox:*** according to Arnold, *"A single process, standardizing the scope, purpose and a set of development actions, has been traditionally associated with systems engineering"* (Arnold, 2000) citing MIL-STD 499B and IEEE 1220. Yet there is no single SEP; each of the process descriptions are different (Bahill and Gissing, 1998; MIL-STD-499A, 1974; EIA 632, 1994; IEEE 1220, 1998) as discussed in Section 9.17.2.
2. ***The roles paradox:*** the *Operational* perspective of the roles of the systems engineer, documented in1969 and 1996 were different (Section 9.18).
3. ***The tools paradox:*** identified in Section 7.3.
4. ***The emergent properties paradox:*** where:
   - On the one hand, some systems thinkers hold that the *emergent behaviour* from the interaction of a set of components *cannot be predicted* (O'Connor and McDermott, 1997: page 6).
   - On the other hand, system design *is predicting* that the *emergent behaviour* from a set of components and the interaction between them (the system) *will meet the requirements for the system*.
5. ***The system optimization paradox:*** was stated by Machol and Miles who wrote, "*the principle of suboptimization states that optimization of each subsystem independently will not lead in general to a system optimum, and that improvement of a particular subsystem actually may worsen the overall system. Since every system is merely a subsystem of some larger system, this principle presents a difficult if not insoluble problem, - one that*

is always present in any major systems design" (Machol and Miles Jnr, 1973: page 39).

6. ***The reductionist paradox:*** reductionism has been considered as poor practise in systems engineering, yet current system views are inherently reductionist since they exclude the metasystem.

9.24.2. The dichotomies

There are similar dichotomies in the literature on complexity and Systems of Systems.

Examples from each side of the dichotomy found in a literature review of complexity in the systems engineering field include:

- Jenkins 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*" (Jenkins, 1969).
- Meyer and Rechtin 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 (Maier and Rechtin, 2000).
- Bar-Yam 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, 2003). Bar-Yam quoted the CHAOS Study (CHAOS, 1995) 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 was that the systemic reason for the challenged projects was poor management!

There are dichotomies in the development of systems in Hitchins' Layer 3 (Section 9.10) sometimes known as Systems of Systems, namely:

- On one hand, civilian organizations can manage so-called Systems of Systems such as fleets of cruise ships, financial and banking networks, oil rigs, airlines, transportation systems and hospitals.
- On the other hand, the US DoD and its contractors, grounded

in Hitchins' Layer 2 processes cannot manage the step up in the hierarchy to Hitchins' Layer 3 and identify new tools and/or adopt the existing tools and techniques appropriate to the Hitchins' Layer 3[8] and are working on extending the methodologies used in Hitchins' Layer 2 into Hitchins' Layer 3 with scant results.

9.25. The different definitions of complexity

The literature on complexity contains different definitions of the term "complexity". For example:

- *"A complex system usually consists of a large number of members, elements or agents, which interact with one another and with the environment"* (ElMaraghy, et al., 2012). According to this definition the only difference between a system and a complex system is in the interpretation of the meaning of the word 'large'.
- ElMaraghy et al. wrote, *"Colwell (Colwell, 2005) defined thirty-two complexity types in twelve different disciplines and domains such as projects, structural, technical, computational, functional, and operational complexity"* (ElMaraghy, et al., 2012).
- Tomiyama et al. introduced two different types of complexity: (i) complexity by design and (ii) the intrinsic complexity of multi-disciplinary, from the viewpoint of knowledge structure (Tomiyama, et al., 2007).
- Suh defined complexity as, *"the measure of uncertainty in achieving the functional requirements (FRs) of a system within their specified design range."* (Suh, 2005). Suh stated the need to abstract out things that were not pertinent to the issues at hand.
- Sillitto distinguished between subjective and objective complexity (Sillitto, 2009).

9.26. Distinguishing between subjective and objective complexity

There do not appear to be unique words that uniquely define the concepts of 'subjective complexity' and 'objective complexity' in the English language. The words 'complex' and 'complicated' have been used for both concepts because their meanings overlap and contain both a subjective and objective meaning. For example consider the following definitions (Dictionary.com, 2013):

- "***Complex***
    a. Composed of many interconnected parts; compound; composite: [e.g.] a complex highway system.

[8] Such as those used in Operations Research.

b. Characterized by a very complicated or involved arrangement of parts, units, etc.: [e.g.] complex machinery.
c. So complicated or intricate as to be hard to understand or deal with: [e.g.] a complex problem.

- ***Complicated***

a. Composed of elaborately interconnected parts; complex: [e.g.] complicated apparatus for measuring brain functions.
b. Difficult to analyse, understand, explain, etc.: [e.g.] a complicated problem".

Hence the literature accordingly uses the words 'complicated' and 'complex' as synonyms to mean both subjective and objective complexity.

Sillitto distinguished between subjective and objective complexity (Sillitto, 2009) as:

- "***subjective complexity*** [9,10] *– which means that people don't understand it and can't get their heads round it – and*
- ***objective complexity*** *– which means that the problem situation or the solution has an intrinsic and measurable degree of complexity* ".

9.27. The different types of objective complexity

The various definitions of objective complexity in the literature can be aggregated into two types of objective complexity as follows:

1. ***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.
2. ***Artificial complexity:*** arising from either poor aggregation or failure to abstract out 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 complexity in the manner of Rube Goldberg or W. Heath Robinson. For example, in today's paradigm, complex drawings are generated that contain lots of information[11] and the observer is supposed to abstract information as necessary from the drawings. The natural complexity

[9] This author highlighted the words 'subjective complexity' and 'objective complexity'.
[10] Which can be quantified into the levels of difficulty discussed in Section 10.12.
[11] The DODAF Operational View (OV) diagrams can be wonderful examples of artificial complexity.

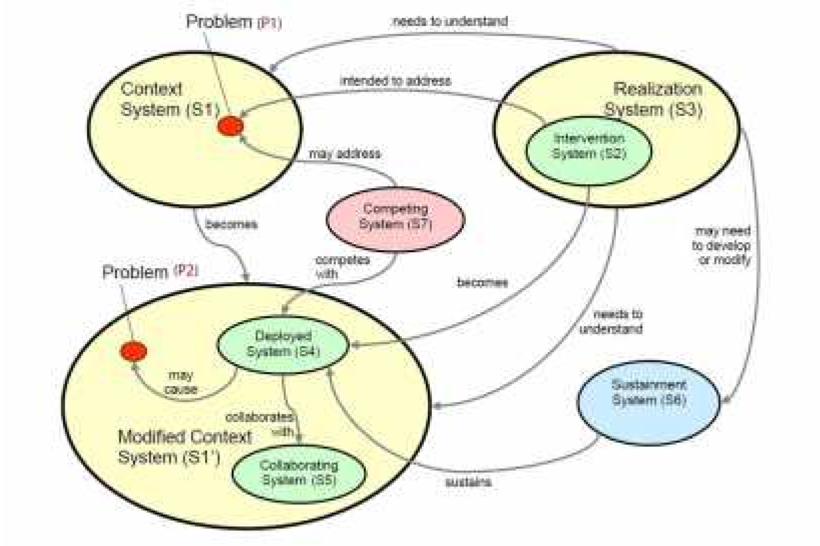


Figure 9.14 The Seven Samurai Systems (Martin, 2004)

of the area of interest is included in the drawings; hence the system is thought to be complex.

9.28. The previous proposed approaches to manage complexity in the INCOSE literature

Since complexity cannot be removed, it must be managed. Given the problem of managing complexity, the first activity was to research the INCOSE literature to identify previous attempts to manage complexity[12]. This section discusses the following three models for managing complexity in the systems development context found in the INCOSE literature.

- The Seven Samurai (Martin, 2004).
- The Whole System Model (Adcock, 2005; Mackley, 2008).
- The Systems Project (Paul and Owunwanne, 2006).

9.28.1. The Seven Samurai

Martin's approach to managing the complexity is grounded in the problem solving camp of systems engineering. Martin starts with a problematic or undesirable situation (Schön, 1991) and ends with a solution system that remedies the problem. Martin stated that, "*the seven different systems must be acknowledged and understood by those who purport to do systems engineering*" (Martin, 2004). Martin likens his seven systems to the seven samurai in the 1954 film (Kursawa, 1954) because just as the seven unemployed

[12] Other approaches outside the INCOSE literature are acknowledged but not cited in this work.

samurai became heroes by saving a poor village under attack, according to Martin, when his seven systems are employed with proper consideration and enthusiasm they will become the heroes of your systems development project. Martin's SOI shown in Figure 9.14, is the seven samurai systems and the 15 interactions between them. Martin's seven samurai systems are:

S1. ***The context system*** is where the problem (P1) resides; namely, the "as-is" situation. Aspects of the *context system* must be analysed to determine the underlying problem.

S2. ***The intervention system*** provides the solution to a real or perceived problem in the context system. The *intervention system* is created by the *realization system*. However, once deployed in the *context system*, the *intervention system* becomes the *deployed system*.

S3. ***The realization system*** consists of all the resources to be applied in causing the *intervention system* to be fully conceived, developed, produced, tested, and deployed. Martin adds that this system is often known as an Enterprise.

S4. ***The deployed system*** which evolves from the *intervention system* and interacts with *collaborating systems* to accomplish its own functions. While the *deployed system* is intended to be the same as the *intervention system*, there generally are differences for various reasons, intentional or otherwise. Once deployed, the system will often change the original ***context system*** into a ***modified context system*** (S1')[13] and might cause a new or modified problem (P2).

S5. ***The collaborating systems*** interact with the deployed system in the modified context system.

S6. ***The sustainment system*** provides services and materials to keep the *deployed system* operational, e.g. fuel, energy, spare parts, training, customer hotline, maintenance, waste removal, refurbishment, retirement etc. In many instances, the *realization system* may need to modify or even develop parts of the *sustainment system*.

S7. ***The competing systems*** which may also solve the original problem or parts of it and compete for resources used by the *deployed system*.

9.28.2. The Whole System Model

The Whole System Model (Adcock, 2005; Mackley, 2008) shown in Figure 9.15 views the problem of managing complexity from two different perspectives; lifecycle and process, considering the SOI as the following

[13] This could be considered as an eighth system.

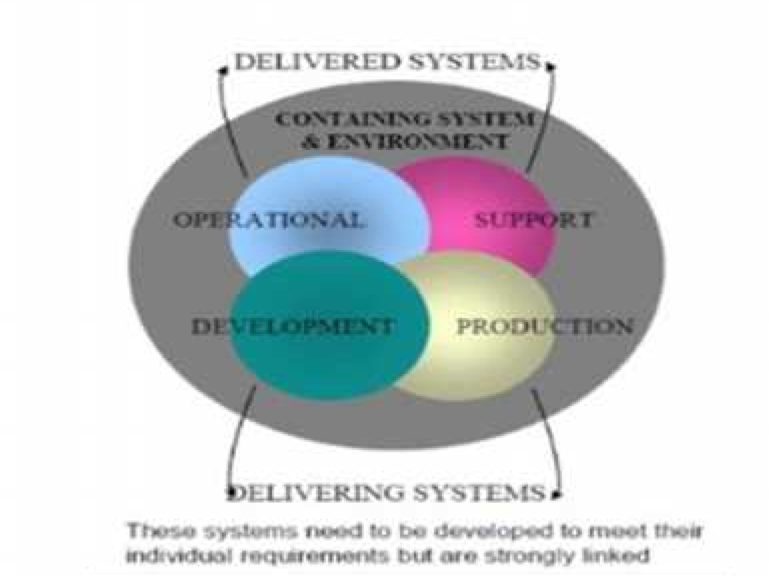


Figure 9.15 The Whole System Model (Adcock, 2005; Mackley, 2008)

five linked systems within "*the bounded system whose lifecycle is under consideration*":

S1. ***Operational system (OS):*** Entities involved in provision of system mission, objective, strategies and plans.
S2. ***Support system (SS):*** Entities involved in maintaining the OS with supply of required resources.
S3. ***Development system (DS):*** The process and associated equipment/tools required for creation, development and certification of the OS design throughout its lifetime.
S4. ***Production system (PS):*** Process and equipment/tools required to create a validated and reproducible OS from the system design.
S5. ***Containing system (CS):*** The related systems and the environment in which the other four systems interact, often known as the acquisition system.

According to Adcock, the Whole System Model illustrates the scope of related system and enabling system relationships which might apply to a given SOI, depending upon which part of the whole system it is from.

9.28.3. The Systems Project

The Systems Project (Paul and Owunwanne, 2006) shown in Figure 9.16:

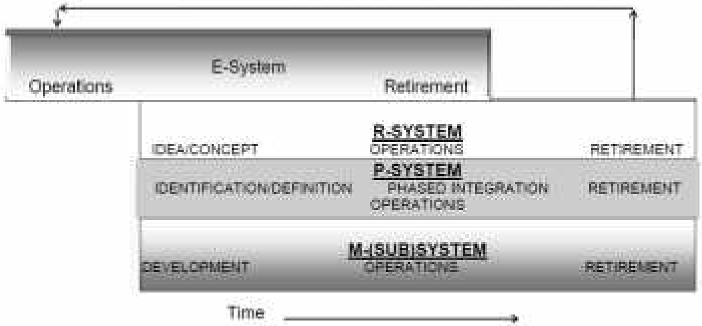


Figure 9.16 The Systems Project (Paul and Owunwanne, 2006)

- "*is a framework for packaging and conducting all of the systems engineering activities associated with developing/managing the product system, the Producing System, the Existing System (if there is one in place, if applicable [I/A]), and the Maintenance and Support System through the life cycle of the product system*".
- Manages the complexity by viewing the SOI from the process perspective.
- Involves the simultaneous development/management of as many as four independent but related systems. These related systems shown in Figure 9.16, encompassed in the SOI are:

  S1. ***The Existing System ("E-System"):*** the system which may be in place and which will be retired upon implementing the R-System.

  S2. ***The Required System ("R-System"):*** the system that is being developed to satisfy a need, alleviate an existing problem situation, or respond to an opportunity.

  S3. ***The Producing System ("P-System"):*** the system that produces the R-*System*. The *P-system* comprises the businesses, individuals, manufacturing plants etc. that must be managed and coordinated to produce and deploy the R-*System* and the *M-System* and decommission and remove the *E-System* when it is replaced by the R-*System*.

  S4. ***The Maintenance and Support System ("M-System")*** which supports the R-System through its life cycle.

If the R-*System* is a new system, there may not be an *E-System* in place. If there is an *E-System*, then, when deployed, *the* R-*system* becomes the new *E-system* and the entire development cycle repeats itself.

9.29. The different ways of assessing competency in systems engineering[14]

Ways of measuring competency are generally quantitative, and so the perceptions of ways of measuring competency are stored in the *Quantitative* perspective.

Competency assessment tends to be performed using competency models which form the foundation for developing curriculum and selecting training materials, and for licensure and certification requirements, job descriptions, recruiting and hiring, and performance reviews (CareerOneStop, 2011). *"These models have competency domains broken down into competency groups and further sub-categorized into sub-competencies. As one continues to the next[15] levels in the hierarchy, the competencies become further focused and specific to the industry, job or occupation, and position"* (Ennis, 2008). A multi-level assessment approach to assessing proficiencies of systems engineers groups the knowledge, traits, abilities and other characteristics of successful systems engineers into a two-dimensional maturity model[16] in accordance with Arnold who wrote, *"at its simplest, competence may be viewed in terms of two dimensions or axes. One axis defines the process, or set of processes, considered relevant to the discipline of interest. The other axis establishes the level of proficiency attained typically using a progression of increasing-value cardinal points that are defined in terms of attainment or performance criteria"* (Arnold, 2000).

The activities performed in the role of a systems engineer in one organisation are different to those performed by a systems engineer in another organization and sometimes even in different parts of the same organisation (Section 9.19). It could thus be expected that different ways of assessing the competency of systems engineers would assess different characteristics. The following four competency models were studied to determine their coverage:

1. Knowledge, Skills, and Abilities (KSA) discussed in Section 10.8.1[17].
2. The INCOSE Certified Systems Engineer Professional (CSEP) Examination (INCOSE, 2008) discussed in Section 10.8.2.
3. The INCOSE UK Systems Engineering Competencies Framework (SECF) (INCOSE UK, 2010) discussed in Section 10.8.3.

[14] While perceptions from the *Continuum* perspective identified the different approaches to assessing competency in systems engineering, assessments are quantitative, so these perceptions are stored in the *Quantitative* perspective.

[15] Next level down or lower levels.

[16] Due to space limitations, where prior work covers a topic in detail, the work is cited and summarized.

[17] The details of the competency models are stored in the *Quantitative* perspective because they are measurement tools.

4. Capacity for Engineering Systems Thinking (CEST) (Frank, 2006) discussed in Section 10.8.4.

Sometime later in the research, the following additional competency models were also studied:

5. The Systems Engineering Competency Taxonomy (SECT) (Squires, et al., 2011) discussed in Section 10.8.5.
6. The NASA 2010 Systems Engineering Competencies (NASA, 2010) discussed in Section 10.8.6.
7. The Jet Propulsion Laboratory (JPL) Systems Engineering Advancement (SEA) project (Jansma and Jones, 2006) discussed in Section 10.8.7.
8. The MITRE 2007 Systems Engineering Competency Model (Metzger and Bender, 2007) discussed in Section 10.8.8.
9. The NDIA proposed systems engineering competency model (Gelosh, 2008) discussed in Section 10.8.9.

9.30. Summary

This Chapter contained perceptions of differences within systems engineering and between systems engineering and other disciplines from the *Continuum* perspective. The key points were:

- There are many different definitions of the word ‘system’.
- There are many different definitions of ‘systems engineering’.
- There are many different definitions of the term ‘requirement’.
- The different meanings of the word ‘problem’.
- The difference between problem formulators and problem solvers.
- The different layers of systems engineering.
- The differences between the problem, solution and implementation domains.
- The different states in the SLC.
- The different camps in systems engineering.
- The different roles of the systems engineer.
- The overlap between systems engineering, project management and other disciplines.
- The ‘A’ and ‘B’ paradigms.
- The paradoxes and dichotomies in systems engineering.
- Distinguishing between objective and subjective complexity.

--oo--

10. The Quantitative perspective

The *Quantitative* perspective incorporates Richmond's quantitative thinking (Richmond, 1993) and:

- Perceives the numbers and measurements associated with the system.
- Indicates that relative comparisons are sometimes more useful than absolute comparisons.
- Is not about the need to measure everything, *"it is more the recognition that numbers must be useful, not necessarily perfect and need not be absolute*" (Richmond, 1993).
- Is about quantification rather than measurement, and leads to the values of parameters in mathematical relationships in models and simulations.

Perceptions of systems engineering from the *Quantitative* perspective included:

1. The definition of a metric discussed in Section 10.1.
2. The amount of education and training opportunities discussed in Section 10.2.
3. Research into the value of systems engineering discussed in Section 10.3.
4. Two ways of measuring project success discussed in Section 10.4.
5. The lack of a metric for the goodness of a requirement as discussed in Section 10.5.
6. The three types of emergent properties discussed in Section 10.6.
7. The Capability Maturity Models (CMM) discussed in Section 10.7.
8. The ways of measuring and improving systems engineering discussed in Section 10.8.
9. The five types of systems engineers discussed in Section 10.9.
10. The number of systems engineers in the different states of the SDP discussed in Section 10.10.
11. A way of assessing technology readiness discussed in Section 10.11.

12. The four levels of difficulty of the problem discussed in Section 10.12
13. The top ten reasons for project success and failure cited in the oft quoted Standish report (CHAOS, 1995). The report also made no mention of process.

10.1. Metrics

Perceptions from the *Quantitative* perspective are used in performing measurements. The ideal unit of measure (Juran, 1988: pages 76-78):

- ***Provides an agreed basis for decision-making:*** different people view things differently, and have different priorities. The metric must allow a meeting of minds.
- ***Is understandable:*** metrics may not be understandable, perhaps because words do not have standardized meanings, or may require an educational background that is lacking.
- ***Applies broadly:*** for use to determine if an improvement has occurred.
- ***Is susceptible to uniform interpretation:*** the units used and types of errors must have been defined with appropriate precision.
- ***Is economical to apply:*** there is a trade-off between the cost of making the measurements and the value of having them. The cost may depend on the precision, so care must be taken to specify the correct precision.
- ***Is compatible with existing designs of sensors:*** if you can't measure it, there is little point in defining it as a metric.

10.2. The amount of available education and training

As perceived from the *Operational* perspective, many systems engineers make use of the many opportunities for education and training.

10.2.1. Education

There were 213 Masters and 59 Doctorate programs in systems engineering world-wide in October 2013 (GradSchools.com, 2013). As of 2013, the number of Master's degrees in systems engineering had grown each year since 2001, with an average annual growth rate of 20% (Lasfer and Pyster, 2013).

10.2.2. Training

There are many training opportunities provided by commercial trainers in many countries. Courses differ in the topics covered and in the number of days per course. Courses with the same name by different providers may also differ in content (Kasser and Arnold, 2014). The content of the courses seem to line up with the different camps in systems engineering discussed in Section 9.17.

10.3. Research into the value of systems engineering

The following two studies of the value of systems engineering based on surveys of practitioners were identified:

1. A Survey of Systems Engineering Effectiveness discussed in Section 10.3.1.
2. Systems engineering Return on Investment (ROI) discussed in Section 10.3.2.

10.3.1. A survey of systems engineering effectiveness

The analysis of the data collected by Elm et al., showed that projects with better systems engineering capabilities delivered better project performance as seen in Figure 10.1 (Elm, et al., 2008). The study looked at the following twelve areas of systems engineering capability, addressing the project's utilization of systems engineering best practices in each area.

1. Project Planning.
2. Project Monitoring and Control.
3. Risk Management.
4. Requirements Development and Management.
5. Trade Studies.
6. Product Architecture.
7. Technical Solution.
8. Product Integration.
9. Verification.
10. Validation.

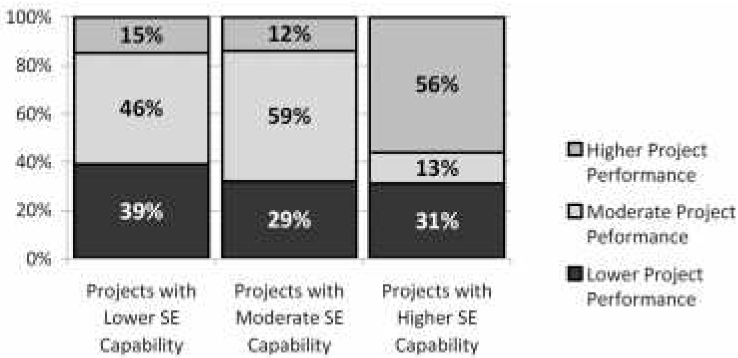


Figure 10.1 Project performance vs. systems engineering capability (Elm, et al., 2008)

11. Configuration Management (CM).
12. Integrated Product Team-Related Capability.

10.3.2. Systems engineering return on investment

Honour performed research into the value of systems engineering. Findings included (Honour, 2013):

- Statistically significant relationships between the amount of systems engineering activities and three success measures:
  a) Cost compliance.
  b) Schedule compliance.
  c) Stakeholder overall success.

- Optimum systems engineering effort for median programs is 14.4% of total program cost.
- The Return on Investment (ROI) in system engineering is as high as 7:1 for programs with little systems engineering effort and 3.5:1 for median programs.

  If a project is spending nothing, then each $1 re-purposed into systems engineering can reduce a potential overrun by $7. If project spending conforms to the average program (about 7.5%), then each $1 re-purposed into systems engineering can reduce the project's potential overrun. If the project is already at the optimum of 14.4%, then each $1 re-purposed into systems engineering gets the project no gain at all, it just increases costs (Honour, 2015).

- Systems engineering activities correlate strongly to program success measures, but do not correlate strongly to the technical quality of the resulting system.

Honour's research considered systems engineering to be the total effort expended across eight system level technical activities based in descriptions in the systems engineering standards discussed in Section 7.4 to define and develop a new system. This effort may be expended by systems engineers or by others including individuals who may be outside the development organization. The eight system level technical activities are:

1. Mission/purpose definition.
2. Requirements engineering.
3. System architecting.
4. System integration.
5. Verification and validation.
6. Technical analysis.
7. Scope management.

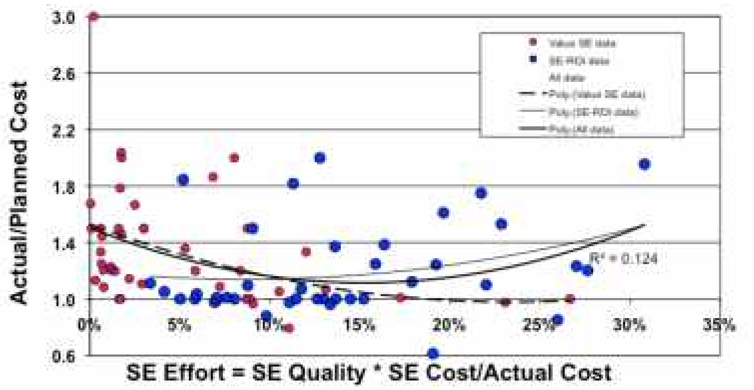


Figure 10.2 Cost overrun vs. systems engineering effort (Honour, 2013)

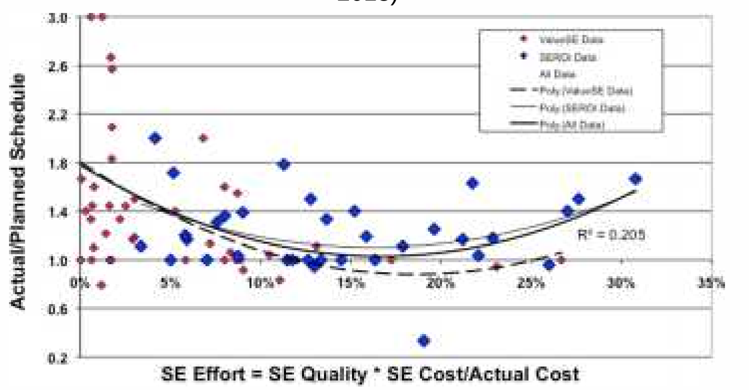


Figure 10.3 Schedule overrun vs. systems engineering effort (Honour, 2013)

8. Technical leadership/management.

Honour presented data showing cost and schedule overruns as a function of systems engineering effort. The relationship between cost overrun and systems engineering is shown in Figure 10.2 while the relationship between schedule overrun and systems engineering effort is shown in Figure 10.3. Honour points out that:

- The data in the figures come from different datasets.
- Few of the programs performed the early state systems engineering activities[1].

[1] of the 'A' paradigm (author's note)

10.4. Two ways of measuring project success

Two ways of measuring project success were identified in the literature, namely:

1. ***Conformance to cost and schedule estimates***. Projects were deemed to have failed if they overran their cost and schedule estimates. This is a project management measurement.
2. ***Satisfying the customer needs*** at the time the system is placed into service, and during the operational lifetime of the system. This is a systems engineering measurement.

10.5. The lack of a metric for the goodness of a requirement

Given consensus that requirements are critical to systems engineering[2], *"It has been known since as early as the 1950s that addressing requirements issues improves the chance of systems development success"* (Buren and Cook, 1998), there does not appear to be a metric for the goodness of a requirement.

While there has been a lot of research into building the right system and doing requirements better (Glass, 1992), much of that research has focused on

- How to state the requirements once they have been obtained.
- Using a Requirements Traceability Matrix (RTM).
- The tools that incorporate a RTM.

However, recognition that systems engineers produce poorly written requirements has been documented at least as early as 1993 (Hooks, 1993) and various approaches have been since proposed to alleviate the situation without much success. For example:

- A 1997 analysis of the software development process performed at Ericsson identified, *"missing understanding of customer needs" as the main obstacle for decreasing fault density and lead-time (Jacobs, 1999).* Related findings were aggregated under the heading, *"no common understanding of 'what to do'"*. The countermeasures to overcome these problems focused on testing the quality of the existing requirements rather than producing good requirements. There was no proposal on how to get clear requirements, nor was there a clear understanding of what a clear requirement was.
- Goldsmith states that the process of *"defining business requirements is the most important and poorest performed part of system development"* (Goldsmith, 2004).

---

[2] At least in the 'B' paradigm

Thus, there is a consensus that good requirements are critical to the success of a project. System engineers have focused on generating requirements to ensure that the as-built system is fit for its intended purpose. However, there is no universally accepted metric for the goodness of requirements either individually or as a set in a specification (Kasser, et al., 2006).

10.6. The three types of emergent properties

Emergent properties can only be achieved by the combination of the subsystems or components and the interactions between them. There seem to be three types of emergent properties (Kasser and Palmer, 2005):

1. Desired.
2. Undesired.
3. Serendipitous.

Since there are three types the perceptions are stored in the *Quantitative* perspective.

The three types of emergent properties are split between known emergent properties at design time and unknown emergent properties at design time, as follows:

- ***Known emergent properties at design time:***
    - ***Desired:*** being the purpose of the system.
    - ***Undesired***: based on experience and are:
        - Designed out.
        - Compensated for if they cannot be designed out.
- ***Unknown emergent properties at design time:***
    - ***Undesired***: functionality performed by the system that is undesired, also sometimes known as 'side effects'.
    - ***Serendipitous***: beneficial and desired once discovered, but not part of the original specifications.

10.7. The Capability Maturity Models (CMM)

Capability Maturity Models (CMM) represent ways of assessing or measuring the systems engineering processes (SEP) in an organisation and accordingly, the perceptions of CMMs are stored in the *Quantitative* perspective. The systems engineering CMMs are based on the Software Capability Maturity Model (CMM, 1995) which provides software organizations with guidance on how to gain control of, and improve, their processes to develop and maintain software. The Software CMM has the following five levels:

Level 1 - Initial: the organisation has few standard processes.

Level 2 - Repeatable: the organisation has basic project management processes that track cost, schedule, and functionality.

Level 3 - Defined, standard and consistent: the processes for management and engineering are documented, standardized, and integrated into a standard software process for the organisation. All projects use an approved, tailored version of the organization's standard software process for developing software.

Level 4 - Managed and predictable: the organisation uses detailed software process and product quality metrics to establish the quantitative evaluation foundation. These metrics allow meaningful variations in process performance to be distinguished from random noise, and the ability to predict trends in process and product quality.

Level 5 - Optimising and continual improvement: the organization has quantitative feedback systems in place to identify process weaknesses and strengthen them proactively.

The Software CMM was adapted for systems engineering in the form of a Systems Engineering CMM (SE-CMM) (Bate, et al., 1995). The SE-CMM has the following six levels of increasing process maturity.

Level 0: The Not Performed level.
Level 1: The Performed Informally level.
Level 2: The Planned and Tracked level.
Level 3: The Well Defined level.
Level 4: The Quantitatively Controlled level.
Level 5: The Continuously Improving level.

The CMM was replaced by the CMM-Integrated (CMMI) which was adapted for different industries with different Key Process Areas (KPA). The basic CMMI levels are shown in Figure 10.4 (Godfrey, 2004).

There was a short-lived National Council on Systems Engineering (NCOSE)[3] Systems Engineering Capability Assessment Model (SECAM) (INCOSE-CAWG, 1996) which was a continuous capability model and contained 19 focus areas distributed across three major categories as shown in Table 10.1. Each focus area is rated on a scale from zero (the default level) to five (optimal performance) as follows:

Level 0: Initial.
Level 1: Performed.
Level 2: Managed.
Level 3: Defined.

[3] Before NCOSE added the 'I' and became the INCOSE

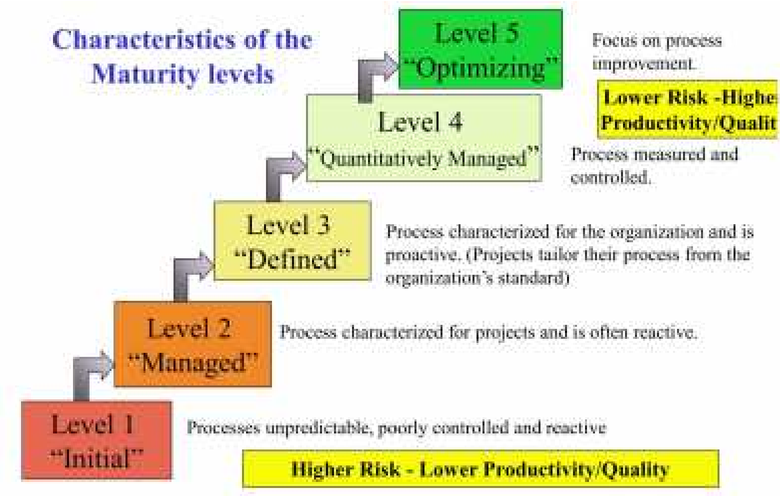


Figure 10.4 Characteristics of maturity levels (Godfrey, 2004)

Level 4: Measured.
Level 5: Optimizing.

The INCOSE SECAM model was retired with the release of the similar but upgraded Systems Engineering Capability Model, EIA/IS-731 (EIA 731, 1998).

10.8. The ways of measuring and improving systems engineering

Ways of measuring and improving systems engineering are generally quantitative, and so the perceptions of ways of measuring and improving systems engineering are stored in the *Quantitative* perspective.

These perceptions included:

- The development of process CMMs discussed in Section 10.7.
- A literature review that revealed that the work on improving systems engineering has focused on improving and developing new SEPs (Swarz and DeRosa, 2006; Goldberg and Assaraf, 2010) and tended to ignore people.
- The literature on excellence focused on people, not process (Rodgers, et al., 1993; Harrington, 1995; Peters and Austin, 1985; Peters and Waterman, 1982).

10.8.1. Knowledge, Skills, and Abilities

Knowledge, Skills, and Abilities (KSA) are one way of assessing the suitability of candidates for job positions according to qualification standards published by the US Office of Personnel Management (OPM). These standards are intended to identify applicants who are likely to perform

Table 10.1 INCOSE SECAM focus areas

1.0 Management	2.0 Organization	3.0 System Engineering
1.1 Planning 1.2 Tracking & Oversight 1.3 Subcontract Management 1.4 Intergroup Co-ordination 1.5 Configuration Management 1.6 Quality Management 1.7 Risk Management 1.8 Data Management	2.1 Process Management & Improvement 2.2 Competency Development 2.3 Technology Management 2.4 Environment & Tool Support	3.1 System Concept Definition 3.2 Requirements & Functional Analysis 3.3 System Design 3.4 Integrated Engineering Analysis 3.5 System Integration 3.6 System Verification 3.7 System Validation

successfully on the job, and to screen out those who are unlikely to do so (OPM, 2009). In practice, KSAs tend to be lists of statements written by, or on behalf of, candidates. These statements are targeted to specific positions and describe a number of situational challenges faced by the candidate and outcomes achieved in previous jobs that are to be used by evaluators in a pass-fail mode when looking for qualified candidates for a specific position.

10.8.2. The INCOSE CSEP Exam

The INCOSE Certified Systems Engineering Professional (CSEP) examination (INCOSE, 2008) is designed to test the applicant's knowledge of the contents of the INCOSE Systems Engineering Handbook (Haskins, 2011; 2006a)[4]. However, the INCOSE CSEP examination is only a part of the three-tier INCOSE approach to certifying the competency of a systems engineer and should not be considered as a stand-alone certification of competency.

10.8.3. The INCOSE UK Systems Engineering Competencies Framework

The INCOSE UK Systems Engineering Competency Framework (SECF) (INCOSE UK, 2010) was initially developed in response to an issue identified by the INCOSE UK Advisory Board (UKAB) (Hudson, 2006). The objective determined by the INCOSE UKAB was, *"to have a*

[4] The handbook and hence the syllabus for the exam has been updated since this was written.

measurable set of competencies for systems engineering which will achieve national recognition and will be useful to the enterprises represented by the UKAB". The focus of the SECF is on the competencies of systems engineering rather than the competencies of a systems engineer.

The SECF competencies are grouped into three themes:

1. Systems Thinking.
2. Holistic Lifecycle View.
3. Systems Engineering Management.

Consider each of them.

- ***Systems Thinking*** contains the underpinning systems concepts and the system/super-system skills including the enterprise and technology environment.
- ***Holistic Lifecycle View*** contains all the skills associated with the SLC from needs identification and requirements through to operation and ultimate disposal.
- ***Systems Engineering Management*** deals with the skills of choosing the appropriate lifecycle and the planning, monitoring and control of the SEP.

According to the SECF, each competency should be assessed in terms of four levels of comprehension and experience defined by "Awareness" through to "Expert".

1. ***Awareness:*** the person is able to understand the key issues and their implications. They are able to ask relevant and constructive questions on the subject. This level is aimed at enterprise roles that interface with systems engineering and therefore require an understanding of the systems engineering role within the enterprise.
2. ***Supervised Practitioner:*** the person displays an understanding of the subject but requires guidance and supervision. This proficiency level defines those engineers who are "in-training" or are inexperienced in that particular competency.
3. ***Practitioner:*** the person displays detailed knowledge of the subject and is capable of providing guidance and advice to others.
4. ***Expert:*** the person displays extensive and substantial practical experience and applied knowledge of the subject.

10.8.4. Capacity for Engineering Systems Thinking (CEST)

The Capacity for Engineering System Thinking (CEST) is a proposed set of high order thinking skills that enable individuals to successfully perform systems engineering tasks (Frank, 2006). A study aimed at identifying the characteristics of successful systems engineers identified 83 characteristics, which were aggregated into four sets of characteristics as follows:

1. Cognitive characteristics related to systems thinking.
2. Systems engineering skills.
3. Individual traits.
4. Multidisciplinary knowledge and experience.

CEST focuses on the cognitive skills, individual traits, capabilities and knowledge and background characteristics of a systems engineer who can examine system failures and identify and remedy system problems (Frank and Waks, 2001).

10.8.5. A systems engineering competency taxonomy (SECT)

Squires et al. have built a Systems Engineering Competency Taxonomy (SECT) from a selected set of existing competency models combined with some systems thinking research (Squires, et al., 2011). The authors combined the following three models into single Experience Accelerator (ExpAcc) competency taxonomy:

1. The Systems Planning, Research, Development, and Engineering (SPRDE) Systems Engineering and Program Systems Engineer competency model, known as the SPRDE-E/PSE (DAU, 2010).
2. The Systems Engineering Research Center (SERC) Technical Lead Competency Model (Gavito, et al., 2010).
3. A Critical/Systems Thinking Competency Model (Squires, 2007).

The final SECT competency taxonomy which covers 87 unique competencies is based on the following three-pronged approach:

1. Systems and critical thinking is the backbone of the model.
2. Technical expertise which comprises technical leadership, technical management, and technical/analytical skills.
3. Project management and other broad-based professional competencies.

Unlike the other competency models studied, SECT also evaluates the ability to deal with complexity in several levels of proficiency.

10.8.6. NASA 2010 Systems Engineering Competencies

NASA 2010 identifies 49 systems engineering competencies which are grouped by competency areas, competencies and competency elements and assessed in four proficiency levels (NASA, 2010).

The ten competency areas are:

1. Concepts and architecture.
2. System design.

3. Production and operations.
4. Technical management.
5. Project management.
6. Internal and external environments.
7. Human capital management.
8. Security and safety.
9. Professional development.
10. Leadership development.

The 35 systems engineering element competencies express the overall knowledge, skills and behaviours that systems engineers are expected to possess and/or perform as a part of their job.

The four proficiency levels are:

1. Technical engineer/project team member.
2. Subsystem lead.
3. Project systems engineer.
4. Program systems engineer.

The model is tailored to the NASA needs. It does not include any overt reference to systems thinking, cognitive competencies and behavioural traits.

10.8.7. The JPL Systems Engineering Advancement (SEA) project

Jansma and Jones developed a systems engineering competency model along three axes; processes, personal behaviours and technical knowledge as part of a project to improve systems engineering at the JPL (Jansma and Jones, 2006). The SEA project utilized a rigorous process to identify a list of highly valued personal behaviours of systems engineers.

The *processes* axis encompasses ten systems engineering functions. The identified personal behaviours fall into five clusters:

1. Leadership skills.
2. Attitudes and attributes.
3. Communication.
4. Problem solving and systems thinking.
5. Technical acumen.

The *technical knowledge* axis encompasses 21 systems engineering disciplines and fields.

10.8.8. MITRE 2007 Systems Engineering Competency Model

The MITRE Systems Engineering competency model (Metzger and Bender, 2007) is based on criteria for successful MITRE systems engineers. The MITRE model has three cumulative levels of proficiency (i.e., levels of proficiency) and consists of 36 competencies organized into the following five sections:

1. Enterprise Perspectives.
2. Systems Engineering Life Cycle.
3. Systems Engineering Planning and Management.
4. Systems Engineering Technical Specialties.
5. Collaboration and Individual Characteristics.

The authors of this model do not claim that their model is a general competency model. They explicitly state that the model is tailored to the MITRE needs. The model was not "scientifically" validated. The authors generally claim that, *"The original draft competencies were based upon information from standards bodies, the MITRE Institute, commercial companies, and Government sources … The model went through numerous revisions with input from many people across MITRE before it reached this form. It will continue to evolve and be upgraded …"*

The MITRE systems engineering competency model has three increasing levels of proficiency:

1. Foundational.
2. Intermediate.
3. Expert.

MITRE assumes that a person's competence at a specific proficiency level is generally the result of education, work experience, job tasks, and specific job roles. A MITRE systems engineer is likely to be expert in some competencies, intermediate in others, and foundational in others. It is not expected, and it would be highly unlikely, that any one person would be expert in all the behaviours and competencies in this model.

10.8.9. The National Defense Industrial Association (NDIA) proposed systems engineering competency model

The NDIA proposed systems engineering competency model groups 50 competencies in the following four areas (Gelosh, 2008):

1. ***Analytical:*** containing 20 competencies covering systems engineering tools and techniques design considerations.
2. ***Technical management:*** containing 15 competencies in the technical management process.
3. ***General:*** containing five competencies pertaining to a total systems view.
4. ***Professional competencies:*** containing 10 competencies covering thinking, problem solving and inter-personal skills.

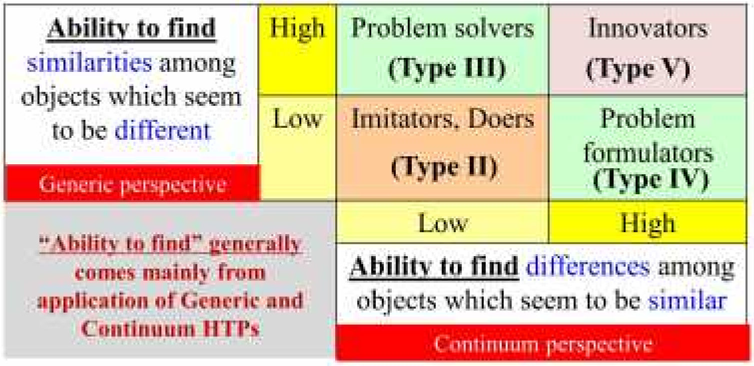


Figure 10.5 Matching Factors conducive to innovation to the five types of systems engineers

The planned approach, according to the presentation, was to develop the competencies based on the roles of systems engineers. Two years later, the model was still a work in progress (NDIA E&T, 2010).

10.9. The five types of systems engineers

Perceptions from the *Quantitative* perspective identified the following five types of systems engineers based on observations of their ability to deal with problems and solutions (Kasser, et al., 2009).

- ***Type I:*** apprentices who have to be told "how" to implement the solution system.
- ***Type II:*** imitators/doers. This type is the most common type of systems engineer. Type IIs have the ability to follow a process to implement a physical solution system once told what to do.
- ***Type III:*** problem solvers. Once given a statement of the problem, this type has the expertise to conceptualize the solution system and to plan the implementation of the solution, namely create the process to realize the solution.
- ***Type IV:*** problem formulators. This type has the ability to examine the situation and define the problem (Wymore, 1993: page 2), but cannot conceptualise a solution.
- ***Type V:*** engineer-leaders or innovators. This type is rare and combines the abilities of the Types III and IV, namely has the ability to examine the situation, define the problem, conceptualise the solution system and plan and manage the implementation of the physical solution.

Four of the five types were then matched to the factors conducive to innovation discussed in Section 9.9 as shown in Figure 10.5. Type IIs tend to rate low in their ability to identify similarities among objects that

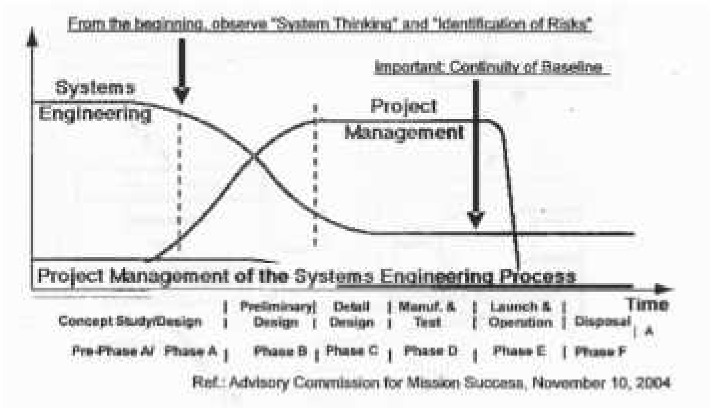


Figure 10.6 Relation between workload in the project lifecycle (JAXA)

appear to be different as well as their ability to identify differences among objects which seem to be similar. Accordingly, Type IIs tend to look for patterns and follow the process for dealing with the pattern.

10.10. The number of systems engineers in the different states of the SDP

The relative number of systems engineers and other disciplines needed in the different phases of the system lifecycle identified by the Japan Aerospace Exploration Agency (JAXA) is shown in Figure 10.6 (JAXA, 2007) Figure 2-2). The crossover point seems to be at the SRR when the SDP transitions from the Requirements State to the System Design State, (Section 9.12). JAXA's interest in the system appears to focus on the early states of the SDP. Since there is some systems engineering activity in the System Test and System Integration States, a more realistic representation might be as shown in Figure 10.7. This figure also shows the relative

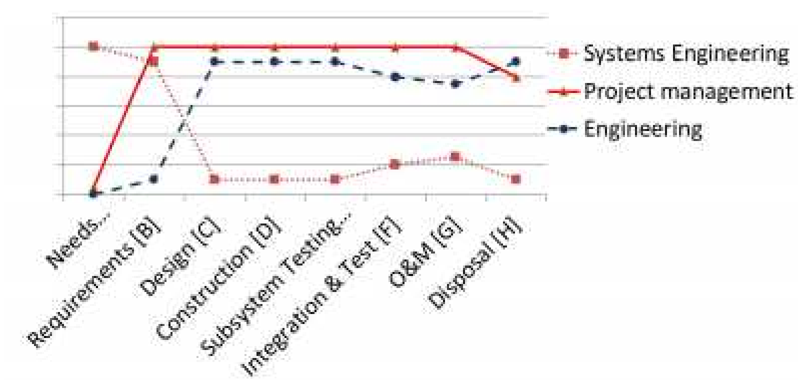


Figure 10.7 Representation of activities in the different states of the SLC

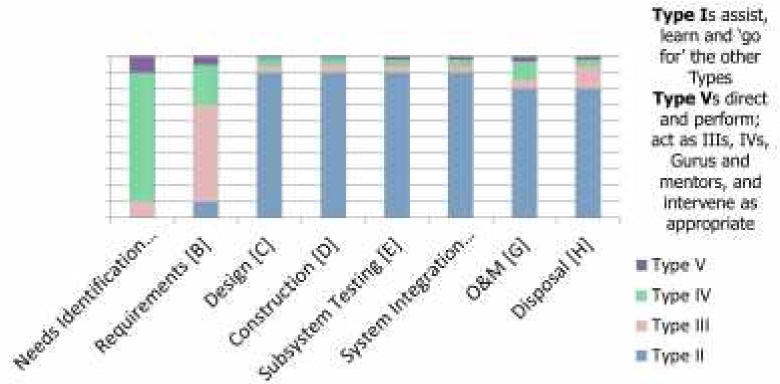


Figure 10.8 Relative number of each type of systems engineer in the states of the SLC

representation of the systems engineering, project management and engineering activities in the different states of the SLC (Section 9.12). As in the JAXA figure, there is a lot of systems engineering in the early states which dwindles during the System Design, Subsystem Construction and Subsystem Testing States and then increases during the System Integration and System Testing States. Once the system is in the O&M State, the amount of systems engineering depends on the amount of changes and upgrades. The amount of engineering is minimal in the early states and picks up in the System Design and Subsystem Construction States. Project Management is ubiquitous once the project Plan (PP) to realise the system has been approved.

Figure 10.8 shows the relative numbers of each type of systems engineer in the SLC. The figure shows that Type Vs are needed in Needs Identification and Requirements States of the SLC where they determine the nature of the problem, conceptualize the solution and create the realization plans and the matched set of specifications for the solution system. From the SRR onwards, Type IIs take over and follow the process designed by the Type Vs.

10.11. A way of assessing technology readiness

Two ways of assessing Technology Readiness Levels (TRL) were found in the literature:

- The NASA TRL discussed in Section 10.11.1.
- The US DoD TRL based on the NSAS TRL discussed in Section 10.11.2.

Table 10.2 NASA's TRLs

9	Actual system "flight proven" through successful mission operations
8	Actual system completed and "flight qualified" through test and demonstration (ground or space)
7	System prototype demonstration in a space environment
6	System/subsystem model or prototype demonstration in a relevant environment (ground or space)
5	Component and/or breadboard validation in relevant environment
4	Component and/or breadboard validation in laboratory environment
3	Analytical and experimental critical function and/or characteristic proof-of concept
2	Technology concept and/or application formulated
1	Basic principles observed and reported

10.11.1. The NASA TRL

The NASA TRL shown in Table 10.2 is a tool that was developed in NASA to provide a "*systematic metric/measurement system that supports assessments of the maturity of a particular technology and the consistent comparison of maturity between different types of technology*" (Mankins, 1995). The project manager could assess various technologies and determine which one to use. TRLs=1, 2, 3, and 4 seem to constitute the research levels, TRL=5 and 6, the development levels and TRL=9, the production level. The TRL was used in NASA and later adopted by the DoD (GAO, 1999) to assess a technology and approve it for use if it was above a certain TRL.

However, a number of deficiencies in the TRL which reduce its fitness for purpose have been pointed out. For example:

- Katz et al. wrote "*Program managers underestimate the time and technical effort needed to mature technologies above TRL=6 to achieve higher levels of maturity*" (Katz, et al., 2014).
- Sauser et al. wrote (Sauser, et al., 2006) "*it has been stated that the TRL:*
  1. *does not provide a complete representation of the (difficulty of) integration of the subject technology or subsystems into an operational system (Dowling and Pardoe, 2005; Mankins, 2002; Meystel, et al., 2003; Valerdi and Kohl, 2004),*
  2. *includes no guidance into the uncertainty that may be expected in moving through the maturation of TRL (Cundiff, 2003; Dowling and Pardoe, 2005; Mankins, 2002; Shishkio, et al., 2003; Smith, 2005; Moorehouse, 2001), and*
  3. *assimilates no comparative analysis technique for alternative TRLs (Cundiff, 2003; Dowling and Pardoe, 2005; Mankins, 2002; Smith, 2005; Valerdi and Kohl, 2004)".*

Table 10.3 DoD Technology Readiness Levels and their definitions (GAO, 1999)

9	Actual system "flight proven" through successful mission operations
8	Actual system completed and "flight qualified" through test and demonstration
7	System prototype demonstration in an operational environment
6	System/subsystem model or prototype demonstration in a relevant environment
5	Component and/or breadboard validation in relevant environment
4	Component and/or breadboard validation in laboratory environment
3	Analytical and experimental critical function and/or characteristic proof of concept
2	Technology concept and/or application formulated
1	Basic principles observed and reported

10.11.2. The DoD TRL

The DoD TRL based on the NASA TRL also established nine levels of TRLs or maturity as shown in Table 10.3 (GAO, 1999) which are almost identical to NASA's TRL shown in Table 10.2. DoD's description of each level (lowest to highest) is:

1. ***Lowest level of technology readiness***. Scientific research begins to be translated into applied research and development. Examples might include paper studies of a technology's basic properties.
2. ***Invention begins***. Once basic principles are observed, practical applications can be invented. The application is speculative and there is no proof or detailed analysis to support the assumption. Examples are still limited to paper studies.
3. ***Active research and development is initiated***. This includes analytical studies and laboratory studies to physically validate analytical predictions of separate elements of the technology. Examples include components that are not yet integrated or representative.
4. ***Basic technological components are integrated*** to establish that the pieces will work together. This is relatively "low fidelity" compared to the eventual system. Examples include integration of "ad hoc" hardware in a laboratory.
5. ***Fidelity of breadboard technology increases significantly***. The basic technological components are integrated with reasonably realistic supporting elements so that the technology can be tested in a simulated environment. Examples include "high fidelity" laboratory integration of components.

6. ***Representative model or prototype system, which is well beyond the breadboard*** tested for TRL 5, is tested in a relevant environment. Represents a major step up in a technology's demonstrated readiness. Examples include testing a prototype in a high fidelity laboratory environment or in simulated operational environment.
7. ***Prototype near or at planned operational system***. Represents a major step up from TRL 6, requiring the demonstration of an actual system prototype in an operational environment, such as in an aircraft, vehicle or space. Examples include testing the prototype in a test bed aircraft.
8. ***Technology has been proven to work in its final form and under expected conditions***. In almost all cases, this TRL represents the end of true system development. Examples include Developmental Test and Evaluation (DT&E) of the system in its intended weapon system has determined that the technology meets its design specifications.
9. ***Actual application of the technology in its final form and under mission conditions,*** such as those encountered in Operational Test and Evaluation (OT&E). In almost all cases, this is the end of the last "bug fixing" aspects of true system development. Examples include using the system under operational mission conditions.

10.12. The four levels of difficulty of the problem

The level of difficulty of a problem tends to be subjective[5]. Ford introduced four categories of increasing order of difficulty for well-structured mathematics and science problems: easy, medium, ugly, and hard (Ford, 2010). These categories may be generalized and defined as follows:

1. ***Easy problems***: which are problems that can be solved in a short time with very little thought.
2. ***Medium problems***: which can be solved after some thought, may take a few more steps to solve than an easy problem and can probably be solved without too much difficulty, perhaps after some practice.
3. ***Ugly problems***: which are ones that will take a while to solve. Solving them involves a lot of thought, many steps and may require the use of several different concepts.
4. ***Hard problems***: which usually involve dealing with one or more unknowns. Solving them involves a lot of thought and some research and may also require iteration through the prob-

[5] A problem that is easy for you to solve may be a difficult problem for someone else.

lem-solving process as learning takes place (as knowledge that was previously unknown becomes known).

Classifying problems by level of difficulty is difficult in itself because difficulty is subjective since one person's easy problem may be another person's medium, ugly or hard problem.

10.13. Summary

This Chapter contained perceptions of systems engineering from the *Quantitative* perspective. The key points were:

- Research has shown there is value in systems engineering.
- While requirements are considered an essential part of systems engineering, there is no metric for measuring the goodness of a requirement.
- There are three types of emergent properties.
- There are ways of measuring and improving systems engineering.
- The TRL.
- The four levels of difficulty of a problem.

--oo--

11. The Temporal perspective

The *Temporal* perspective is a progressive perspective which incorporates Richmond's dynamic thinking (Richmond, 1993) and perceives the system as it was in the past, is in the present and as it will be in the future. If the system exists, past patterns of behaviour are perceived and future patterns are predicted using this perspective.

Perceptions of systems engineering from the *Temporal* perspective included:

1. The history and origin of systems engineering discussed in 11.1.
2. The evolution of the role of the systems engineer discussed in Section 11.2.
3. The evolution of requirements engineering discussed in Section 11.3.
4. The use of models in systems engineering discussed in Section 11.4.
5. The introduction of the V diagram discussed in Section 11.5.

11.1. The history and origin of systems engineering

Systems engineering was used when building the pyramids in ancient Egypt around 2575 BC.

While software engineering claims to have invented the object-oriented paradigm, it was documented as an analysis methodology at least as early as the 12th Century when Maimonides (Maimonides, circa 1200: pages 69-70) wrote, "An object is characterized by its:

- Definition.
- Part of its definition, namely what it inherits from a parent.
- Attributes.
- Relationships with other objects.
- Internal actions.

Engineers ran organisations before managers, for example Frederick W. Taylor presented his paper on "Shop Management" to a meeting of the American Society of Mechanical Engineers (George, 1972: page 92) and other early 20th century leaders of management thought such as Harrington Emerson and Henry Gantt published in engineering journals (George, 1972: pages 104-107).

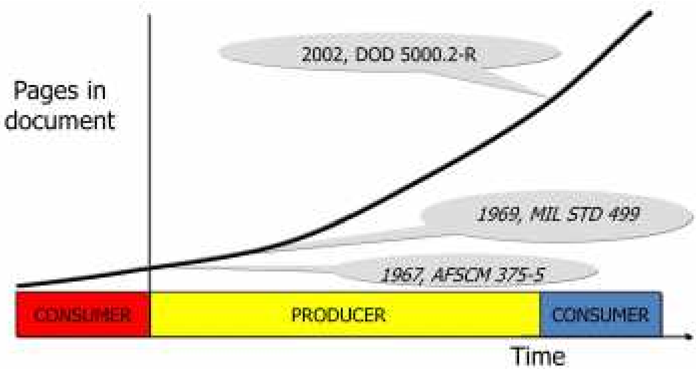


Figure 11.1 The increase in the degree of micromanagement in the Standards

The term 'systems engineering' has only been in existence since the middle of the 20th century (Johnson, 1997; Jackson and Keys, 1984; Hall, 1962).

In the 1960's systems engineering was a discipline dealing with complexity over the whole SLC (Jenkins, 1969; Chapanis, 1960). Since then systems engineering devolved into the 'A' and 'B' paradigms discussed in Section 9.21. In the DoD where systems engineering followed the 'B' paradigm, the whole set of activities performed in the Needs Identification State of the SDP were removed from systems engineering in DoD 5000[1] where:

- DoD 5000.1 required the use of systems engineering.
- DoD 5000.2 emphasized the use of systems engineering and assigned the Needs Identification State systems engineering activities to CAIV to be performed by IPTs (DOD, 1998; DOD 5000.2-R, 2002).

The degree of micromanagement in the Standards increased exponentially over time from the AFCM 365-5 in 1967 (Gelbwaks, 1967) to the DoD Architecture Framework (DODAF) in 2004 (DoDAF, 2004)[2] as illustrated in Figure 11.1.

11.2. The evolution of the role of the systems engineer

In 1969 Jenkins listed the following 12 roles of the systems engineer (Jenkins, 1969: page 164):

---

[1] This removal was documented in 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.

[2] Based on the page count of the documentation

1. "*He tries to distinguish the wood from the trees – what's it all about?*
2. *He stimulates discussion about objectives – obtains agreement about objectives.*
3. *He communicates the finally agreed objectives to all concerned so that their co-operation can be relied upon.*
4. *He always takes an overall view of the project and sees that techniques are used sensibly.*
5. *By his overall approach, he ties together the various specializations needed for model building.*
6. *He decides carefully when an activity stops.*
7. *He asks for more work to be done in areas which are sensitive to cost.*
8. *He challenges the assumptions on which the optimization is based.*
9. *He sees that the project is planned to a schedule, that priorities are decided, tasks allocated, and above all that the project is finished on time.*
10. *He takes great pains to explain carefully what the systems project has achieved, and presents a well-argued and well-documented case for implementation.*
11. *He ensures that the users of the operational system are properly briefed and well trained.*
12. *He makes a thorough retrospective analysis of systems performance*".

Seven of these roles of the systems engineer (activities performed by a person with the title systems engineer) overlap the role of the project manager (activities performed by a person with the title project manager).

Almost 20 years later, Sheard documented the following different twelve systems engineering roles (Sheard, 1996):

1. "**Requirements Owner Role.** *Requirements Owner/requirements manager, allocator, and maintainer/specifications writer or owner/developer of functional architecture/developer of system and subsystem requirements from customer needs.*
2. **System Designer Role.** *System Designer/owner of "system" product/chief engineer/system architect/developer of design architecture/specialty engineer (some, such as human-computer interface designers)/"keepers of the holy vision" (Boehm, 1994).*
3. **System Analyst Role.** *System Analyst/performance modeller/keeper of technical budgets/system modeller and simulator/risk modeller/specialty engineer (some, such as electromagnetic compatibility analysts).*
4. **Validation and Verification Role.** *Validation and Verification engineer/test planner/owner of system test program/system selloff engineer. Validation and Verification engineers plan and implement the system*
5. **Logistics and Operations Role.** *Logistics, Operations, maintenance, and disposal engineer/developer of users' manuals and operator training materials.*

6. **Glue Role.** *Owner of "Glue" among subsystems/system integrator/owner of internal interfaces/seeker of issues that fall "in the cracks"/risk identifier/"technical conscience of the program".*
7. **Customer Interface Role.** *Customer Interface/customer advocate/customer surrogate/customer contact.*
8. **Technical Manager Role.** *Technical Manager/planner, scheduler, and tracker of technical tasks/ owner of risk management plan/product manager/product engineer.*
9. **Information Manager Role.** *Information Manager (including Configuration Management, data management, and metrics).*
10. **Process Engineer Role.** *Process engineer/business process reengineer/business analyst/owner of the SEP.*
11. **Coordinator Role.** *Coordinator of the disciplines/tiger team[3] head/head of IPTs/system issue resolver.*
12. **"Classified Ads Systems Engineering" Role.** *This role was added to the first eleven in response to frustration encountered when scanning the classified ads, looking for the INCOSE -type of systems engineering jobs*".

Some of the evolution in systems engineering can be seen in the very little overlap between the 12 roles documented by Jenkins and the 12 systems engineering roles documented by Sheard. Jenkins' roles relate to conceiving and planning the solution system while almost 30 years later, few of Sheard's roles address the original systems engineering approach to conceiving and planning the solution system. Sheard's set of roles relate to interpersonal relationships between the practitioners of disparate skills and disciplines implementing the solution system. Furthermore, according to both Jenkins and Sheard the role of the systems engineer (the activities performed by a person with the title systems engineer) overlaps activities performed (the roles) by people from other professions[4].

11.3. The evolution of requirements engineering

Jarke wrote that requirements engineering is a discipline that seems to be evolving from its traditional role as a mere front-end to the systems lifecycle towards a central focus of change management in system-intensive organizations (Jarke, 1996). Two definitions of requirements engineering that support this statement are:

- In 1990, the definition of requirements engineering was, *"the science and discipline concerned with analysing and documenting requirements"* (Dorfman and Thayer, 1990).

---

[3] A temporary team created to solve specific urgent problem.

[4] A different set of activities, as seen across the years.

- In 2000, the definition of requirements engineering was, *"the systematic process of eliciting, understanding, analysing, documenting and managing requirements"* (Kotonya and Summerville, 2000).

The 1990 definition is consistent with the 'A' paradigm and the 2000 definition is consistent with the 'B' paradigm (Section 9.21).

11.4. The use of models in systems engineering

The use of models in systems engineering is not a new concept. Models have been used in systems engineering since its earliest days. For example:

- The CONOPS has been a part of systems engineering since the early days of the 'A' paradigm.
- Modelling has long been used in the form of schematics, prototypes and scale models. For example, *"during the 1950s and 1960s electronic and hybrid analogue computers were at the heart of modelling such technological systems as aerospace and industrial plant control"* (Bissell, 2004).
- Arthur D. Hall discusses the use of models and simulations (Hall, 1962: page 131).
- Jenkins's fifth role in 1969, "*By his overall approach, he ties together the various specializations needed for model building*" in Section 11.2.
- The US Air Force (USAF) recognized the need for semantic models to represent conceptual schemas in the mid 1970's as a result of the Integrated Computer Aided Manufacturing (ICAM) Program. The ICAM program developed a series of techniques known as the ICAM Definition (IDEF) methods (IEEE 1320, 1998), which included the following:
  - IDEF0, a technique used to produce a "function model" which is a structured representation of the activities or processes within the environment or system.
  - IDEF1, a technique used to produce an "information model" which represents the structure and semantics of information within the environment or system.
  - IDEF2, a technique used to produce a "dynamics model" which represents the time-varying behavioural characteristics of the environment or system.
- Hately and Pirbhai's methodology employs three models, Requirements, Architecture and Specification (Hately and Pirbhai, 1987).
- The conceptual model in Checkland's SSM (Checkland and Scholes, 1990).

- The early mathematical models (Wymore, 1993; Chapman, et al., 1992; Saaty and Alexander, 1981).
- Menzes et al. discuss viewpoint-based requirements engineering, and its advantages (Menzes, et al., 1999).
- Lagakos et al. write that, "*The object-model formulation views a system as a group of interacting objects that work together to accomplish system objectives and satisfy system requirements. Use-case and domain models provide a visual representation for high-level system functionality and system design*" (Lagakos, et al., 2001).
- The DODAF calls out 26 different views (DoDAF, 2004).

Before the personal computer became ubiquitous, systems engineers working in one of the states of the SDP produced documents which became inputs to the subsequent state or states. For example in the 'A' paradigm of systems engineering (Section 9.21.1), the CONOPS document was an input to the Requirements State; the System Requirements Document (SRD) was an input to the System Design State and so on. These documents were typewritten on paper originals. Copies for distribution were made using appropriate duplication technology and version control was performed using Configuration Management (CM). As technology advanced, in the latter years of the 20th century, information technology provided word processors, databases, spreadsheets and electronic storage. Even with electronic storage capability the paradigm hardly changed; information was still stored as separate documents and copies were printed as needed. Pioneers of the use of information technology realized that documents could be considered not as information per sé, but as selected displays or views of part of the information in an underlying database. This paradigm shift changed the storage of project information from separate documents or files to interlinked files where information is stored in one place, linked to information in another place and viewed from different perspectives. The concept behind the DODAF (DoDAF, 2004) is but one such example.

First generation project management tools allowed schedules, cost estimates and other project information to be stored in databases and automated the process of producing charts and reports. Similarly first generation Computer Enhanced Systems Engineering (CESE) tools provided storage for product or the system-to-be-realized information and some useful functionality. For example, the tools allowed the user to link requirements to sources providing traceability. Second generation Computer Aided Software Engineering (CASE) and CESE tools allowed systems engineers to build executable models of proposed conceptual systems as well as software-based models and simulations. MBSE is but one way of applying CESE tools.

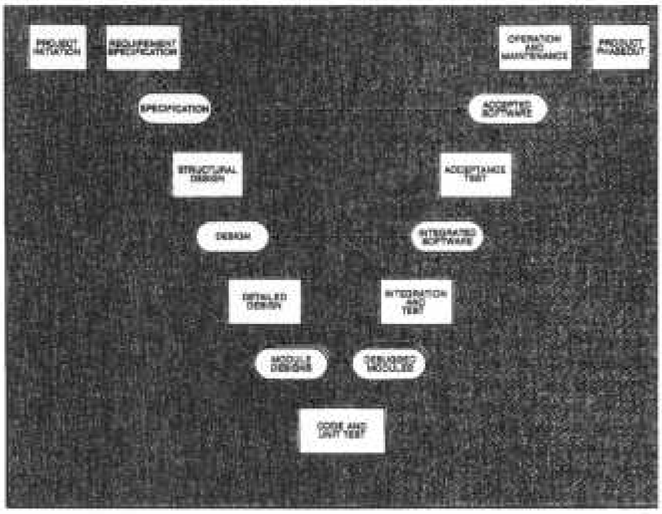


Fig. 3 The stages in software development confidence

Figure 11.2 The V diagram for software development (Rook, 1986)

11.5. The introduction of the V diagram

The V diagram is often described as a depiction of the systems engineering process. Practitioners however tend to forget or are unaware that it is a three dimensional model and in its two-dimensional representation it is only an overview of some of the aspects of the SDP relating development to Test and Evaluation (T&E) at the various states of the SDP while abstracting out all other information. The V diagram was initially introduced into both software and systems engineering as a project management tool.

A literature search found the first mention of the V diagram (Rook, 1986) where it was introduced as a software project management tool illustrating the concept of verification the process-products at established milestones. The original figure shown in Figure 11.2 was captioned "*the stages in software development confidence*". It was drawn to show that the intermediate process products produced at each state of the software development process were to be verified against previous baselines before starting work on the subsequent state.

The V diagram seems to have been introduced to the systems engineering community (Forsberg and Mooz, 1991) as a project management tool. Forsberg, Mooz and Rook all state that the simplistic view of the product development cycle is not to be interpreted as a waterfall namely that each state is to be completed before the next begins. They agree that explanatory work on subsequent states is often required before a state is

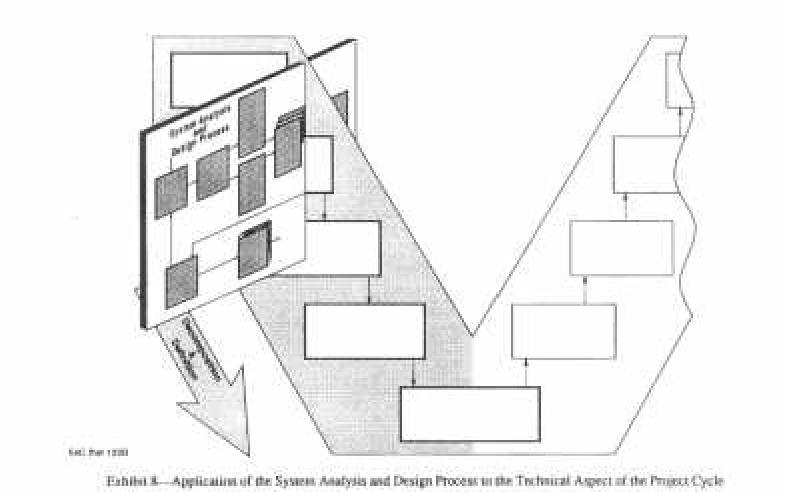


Figure 11.3 The three dimensions to the V diagram (Forsberg and Mooz, 1991)

complete and there is a third dimension to the model. Forsberg and Mooz include a representation of that third dimension in their paper and one of their figures, extracted from their paper is shown in Figure 11.3.

11.6. Summary

This Chapter contained perceptions of systems engineering from the *Temporal* perspective. The key points were:

- Systems engineering as a discipline has only existed since the middle of the 20th century.
- The evolution of the role of the systems engineer.
- The evolution of requirements engineering.
- The use of models in systems engineering is not a new concept.
- The introduction of, and increase in the degree of micromanagement in, the Standards for systems engineering.

--oo--

PART III

The *Scientific* Perspective

Part III is where the outcomes of the analysis of the information in the descriptive perspectives on ways to improve systems engineering, namely the inferences and insights from the *Scientific* perspective begin and continue through Parts IV and V.

The *Scientific* perspective is where the inferences, insights and ideas produced by the analysis of the descriptive perceptions and observed facts documented in Chapters 4 to 11 are presented. This approach separates the facts from the opinions.

The *Scientific* perspective:

- Incorporates Richmond's scientific thinking (Richmond, 1993) and is the output of the analysis process; namely, the opinions, insights and inferences made by the analysis using critical thinking.
- Contains lessons learned, a statement of the problem, the design of the solution or the guess, etc.

Specifically:

- Chapter 12 contributes to the improvement of systems engineering by containing some insights, inferences and explanations from analysing the information in Chapters 4 to 11. This Chapter and the following chapters should invoke discussions and debates between systems engineers with different perspectives from single viewpoints of systems engineering.
- Chapter 13 contributes to the improvement of systems engineering by perceiving the System Lifecycle (SLC) as a State Machine producing some innovative insights.

- Effective workmen sharpen their tools. Effective systems engineers not only sharpen their tools they are also always on the lookout of new tools that they can adopt or modify for their own use. Chapter 14 contains a selection of tools and frameworks for improving the practice of systems engineering which have been conceptualised, prototyped and found to be useful.
- Chapter 15 presents an underpinning axiom for systems engineering. The principles within the axiom apply to the solution system, production of which is the common goal of all the systems engineering camps (Section 9.17). As a consequence, the axiom has the potential to improve systems engineering by uniting the disparate systems engineering camps by allowing them to agree on the principles applying to the solution system which will then enable the practice of systems engineering to repeat the successes it achieved in the NASA environment in the 1960's and 1970's in many current and future application domains.

--oo--

12. Insights, inferences and explanations

This Chapter is the first chapter in the *Scientific* perspective where the inferences, insights and ideas produced by the analysis of the descriptive perceptions and observed facts documented in Chapters 4 to 11 are presented. The inferences, insights and idea presented in the Chapter are:

1. Some reasons why systems engineers cannot agree on the nature of systems engineering discussed in Section 12.1.
2. The three types of systems engineering discussed in Section 12.2.
3. The myth of the single SEP discussed in Section 12.3.
4. The implementation domain discussed in Section 12.4.
5. The devolution of systems engineering discussed in Section 12.5.
6. Explaining the similarity between the SEP, the problem-solving process and the decision-making process discussed in Section 12.6.
7. Resolving the overlap between systems engineering and project management; discussed in Section 12.7.
8. The 'B' paradigm is inherently flawed; discussed in Section 12.8.
9. The question "Is there an alternative to "requirements"?; discussed in Section 12.9.
10. Five reasons for the failures of systems engineering discussed in Section 12.10.
11. Reasons for the success of systems engineering discussed in Section 12.11.
12. Problem solving is not taught very well; discussed in Section 12.12.
13. The reason for the different descriptions of the problem-solving process discussed in Section 12.13.
14. Changing the SDP from a single waterfall to a series of waterfalls at project planning time; discussed in Section 12.14.
15. Consider the SLC as a State Machine; discussed in Section 12.15.
16. The two interdependent sequential SEPs discussed in Section 12.16.
17. Systems engineering is a discipline; discussed in Section 12.17.
18. Systems engineering is demonstrating the symptoms of a discipline in its early stages; discussed in Section 12.18.
19. Resolving the paradoxes; discussed in Section 12.19.

20. The two processes for creating a system discussed in Section 12.20.
21. MBSE discussed in Section 12.21.
22. Previous ways of dealing with complexity in the INCOSE literature discussed in Section 12.22.
23. The need to focus on people as well as process discussed in Section 12.23.
24. Systems engineering is more than just applying process standards; discussed in Section 12.24.
25. Ways of assessing competency in systems engineering discussed in Section 12.25.
26. Improving the practice of systems engineering by adjusting the terminology discussed in Section 12.26.
27. The Standards for systems engineering are a myth; discussed in Section 12.27.
28. Systems of Systems are a different class of problem and need new tools and techniques is a myth; discussed in Section 12.28.
29. The existence of myths of, and defects in, systems engineering (Kasser, 2007; 2010b). Three of these myths are:
    a) The myth of the single SEP discussed in Section 12.3.
    b) The myth of the systems engineering Standards discussed in Section 12.27.
    c) The myth of Systems of Systems discussed in Section 12.28.
30. Aspects of detailed design decisions discussed in Section 12.29.
31. The many definitions of a system including those listed in Section 9.1 are formulations of problem statements by the persons who wrote the definitions (Kasser and Palmer, 2005). They defined their system to suit their problem in accordance with Beer and Churchman (Beer, 1994; Churchman, 1979: page 91).
32. The waterfall, V and spiral views/models are different views of the same sequential process as partly shown in Figure 12.1 where:
    - The waterfall (Royce, 1970) is a planning view perceiving the process from before it starts.
    - The V is a view relating development activities to testing activities is a waterfall view with the latter sections raised to the corresponding level of earlier sections as discussed in Section 11.5.
    - The spiral model (Boehm, 1988: pages 61-72) is the waterfall curved into a spiral with an emphasis on risk management.

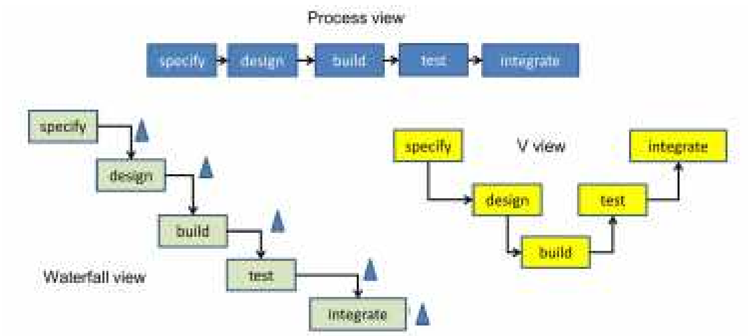


Figure 12.1 Three different views of the same sequential process

12.1. Some reasons why systems engineers cannot agree on the nature of systems engineering

Some reasons why systems engineers cannot agree on the nature of systems engineering are based on:

- The different roles performed by systems engineers in the workplace discussed in Section 12.1.1.
- The difference between the roles and activities discussed in Section 12.1.2.

12.1.1. The different roles performed by systems engineers

The different roles performed by systems engineers and the different types of products developed in different States of the SDP in different areas of the Hitchins-Kasser-Massie Framework (HKMF) (Section 14.4) are two of the reasons why systems engineers, using different single viewpoints from their area, could not agree on the nature of systems engineering. Systems engineering activities are different in each area of the HKMF. Consequently, when systems engineers working in different areas of the HKMF discuss what they do, they are discussing a different set of activities. From the *Generic* perspective, this is a similar situation to the fable of the blind men studying the elephant and drawing different conclusions (Yen, 2008).

12.1.2. The difference between the roles and activities

The two dimensions of the Roles Rectangle shown in Figure 9.11 provide a simplified representation of the four roles from the perspective of planning and implementing the product and the development process producing the product. Each architecting role requires knowledge of the activities of both implementation activities since for example:

- ***Product:*** there is little pointing designing a product that cannot be produced either because the specifications are unachievable (e.g., requirements to travel faster than the speed of light are not achievable with today's technology).

- ***Process:*** there is little point in setting schedules that are not feasible due to lack of resources, or time.

In addition, each implementation role also performs some planning.

Each quadrant in the Roles Rectangle shown in Figure 9.11 contains activities performed by the speciality disciplines. For example, Eisner describes 38 speciality disciplines in systems engineering alone (Eisner, 1988). Some of these disciplines are present in the other three quadrants. For example risk management is an activity that takes place in all four quadrants (Section 12.7).

If there was a one-to-one mapping of the roles to the activities, then there would be little discussion as to the differences between roles of the systems engineer and the roles of the project manager. All the activities in the systems engineering quadrant would be performed by the systems engineer, and the project manager would perform all the activities in the project management quadrant. It is when the boundaries of a role, as defined by the job description, contain activities located in another quadrant that discussions arise.

The work in developing systems is interdisciplinary. It incorporates a large number of engineering, management, and other activities that have to be performed (e.g. requirements management, design, decision making, problem solving, validation and verification, test and evaluation, risk management, reliability, and logistics, process design and improvement, etc.) (Watts and Mar, 1997). In small projects, one person might perform all of the activities. On larger projects, the activities tend to be grouped (slightly) differently in different organisations in different jobs that are not exactly aligned with the organisational roles. Thus a systems engineer's job does not exactly align with the activities of systems engineering. As both Roe and Sheard noted, a systems engineer can perform some systems engineering activities and also perform some project management activities (Roe, 1995; Sheard, 1996). They can also perform architecting activities, yet the job description is "Systems Engineer". However, in a different organisation, the partition of work into different jobs is also not exactly aligned with the activities but in a different way. This means that in different organisations, the partition of work between the jobs of (the roles of) Systems Engineer, Project Manager, and Systems Architect will probably be different. Thus the same person known as a "Systems Engineer" or an "Engineering Specialist" might perform one mixture of systems engineering, project management, and systems architecting as shown in Figure 12.2. In a different organisation, the person with the same role might perform a different mixture of systems engi-

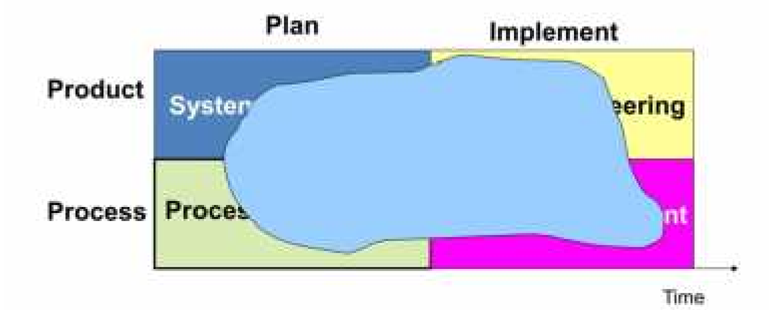


Figure 12.2 Role of the systems engineer in one organisation

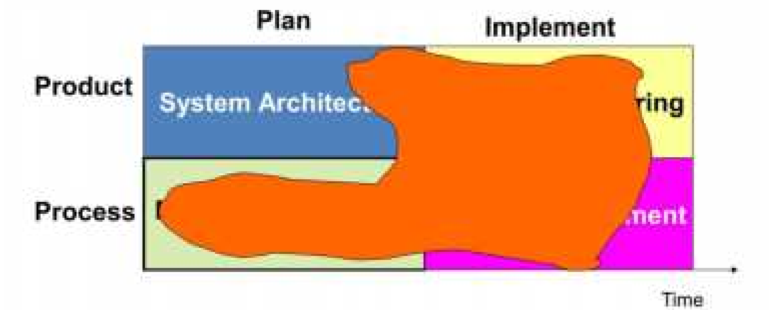


Figure 12.3 Role of the system engineer in another organisation

neering, project management, and systems architecting activities as shown in Figure 12.3. This situation means that in any one organisation, in general, the roles performed by the jobs of systems engineer, systems architect and project manager while they do not map directly into their corresponding activities, also do not overlap each other's roles (unless there is a turf war in progress). The activities only overlap job roles when compared across different organisations.

12.2. The three types of system engineering

This framework was created to sort information perceived during a benchmarking study of the stated content[1] of the required courses in 10 systems engineering Master's degrees in 2013 (Kasser and Arnold, 2014).

The *Generic* perspective provided the perception of the similarity between systems engineering and mathematics (Section 8); two disciplines providing tools used to solve certain types of problems in other disciplines. Mathematics is divided into two parts; pure and applied mathematics, systems engineering can similarly be divided into three parts; pure, applied and domain systems engineering where:

---

[1] Recognizing that what was actually taught may not be what was stated on the institution's Web site.

- ***Applied systems engineering*** consists of the activities performed in the operational scenarios in which systems engineering is performed discussed in Chapter 5.
- ***Pure systems engineering*** consists of the functions performed in the activities discussed in Chapter 6.
- ***Domain systems engineering*** consist of the knowledge pertinent to the application of systems engineering in a domain and is usually found in the domain literature rather than in the systems engineering literature.

Before defining the three types of systems engineering, it was difficult to see any patterns in the research findings. Once sorted by the three types of systems engineering it was easy to infer information including:

- Different postgraduate courses had different mixtures of pure systems engineering, applied systems engineering and domain systems engineering.
- The differences in the content of textbooks on systems engineering discussed in Section 9.22 was because each textbook focused on a different mixture of pure systems engineering, applied systems engineering and domain systems engineering.

This framework simplifies the problem of creating a Systems Engineering Body of Knowledge (SEBOK) since:

- In the SETR paradigm, the SEBOK encompasses systems engineering knowledge and the body of knowledge used in each role of the systems engineer and domain systems engineer. As such, the body of knowledge needs to contain knowledge from the other disciplines according to the activities performed in the role which associates with the discipline and meta-discipline camp (Section 9.17.4).
- In the SETA paradigm, the SEBOK could be structured as pure and applied systems engineering. Although domain knowledge is critical to understanding the domain and the correctness of decisions, the problem of providing domain knowledge is the province of the party providing education to the domain.

  This framework has changed the boundaries of the problem of creating a SEBOK.

12.3. The myth of the single systems engineering process

According to Arnold, *"A single process, standardizing the scope, purpose and a set of development actions, has been traditionally associated with systems engineering"* (Arnold, 2000) citing (MIL-STD-499B, 1993) and (IEEE 1220, 1998). However, from the *Continuum* perspective there is no single widely agreed

Table 12.1 Focus of Standards – chronological order

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	No	✓	✓	✓	✓
System implementation	✓	✓	No	✓	✓
Technical analysis	✓	✓	✓	✓	✓
Technical management/ leadership	✓	✓	✓	✓	✓
Verification & validation	✓	✓	✓	✓	✓

upon SEP since over the years, the SEP has been stated in many different ways as discussed in Section 9.17.2. The key insight to understanding the reason for the variety of SEPs may lie with Biemer and Sage who state that *"the systems engineer creates a unique process for his or her particular development effort"* (Biemer and Sage, 2009: page 153). So, perceptions from the *Scientific* perspective, consider each published version of the SEP as the **unique process created for their particular development effort** by someone or some group at some point in time, at some point in the system lifecycle, in the context of what they defined as a systems engineering problem and subsequently documented as their SEP.

Using the *Generic* perspective to look for patterns in the various versions of the SEP as well as others in the literature summarized in Table 12.1 which contains data extracted from Table 5 in Honour and Valerdi (Honour and Valerdi, 2006) and rearranged in chronological order[2], one can identity versions of the SEP that focus on:

- Early state systems engineering where the problem is explored and conceptual solutions developed starting with mission/purpose definition.
- Engineering the system and realizing the solution.
- Both aspects.

12.4. The implementation domain

The implementation domain identified in Section 9.11 sets constraints on both the process and solution systems. For example, the development system for a software system constitutes part of the implementation domain. Implementation domain knowledge relates to the properties of the

[2] 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.

compiler as well as the characteristics (especially limitations) of the development hardware. In another environment such as aerospace, the implementation domain might include thermal vacuum chambers and other equipment used to partially or fully develop and test the solution system.

In the meta-discipline camp (Section 9.17.4) SETR paradigm (Section 9.18) the large number of implementation, problem and solution domains in which systems engineering takes place also requires a corresponding large Systems Engineering Body of Knowledge (SEBOK) which is not necessarily applicable to all systems engineers (Section 12.2). Consequently, the knowledge must be tailored to the specific problem, solution and implementation domains and the phase of the system lifecycle. For example:

- ***Requirements analysis.*** Systems engineers performing requirements analysis will need to know how to develop a matching set of specifications that describe the mission and support functions of the solution system in its fielded operational context; a different subset of systems engineering knowledge to that needed by systems engineers performing test and evaluation.
- ***Control or operations and maintenance environment.*** Systems engineers working in a control or operations and maintenance environment will need knowledge of the software development process and the tools, and the properties of the underlying development hardware platforms as well as the solution domain in which the system is to be fielded.
- ***Electro-optical engineering.*** Systems engineers working in an electro-optical engineering factory will need knowledge of how the various components can be configured without disturbing the performance of the system.
- ***Socio-technical systems.*** Systems engineers working on socio-technical systems will need the appropriate knowledge of human behaviour and how humans interact with technology and each other.
- And so on.

12.5. The devolution of systems engineering

Research shows that the systems engineers of the 1950's and 1960's tended to focus on identifying the problem (Wymore, 1993) and finding an optimal solution (Hall, 1962; Goode and Machol, 1959). These systems engineers were of Type III, Type IV, and Type V (Section 10.9), while the systems engineers who came later tended to focus on processes (Type II)'s. Back in the "good old days" of systems engineering Type III, Type IV and Type V systems engineers solved/resolved/dissolved the problem in the first SEP (Section 12.16) addressing the conceptual solution in the Needs Identification State of the SDP, then initiated the implementation of the solution, and moved on to the next contract (project), leaving the Type IIs to continue assisting the development of the solution system in the second SEP (Section 12.16).

There then came a time when there was a lack of new projects and so many of the Type III, Type IV and Type Vs were laid off and lost to the discipline. When the need for systems engineers in the US picked up again, in general only the Type II systems engineers were left and they took over systems engineering. They had missed the activities of the first SEP (Section 12.16) in the Needs Identification State and so their focus was on the second SEP. They wrote the Standards used in systems engineering (MIL-STD-499, 1969; MIL-STD-499A, 1974; EIA 632, 1994; IEEE 1220, 1998) for other Type II systems engineers to follow. These Standards in turn became the foundation for educating systems engineers. The 499, 499A, 632, 1220, and 15288 Standards cover the SEP and engineering management rather than systems engineering because there is actually very little systems engineering (SETA not SETR) in the System Design, Subsystem Construction, and Subsystem Testing States of the SDP for a single system in isolation. The mantra became 'follow the process and all will be well'. The term GIGO - garbage in, garbage out, was acknowledged but ignored.

12.6. Explaining the similarity between the systems engineering process the problem-solving process and the decision-making process

The similarity between the SEP, the problem-solving process and the decision-making process can be explained by recognizing:

- IEEE 1220 stated, *"The systems engineering process is a generic problem-solving process"* (IEEE 1220, 1998) Section 4.1). IEEE 1220's replacement of the term "the problem-solving process" by the term "the SEP " seems to have led to today's focus on process; specifically the second SEP (Section 12.16). Had the Standard instead stated, 'Systems engineers apply the generic problem-solving process', the focus of DoD-based and INCOSE-based systems engineering might have remained on the original focus (according to Jenkins) of managing complex problems (Jenkins, 1969) rather than devolving into the focus on process and developing new processes for each type of problem[3].
- As shown in Figure 12.4, when the problem is:

[3] An example of an unknown undesired emergent property at the time the Standard was created.

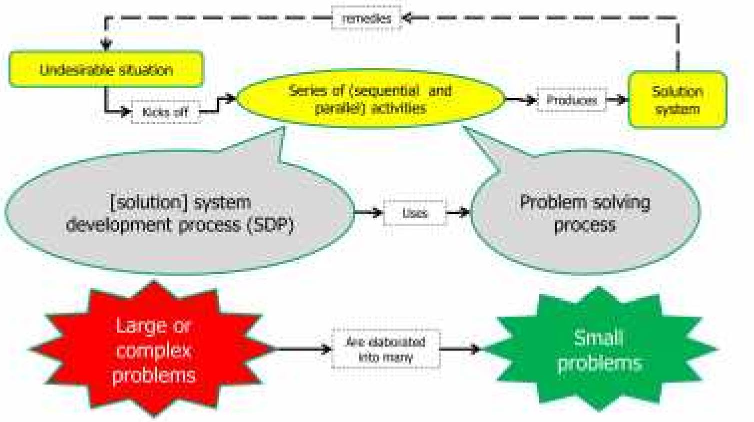


Figure 12.4. The SDP and the problem-solving process

- ***Small or non-complex*** the sequence of activities in the remedial action is known as the "problem-solving process".
- ***Large or complex,*** the sequence of activities in the remedial action is often known as the SEP instead of the SDP.

12.7. Resolving the overlap between systems engineering and project management

Research into the reason for the overlapping of the disciplines turned up information as to how the overlap originated in the form of the following statement, *"Driven by cold war pressures to develop new military systems rapidly, Operations Research, systems engineering, and project management resulted from a growing recognition by scientists, engineers and managers that technological systems had grown too complex for traditional methods of management and development"* (Johnson, 1997). Thus systems engineering, project management and Operations Research can be seen as three solutions to the problems posed by developing complex systems in the Cold War by three different communities of practice that have continued to evolve and overlap. The way to resolve the apparent overlap between systems engineering and project management (Section 9.20) is to recognise that the problem only exists in the SETR paradigm. This is because the role of the systems engineer in the workplace has evolved over time so that it is different in practically every organisation and has various degrees of overlap with the roles of project managers and personnel in other disciplines (Kasser and Massie, 2001; Kasser and Hitchins, 2009); the overlap does not exist in the SETA paradigm (Section 9.18).

The non-overlapping relationships between the SETA, project management and engineering activities performed to realize a solution system are shown in Figure 12.5 (Kasser and Hitchins, 2013). An entity associat-

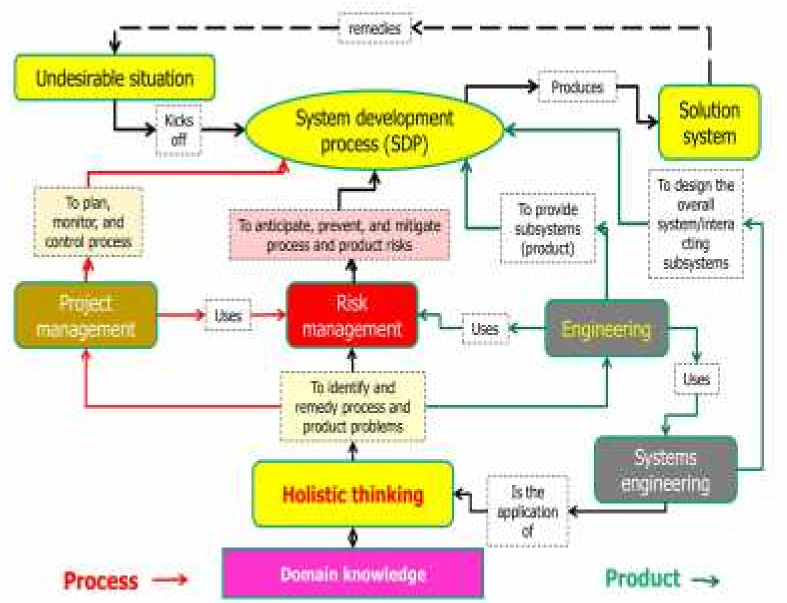


Figure 12.5. The relationships between the activities in the SDP in the SETA paradigm

ed with the undesirable situation initiates or kicks off the SDP which consists of a set of activities performed in series and in parallel (a process) which produces a solution system which is designed to remedy the undesirable situation. The activities in the SDP can be divided into project or engineering management, systems engineering and engineering where:

- ***Domain knowledge*** is the underpinning information used by the critical thinking element in holistic thinking in the performance of the activities performed in a SDP which require knowledge of the problem, solution and implementation domains (Section 9.11).
- ***Holistic thinking*** is pure systems engineering; the use of the thinking tools that use the domain knowledge to identify and remedy problems in undesirable situations in the activities known as systems engineering, project management, engineering and risk management.
- ***Risk management*** is the set of activities that anticipate, prevent and mitigate risks in the problem, solution (product and process) and implementation domains used in the activities known as project management, systems engineering and engineering.
- ***Project management*** is the set of activities known as planning,

organizing, directing and controlling (Fayol, 1949: page 8) the SDP. Project management incorporates risk management to manage process risks. Some of these activities are also currently known as systems engineering management.

- ***Systems engineering*** (SETA) is applied systems engineering; the set of activities performed by the systems engineers which incorporates risk management to manage system level product risks when designing and integrating the overall system/interacting subsystems.
- ***Engineering*** is the set of non-SETA engineering activities that incorporate risk management to manage product risks. These activities may be creating, by building, by purchasing Commercial-Off-The-Shelf (COTS) products, by changing a process, by reorganizing a human activity system or by a combination of all or some of the above.

In the SETR paradigm, the role or job of the system engineer, engineer and project manager is to perform an appropriate mixture of the activities known as systems engineering, engineering and project management as well as any other pertinent activities to the project. Due to the various ways in which SETA and non-SETA have been allocated to personnel[4] performing SETR and non-SETR, in any specific organisation at any specific time, roles and activities do not overlap 100%. Thus a person with the role or job title of systems engineer will perform a number of activities that include systems engineering, project or engineering management, and engineering. And an engineer might perform a mixture of engineering and systems engineering. This is why, project management, systems engineering and engineering have been perceived as being overlapping.

12.8. The 'B' paradigm is inherently flawed

The 'B' paradigm discussed in Section 9.21.2 is inherently flawed. This is because even if systems and software engineers working in a paradigm that begins in the Requirements State of the SDP could write perfectly good requirements, the CONOPS is derived from the requirements (Denzler and Mackey, 1994; Guo, 2010), and there is no way to determine if the requirements and associated information are correct and complete because there is no reference for comparison to test for the completeness. Consequently, efforts expended on producing better requirements have not, and will not, alleviate the situation. The situation cannot be alleviated because the situation is akin to participating in Dem-

[4] The word 'personnel' is used to avoid the semantically loaded terms engineers, systems engineers, project manager, etc.

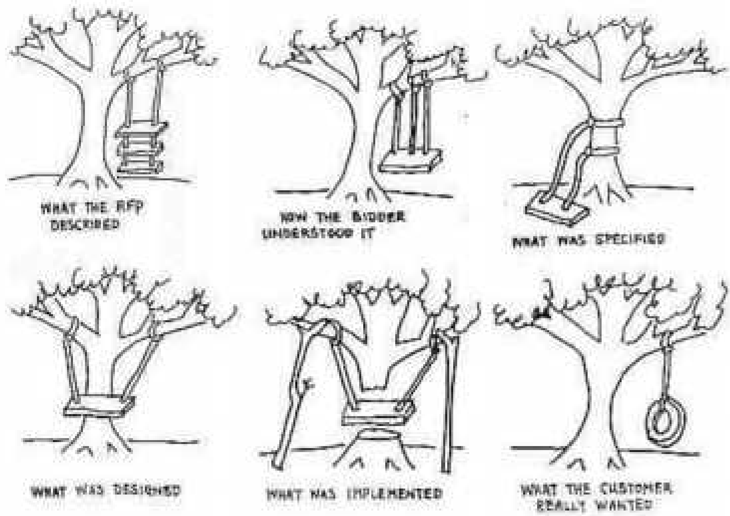


Figure 12.6 The consequences of not developing a CONOPS in the Needs Identification State of the SDP

ing's red bead experiment, which demonstrates that errors caused by workers operating in a process are caused by the system rather than the fault of the workers (Deming, 1993: page 158). The 'B' paradigm is best characterised by Figure 12.6[5] which does not describe a process, it reflects a lack of a common vision of what is to be produced, namely the consequences of not developing a CONOPS in the Needs Identification State of the SDP.

Recognition that the 'B' paradigm is inherently flawed is not a new observation. For example:

- Sutcliffe et al. proposed reducing human error in producing requirements by analysing requirements using an approach of creating scenarios as threads of behaviour through a Use Case, and adopting an object-oriented approach (Sutcliffe, et al., 1999); namely they proposed a return to the 'A' paradigm.
- Daniels et al. point out that standalone requirements make it difficult for people to understand the context and dependencies among the requirements, especially for large systems and suggest using Use Cases to define scenarios (Daniels, et al., 2005).
- MBSE is an attempt to return to, or recreate, the 'A' paradigm.

[5] I found this drawing in 1970 as a process description and it was old then. It has evolved somewhat in the intervening 40 years but the message it contains has not changed.

12.9. Is there an alternative to "requirements"?

Traditional systems engineering is focused on dealing with well-structured problems (Jackson and Keys, 1984). However, the problem of poor requirements is complicated and ill structured, and hence not solvable by the traditional SDP. A text-mode requirement should just be a simple sentence. Yet there are problems in the way requirement sentences are structured (Scott, et al., 2006). Contemporary requirements management practice irrespective of the paradigm used to generate the requirements is far from ideal, producing:

- ***Vague and unverifiable requirements:*** due to poor phrasing of the written text.
- ***Incompletely articulated requirements:*** due to a poor requirements elicitation process.
- ***Incomplete requirements:*** due to various factors including domain inexperience, and the lack of expertise in eliciting and writing requirements by technical staff.
- ***Poor management of the effect of changing user needs during the time that the system is under construction:*** due to lack of the understanding of the need for change management, and use of appropriate tools to do the function in an effective manner.

In conjunction with improving the writing of requirements, there also has been recognition that a requirement is more than just the imperative statement having additional properties (e.g. priority and traceability) (Alexander and Stevens, 2002; Hull, et al., 2002). The IEEE Computer Society Computing Curriculum - Software Engineering --- Public Draft 1 --- (July 17, 2003) Software Engineering Education Knowledge Software expands on the earlier IEEE 610 definition of a requirement as follows, "*Requirements identify the purpose of a system and the contexts in which it will be used. Requirements act as the bridge between the real world needs of users, customers and other stakeholders affected by the system and the capabilities and opportunities afforded by software and computing technologies. The construction of requirements includes an analysis of the feasibility of the desired system, elicitation and analysis of stakeholders' needs, the creation of a precise description of what the system should and should not do along with any constraints on its operation and implementation, and the validation of this description or specification by the stakeholders. These requirements must then be managed to consistently evolve with the resulting system during its lifetime*".

However, in practice, there is difficulty in adding these additional properties to the traditional requirement document or database and then managing them. This is because the current systems and software development paradigm generally divides the work in a project into three independent streams as shown in Figure 9.2. Thus requirements engineering tools contain information related to the Development and Test streams (the requirements) while the additional properties tend to be separated in several different tools, (e.g. Requirements Management, Project Management, Work Breakdown Structures, Configuration Control, and Cost Estimation, etc.).

As "requirements" are still poorly implemented after all these years, perhaps they should be eliminated or bypassed (automated). Kasser proposed and developed a prototype tool to improve the wording of requirements[6], but a greater degree of improvement should be achievable by replacing written requirements (Kasser, 2002d). Consider ways in which this might occur.

Gabb et al. define a requirement as, "*An expression of a perceived need that something be accomplished or realized*" (Gabb, et al., 2001). The focus should be on user needs, not on requirements. Van Gaasbeek and Martin quote Dahlberg as stating, "*We don't perform systems engineering to get requirements*" (Van Gaasbeek and Martin, 2001) and add, "*We perform systems engineering to get systems that meet specific needs and expectations*." What systems engineering appears to have forgotten is that requirements are used to document user needs in a verifiable manner, Requirements are a means, not an end. There is nothing divine about requirements; they are just a convenient poorly-used tool for translating customers' needs into a system that should be built.

Requirements are developed as an intermediate work product in the SDP, and are developed to provide formal communication between the stakeholders. Writing text-based unambiguous requirements for combinatorial and sequential scenarios in the form of imperative construct statements is difficult. Timing and state diagrams are often used within the context of the SRD to provide the necessary information. Thus the concept of stating user needs (under certain circumstances) via diagrams is already in use in systems engineering.

Sutcliff et al. proposed reducing human error in generating requirements by analysing requirements using an approach of creating scenarios as threads of behaviour through a Use Case, and adopting an object-oriented approach (Sutcliffe, et al., 1999). So, if Use Cases can represent the user's needs in a manner verifiable by all stakeholders, then an improvement on the current text-mode based requirements paradigm will have been made. The use of Use Cases driving an object-oriented ap-

---

[6] The tool, FRED, evolved into Tiger Pro, available at http://therightrequirement.com/TigerPro/TigerPro.html

proach describing properties of components can provide the same representation of user needs as that of "requirements" if each property consists of an attribute and a value.

Schach partitions requirements into functional and non-functional and adds the properties of traceability and priority (Schach, 2002: page 294).

12.9.1. The properties of an object-oriented requirement

Consider "property" as the totality of the attribute and its value. Then requirements can be stated as the properties needed, and capability can be stated as the properties measured or exhibited by the object. The words functional and non-functional requirements no longer have to be used. When the system is broken down into subsystems each property (attribute and value) is allocated to subsystem elements[7]. Traceability of properties (functional and non-functional) is built into the approach.

The major question is "what are the properties of an object-oriented requirement? The answer is not simple. Before attempting to identify the properties of object-oriented requirements, a set of rules were established based on the maxim that a good requirement has the following three characteristics:

1. ***It describes something about the physical system that will meet the needs of the customer.*** This is the traditional text-based sentence that covers the functional and non-functional aspects of the system being produced.
2. ***It facilitates (or rather not does not impede) the production process***. This characteristic is derived from Total Quality Management (TQM) which is defined by NASA as the application of systems engineering to the work environment) (NASA, 1992) and is concerned with the effectiveness of the production process. While requirements define a need, they can also be viewed from the contractual perspective. The cost of realising a system is based on the work and materials needed to transform needs into systems. Properties based on this characteristic include vagueness, understandability and ambiguity, namely properties that lead to cost escalations, schedule delays, or the provision of undesired functionality.
3. ***It is something the customer really wants.*** This is the most difficult characteristic since customers do not always to state the real requirement.

---

7 Desired emergent properties can be allocated to a virtual subsystem.

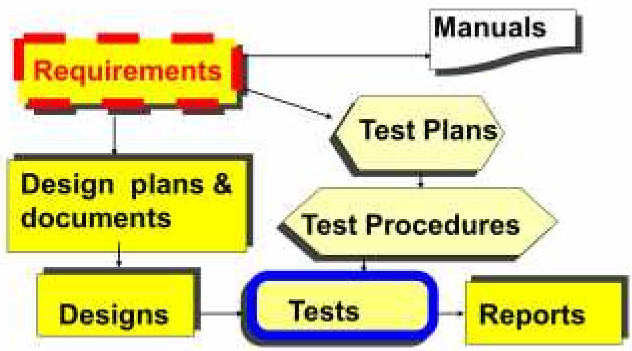


Figure 12.7 Requirements drive the work (Kasser, 1995)

The first avenue to be explored on the journey to identify the properties is the usage of requirements in the SDP and to explore how the object-oriented paradigm can improve the current situation.

12.9.2. Requirements drive the work

The requirements elicitation process produces a set of requirements, which represent the performance of a system that will meet the customer's needs when placed into service in the FCFDS. Consequently, every element of the work ought to be traceable (and chargeable) to a specific requirement or set of requirements. The work to realise a system specified by the requirement) takes place takes place in the three streams of activities (management, test and evaluation, and development) shown in Figure 9.2; hence, every requirement can be thought of as having properties driving the work in any of the three streams. This produces a view of a requirement statement as the tip of an iceberg, where the statement can be seen, but the underlying work to produce the capability that meets the requirement is hidden. An alternative view, a more traditional perspective in the form of an overview of the documentation tree, in which requirements drive the work to produce the various process-product documents, is presented in Figure 12.7.

Effectively, object-oriented requirements engineering and management not only performs the requirements engineering at the front end of the SDP, but also provides integrated information for the functions of project management, design, development, test and evaluation, and operations and maintenance as performed in the current paradigm. As in the current paradigm, the implementation work plan can be published in three documents, namely:

- ***The SRD:*** specifies the system to be realised in the development stream of activities.

- ***The SEMP:*** guides the management of the streams of activities.
- ***The TEMP:*** guides the test stream of activities.

These documents may be perceived as partial views of a requirements database in which each only contains the properties of the requirements appropriate to the stream of work pertaining to the document. Consider the contents of each of the documents. For each document, the following generic properties of the requirement apply.

- ***The unique identification number:*** to clearly identify the requirement.
- ***The text of the requirement statement.***
- ***Version number:*** identifying the version of the requirement.
- ***Date of acceptance***.

The other properties of the requirements are document specific as follows:

- ***The system requirements document*** (SRD) contains the documented solution of what has to be done to provide a remedy to the customer's undesirable situation. The SRD should contain the following properties of each requirement:
  - ***Traceability to source(s):*** where the requirement came from, i.e., the CONOPS, regulations, specific people, etc.
  - ***Rationale for requirement:*** to communicate the reason why the requirement was included in the first place. This information is important for considering change requests during the O&M State of the SLC. This information is sometimes included as comments in the current paradigm, but is not required.
  - ***Traceability sideways:*** to other documents (or databases) at the same level of decomposition of the system. This provides information when considering the impact of requested changes.
- ***The Test and Evaluation Master Plan*** (TEMP) guides the T&E process and should contain the following properties of each requirement:
  - ***Acceptance criteria:*** which are provided in response to the question "*How will we know that the requirement has been met by the system*?"
  - ***Planned verification methodology(s):*** demonstration, analysis, etc. Not all requirements can be tested. For example, how do you test a fuse?

  - ***Testing parameters:*** the sections of the test plans and procedures that verify the system meets the requirement.
  - ***Resources needed for the tests:*** people, equipment, time, etc.
- ***The Systems Engineering Management Plan*** (SEMP) contains the planned resources and schedule necessary to perform the design and testing activities. The SEMP should contain the following information for each requirement:
  - ***Traceability to implementation:*** identifies the Iteration or Build in which the requirement is scheduled to be implemented.
  - ***The priority*** of the requirement.
  - ***The estimated cost*** to construct and test the elements of the system that provided the functionality specified by the requirement.
  - ***The level of confidence in the cost estimate***.
  - ***Risks:*** implementation, programmatic, and any other identified. Risk mitigation approaches are an attribute of the risk.
  - ***Production parameters:*** the Work Package (WP) for the work to be done to meet the requirement.
  - ***Required resources*** for the work, when they will be required and for how long.

This information was presented in the form of a set of Quality Systems Elements (QSE) (Kasser, 2000) in an integrated Information Environment (IIE) concept (Cook, et al., 2001; Kasser, 2013b: pages 97-104) as being necessary for effective system and software development The QSE are not new. They are known and have been used independently in project management and systems engineering for many years. For example, MIL-STD 2167A prescribed a set of software development folders that shall include (directly or by reference) the following information (MIL-STD-2167A, 1998):

- Design considerations and constraints.
- Design documentation and data.
- Schedule and status information.
- Test requirements and responsibilities.
- Test cases, procedures, and results.

Some of the QSE have also been incorporated as fields in requirements management tools from time to time. However, these instances seem to be the exception rather than the rule. The QSE also do not seem to have been used together in an integrated manner. The object-oriented approach integrates them. Consider the QSE as the initial set of candidate properties of requirements and thus at least improve on the current paradigm by providing a place to store those additional properties in an IIE containing information about the process and product.

12.9.3. Other object–oriented properties

So far the QSE database has been populated from the content of the three documents. Software engineering articulated the object-oriented approach as a way of encapsulating data and processes in ways that were not tied to physical implementations. Consider the addition of other object-oriented properties such as:

- ***Non-functional elements*** of capability needed – survivability, reliability, maintainability, etc.
- ***Access control:*** to control access to the requirement or selected properties. This might be used in classified situations or in corporate situations where two companies share partial information.
- ***Version control:*** identifying the version of the database.
- ***Archived earlier versions:*** for tracing changes in case incompatibility issues show up in the later states of the SLC.

12.9.4. Populating the properties of the requirement

The process of accepting requirements may be represented as shown in Figure 12.8. The customer's need (source of the requirement) is represented as a request for capability (requirement request) and allocated a unique identification number. The requirement request is then assessed for priority, and cost and schedule impact on the SDP, as well as for risks and conflicts with existing requirements. The result of the impact assessment is presented to the customer who then decides to accept, reject, or modify the requirement request. However, some of these impact assessments are generally not performed in the current paradigm.

The entire set of properties cannot be populated at the same time. Population begins as shown in Figure 12.8. The initial set of properties of a requirement request submitted to the Configuration Change Board (CCB) consists of the:

- Requirement statement or representation.
- Source of the requirement (traceability).
- Key words.
- Rationale for the requirement.
- Acceptance criteria.
- Non-functional properties (e.g. reliability, maintainability and survivability).
- Priority of the need for the capability.

Figure 12.8 Populating the properties of the requirement

The requirement request is allocated a unique identification number. After performing the total impact assessment, the next set of properties is populated (estimated cost to implement, risk (implementation and programmatic and their mitigation) and the Iteration or Build in which the requirement will be implemented). The impact assessment is then negotiated with the customer who may accept, reject or modify the assessment. If the requirement request is accepted the properties are incorporated in the QSE database. If the requirement request rejected, the reason for the rejection is also documented in case the same request shows up at a later time. Later on in the SDP, the test properties are populated as the test team develops the test plans etc. When the Build Plan is developed the implementation properties are further populated, and so on.

12.9.5. The object-oriented approach

The objective properties viewer shown in Figure 12.9 lists the properties of a component[8]. Each property has an attribute, which has some value. In the example of a communications object, as far as performance attributes are concerned, the data input attribute has a value of 1000±10 units, the data output attribute a value of 1000±10 units, etc. The services (functions) performed by the object have to do with ingesting the data input, performing some action on the data and then forwarding processed data. A few of the non-functional reliability attributes such as Mean Time between Failures (MTBF), and Mean Time to Repair (MTTR) and operating temperatures are also shown.

[8] This is a typical Delphi display of the properties of a software component being used to display the properties of a subsystem.

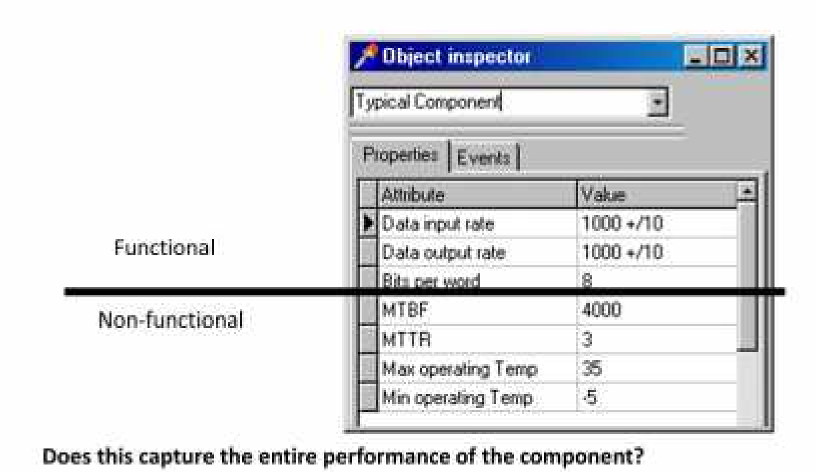


Figure 12.9 The object view

The use of Use Cases within a CONOPS in an object-oriented approach describing 'properties of', and 'services' (functions) provided by, components, can often provide the same representation of user needs as that of "requirements" if each property consists of an attribute and a value as discussed above. Other non-functional attributes such as colour and weight associated with a component can also be shown in the property viewer. The object-oriented approach also provides for inheritance of attributes of various classes of components which helps to maximise the completeness of the information in the CONOPS.

Systems engineering is all about ensuring that all the properties of the system as delivered (system capability) are at least equal to the properties of the system needed (system needed). Thus systems engineering can be requirements free.

12.10. Five reasons for the failures of systems engineering

The following reasons are hypothesized (there may be others not mentioned):

1. The use of the wrong type of systems engineer for the job discussed in Section 12.10.1
2. The use of the 'B' paradigm discussed in Section 12.10.2.
3. Managers who make decisions ignoring the recommendations of the systems engineer discussed in Section 12.10.3.
4. Problem solving is not taught very well discussed in Section 12.12.
5. Decisions made for political and other non-technical reasons lead to the failure but systems engineering gets the blame.

Table 12.2 Failure data from GAO report 06-368, 2006

Cost and Schedule Outcomes Sorted by Percent of Product Development Remaining

Programs	Percent cost growth*	Schedule growth, in months	Percent of development remaining
Aerial Common Sensor	45%	24	85%
Future Combat System	48%	48	78%
Joint Strike Fighter	30%	23	60%
Expeditionary Fighting Vehicle	61%	48	49%
C-130 Avionics Modernization Program	122%	Delays anticipated	Undetermined
Global Hawk (RQ-4B)	166%	Delays anticipated	Undetermined

Sources: DOD (data); GAO (analysis and presentation).

*Cost growth is expressed as the percent change in program development cost estimates in 2005 base year dollars.

12.10.1. The use of the wrong type of systems engineer for the job

The approach to characterizing systems engineers into five types discussed in Section 10.9 provides a hypothesis for a reason for the failure of systems engineering in the early stages of large projects (Hiremath, 2008) and other examples of poor systems engineering implementation (GAO, 2006). Namely the early stage systems engineering failed because the lead systems engineers were not Type Vs. For example, the cost and schedule overruns in the Joint Strike fighter (JSF) development project shown in Table 12.2 were predicted (Kasser, 2001) and hence some were probably preventable. Had Type V systems engineers been working on the states of the JSF project in the Needs Identification State of the SLC, the factors identified as potential causes of cost and schedule overruns leading to the would have probably been identified as risks. Appropriate risk management techniques would then have been recommended and if these risk management techniques had been implemented[9], the ensuring cost and schedule overruns would have been reduced.

Based on a combination of the five types of systems engineers and the history of systems engineering paraphrased in terms of those five types, the hypothesis is that a current cause of failures in systems engineering is the assignment of Type II systems engineers or higher types trained in a Type II process thinking paradigm to tasks that need the problem/solution characteristics of the Type III, Type IV and Type V systems engineers. The associated prediction to test the hypothesis is that the cost and schedule overruns and other failures will continue in spite of all the funding being allocated to systems engineering education if the education of systems engineers remains in the Type II and 'B' paradigms and continue to start with the activities in the Requirements State of the SDP.

[9] A big "if" since political considerations in the Type I I process paradigm would probably have precluded the risk mitigation activities.

12.10.2. The use of the 'B' paradigm

The 'B' paradigm is inherently flawed as discussed in Section 12.8. As such any project using the 'B' paradigm has a high probability of failing to deliver a solution system that remedies the undesirable situation.

12.10.3. Managers who make decisions ignoring the recommendations of the systems engineer

This situation was recognized almost at the dawn of systems engineering by Goode and Machol who wrote, "*The most difficult obstacle that may be encountered by an [systems] engineer is not the problem but a management which is unsympathetic or lacking in understanding*" (Goode and Machol, 1959: page 513).

The optimal management method is said to be Management by Walking Around (MBWA) (Peters and Austin, 1985). Yet Deming wrote, *"MBWA is hardly ever effective. The reason is that someone in management, walking around, has little idea about what questions to ask, and usually does not pause long enough at any spot to get the right answer"* (Deming, 1986: page 22). And the situation continues into the 21st century as satirized by Scott Adams in his Dilbert cartoons. Think of the cost of the waste and the work expended to implement and then correct the results of poor decisions.

12.11. Reasons for the success of systems engineering

Two reasons for the successes of systems engineering are:

1. The projects followed the 'A' paradigm.
2. Projects succeeded when the Chief Systems Engineer was a Type V (Section 10.9) and management did not interfere with the technical side of the operation.

Honour's research findings showed, "*Systems engineering activities correlate strongly to program success measures, but do not correlate strongly to the technical quality of the resulting system*" (Section 10.3.2). One might speculate that the observed lack of correlation to the technical quality of the resulting system may have been caused by:

- A variable that was not considered in the research such as the type of Chief Systems Engineer on the project; an example of Simpson's paradox (Savage, 2009).
- Most of the systems did not perform the early states systems engineering act ivies, i.e., they used the 'B' paradigm.

12.12. Problem-solving is not taught very well

One of the reasons systems engineering has failed to deal with complexity may be that the problem-solving process as it is generally taught and used does not equip systems engineers with the necessary skills to identify and solve complex problems for reasons that include:

1. The use of the shortened problem-solving process discussed in Section 6.1.1.
2. Not understanding the time delays in realizing solutions discussed in Section 6.1.2.
3. The structure of the problem discussed in Section 7.6.
4. The multiple meanings of the word problem discussed in Section 9.3.
5. The focus on a single correct solution instead of realising that there may be more than one acceptable solution to a problem as discussed in Section 9.4.
6. The levels of difficulty of the problem discussed in Section 10.12.
7. The way of dealing with Wicked problems discussed in Section 12.12.1.

12.12.1. Dealing with Wicked problems

From the *Generic* perspective, Wicked situations may manifest themselves in the first step of the Scientific Method problem solving process even if nobody is consciously using the Scientific Method to address the problem. That is, the current situation is under observation, but a working hypothesis to explain the causes of the observations (desirable and undesirable) has yet to be developed. For example, the state of the art of chemistry before the development of the periodic table of the elements by Mendeleev could have been considered as a Wicked situation, as could the state of electrical engineering before the development of Ohm's Law. As such, the way to deal with a Wicked problem is to use the Scientific Method to convert the Wicked problem to a well-structured problem, or a set of well-structured problems, remedy the well-structured problems and repeat the observations as shown in Figure 6.1.

12.13. The reason for the different descriptions of the problem-solving process

The reason for the different descriptions of the problem-solving process discussed in Section 8.1 is because each problem-solving approach documented a version of the problem-solving process that was done to tackle a specific problem at some point of time by some people; as such there is no reason for each approach to cover the entire process. From the *Generic* perspective it is the same reason as the reason for the different deceptions of the SEP discussed in Section 9.17.2.

12.14. Changing the SDP from a single waterfall to a series of waterfalls at project planning time

The process camp focus on a single pass through the SDP as a result of teaching systems engineering using the waterfall and V views since, while not representative of the real world, they are simple to explain (Biemer and Sage, 2009: pages 152-153). The traditional SDP is based on a single pass through the waterfall. Given that most complex systems development applies to Shenhar and Bonen's type B, C and D systems (Section 9.16), the traditional project timeline needs to be changed from a single pass through the waterfall to at least two passes for Type B projects and three or more for Type C and D projects in accordance with Shenhar and Bonen's recommendations based on the different levels of technological uncertainty.

In addition, all passes through the waterfall after the first shall take in account changes in user needs during the development time in the manner of the PRINCE 2 project management methodology (Bentley, 1997).

12.15. Consider the SLC as a State Machine

The traditional view of the SLC is that it passes through a number of phases in a sequential manner, often represented by the waterfall. Each phase starts and ends at a formal milestone. However, the SLC can also be treated as a State Machine (Wymore, 1993). The SLC State Machine perspective is similar but the State Machine model facilitates understanding the effect of changes during the SLC. Chapter 13 perceives the SLC from the following perspectives:

1. The SLC as a State Machine.
2. The SDP.
3. The relationship between the SDP and the O&M State.
4. The operate, modify and upgrade cycle in the O&M State.
5. The SLC as a multi-phased time-ordered parallel-processing recursive paradigm.

12.16. The two interdependent sequential systems engineering processes

There seem to be two interdependent sequential SEPs:

- ***The traditional 'doing' SEP:*** in which HKMF Layer 2 systems engineering is performed. This is the unique SEP which is constructed for the realization of a specific system. This process is identified as S5 in the Nine-System Model (Section 16.5). The activities performed in the unique SEP will depend on the problem-identification-solution -realization activities that have and have not been done up to the time the unique SEP is constructed.

- ***The planning SEP:*** the process followed by the systems engineer to create the unique SEP (Biemer and Sage, 2009: page 153). This process is identified as S4 in the Nine-System Model (Section 16.4). When designing/planning the unique SEP for the realization of a system, systems engineers use implementation domain knowledge based on experience and the activities functions and processes which can be found in the processes, Standards and the literature.

12.17. Systems engineering is a discipline

Perceptions from the *Structural* perspective discussed in Section 7.1 show that systems engineering is indeed a discipline. The question is what kind of a discipline? Perceptions from the *Continuum* perspective indicate that systems engineering could be:

- ***A meta-discipline:*** The SETR proponents in the meta-discipline camp (Section 9.17.4) consider systems engineering to be a meta-discipline.
- ***An enabling discipline:*** The SETA proponents in the enabler camp (Section 9.17.8) consider systems engineering to be an enabling discipline.

12.18. Systems engineering is demonstrating the symptoms of a discipline in its early stages

Systems engineering is demonstrating the symptoms of a discipline in its early stages. Disciplines in their early stages are characterised by several factors including:

- Researchers and practitioners viewing the discipline from different single viewpoints and drawing different conclusions about the nature of the discipline, which results in,
  - Different camps; the different camps in systems engineering (Section 9.17).
  - Simultaneous discoveries or inventions of the same or very similar concepts, documented using different terminology leading to confusion and complexity. For example, many systems engineers do not realise that the waterfall, V and spiral models are different views of the same sequential series of activities (process), each view originally being used for a different purpose as partially shown in Figure 12.1.
  - Debates with parties speaking but not listening.
  - The development of a toolbox of processes, that when followed, produce expected results some of the time.

- The lack of fundamental frameworks and underpinning theories.

12.19. Resolving the paradoxes

The paradoxes identified in Section 9.24 can be resolved or dissolved as follows:

1. The process paradox discussed in Section 12.19.1.
2. The roles paradox discussed in Section 12.19.2.
3. The emergent properties paradox discussed in Section 12.19.3.
4. The tools paradox discussed in Section 12.19.4.
5. The system optimization paradox discussed in Sections 12.26.3 and 18.7.
6. The reductionist paradox may be dissolved by the use of the Nine-System Model discussed in Chapter 16.

12.19.1. The process paradox

The different representations of the problem-solving and SEPs are all different aggregations of a set of activities which constitute the extended holistic problem-solving process; some representations being incomplete. The reason for the different descriptions of the problem-solving process discussed in Section 8.1 is because each problem-solving approach documented something that was done to tackle a specific problem at some point of time by some people; there is no reason for each approach to cover the entire process.

12.19.2. The roles paradox

Perceptions from the *Temporal* perspective discussed in Section 11.1 turned up information as to how the paradox originated. Thus systems engineering, project management and Operations Research can be seen as three solutions to the problems posed by complex systems in the Cold War by three different communities of practice (Johnson, 1997) that have continued to evolve by performing activities there were not performed in specific situations and the roles then overlapped. For example:

- Lewis provides Case Studies in software IV&V (Lewis, 1992). Yet the words "IV&V engineers" in those Case Studies could be replaced by the words "systems engineers" and the cases would be just as appropriate in a book on systems engineering instead of a book on IV&V.
- There are two types of configuration audits within Configuration Management (CM) (MIL-HDBK-61A, 2001) which overlap systems engineering activities as discussed in Section 9.20.

12.19.3. The emergent properties paradox

The emergent properties paradox identified in Section 9.24 can be dissolved by realising that each side of the dispute is referring to a different subset of emergent properties, namely predictable and unpredictable:

- If someone else has already connected the set of components in the same way, under the same conditions, then the emergent behaviour can be predicted as being the same as previously observed[10].
- The first time a set of components (a system) is connected together:
  - The total amount of emergent behaviour cannot be predicted.
  - Some emergent behaviour can be inferred using perceptions from the *Generic* perspective, namely the emergent behaviour should be similar to that of an existing similar system.

12.19.4. The tools paradox

Relating the descriptions of the tools to the 'A' and 'B' paradigms dissolves the tools paradox identified in Section 7.3. The tools of the 1960's were mainly used in the Needs Identification State of the SDP in the 'A' paradigm, while the tools of 2000's are used in the remaining states of the SDP in the 'B' paradigm.

12.20. The two processes for creating a system

The two processes for creating a system found in the literature were discussed in Section 9.23. Unfortunately, while O'Connor and McDermott's guidelines are interesting and useful, they can lead to unnecessary complexity and errors in the creation of the system. For example in O'Connor and McDermott's set of guidelines (O'Connor and McDermott, 1997: page 166):

- "*1. Draw from your experience and viewpoint*" can lead to the Not Invented Here (NIH) syndrome because it:
  - Ignores the wealth of experience offered by others.
  - Has a tendency to suffer from the 'mine is better' habit that hinders thinking (Ruggiero, 2012: pages 54 to 61).
- "*7. Only include elements that can change when influenced by another element*" ignores elements that influence the system but do not change. For example a closed systems view of the pendulum clock ignores the effect of gravity because while it is there, it re-

[10] This is one of the principles underpinning the Scientific Method.

mains constant in a specific location, and may be ignored. However, if the clock is moved into a different gravitational field, the mass on the end of the pendulum will need to be adjusted or replaced to compensate. As a possible second example, the Lunar Surface Gravimeter experiment flown to the moon in the Apollo 17 mission did not perform as expected on the moon (Giganti, et al., 1971) and may have suffered from the lack of compensation for the difference between terrestrial and lunar gravity.

Chapter 18 introduces some better rules for creating systems in a manner to manage complexity together with some examples of optimizing the system at design time.

12.21. Model Based Systems Engineering

In the last few years, the process camp (Section 9.17.2) has produced MBSE by applying 21st century technology in their 20th century SEP paradigm. Inferences from the *Scientific* perspective include that in its current form:

1. MBSE conflates two distinct and different models discussed in Section 12.21.1.
2. MBSE is a poor choice of terminology for the concepts it contains discussed in Section 12.21.2.
3. MBSE suffers from a lack of holistic thinking discussed in Section 12.21.3.
4. MBSE is a return to the 'A' paradigm discussed in Section 12.21.4.
5. MBSE is reinventing old concepts discussed in Section 12.21.4.1.

12.21.1. MBSE conflates two distinct and different models

MBSE conflates two distinct and very different information models, namely:

1. ***A conceptual model*** in the form of a vision of a FCFDS in operation discussed in Section 12.21.1.1.
2. ***The integrated interdependent information*** pertaining to both the process (project management) and product (systems and non-systems engineering) in the project that realizes the solution system which currently exists in the form of unconnected un-integrated databases and documents discussed in Section 12.21.1.2.

12.21.1.1. A conceptual model in the form of a vision of a FCDS in operation

The 'A' paradigm in systems engineering discussed in Section 9.21.1 has been using these types of models in prototype, document and simulations for at least 50 years. Indeed, the success of systems engineering in the NASA environment in the 1960's and 1970's was attributed to a set of eight principles (Hitchins, 2007: page 85) which were updated for the 21st century (Kasser and Hitchins, 2011; Kasser, 2013c: pages 427-437; Chapter 15) one of which is:

"*There shall be a concept of operations (CONOPS) from start to finish of the mission describing the normal and contingency mission functions as well as the normal and contingency support functions performed by the solution system that remedies the problem*".

Now that second generation CESE tools provide ways to create a CONOPS in the form of interactive simulations and executable models, MBSE seems to be restating that capability using application language as in the definitions by Friedenthal et al. (Friedenthal, et al., 2007) and INCOSE's Vision 2020 (INCOSE, 2007) included in Section 9.6. When the benefits of MBSE are summarised as in the following extract (Long, 2013), the author is really summarizing the benefits of the 'A' paradigm.

- *"Early identification of requirements issues.*
- *Missing requirements, conflicting requirements, and general defects.*
- *Enhanced stakeholder communication to enable better validation.*
- *'We fail more often because we solve the wrong problem than because we get the wrong solution to the right problem' (Ackoff)*[11]*.*
- *Disciplined (and defensible) basis for decision making.*
- *Moving beyond "a miracle occurs here" analysis.*
- *Enhanced visibility into information gaps and system design integrity.*
- *Model-driven consistency vs. document-driven hope.*
- *Improved specification of allocated requirements to Hardware/Software.*
- *Reduction in errors reaching integration and test.*
- *Rigorous traceability from need through solution.*
- *Improved alignment of collective team understanding.*
- *One high-visibility version of truth.*
- *Reduction of rework.*
- *Improved communication & insight.*
- *Improved impact analysis of requirements changes.*
- *Knowing when you are done!"*

12.21.1.2. The integrated interdependent information pertaining to a project

The concept in the MBSE information model is to replace the 20th century independent document-centric paradigm by a paradigm in which in-

[11] (Ackoff, 1974)

formation is stored electronically in interdependent databases where documents are views or printouts of the contents of the databases. This is a desired characteristic of the DODAF (DoDAF, 2004) as but one example. A better term for this model, in functional language, might be an Integrated Information Environment (IIE) for the repository of project information (Cook, et al., 2001; Kasser, 2013b: pages 97-104) as discussed in Section 12.21.4.1.

12.21.2. MBSE is a poor choice of terminology for the concepts it contains

MBSE is more than just developing and using models, for example, it is also an effort to address the following process issues (Shoshani, 2010):

- ***Communication and understandability:*** well-structured models improve the ability to convey meaning to different stakeholders.
- ***Traceability:*** linked and repository-based models allow for better traceability and consistent models.
- ***Early knowledge:*** early executable models allow eliciting knowledge earlier in the SDP.
- ***Reduced Time To Market (TTM):*** model based analysis and design takes less time than the textual process. Models that create software deliveries automatically, improve TTM.
- ***Reuse:*** well-structured model parts can be reused in product lines or component based development, again shortening the development cycle and cost.
- ***Formal proofs:*** models can be validated early and fully, models that are turned into software code are considered proven by construct. This is useful where system high reliability is required.
- ***Maintenance:*** a model of the system captures all the data needed for change thus allowing for easier maintenance.

The term MBSE is in application language. Application language focuses on the instance described by the application such as stating the need for a 'car' where the need is for the 'transportation' function[12]. The use of application language is limiting since it focuses on the instance rather than the function as well as being open to multiple interpretations in this situation as discussed above.

[12] Another example of application language is the use of 'network centric' when the function is information distribution and the application is via networks.

12.21.3. MBSE suffers from a lack of holistic thinking

MBSE suffers from a lack of holistic thinking[13] in the following ways:

1. MBSE is the future of systems engineering discussed in Section 12.21.3.1.
2. The focus on a single modelling language engineering discussed in Section 12.21.3.2.
3. The focus on requirements engineering discussed in Section 12.21.3.3.
4. MBSE is constrained by the 'B' paradigm engineering discussed in Section 12.21.3.4.

12.21.3.1. MBSE is the future of systems engineering

MBSE proponents claim that MBSE is the future of systems engineering. However, perceptions from the *Continuum* and *Generic* perspectives show that there is generally more than one way to perform a function, so from the *Generic* perspective, personnel who claim that MBSE is the way or is the future of systems engineering are members of the non-systems thinking camp (Section 9.17.5) who claim their tool will fit all current and future situations. This behaviour matches Maslow's observation of non-systems thinking human behaviour which was, "*I suppose it is tempting, if the only tool you have is a hammer, to treat everything as if it were a nail*" (Maslow, 1966: pages 15-16).

12.21.3.2. The focus on a single modelling language

Holistic thinking and its application to systems engineering is about choices.

- Perceptions from the *Operational* perspective suggest that the stakeholders should see the information in the model in a way that makes sense to them, rather than have to learn a modelling language.
- Perceptions from the *Structural* perspective point out that since the UML was designed to be extendable there was no need to develop Systems Modeling Language (SysML). All that was needed to be developed were extensions to UML, which had already been done (Holt, 2001) as discussed in Section 8.2.
- Perceptions from the *Continuum* HTP show that:
  - There are other modelling languages and other ways of performing the same function such as Object-Process Methodology (OPM) (Grobshtein and Dori, 2010).

[13] As summarized in Chapter 2

- BPR uses models in describing the 'as-is' and 'to-be' models.
- There are ways to communicate the FCFDS in the customer's language using videos, pictures and other non-language methods as demonstrated in the Operations Concept Harbinger (OCH) which may be thought of as a multimedia combined CONOPS and system requirements repository that also contains Measures of Effectiveness (MOE) for each operational scenario which was developed, prototyped and demonstrated for a Force Level Systems Engineering (FLSE) application in the SEEC at the UniSA (Kasser, et al., 2002).
- While it generally is possible to use a single language to program all applications all tasks, different languages are better suited for specific applications. For example, in the early days of computing FORTRAN (FORmula TRANslation) was a language used for programming mathematical tasks, while COBOL (Common Business Oriented Language) was used to program business tasks.

- Perceptions from the *Temporal* perspective:
  - Indicate that new languages and methods will arise in the course of time.
  - Provide the lesson not learned that attempts to standardize on a single computer language have failed in the past; ADA being but one example.

12.21.3.3. The focus on requirements

Requirements are a means, not an end. The focus on requirements as an end in itself rather than as a means to an end stems from the roots of MBSE in the 'B' paradigm (Section 9.21.2). If, for example, the performance of a system can be documented in an executable model of a CONOPS as in the 'A' paradigm, there is no need for most of the requirements that exist in the 'B' paradigm as discussed in Section 12.9.

Perceptions from the *Functional* perspective of the 'A' paradigm show that the information in the CONOPS model is translated to requirements which are then passed on to the hardware and software design teams. The software design teams then produce Use Cases based on the requirements. Since the CONOPS model contains the Use Cases, there is no need to develop most of the functional and performance requirements; the functions can be tagged with performance properties as discussed in Section 12.9. Just allow the software designers to access the tagged CONOPS model and eliminate one of the major contributors to project failure, namely poor requirements.

12.21.3.4. MBSE is constrained by the 'B' paradigm

MBSE seems to be focused on improving the 20th century process focused 'B' paradigm (Section 9.21.2) rather than applying the technology to upgrade systems engineering to an improved 21st century 'A' paradigm.

12.21.4. MBSE is a return to the 'A' paradigm

Perceptions from the *Generic* perspective include:

1. MBSE is reinventing old concepts discussed in Section 12.21.4.1.

12.21.4.1. MBSE is reinventing old concepts

MBSE is reinventing concepts which may be new to the process focused practitioners of the 'B' paradigm but are well known, and have been in use, outside their box. This situation may be an example of the inadvertent use of a flawed approach to problem solving in which the research activities shown in Figure 6.1 have been omitted resulting in a flawed hypothesis or solution. This lack of the research step often results in the 'Not Invented Here' (NIH) syndrome. The flawed process is often inadvertent because the experts in one domain are used to producing solutions without performing the research step. Consequently, when faced with a problem for which they have no immediate solution, they forget to pose the question "who has faced a similar problem?" and perform the research step to see if indeed anyone else has faced and remedied a similar problem. Thus, MBSE is rediscovering concepts that have been explored and published in prior years both within and outside of systems engineering. These concepts include:

1. The models used in Operations Research in HKMF Layer 3; specifically area 3G, discussed in Section 12.21.4.1.1.
2. The use of models and simulations discussed in Section 12.21.4.1.2.
3. Use of interdependent databases rather than independent documents discussed in Section 12.21.4.1.3.
4. The concept of an electronic executable model discussed in Section 12.21.4.1.4.

12.21.4.1.1. *Operations Research*

The INCOSE definition of MBSE is, "*Model-based systems engineering (MBSE) is the formalized application of modeling to support system requirements, design, analysis, verification and validation activities beginning in the conceptual design phase and continuing throughout development and later life cycle phases*" (INCOSE, 2007). Use perceptions from the *Generic* perspective to compare this definition with some definitions of Operations Research:

- *"Operations Research is a scientific method of providing executive departments with a quantitative basis for decisions regarding the operations under their control*" (Morse and Kimball, 1951).
- "*Operations Research is concerned with the heart of this control problem – how to make sure that the whole systems works with maximum effectiveness and least cost*" (Johnson, 1954: page xi).
- "*Operations Research is the application of scientific and especially mathematical methods to the study and analysis of problems involving complex systems* (Webster, 2013).

Operations Research is most often used to analyse complex real-world systems in HKMF area 3G, typically with the goal of improving or optimizing performance. The overlap between Operations Research and systems engineering can be seen as early as 1954 in Johnson's quoted definition above. It is a goal that many modern systems engineers would apply to systems engineering. Goode and Machol wrote that the steps of the Operations Research process and the SEPs have much in common however there is a fundamental difference in approach namely, "*the operations analyst is primarily interested in making procedural changes, while the systems engineer is primarily interested in making equipment changes"* (Goode and Machol, 1959: page 130). Roy noted a lasting difference as, "*Operations Research is more likely to be concerned with systems in being than with operations in prospect*" (Roy, 1960: page 22).

12.21.4.1.2. *The use of models and simulations*

The use of models and simulations pertaining to the FCFDS is an old concept as discussed in Section 11.4.

12.21.4.1.3. *Use of interdependent databases rather than independent documents*

One MBSE concept seems to be an IIE with a number of independent and interdependent software agents, where each agent acts on the same underlying data at different stages in the SDP. Kasser and Cook discussed a proposed architecture for a Frame-Based third generation Requirements Tool (FBRET) embodying a superset of current MBSE concepts (Cook, et al., 2001; Kasser, 2013b: pages 97-104) including automated approaches to requirements engineering providing features such as reasoning, structured knowledge capture, generic design solutions and elicitation assistance which were then, and still are, becoming increasingly important as the systems and the environments that they interact with, become increasingly complex.

12.21.4.1.4. *The concept of an electronic executable model*

The concept of an electronic executable model is nothing new. As but one example, a prototype of an OCH was developed, prototyped and demonstrated by the SEEC at the UniSA (Kasser, et al., 2002). In early 2002, the Joint Systems Branch (JSB) of the Australian Defence Science and Technology Organisation (DSTO) was carrying out a number of initiatives in architecture and systems engineering for joint C4ISREW systems analysis, future capability studies and force development. The teams working in these initiatives had produced many interesting and important findings, useful data and references for supporting force level defence capability planning and management, and had gained good experience in using various tools for architecture development and analysis. JSB was then faced with the problem of how to bring these outcomes together and indicate the feasibility of FLSE across the strategic and capability disciplines in a relatively simple demonstration. The dimensions of the problem included:

- Process and process interactions.
- Tools required at each level.
- Communications and information flows.
- Data, change and Configuration Management.
- Organizational cultural imperatives.

The solution developed at SEEC was to present the outcomes in the form of a prototype *F*orce-*L*evel *A*ustralian Defence Force *S*ystems *H*arbinger (FLASH) based on the OCH concept. The FLASH was not based on any specific language; rather it was based on the concept of storing information in an underlying database and providing stakeholders with various views in their own languages.

12.21.5. Conclusions about MBSE

In conclusion:

- MBSE is an instance in HKMF Layer 2 (Section 9.10) of class Operations Research – in object language.
- MBSE seems to be much ado about nothing new.

12.22. Previous ways of dealing with complexity in the INCOSE literature

Perceptions from the *Continuum* perspective identified three models for managing complexity in the systems development context found in the INCOSE literature as discussed in Sections 9.28[14]. When the models are compared in Table 12.3 it can be seen that:

- Each model is a different set of systems.

[14] Other approaches are recognized but documented in this book

Table 12.3 Comparison of the three models

Systems addressed by the models	**Seven Samurai**	**Whole System Model**	**Systems Project**
Existing "as-is" **situation**	Context (S1)	-	-
Existing **system** in "as-is" situation	-	-	E-system
Process to develop conceptual solution system	-	-	-
Conceptual solution **system** at time development begins	Intervention (S2)	-	R-System
Process to plan transition from existing situation to situation in which the solution system will be deployed	Implied in Realization (S3)	Implied in Production	Implied in P-System
Process to realize solution system	Realization (S3)	Production	P-System
Resources to be applied to realize the solution system	Realization (S3)	Development	P-System
Solution **system** at and after time of deployment	Deployed (S4)	Operational	R-System
New situation after solution system has been deployed	-	-	-
Adjacent **systems** operating in association with the solution system at and after time of deployment	Collaborating (S5)	-	-
System or systems that keeps the solution system operational at and after deployment	Sustainment (S6)	Support	M-System
Process to determine situation after deployment of solution system contains no undesirable elements	Realization (S3)	Production	P-System
Alternative solution systems	Competing (S7)	-	-
Enterprise and environment	Realization (S3)	Containing	-

- Each system may be an organization, situation, process or technological system.
- Each model is incomplete since other models may contain systems that the model does not.
- Each model contains perceptions from the *Temporal* perspective (considers the time to realize the solution system) but in different ways.
- The situation after the solution system has been deployed is not considered in any of the three models, although Martin does refer to it as a *modified context system* (S1').

Chapter 16 introduces the Nine-Systems Model, a more complete model that covers all the systems and more.

12.23. The need to focus on people as well as process

There is a need to focus on people as well as process. Many systems engineers perceive systems engineering as a process and there is a major focus on process standards but the contribution of effective people and the difference they can make is generally overlooked.

Henry Ford wrote, "*the best results can and will be brought about by individual initiative and ingenuity – by intelligent individual leadership*" (Ford and Crowther, 1922). The contribution of good people in an organisation is discussed in Section 7.5.

Bungay in summarising the people in the Battle of Britain discusses the differences between Air Vice-Marshalls Keith Park and Trafford Leigh-Mallory who commanded different Fighter Groups. Bungay then continues, "*What Park achieved in the Battle of Britain is in itself enough to place him amongst the great commanders of history. But his performance in 1940 was not a one-off. In 1942 in Malta, Park took the offensive and turned Kesselring's defeat into a rout. After that, he directed the air operations that enabled Slim to expel the Japanese from Burma. He was as adept at offence as he was at defence, and, like Wellington, he never lost a battle. His record makes him today, without rival, the greatest fighter commander in the short history of air warfare*" (Bungay, 2000: page 383). In 1940 Park and Leigh-Mallory had the same processes based on (RAF tactics and doctrine), yet it was not the superiority of the RAF process to that of the Luftwaffe that made the difference[1], it was the person who made the difference[2]. One was an administrator, the other a leader!

---

1 Bungay points out, that the RAF tactics for fighter formations were inferior to that of the Luftwaffe and cost the lives of many pilots until the survivors learnt to ignore the RAF tactics.

2 As another example, consider the service at your favourite restaurant. Do all table staff provide the same level of service, or are some better than others?

The literature is full of advice as to how to make projects succeed; typical examples are (Rodgers, et al., 1993; Peters and Waterman, 1982; Peters and Austin, 1985; Peters, 1987; Harrington, 1995) which in general tend to ignore process and focus on people. Systems engineers focus on developing processes for organisations – 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). Excellence is in the person not the process. This was recognised as early by Hall's specifications or traits for an "Ideal Systems Engineer" (Section 7.5). In the intervening years, process standards such as ISO 9000 and the various CMMs have proliferated. Yet the standards do not provide metrics that can predict the failure of a project.

12.24. Systems engineering is more than just applying process standards

Systems engineering is more than just applying process standards. Process standards document observed activities that have led to successes. The standards need to be tailored to suit the specific project in the specific organisation at the specific time. Instead of blindly following the process, systems engineers need to know when to go by the book and when to write the book. The literature on excellence has little if anything to say about complying with processes (Peters and Waterman, 1982; Peters and Austin, 1985; Rodgers, et al., 1993). The literature discusses the need for knowledgeable people to get things done. An improvement will usually be observed when going from chaos to order. That takes an organisation to CMM (Section 10.7) Level 5 or to ISO 9000 compliance, but what then? Standing at the bottom of the process improvement mountain you only see the foothills leading to the plateau at Level 5 as shown in Figure 12.10. Level 1 is categorised by having success achieved by heroes. Levels 2-5 discourage heroes and focus on orderly processes. However, it will take heroes working within the organised organisation structure to effect further improvements beyond Level 5 and improve the competitive edge. Companies don't want employees who can follow rules; they want people who can make the rules (Hammer and Champy, 1993: page 70). Winning (world-class) organisations need to focus on individual excellence and reward individuals for their achievements and the risks that they are willing to take (Harrington, 2000).

12.25. Ways of assessing competency in system engineering

Perceptions of nine ways of assessing competency in systems engineering were stored by the:

- *Continuum* perspective in Section 9.29.
- *Quantitative* perspective in Section 10.8.

Figure 12.10 The process improvement mountain

The authors of each of the competency models found in the INCOSE literature drew different SOIs from different perspectives to develop their models.

The systems addressed by the models, used as the basis for comparison, shown in Table 12.3 are derived from the extended holistic problem-solving process shown Figure 6.5. They are not derived from the systems engineering standards because the standards only document the activities performed by different groups at different times in different locations. As such there is no way to ensure that the set of systems is complete using a standards-based reference.

Some comments and conclusions from examining the description of several of the following competency models are:

- Current approaches for constructing and using competency models are based on observations of what systems engineers do in specific organisations. However, basing a measurement on what is being observed, does not guarantee that the systems engineers are doing everything they should be doing; namely the approach has tendency to suffer from errors of omission (Ackoff and Addison, 2006). That is, the approach cannot provide any information as to whether something that systems engineers should be doing is not being done. As such, there is no way to verify if indeed the modules are fit for purpose (Kasser, et al., 2013).
- ***The Capacity for Engineering Systems Thinking***. CEST may be useful for assessing some aspects of the competency of systems engineers. However, at this time, CEST is still in its research stages.
- ***Knowledge, Skills, and Abilities***. KSAs might be an im-

provement over résumés written as job descriptions citing years of experience that state nothing about the achievements of the person. Moreover, being descriptive, KSAs do not seem to be generally suitable for assessing the difference between a Type II person (Section 10.9) who does not understand the underlying fundamentals and just follows a process to reach a successful conclusion and a Type V person who understands what needs to be done and can create and implement a process to do it successfully. Lastly, while KSAs can provide a multi-level assessment of the proficiency of a systems engineer, there is no standard definition for any such levels.

- ***INCOSE CSEP Exam.*** The INCOSE handbook focuses on processes according to ISO/IEC 15288 and only addresses a limited body of declarative and procedural knowledge (Section 9.5) and does not address the cognitive skills and the individual traits in an objective manner. These skills and traits are addressed in a subjective manner in the follow up evaluation of the career experience of the candidate. The CSEP examination may be considered as being a minimal measurement of systems engineering competency.
- ***INCOSE UK Systems Engineering Competencies Framework***. While the SECF is a worthwhile effort, there seem to be a number of inconsistencies in the document including:
  - The four levels of proficiency are not in the same dimension: while the last three levels are attributable to increasing levels of proficiency of systems engineers, the 'awareness' level is applicable to people who work with systems engineers at high levels in an organization and as such there is an assumption that these people should have some knowledge of systems engineering.
  - The allocation of knowledge to the systems thinking competency theme does not match the way the term cognitive skills is used in the systems thinking and critical thinking literature. This is a potential cause of confusion.
  - While lists of abilities within the competencies make it easy to assess compliance by checking off experience against the items on the list, the method has the same intrinsic defect as the use of KSAs. Namely, it does not seem to be generally suitable for assessing the difference between a Type II person who does not understand the underlying fundamentals and just follows a process to reach a successful conclusion and a Type V person who understands what needs to be

done and can create and implement a process to do it successfully.

The SECF does however provide a way of setting the systems engineering role proficiency requirements for jobs in a process-oriented work environment, namely meets the one of the purposes for competency models produced by human resource professionals. Nevertheless, it should be used with care for assessing the competencies of individuals due to:

- Its lack of an objective way of assessing cognitive skills and individual traits.
- Its being prone to errors of omission since it is based on the observed role of a systems engineer in a number of UK organisations; namely the knowledge that systems engineers in the UK have, rather than instead of the knowledge systems engineers need to have.

▪ ***The NDIA proposed systems engineering competency model.*** The planned approach was to develop the competencies based on the roles of systems engineers. Two years later, the model was still a work in progress (NDIA E&T, 2010). For example, the first of the proposed 2011 tasks was to survey existing, freely available systems engineering competency models for entry-level systems engineers to develop the minimum requirements for an individual to be called a systems engineer. Reasons for this lack of progress may include:

- The difficulty of defining a SETR-SEBOK due to the broad range of non-systems engineering activities performed by systems engineers in their role in the workplace that require knowledge from other disciplines (Section 12.2).
- The different opinions on the nature of systems engineering from the different camps discussed in Section 9.17 that preclude obtaining consensus with respect to a SEBOK for systems engineering.

And without consensus on a SEBOK, the committee cannot produce even a minimal objective traceable set of generic requirements for the competency of a systems engineer.

As expected, each of the competency models described above:

- Was developed to provide a solution to a different problem and contains different bodies of knowledge. This is in accordance with general industry practice for the design and use of competency models (Ennis, 2008).
- Was **not** presented in a format compatible with the nine-tier US Employment and Training Administration (ETA) Competency Model Clearinghouse's General Competency Model Framework (ETA, 2010).
- Identified a large number of competencies and then grouped the competencies into smaller manageable but different groups that while meeting the need of the time and place, make comparing the assessment approaches difficult as shown in the summary in Table 12.4.

At the detailed level, NDIA aggregates 'requirements management' into 'technical competencies' (Gelosh, 2008) while MITRE groups the same function into 'systems engineering life cycle' (Metzger and Bender, 2007). SECT allocates 'requirements analysis' to 'Technical/Analytical Competencies' (Squires, et al., 2011)' while MITRE incorporates the function into 'requirements engineering' which is allocated to 'systems lifecycle'. Thus, a common framework that could encompass all the assessment approaches is needed to compare the different competency models. This framework would allow owners and users of each of the competency models to benchmark their competency model against the others, perhaps identify gaps, and upgrade their approach.

Some of the competencies being assessed fall into the category of cognitive characteristics. The traditional academic approach to measuring cognitive characteristics is based on the revised Bloom's taxonomy which combines systems thinking and critical thinking (Anderson, et al., 2000).

Research into the psychology domain identified an alternative approach based on the types of knowledge discussed in Section 9.5 which unlike Bloom's taxonomy, allows for the systems thinking and critical thinking skills to be assessed separately (Kasser, 2010a: pages 134-136).

The levels of ability in each in each of the nine competency models studied are also different, some models only recognise one level, some models assess skill proficiencies and some assess necessary proficiencies for job positions (roles) at specific levels in the organisational hierarchy as shown in Table 12.5. For example, Dreyfus and Dreyfus (Dreyfus and Dreyfus, 1986) quoted by Ennis (Ennis, 2008) describe levels of proficiency that include novice, experienced beginner, practitioner, knowledgeable practitioner, expert, virtuoso, and maestro. From the novices who are focused on rules and are limited or inflexible in their behaviour to the individual who is willing to break rules to provide creative and innovative solutions to business problems.

The existing competency models seem to have been populated based on observing the role of the systems engineer, namely what systems engineers do in the workplace, and researching the literature for additional requirements. These competency models may suffer from errors of omission because the development methodology does not include a validation function to determine if something that should be done is not being done (and the effect of that lack may not show up for some months or even years).

Indeed, this research has identified an error of omission in all of the nine competency models studied, namely the lack of competencies in the implementation domain. In addition, benchmarking used alone produces followers, not leaders. Benchmarking should be used only as a check to make sure a competency[3] is not lacking some necessary competency.

So while the models represent some aspects of the complexity of the situation they do not provide much help in managing the overall complexity.

A Competency Model Maturity Framework (CMF) that can be used to benchmark other competency models and as a competency model itself is proposed in Section 14.2.

12.26. Improving the practice of systems engineering by adjusting the terminology

This section looks at the following three ways of improving systems engineering by adjusting the terminology concerning:

1. 'Open' and 'closed' systems discussed in Section 12.26.1.
2. Containing systems and meta-systems discussed in Section 12.26.2.
3. The challenge of systems optimization discussed in Section 12.26.3.

12.26.1. Open and closed systems

This section discusses open and closed systems pointing out that from the HTPs these views are only partial views of a system with information abstracted out and proposes that these views should be considered as two partial views of a system, each used for an appropriate purpose.

Currently, open and closed systems are generally treated as different types of systems. Perhaps this is because the SOI is generally viewed from only one perspective; the other not being relevant at the time.

[3] Or anything else you are creating and wish to benchmark

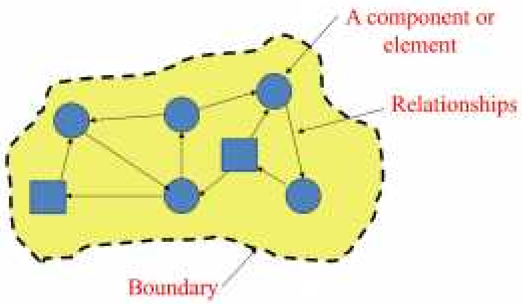


Figure 12.11 Closed system view

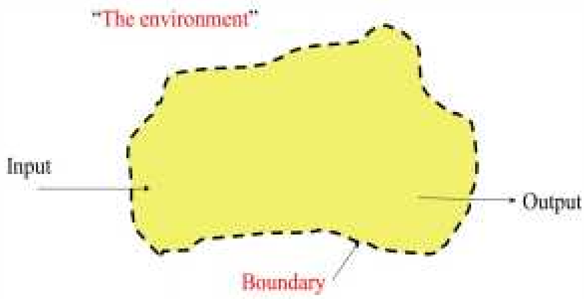


Figure 12.12 Open system view

However, closed systems such as the solar system or a pendulum clock are not really closed as discussed in Section 12.20.

From the HTPs, the same group of objects can be considered as an open and a closed system for different purposes; namely, they are two views of the same system each focusing on certain pertinent aspects and abstracting out non-pertinent aspects.

Internal and external views provide different information, the total of which provides understanding about the system and leads to insight. Separating the views reduces the complexity of the information inherent in a single view. Thus:

- ***Closed systems*** are white box views of the internal parts of a system for the purpose of studying the internal aspects of the system such as the one shown in Figure 12.11. They can be views from the *Structural* and *Functional* perspectives.

  Considering a closed system view without acknowledging that it is part of a metasystem is reductionist, may lead to incorrect conclusions and should be discouraged.

Table 12.4 Arrangement of competencies in the nine competency models

KSAs	INCOSE CSEP Exam	SECF	CEST	SECT	NASA 2010	JPL SEA	MITRE	NDIA
N/A	Systems Engineering Overview	Systems Thinking	Cognitive Characteristics	Systems and Critical Thinking	Concepts and Architecture	Processes	Enterprise Perspectives	Analytical
	General Lifecycle Stages	Holistic Lifecycle View	Systems Engineering Skills	Technical Expertise	System Design	Personal Behaviors	Systems Engineering Life Cycle	Technical Management
	Technical Processes	Systems Engineering Management	Individual Traits	Project Management	Production and Operations	Technical Knowledge	Systems Engineering Planning and Management	General
	Project Processes		Multidisciplinary Knowledge		Technical Management		Systems Engineering Technical Specialties	Professional Competencies
	Agreement Processes				Project Management		Collaboration and Individual Characteristics	

KSAs	INCOSE CSEP Exam	SECF	CEST	SECT	NASA 2010	JPL SEA	MITRE	NDIA
	Organizational Project Enabling Processes				Internal and External Environments			
	Tailoring Processes				Human Capital Management			
	Specialty Engineering Activities				Security and Safety			
					Professional and Leadership Development			

Table 12.5 Comparison of proficiency levels in the competency models

KSAs	INCOSE CSEP Exam	SECF	CEST	SECT	NASA 2010	JPL SEA	MITRE	NDIA
N/A	N/A	Awareness	N/A	None or Aware Only	Technical Engineer/Project Team Member	N/A	Foundational	N/A
		Supervised Practitioner		Apply with Guidance	Subsystem Lead		Intermediate	
		Practitioner		Apply	Project Systems engineer		Expert	
		Expert		Manage or Lead	Program systems engineer			
				Advance State of Art				

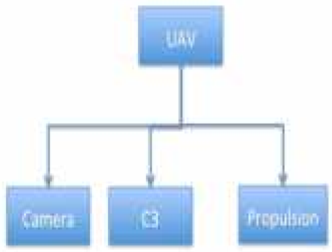


Figure 12.13 Partial view of the functions of a UAV

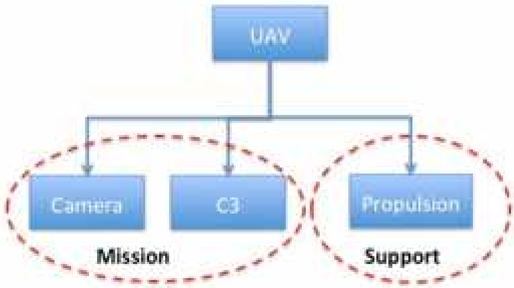


Figure 12.14 UAV functions grouped into functional subsystems (partial)

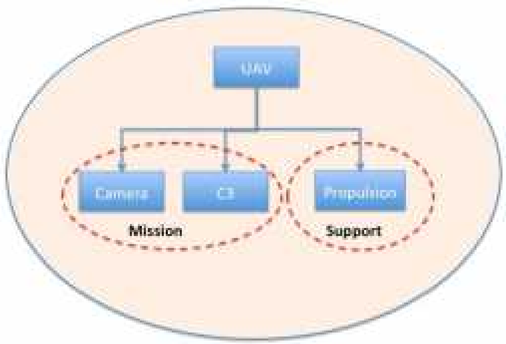


Figure 12.15 UAV functions (partial) with representation of containing system

- ***Open systems*** are black box views of the systems from an external perspective such as the one shown in Figure 12.12. Open systems views are generally associated with thinking about the big picture and the operation of the system in its context, namely views from the *Big Picture* and *Operational* perspectives.

  Accordingly, systems engineers should:

  - Consider open and closed systems as being two of several views of a system, where each view is used for an appropriate purpose.

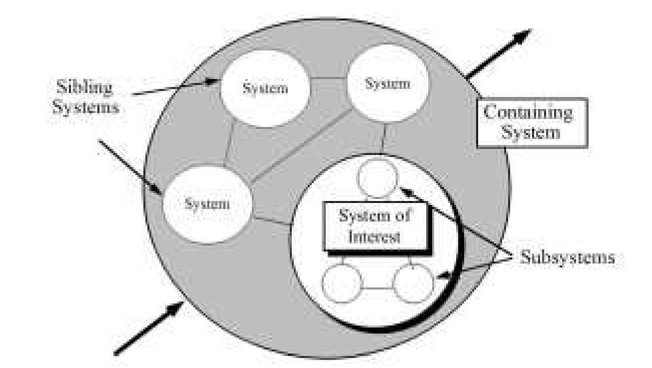


Figure 12.16 A nesting model of systems (Hitchins, 2013)

By doing so, they will abstract out much of the complexity in current views such as those used in the DODAF (DOD, 2010).

- Realize that open and closed systems are views of a system where:
  - The open system view treats the system as a black box and looks at the big picture and missions performed by the system, namely the *Big Picture* and *Operational* perspectives.
  - The closed system view looks at the internal workings of the system, namely the *Functional* and *Structural* perspectives.
- Stop using the terms 'open systems' and 'closed systems' and use the terms 'black box' or 'operational view' and 'white box' or 'functional/structural view' as appropriate instead.

12.26.2. Containing systems and meta-systems

This section discusses the implications of the finding that with all the admonishments not to be reductionist, the current graphical representation of a system is inherently reductionist. For example, a typical hierarchical view of the functions of a system is shown in Figure 12.13. The example is a partial view of the functions in an unmanned aerial vehicle (UAV) created for a purpose. When considering the UAV we should group the functional subsystems into mission and support subsystems, and so in the current way of doing things, we might add dotted lines as shown in Figure 12.14. While Figure 12.14 helps us relate to the mission and support functions, when used on its own, Figure 12.14 is a reductionist view since it leaves out the containing system and the adjacent functional systems. Recognizing that oversight, we may add the containing system as shown in Figure 12.15, this still leaves out the adjacent subsystems of the UAV.

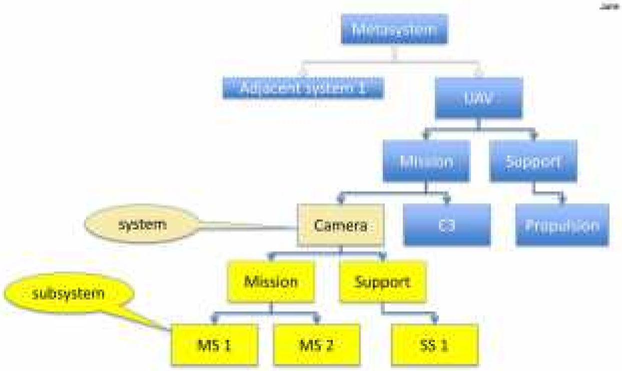


Figure 12.17 UAV Camera [sub]system

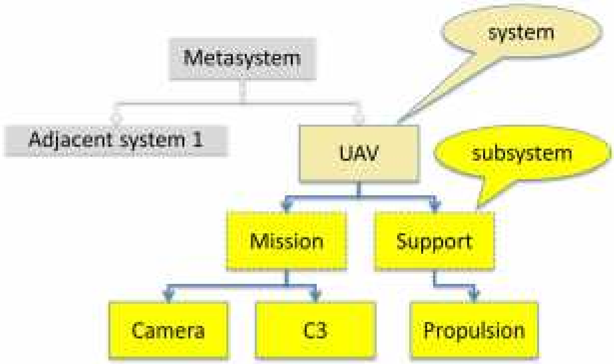


Figure 12.18 UAV [sub]system

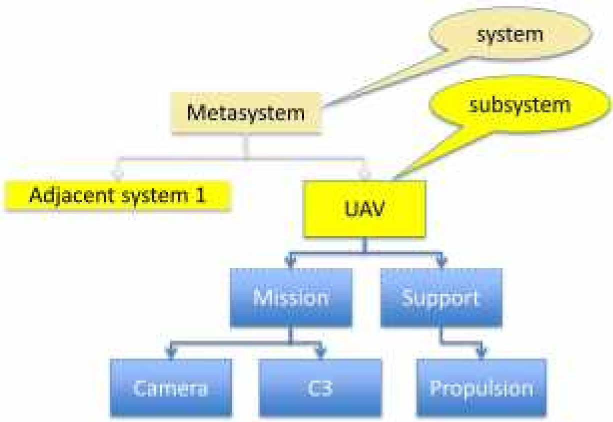


Figure 12.19 UAV Metasystem

A nesting view of a SOI and its adjacent systems inside a containing system is shown in Figure 12.16 (Hitchins, 2013) which has the same format as Figure 12.11. However, Figure 12.16 looks into two levels of the hierarchy from the top for the purpose of illustrating a nesting model of systems within systems within systems showing how:

- The SOI contains intra-connected sub-systems, which are systems in their own right, existing within their own environment.
- The SOI is interconnected to other sibling (adjacent) systems within their mutual environment, all within a containing system.
- The containing system is similarly connected to its sibling systems (not shown) all existing within their environment, and so on.
- "*Some siblings may be interconnected through the containing system boundary to systems within other containing systems*" (Hitchins, 2013).

While Figure 12.16 is an excellent view of a specific situation; namely looking into two levels of the hierarchy from the top for the purpose of illustrating a nesting model to show that sub-systems of the SOI can be systems in their own right and that the SOI has sibling systems, the use of has a tendency to lead to artificial complexity (Section 9.27). For example when the SOI and one or more of its sibling systems are considered together using this type of representation, they are often drawn with overlapping containing/encompassing system circles producing complex and complicated drawings such as those often seen in drawings associated with the DODAF operational views.

12.26.3. The challenge of system optimization

This section discusses the challenge of systems optimization. Optimizing complex systems represents a challenge for reasons that include:

- The systems optimization paradox (Section 9.24.1).
- There will usually be different viewpoints on what should be optimized.
- Traditional approaches to complex systems development either ignore the issue or optimize subsystems.

Wymore stated, "*Conventional systems engineering wisdom has it that if subsystems are optimized, then the system cannot be optimum*" (Wymore, 1997) and then used a mathematical approach to show that conventional wisdom was mistaken and how it was possible for systems engineering to ensure that optimum design of the subsystems can result in optimum design of the system. The principle of hierarchies also indicates that conventional wisdom is wrong but in a graphical manner as indicated in Figure 12.17, Figure 12.18 and Figure 12.19. System optimization at one level is always a subsystem optimization of the metasystem. If any system is a subsystem of the containing or metasystem, then where does the optimization take place? The answer is that system optimization at any level optimizes the interactions between the subsystems of that system level within the constraints imposed by the systems engineer of the metasystem, often via the "*the proper allocation of the system requirements to the subsystems*" (Wymore, 1997). For example:

- In Figure 12.17, the camera system systems engineer performs the trade-offs to optimise the mission and support subsystems for the camera subsystem; the mission system engineer performs the trade-offs to optimise MS1, MS2 and the other mission subsystems, the support systems engineer performs the trade-offs to optimise the support subsystems.
- In Figure 12.18, the UAV system systems engineer performs the trade-offs to optimise the mission and support systems of the UAV. The mission subsystems include the camera and the Communications, Command, and Control (C3) subsystems.
- In Figure 12.19, the metasystem systems engineer performs the trade-offs to optimise the combination of the UAV and its adjacent systems.

Systems engineering focuses on the performance of the SOI which is a combination of the performance of the subsystems and the interactions between the subsystems. The systems engineer is concerned with the system, the metasystem and the subsystem and should use a holistic approach that optimizes the interactions between the subsystems at design time rather than trying to optimize the subsystems as discussed in Sections 12.26.3 and 18.7.

12.27. The Standards for systems engineering are a myth

Consider the myth and corresponding reality.

The myth is that MIL-STD 499, EIA 632, IEEE 1220 and ISO/IEC 15288 (MIL-STD-499, 1969; EIA 632, 1994; IEEE 1220, 1998; Arnold, 2002) are commonly thought of as systems engineering standards (Section 7.4).

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 engi-

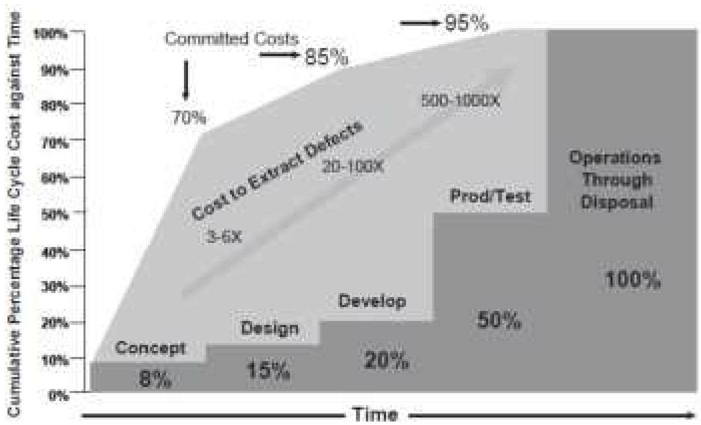


Figure 12.20 When costs are committed in the SLC (DAU)

neering" 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 (SETR) rather than to the activities known as systems engineering (SETA). In addition, many of the activities in ISO/IEC 15288 also overlap those of project management.

In addition, the standards commonly used/taught in systems engineering (MIL-STD-499, 1969; MIL-STD-499A, 1974; DOD 5000.2-R, 2002: pages 83-84) ignore most of the activities allocated to the Needs Identification State in the SLC resulting in the critical first SEP (Section 12.16) addressing the conceptual solution being out of mainstream Type II systems engineering. Table 12.1 shows the lack of coverage of the mission purpose/definition activities of the first SEP (Section 12.16) in MIL-STD-499 and ANSI EIA 632. The top row in Table 12.1 shows that MIL-STD-499 and ANSI EIA 632 do not cover the conceptual activities in early stage systems engineering and while the Systems Engineering CMM, 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 first SEP (Section 12.16).

This is a critical omission since studies have shown that the cost of a system is determined in its early stages. A typical example shown in Figure 12.20 is a Defense Acquisition University study quoted in the INCOSE systems engineering handbook (Haskins, 2006b: 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 SLC.

The purpose of the DODAF was to be used to "*provide correct and timely information to decision makers involved in future acquisitions of communications equipment*" (DoDAF, 2004). 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 11.1 which roughly corresponds to the same curve as the cost to fix a defect as a function of the time the defect is discovered[1]. The early states of systems engineering to the left of the vertical axis in Figure 11.1 are not covered by the standards. While DoD 5000 (DOD 5000.2-R, 2002: pages 73-74) does call out some of the early state activities, those activities are called out as part of the separate independent 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 lifecycle 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 (performed in the Needs Identification State of the SDP) are fourfold[2]:

1. To perform concept studies to investigate different solutions.
2. To evaluate these different concepts.
3. To perform trade-off studies.
4. To define the requirements for the remainder of the acquisition program.

[1] No correlation between the parameters is implied.
[2] 'A' paradigm activities

Standards continue to appear[3] as do systems engineering failures. It seems that **standards may be useful in helping you to produce the wrong system more effectively**. Systems engineers 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 (Section 10.9).

12.28. Systems of Systems are a different class of problem and need new tools and techniques is a myth

There is a dichotomy on the issue similar to the dichotomy on complexity (Section 9.24). The earliest reference to System of Systems found in the literature was Jackson and Keys who wrote that a problem solver needs a methodology for [selecting the appropriate methodology for] solving a problem (Jackson and Keys, 1984) which has nothing to do with the use of the term in modern systems engineering.

The myth is typified by the definition of 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*" (Allison and Cook, 1998). They categorized System of Systems as being:

- ***Permanent:*** such as airlines and national Defence forces,
- ***Temporary:*** ephemeral or virtual; examples of such are multinational peace keeping forces and project teams.

Cook stated, "*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*" (Cook, 2001).

Sillitto stated, "*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 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" (Sillitto, 2008).

[3] And existing standards are occasionally updated.

The reality is that 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 as discussed in the principle of hierarchies (Section 7.2). The characteristics of Systems described by Sillitto are the characteristics of systems in Layer 3 of the HKMF. For example, Sillitto's description applies 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].

Another use of the term System of Systems describes 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

The problems being addressed are those that Operations Research was set up to address in the 1940's and the tools and techniques exist and have existed for the last 50 years such as the tools for systems engineering in the 1950's and 1960's (Section 7.3).

The myth arose when systems engineers educated and practicing in the HKMF Layer 2 US DoD systems engineering paradigm (DOD 5000.2-R, 2002) lacking the tools of the 1950s and 1960s attempted to tackle HKMF Layer 3 problems. "*Complexity*[5] *is in the eye of the beholder*" (Jackson and Keys, 1984); yes, it is a new class of problem to the HKMF 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[6]; such tools do indeed exist and have existed for more than 50 years.

12.29. Aspects of detailed design decisions

This section contains two insights concerning detailed design decisions.

- Detailed design decisions should be made on a Just in Time (JIT) basis discussed in Section 12.29.1.
- Design decisions must also maximize the "don't care's" as well discussed in Section 12.29.2.

12.29.1. Detailed design decisions should be made on a just-in-time (JIT) basis.

There is no need to complete the detailed physical design before starting a Build or Iteration (Section 13.4). However, the detailed design must be feasible. The risk here is in determining the feasibility of the design. For example, consider a case where there is an iffy[7] need for synchronous

[4] Operational Analysis in the UK

[5] I'd use the word 'complicated' to point out that it is subjective complexity.

[6] At least without examining the current set of tools

[7] The requirement is not absolutely certain but included as part of risk management.

voice communications between two places. Since an initial assessment shows that the need can be met using the conventional telephone service or by the use of voice over the Internet, there is no need to make that decision early in the design cycle. The characteristics of the telephone link are known. The characteristics of Internet voice links are also known. Prototyping experiments can take place and the actual decision made in a Just in Time (JIT) manner as to which technology to use to implement the communications links. Since there is a possibility that the requirement for synchronous communications may be deleted in the future, any detailed design effort made earlier would be wasted if the requirement were eliminated. In addition, if the requirement is not eliminated, then advantages can be taken of improvement in technology and/or cost reductions during the time before the decision has to be made.

12.29.2. Design decisions must also maximize the "don't care's" as well.

The example in Section 12.29.1 is Internet voice works (risk minimal) but the actual choice of how to implement the communications subsystem can wait for a while. A better example is from the LuZ Solar Electrical Power Generating System (SEGS) sun sensor glue case in which:

- ***The undesirable situation:*** in the mid-1980's where the LuZ Group, a start-up joint Israel-American venture was developing the world's first commercial Solar Electrical Power Generating System (SEGS-1) (Kasser, 1984). As the first of its kind, SEGS-1 initially only existed then as a vague concept. The station was to be integrated at the installation site in the Mojave Desert in California and the research and development was to be done in Jerusalem.
- ***The FCFDS:*** SEGS-1 which would generate electrical power from the sun by focusing the sun's rays on about 600 parabolic mirror trough reflector collectors each about 40 meters long. The system architecture was that of a distrusted network with a central control microcomputer system controller and operator interface. The operation of each parabolic trough reflector was

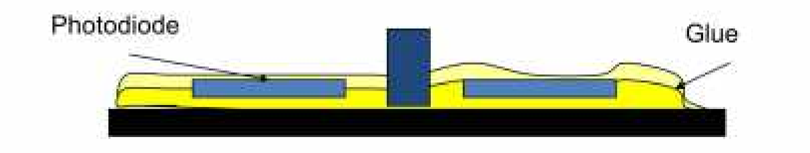


Figure 12.21 The Luz sun sensor

monitored and controlled by a microprocessor based Local Controller (LOC).

- ***The problem:*** to conceptualize and realize the system.
- ***The solution:*** a distributed system. Each LOC controlled a motor that positioned the parabola, and received information about the angle of elevation and the temperature of the oil in the pipe positioned at the focus of the trough. Oil was pumped through the piping, and as long as the LOC kept the reflector pointed at the sun within an accuracy of ±0.2 degrees, the oil was heated. The hot oil was pumped thorough a heat exchanger to generate steam. The steam drove a turbine that generated up to 15 Megawatts of electrical power. Although it was a complex system, it still had a conversion efficiency of about 40%, greater than any alternative method of harnessing solar energy at the time.

The sun sensor provided an example of what can go wrong when "don't care" situations are not considered. The sun sensor used a lens to focus the sun onto a pair of photo diodes with a separating spacer as shown in Figure 12.21. During the assembly process, the diodes were glued to a base plate with transparent glue. The physics department who had created and were building the sun sensors did not place a requirement that there be no glue on the front side of the diode which was illuminated by the sun. After all, the glue was transparent so it was a "don't care" situation. A year or so later, they found that the glue slowly became opaque when subjected daily to the very high temperature at the focal point of the lens. This phenomenon resulted in the need to replace all the sun sensors. From a manufacturing perspective, there was little difference in mounting the diodes if the glue could or could not be allowed to cover the face of the diode, just a matter of care and a few extra minutes of time[8]. Nobody asked about possible changes to the characteristics of the glue over long periods of time under high temperature. If the requirement had been placed on the process, not to allow glue on the face of the diode, the characteristics of the glue under the high temperature conditions would not have mattered and the expensive sun-sensor replacements would have been avoided (Kasser, 1995). This is an example of introducing an unnecessary failure mode by not utilizing the "don't cares". Thus the lesson learned is that if it doesn't make any difference don't do it.

---

[8] A few minutes multiplied 600+ times is still less than two days. The 600+ include the spares also produced in the initial batch.

12.30. Summary

This Chapter contained inferences and insights on systems engineering from the *Scientific* perspective. The key points were:

- Systems engineering is more than just applying process standards.
- Some reasons why systems engineers cannot agree on the nature of systems engineering.
- There are three types of systems engineering, pure, applied and domain.
- The implementation domain needs to be considered.
- The devolution of systems engineering.
- Resolving the overlap between systems engineering and project management.
- The need to focus on people as well as process.
- The 'B' paradigm is inherently flawed.
- Five reasons for the failure of systems engineering.
- One reason for the success of systems engineering.
- Dealing with problems.
- The need to change the SDP from a single waterfall to a series of cataracts at process design time.
- While there is a consensus that systems engineering is a discipline there does not seem to be consensus as to what type of discipline.
- The process, roles, emergent properties, tools and optimisation paradoxes were resolved.
- MBSE is:
  - Much ado about nothing new.
  - Is a return to the 'A' paradigm.
- The different ways of assessing systems engineering competency are specific to the originators and not really suitable for general use.
- Systems engineering can be improved by adjusting the terminology.
- Stop using the terms 'open systems' and 'closed systems' and use the terms 'black box or operational view' and 'white box or functional/structural view as appropriate' instead.
- Three of the myths of systems engineering are:
  - There is a single systems engineering process (SEP).
  - There are standards for systems engineering.

    - Systems of Systems are a different class of problem and need new tools and techniques.
- Detailed design decisions shall:
    - Be made on a Just in Time (JIT) basis.
    - Maximize the "don't care's".

--oo--

13. Perceptions of the System Lifecycle

This Chapter suggests improvements to systems engineering based on insights and inferences from perceptions of the System Lifecycle (SLC) from the following insights:

- The SLC as a State Machine discussed in Section 13.1.
- The SDP discussed in Section 13.2.
- The relationship between the SDP and the O&M State discussed in Section 13.3.
- The operate, modify and upgrade cycle in the O&M State discussed in Section 13.4.

13.1. The SLC as a State Machine

The traditional view of the SLC is that it passes through a number of phases in a sequential manner, often represented by the waterfall. Each phase starts and ends at a formal milestone. However, the SLC can also be treated as a State Machine (Wymore, 1993). The SLC State Machine perspective is similar but the State Machine Model facilitates understanding the effect of changes during the SLC as discussed in this section.

As perceived in Section 9.12, each state of the SLC starts and ends at a major milestone (Section 9.14). The output of each state becomes the input to the subsequent state. When each state is framed in the problem formulation template (Section 14.5) the solution output of any state becomes the problem input to the subsequent state. This situation, shown in Figure 9.4, is often referred to as the:

- ***"What's":*** which refer to what needs to be done, or the problem.
- ***"How's":*** which refer to how it is done, or the solution.

The SLC State Machine perspective:

- Shows the SLC as a cycle in which the O&M State transitions back to the Needs Identification State as shown in Figure 13.1 which represents an ideal notional situation in which no changes take place during the SDP.
- Acknowledges the reality that once the system is in operation it undergoes changes because as time goes by:

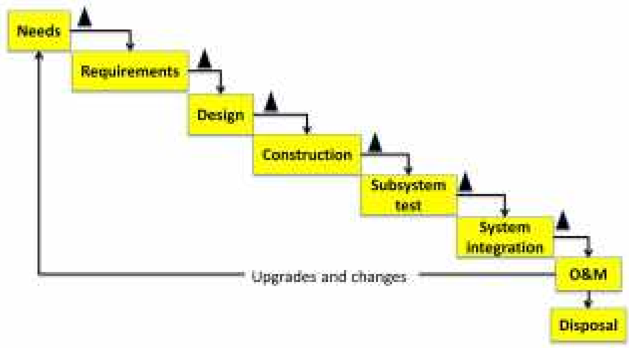


Figure 13.1 Cyclic waterfall view of the SDP in the SLC

- Something that is not being done at all becomes needed resulting in a new system, modified current system(s), or a combination.
- Something that is being done is no longer needed resulting the system being retired or the disposal of [all or parts of] current system(s).
- Something needs to be done better [or worse] resulting in a new system, modified current system(s), or a combination.
- New technology comes into the market which allows something to be done that could not be done before, e.g. new functionality, reduction in size/weight/cost of existing functionality, etc.

What the *Functional* perspective in Figure 13.1 does not show is that when the O&M State transitions back to the Needs Identification State and starts a new cycle, the O&M State may continue in existence in parallel with the new cycle SDP until a new O&M State replaces the existing one. This situation as perceived from the *Temporal* perspective is best shown as a schedule in Gantt chart format as in Figure 13.2 in which the time in each state is for educational use only and does not represent real-world ratios.

Figure 13.2 shows the SLC starting at the initial Needs Identification State and proceeding down the waterfall until the system becomes operational in the initial O&M State. Sometime into the O&M State, a change is approved and the impact of the change is assessed in a second Needs Identification State that co-exists in time with the initial O&M State. The impact of the change propagates through the SLC states shown as a second waterfall SDP coexisting with the initial O&M State until a transition takes place and the modified system becomes active in the second O&M State. The second O&M State is shown as briefly co-existing with the first O&M State while the modified system is validated. Once the modi-

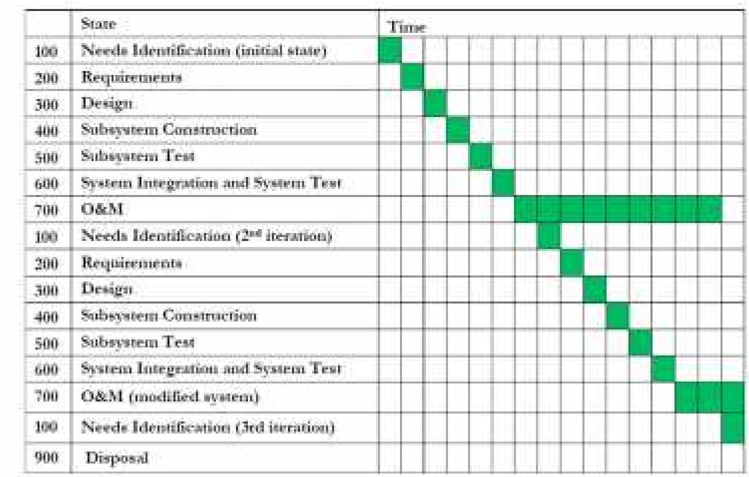


Figure 13.2 Notional SLC in schedule format

fied system is deemed fully operational, the initial system is taken out of service[1] and the SLC rests in the second O&M State. This iteration is known as the change cycle. When another change is approved the SLC iterates through the subsequent change cycle (Section 17.1.8).

Each state in the SLC has two exit conditions:

1. The normal planned exit at the end of state milestone review which documents consensus that the system is ready to transition to the subsequent state.
2. An anticipated abnormal exit anywhere in the state that can happen at any time in any state and necessitates a return to an earlier state in the SLC due to:

    a) A defect that requires rework found during DT&E. changes to the requirements, design, etc. Namely any change other than replacing a defective item.
    b) An approved change.

    This type of exit is recognized and is often depicted in the chaotic views of the SLC similar to Figure 13.3 which shows every state connected to every other state.

13.2. The SDP

The SDP:

- Consists of the first seven states of the SLC beginning with the Needs Identification State and ending when the system becomes

[1] Not always true, see Section 13.4

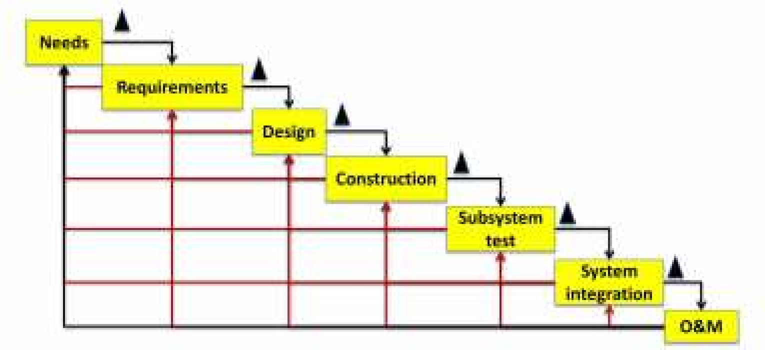


Figure 13.3 The Chaotic view of the SDP

operational at the start of the initial O&M State.

- May be:
  1. An acquisition process to purchase COTS equipment.
  2. A development process to design, built, test and integrate a system.
  3. An integration process to integrate equipment (subsystems) acquired from one or more vendors.
  4. A combination of the above.

13.2.1. Framing the SDP

The SDP can be framed in the problem formulation template (Section 14.5) as follows.

- ***The undesirable situation:*** a situation containing indications that something needs to be done because a desired or needed function cannot be achieved. This need may have:
  - Appeared since the system in the undesirable situation was placed into service.
  - Arisen because a desired function that was not feasible when the system was originally placed into service has only now become feasible by the development of new technology.
  - Something is no longer needed and needs to be removed because it is causing the situation to exhibit undesirable behaviour.
  - A combination of the above.
- ***The FCFDS:*** the same as undesirable situation but without the undesirable aspects of undesirable situation with perhaps added desirable functionality.
- ***The problem:*** to effect a transition from the undesirable situa-

tion to the FCFDS. This well-structured complex problem is generally elaborated into several sequential well-structured problems including:

- Determining the root causes of the undesirability in undesirable situation.
- Determining how to transition from the undesirable situation to the actual future desirable situation by formulating the strategies and plans to realize the solution system.
- Designing and realizing the solution system in its context in accordance with the plan.
- Deploying the solution system to complete the transition from the undesirable situation to the future desirable situation.
- Verifying that the created desirable situation remedies the original and evolved needs and does not seem to contain any undesirable characteristics; this is performed in OT&E.

- ***The solution:*** the solution system operating in its context sometime in the future when the FCFDS has evolved into the actual future desirable situation.

13.2.2. The States in the SDP

Consider each framing each state briefly:

13.2.2.1. The Needs Identification State

The Needs Identification State discussed in Section 9.12.1 can be framed as follows:

- ***The undesirable situation*** at the start of the State is the situation that is the cause of the project initiation.
- ***The FCFDS*** is stakeholder consensus that the CONOPS of a conceptual system operating in its future context will constitute a future situation without any undesirable characteristics.
- ***The problem*** includes:
  - Gaining an understanding of the causes of undesirability of the undesirable situation.
  - Gaining consensus on those causes by the stakeholders.
  - Articulating the conceptual solution system that when operating in its context will remedy the undesirable aspects of the situation.
  - Gaining stakeholder consensus that that conceptual solution system, when operating in its context, will remedy the undesirable aspects of the situation.

  - Determining that realization of the conceptual solution system is feasible.

- ***The solution*** at the end of the Needs Identification State is an articulation of the proposed conceptual system and a way to make it happen in the form of an approved CONOPS and feasibility study.

13.2.2.2. The Requirements State

Framing the problem in the Requirements State discussed in Section 9.12.2 depends on the paradigm (Section 9.21). The 'A' paradigm Requirements State can be framed as follows:

- ***The undesirable situation*** at the start of the State is the lack of:
  - The complete set of matched specifications for the conceptual solution system.
  - The detailed strategies and plans to implement the transition from the undesirable situation to the future situation without the undesirable characteristics.
- ***The FCFDS*** is a complete set of:
  - Matched specifications for the solution system.
  - Detailed strategies and plans for the process to implement the transition from undesirable situation to the future situation without the undesirable characteristics; namely accepted versions of the System Engineering Plan (SEP), Systems Engineering Management Plan (SEMP), Test and Evaluation Master Plan (TEMP), etc. as appropriate.
- ***The problem*** is to create the FCFDS from the CONOPS and feasibility study incorporating any changes that occur during the Requirements State.

The 'B' paradigm Requirements State can be framed in the same way as for the 'A' paradigm, however:

- ***The problem*** is to create the FCFDS as well as the CONOPS and perform the feasibility studies that should have been created in the non-existent Needs Identification State.
- ***The solution*** at the end of the Requirements State is the FCFDS.

13.2.2.3. The System Design State

The System Design State discussed in Section 9.12.3 is split into two parts:

- The Preliminary System Design sub-state (SRR to PDR).
- The Detailed System Design sub-state (PDR-CDR).

In the 'A' paradigm, the Preliminary System Design sub-state (SRR to PDR) can be framed as follows:

- ***The undesirable situation*** at the end of the SRR is the lack of preliminary designs for the solution system that meets the matched set of specifications accepted at SRR.
- ***The FCFDS*** is consensus that that a feasible preliminary design for the solution system:
  - Meets the matched set of specifications accepted at SRR.
  - Remedies the original and evolved undesirable situation.
- ***The problem*** is to create the FCFDS by:
  - Converting the matched set of specifications to a preliminary design.
  - Gaining the consensus.
- ***The solution*** at the end of the PDR is the FCFDS.

In the 'B' paradigm, the Preliminary System Design sub-state (SRR to PDR) the undesirable situation is different and can be framed as follows:

- ***The undesirable situation*** at the end of the SRR is:
  - The lack of a functional design and CONOPS[2].
  - The lack of preliminary designs for the solution system that meets the matched set of specifications accepted at SRR.

The Detailed System Design sub-state (PDR to CDR) is the same in both paradigms and can be framed as follows:

- ***The undesirable situation*** at the end of the PDR is the lack of a final design for the solution system that meets the matched set of specifications accepted at SRR.
- ***The FCFDS*** is consensus that that a final design for the solution system:
  - Meets the matched set of specifications accepted at SRR.
  - Remedies the original and evolved undesirable situation.
  - Is feasible

---

[2] Assuming they were not created during the Requirements State.

- ***The problem*** is to create the FCFDS by:
  - Converting the preliminary design to the feasible documented critical or final design.
  - Gaining the consensus.
- ***The solution*** at the end of the System Design State is the FCFDS.

13.2.2.4. The Subsystem Construction State[3]

The Subsystem Construction State discussed in Section 9.12.4 can be framed as follows:

- ***The undesirable situation*** at the start of the State is the need to construct each subsystem, in isolation, according to the final design approved at the CDR.
- ***The FCFDS*** is the set of subsystems, constructed in isolation, according to the final design approved at the CDR.
- ***The problem*** is to construct each subsystem in isolation according to the final design approved at CDR in such a manner that the system should meet all its specifications once all the subsystems will have been integrated.
- ***The solution*** at the end of the Subsystem Construction State is the FCFDS.

13.2.2.5. The Subsystem Testing State[4]

The Subsystem Testing State discussed in Section 9.12.5 can be framed as follows:

- ***The undesirable situation*** at the start of the State is the need to validate each of the subsystems, in isolation, as being compliant to its requirements.
- ***The FCFDS*** is when the complete set of subsystems has been validated in isolation as being compliant to their requirements.
- ***The problem*** is to ensure the set of subsystem tests:
  - Validate that each of the subsystems, in isolation, is compliant to its requirements.

[3] The scope of the activities will depend on the nature of the system being developed. If the system is a process, a COTS -based product or a human system, then the nature and scope of the activities will be different.

[4] The scope of the activities will depend on the nature of the system being developed. If the system is a process, a COTS -based product or a human system, then the nature and scope of the activities will be different.

  - Are performed in a sequential order that will facilitate system integration.

- ***The solution*** at the end of the Subsystem Testing State is the FCFDS. Note that subsystem testing may continue after the IRR should the integration be phased, as long as the subsystem testing for a subsystem is completed before that subsystem is scheduled to be integrated into the system.

13.2.2.6. The System Integration and System Test States

The System Integration and Testing States discussed in Section 9.12.6 can be framed as follows:

- ***The undesirable situation*** at the start of the States is:
  - The combination of the subsystems which have been developed and have passed their stand-alone tests in isolation (hopefully) have not been integrated into the solutions system.
  - The performance of the whole solution system, with optimum effectiveness, in its operational context, under test conditions, has not been established.
- ***The FCFDS*** is when the performance of the whole solution system, with optimum effectiveness, in its operational context, under test conditions, has been established and shown to meet or exceed the specifications as they exist at the end of the System Integration and System Test States.
- ***The problem*** is to integrate and validate the solution system according to the approved plans.
- ***The solution*** at the end of the System Integration and System Test States is the successful completion of the set of activities that:
  - Combines the parts, subsystems, interactions, etc., to constitute the solution system.
  - Establishes, under test conditions, the performance of the whole solution system, with optimum effectiveness, in its operational context.

13.2.2.7. The Operations and Maintenance State

The O&M State discussed in Section 9.12.7 can be framed as follows:

- ***The undesirable situation*** is that the system is doing something it should not be doing, or not doing something it should be doing.

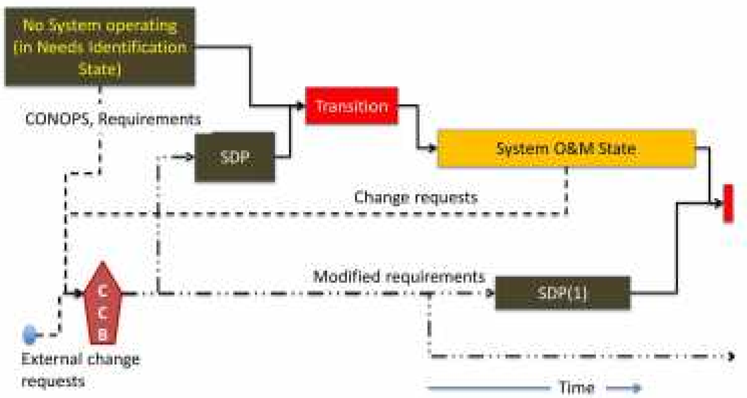


Figure 13.4 The SDP in the O&M State

- ***The FCFDS*** is when the system is doing what it should be doing. In the event that the system cannot be modified to meet the need, then the system transits to the Disposal State and a replacement project is initiated.
- ***The problem*** is to figure out what needs to be changed, and make the change if it is affordable. If it is not affordable then the problem is absolved but kept in view until it becomes affordable.
- ***The solution*** at the end of the O&M State is the FCFDS.

13.2.2.8. The Disposal State

The Disposal State discussed in Section 9.12.8 can be framed as follows:

- ***The undesirable situation*** at the start of the State is the solution system is not needed and needs to be disposed of.
- ***The FCFDS*** is the situation without the system, often containing a replacement system.
- ***The problem*** is to remove the system from service in an orderly manner with minimal impact on the situation.
- ***The solution*** at the end of the Disposal State is the absence of the system in the situation.

13.3. The relationship between the SDP and the O&M State

Perceptions from the *Temporal* perspective identify the notional relationship between the SDP and the O&M State shown in Gantt chart format in Figure 13.2. When perceived from the *Functional* perspective, the relationship is shown[5] using a flow format as in Figure 13.4 which depicts

[5] Note the example of the use of different views for different purposes wherein each view contains information pertinent to the purpose and abstracts out all other information.

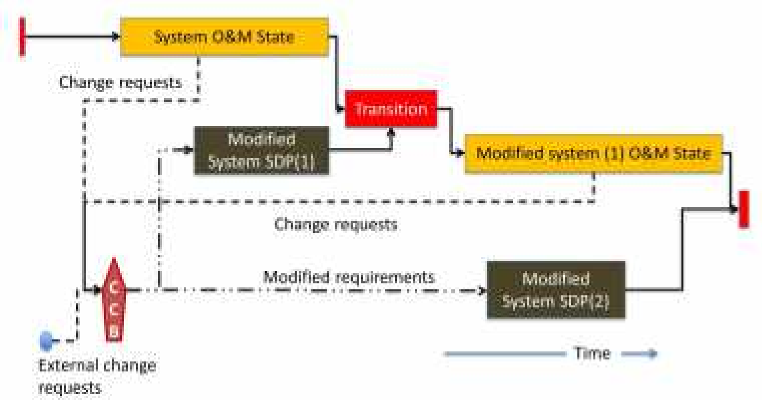


Figure 13.5 O&M State with continuous updates

the situation in which there is no system during the initial states of the SDP and the initial version of the system comes into existence during the transition or deployment into the initial O&M State. During the SDP, change requests may be received from within the SDP or from external sources. In summary:

- All change requests are managed by a joint team consisting of customer[6] and developer (often a contractor) representatives. This joint team is often called a Configuration Control Board (CCB).
- The decision to accept or reject a change request is made by the customer (Section 17.1.8).
- All accepted changes impact:
  - The work; either by adding or removing work and accordingly affect cost and schedule.
  - The functionality of the system; either by adding or removing functionality.
- The work to implement the change may be assigned to the SDP in progress, or delayed to a subsequent SDP shown as SDP(1) in Figure 13.4 depending on the urgency, nature and scope of the impact of the change.

Once the SLC is in the initial O&M State, further change requests may be generated as shown in Figure 13.4 and assigned to the SDP (1) which is proceeding in parallel in time with the initial O&M State or to SDP(2), a subsequent SDP which will take place after the initial O&M State transitions to the first modified O&M State shown in Figure 13.5.

---

[6] In this situation the customer is defined as the entity paying for the system to be developed or for the change to be implemented.

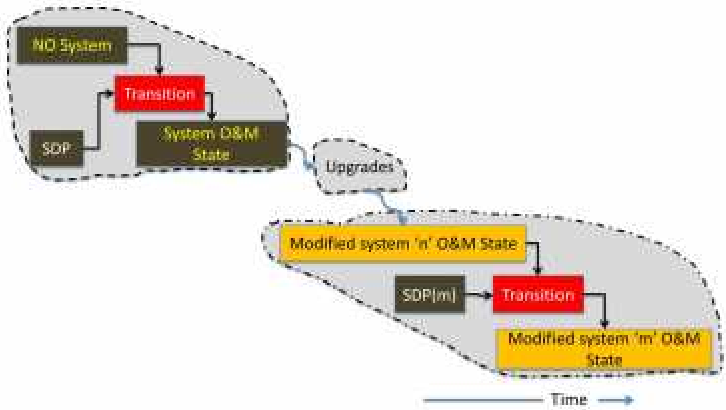


Figure 13.6 The SLC (extended state machine perspective)

Figure 13.5 provides a more generic view of the SLC following the initial O&M State showing the system in the O&M State of the SLC while a SDP coexists in time. This coexisting SDP is where the changes are implemented and when the SDP ends, the system transitions from one version to the subsequent version. At that point there is an operational system and a new SDP coexisting together. This upgrade and transition cycle continues until the system can no longer meet the needs and makes a transitional to the Disposal State.

If Figure 13.4 and Figure 13.5 are combined, the SLC can be shown in the format presented in Figure 13.6 a higher-level representation[7]. The initial SDP of Figure 13.4 is shown in the top of the figure, followed by updates which are processed as shown in Figure 13.5 and the system operate, modify and upgrade cycle shown in the lower half of Figure 13.6 where the system coexists in the modified 'n' version and SDP(m) States and transitions to the modified system 'm' State at the appropriate scheduled time. At which point in time a new SDP cycle will also begin.

With respect to of Figure 13.6:

- The typical Defence SDP is thought of as starting at the top, at the start of the initial SDP where there is no existing system.
- The focus of many courses on systems engineering is limited to the initial SDP.
- The typical commercial starting point is the O&M State represented by the modified system 'n' O&M State in the lower half

[7] While the figure does break the rule managing complexity by not showing internal and external views in the same drawing (Section 18.6.1), (a) it does conform to Miller's rule of 7±2 (Miller, 1956), and (b) its use is similar to the use of Figure 12.16 in its original context.

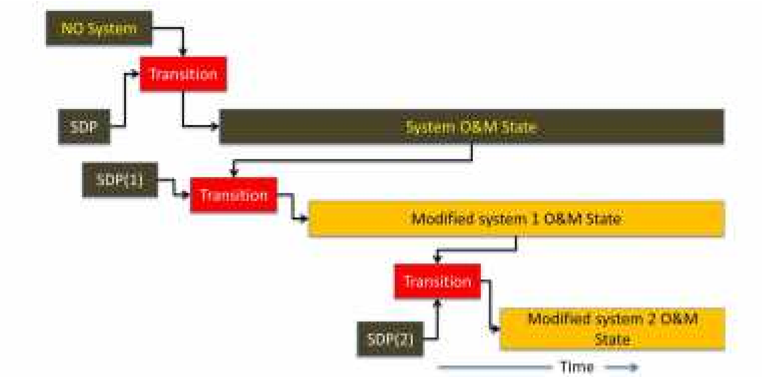


Figure 13.7 The generic extended SLC

of the figure This point can be considered as the typical upgrade/replacement starting point to state.

- The operate, modify and upgrade cycle generally has more constraints than the initial SDP due to the need to be compatible with earlier versions of the system.

13.4. The O&M product support sub-state

Until now, the SLC has been discussed with the assumption that in the O&M State, the current version of the system is replaced by an upgraded version and the current system is then taken out of service. This assumption is not necessarily valid in many commercial and military situations where older versions of a product (HKMF Layer 2) or system (HMKF Layer 3) remain in service for extended periods of time. This situation is represented by the generic extended SLC in Figure 13.7 where:

- Different versions of system are operating and need to be supported.
- Versions of the system may be phased in and out during the O&M State until whole set is obsolete and needs to be disposed of.
- Each modified version coexists with a SDP.
- A Configuration Control/Management system is critical to managing the situation.
- The generic extended SLC is a multi-phased time-ordered parallel-processing recursive paradigm (Kasser, 2002a).

13.5. Summary

This Chapter suggested improvements to systems engineering based on insights and inferences from perceptions of the SLC. Key insights include:

- The SLC as a State Machine.
- The “what’s” and the “how’s” of system engineering match the problem-remedy/solution model.
- The way each state is described via the problem formulation template.
- The generic extended SLC.

--oo--

14. Conceptual tools for systems engineering

Effective workmen sharpen their tools. Effective systems engineers not only sharpen their tools they are also always on the lookout of new tools that they can adopt or modify for their own use. This Chapter is a continuation of the *Scientific* perspective and contains a selection of tools and frameworks for improving the practice of systems engineering. Specifically:

1. Predicting technology availability discussed in Section 14.1.
2. A Competency Model Maturity Framework (CMMF) discussed in Section 14.2.
3. Using the principle of hierarchies to manage complexity; discussed in Section 14.3.
4. The HKMF discussed in Section 14.4.
5. A problem formulation template discussed in Section 14.5.
6. A problem classification framework discussed in Section 14.6.

14.1. Predicting technology availability

The Technology Readiness Level (TRL) as one way of assessing technology readiness was discussed in Section 10.11. This section discusses how holistic thinking can produce a different answer to the same problem by comparing the traditional approach with the holistic thinking approach.

14.1.1. The traditional approach

Consider the traditional approach employed in the 1990's which produced the TRL as a baseline reference. Framing the problem:

- ***The undesirable situation*** is articulated in a focused manner as follows:
  - It is 1998.
  - A system under development is to be deployed in 1999 to meet a projected need.
  - There is no current suitable technology that can be employed for realizing that system.
  - There is no systemic and systematic way to determine the readiness of a technology for use in a product other than seeing it incorporated in current products (GAO, 1999).

- ***The FCFDS*** is the technology is ready in 1999 when needed and is in use in a fully operational deployed product or system.
- ***The problem*** is to create a tool or a methodology (or both), that a decision-maker, the project manager and systems engineer, can use to determine if a technology is mature enough to integrate into the system under development so that the FCFDS will be created in a timely manner.
- ***The solution*** in 1998 was the TRL shown in Table 10.2.

14.1.2. The holistic thinking approach

Now use the holistic thinking approach. Framing the problem:

14.1.2.1. The undesirable situation

The undesirable situation is the same as in the traditional approach in Section 14.1.1. The holistic approach perceives the specific instance of the undesirable situation from each HTP namely:

14.1.2.1.1. *Big Picture perspective*

The same perceptions from the *Big Picture* perspective are articulated as in Section 14.1.2.1.1. In addition, other *Big Picture* perceptions include:

- Any other assumptions about the technology.
- A description about need of technology for the product (system).
- A description of environment in which product incorporating the technology will be used.
- A list and description of the known users of the product.
- A description of the adjacent systems interfacing with the product.

14.1.2.1.2. *Operational perspective*

Perceptions from the *Operational* perspective include scenarios of the different types of missions the product using the technology will perform. Typical generic scenarios include the use of the product incorporating the technology in different categories of missions such as:

- One-of-a-kind, single use, short and long term missions such as the NASA planetary space explorers in the 20^{th} Century. NASA generally developed the technology for a spacecraft for a mission. Since the number of spacecraft were small, the technology could be used at TRL=6. A small number of spacecraft could be crafted and deployed without being placed in mass production.
- One-of-a-kind military targets of opportunity such as Operation

Chastise which went operational on 16 May 1943 at TRL=6. The special purpose dam-busting bombs were crafted and deployed for that specific mission without being placed in mass production and being made available for other types of missions.

- Examples of various uses of the technology in considerable numbers of commercial and military products over a long period of time. This type of deployment requires TRL=9 to guarantee availability of the technology when needed.
- Various in-between scenarios.

14.1.2.1.3. *Functional perspective*

Perceptions from the *Functional* perspective describe how the technology functions.

14.1.2.1.4. *Structural perspective*

Perceptions from the *Structural* perspective include limitations of the technology imposed by its physical structure. Examples might include sensitivity to vibration and humidity (e.g. cannot be used in humid environments).

14.1.2.1.5. *Quantitative perspective*

Perceptions from the *Quantitative* perspective indicate:

- The maturity of the technology can be represented in monotonically increasing levels of technology ranging from a 'concept that needs to be developed' to 'being incorporated in significant quantities of production items'. The US Government Accounting Office (GAO) relates TRL to programmatic risk as shown in Figure 14.1 (GAO, 1999).
- The nine levels of technical maturity shown in Table 10.2 and Table 10.3 comply with Miller's rule of 7±2 (Miller, 1956) for comprehension of an issue.

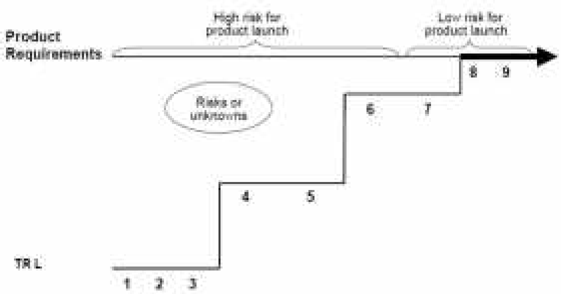


Figure 14.1 Programmatic risk as a function of TRL (GAO, 1999).

14.1.2.1.6. *Generic perspective*

Perceptions from the *Generic* perspective include other ways of assessing readiness and capability to do something, including:

- Capability maturity models.
- Competency models.
- ISO 9001.
- Risk assessment rectangles.
- Temperature thermometers or other meters with useable range markings.
- The 'S' curve which illustrates the introduction, growth and maturation of innovations and technology.

14.1.2.1.7. *Continuum perspective*

Perceptions from the *Continuum* perspective include:

- The differences in the types of operational uses for the technology mentioned in Section 14.1.2.1.2.
- The different types of missions which are described in the *Operational* perspective.
- The differences between:
  - a) using a methodology and a tool to assess the current state of something, and
  - b) using a tool to predict the future state of the same thing.
- Risks and risk management pertaining to the misuse of the methodology and tool.

14.1.2.1.8. *Temporal perspective*

Perceptions from the *Temporal* perspective include:

- Technology maturity and obsolescence are currently considered independently in the technology life cycle. This is a key observation leading to the inference in the *Scientific* perspective to change the problem from 'technology readiness' to 'technology availability'.
- The technology life cycle which has been drawn in the form of the whale diagram shown in Figure 9.6 (Nolte, 2005).
- Once ready for use in products, technology is only readily available during the adulthood and maturity phases of the technology lifecycle.

14.1.2.1.9. *Scientific perspective*

Framing the problem, insights from the *Scientific* perspective infer:

- ***The undesirable situation*** is that while a single TRL number can provide information on the current maturity level of the technology, it cannot, and should not, be used to predict the maturity level of the technology at a future date.
- ***The FCFDS*** is that the decision-maker has a tool or methodology to determine if a specific technology will be available when needed for the duration of all categories of missions.
- ***The problem***: the inference from the *Scientific* perspective of the FCFDS is to restate the problem as "to create a tool or methodology (or both) to allow the project manager to determine if the technology will be available when needed for the duration of all categories of missions". Accordingly, the tool or methodology will need to take into account at least the following:
    - Time to advance maturity to a level suitable for use in the project which will depend on category of mission (single, one-of-a-kind, use or mass production).
    - The period of time in which the technology is available for use in products and systems before it becomes obsolete.
    - Obsolescence issues now considered separately as Diminishing Manufacturing Sources and Material Shortages (DMSMS).
- ***The solution***: to be determined. In the extended holistic problem-solving process, at least two solutions (tools and/or methodologies) would be conceptualized and a selection would be made to determine the most acceptable solution. For the purposes of this example, consider the conceptualization of one of those solutions the Technology Availability Window of Opportunity (TAWOO), discussed in Section 14.1.3.

This rephrasing of the problem statement has altered the scope of the problem in a significant way. It leads the project manager into going beyond systems thinking and using the *Temporal* perspective to consider:

- The rate of change of technology maturity during its development.
- The wider issues pertaining to the obsolescence of the technology after deployment.

14.1.3. The Technology Availability Window of Opportunity

The Technology Availability Window of Opportunity (TAWOO) is one conceptual solution to the problem of determining the availability of technology. Going beyond systems thinking, consider the TAWOO from the appropriate progressive and other perspectives.

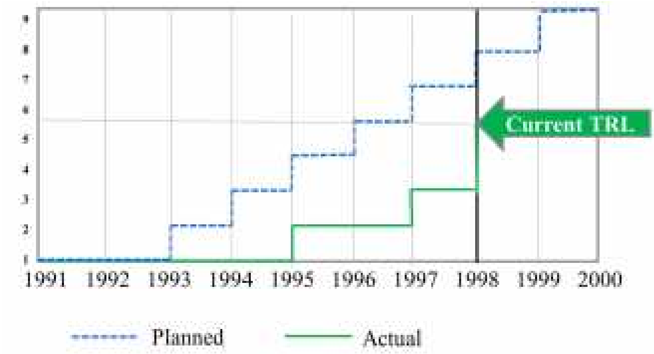


Figure 14.2 TRL 1991-2001

14.1.3.1. The Temporal perspective

According to Crépin et al., "*Although TRL is commonly used, it is not common for agencies and contractors to archive and make available data on the timeline to transition between TRLs*" (Crépin, et al., 2012). Perceptions from the *Temporal* perspective suggest that the data should be archived and used to estimate/predict maturity. If that data were available, one could infer from the *Scientific* perspective that one could consider the rate of change of TRL in a manner similar to Figure 14.1 such as shown in Figure 14.2. Figure 14.2 shows that the technology was conceptualized in 1991 and the development was planned to advance one TRL each year starting in 1993 for production in 1999. However, the development did not go according to plan. The technology did not get to TRL=2 until 1995 advancing to TRL=3 two years later in 1997 and jumping to TRL=6 in 1998. So can the technology be approved for a project due to go into service in 1999? It depends. If the project can use the technology when TRL=6, then yes. But, if the product using the technology is to go into mass production, the answer cannot be determined because there is insufficient information to predict when the technology will be at TRL=9. The project's decision-maker will have to obtain more information about the factors affecting the change in TRL.

14.1.3.2. The Generic perspective

Perceptions from the *Generic* perspective indicate that projects use Earned Value Analysis (EVA) and display budgeted/planned and actual cost information in graphs as the information changes over time as the project progresses through the states of the SDP.

14.1.3.3. The Scientific perspective

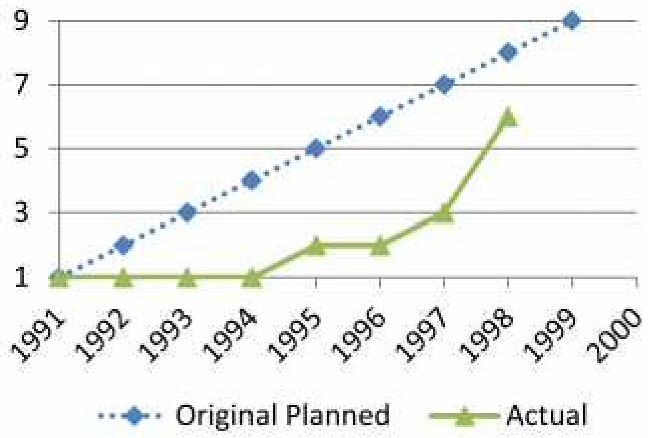


Figure 14.3 The dynamic TRL (dTRL)

When the rate of change of TRL is displayed in the form of an EVA graph as shown in Figure 14.3 instead of as in Figure 14.2, one additional significant item of information is obtained. Assuming nothing changes and progress continues at the same rate as in 1997-1998, the technology should reach TRL=9 by 1999. However, the reason for the rate of change between 1996-1997 and 1997-1998 is unknown. Determining the reason for the change provides the decision-maker with some initial questions to ask the technology developers before making the decision to adopt the technology. The static single value TRL has become a dynamic TRL (dTRL) (Kasser and Sen, 2013). The dTRL component would make adoption choices simpler. Prospective users of the technology could look at their need by date, the planned date for the technology to achieve TRL=9 and the past progress through the various TRLs. Then the prospective users could make an informed decision based on the graph in their version of Figure 14.3. If the rate of change projects that the desired TRL will not be achieved when needed and they really need the technology, they could investigate further and determine if they could help increase the rate of change of TRL so the technology will be available when needed.

Insight from the *Temporal* and *Generic* perspectives has conceptualized the use of a dTRL to help to predict when a technology will achieve a certain TRL. The need for a dTRL has been recognized in practice and there has been research into estimating the rate of change of technology maturity (El-Khoury, 2012). The dTRL concept was used for quite a few years the US aerospace and defense industry beginning in the Strategic Defense Initiative era (early 1990's) and took the form of waterfall charts that tracked the TRL (Benjamin, 2006).

Table 14.1 TAWOO Levels

TAWOO State	Level	Comments
Research and Development	1-8	Same as for TRL, but with a dynamic rate of change component.
Operational	9	Same as for TRL. Available for use in new products (in general).
Approaching obsolescence	10	Use in existing products but not in new products. Plan for replacement of products using the technology.
Obsolete	11	Some spares available, maintenance is feasible.
Antique	12	Few if any spares available in used equipment market. Phase out products or operate until spares are no longer available.

While the dTRL addresses the front end of the technology lifecycle, the issues pertaining to the other phases of the technology lifecycle may have to be addressed in a different manner. One framework might be the TAWOO levels as shown in Table 14.1 which extend the TRL through the whole technology lifecycle. However, should the dynamic aspect of levels 1-8 be overlooked, the TAWOO will become just as useless as the TRL for predicting the availability of the technology after deployment.

The TAWOO:

- Superimposed on the whale diagram (Nolte, 2005) is shown in Figure 14.4.
- Provides information about the availability of the technology in the remaining stages of the technology lifecycle.

14.1.4. Summary

Section 14.1 has provided an example of applying holistic thinking to the problem of predicting the availability of a technology for use in a system over its operational lifecycle and illustrated a number of improvements

Figure 14.4 TAWOO superimposed on the Whale diagram

over the traditional TRL approach namely:

- The holistic approach redraws the boundary of the problem posing different questions to those posed by the traditional approach. For example:
  a) The traditional question "what is the maturity of the technology?" produced the TRL.
  b) The question then changed to "when will the technology be ready for use?" and produced the dTRL.
  c) The resulting holistic (lifecycle) question "when will the technology be available for use?" produced the TAWOO which might predict when the technology will be ready as well as the length of time it would be available.
- Questions b) and c) were posed as the result of a change of perspective.
- The whole lifecycle solution came from a combination of the *Generic* and *Tempora*l perspectives, namely a result of going beyond systems thinking

The benefit of the holistic thinking approach in system analysis has also been shown in the case of the Royal Air Force (RAF) Battle of Britain Air Defence System that was used to foil the Luftwaffe's attempt to gain control of the sky over southern England in 1940 (Kasser, 2013c: pages 168-174). There the use of the HTPs identified two preventable failure modes in the system which unfortunately were only identified after the fact.

14.2. A Competency Model Maturity Framework (CMMF) for benchmarking the competency models of systems engineers

This Section introduces a two-dimensional Competency Model Maturity Framework (CMMF) for benchmarking different competency models including those discussed in Section 10.8.

14.2.1. The vertical dimension

The vertical dimension is based on three categories:

1. Knowledge.
2. Cognitive characteristics.
3. Individual traits.

14.2.1.1. Category 1: Knowledge

The knowledge category covers the application of systems engineering (Section 12.2) in the three domains, problem, solution and implementation discussed in Section 9.11.

The different camps of systems engineering discussed in Section 9.17 provide different opinions on what constitutes systems engineering; each opinion will have a different vision of the knowledge content. This was reflected in the different ways of assessing systems engineering proficiency discussed in Section 9.29. In addition, since systems engineers apply their skills in different domains (e.g. aerospace, land and marine transportation, information technology, defence, etc.), there is an assumption that to work in any specific domain, the systems engineer will need the appropriate problem, solution and implementation domain knowledge (Section 9.11).

Knowledge of the SEP and systems engineering tools is considered as part of systems engineering rather than the implementation domain.

14.2.1.2. Category 2: Cognitive characteristics

Cognitive characteristics, namely systems thinking and critical thinking provide the pure systems engineering (Section 12.2) problem identification and solving skills[1] to think, identify and tackle problems by solving, resolving, dissolving or absolving problems (Ackoff, 1999: page 115), in both the conceptual and physical domains. Perceived from the *Continuum* perspective, problem identification and solving competency is not the same thing as problem domain competency.

The approach to the assessment of systems thinking was is based on the HTPs discussed in Section 2.2.2.

A literature review showed that the problem of assessing the degree of critical thinking in students seemed to have already been solved several times over in different ways depending on the definition of critical thinking. Readers are advised to refer to Chapter 5 in Volume 1 of the series for more information (Kasser, 2013c: pages 123-141).

The approach selected for the CMMF is based on Wolcott and Gray's five levels of critical thinking (Section 9.5). Perceptions from the *Generic* perspective indicated that Wolcott's method for assessing a critical thinking level was very similar to that used by Biggs for assessing deep learning in the education domain (Biggs, 1999). Since a tailored version of the Biggs criteria had been used successfully at the University of South Australia for assessing student's work in postgraduate classes on systems engineering (Kasser, et al., 2005), Wolcott's method was adopted for the CMMF. Wolcott's five levels (from lowest to highest) are:

- Confused fact finder.
- Biased jumper.

[1] Problem solving and identification skills have been listed separately to map into Type I V and V as discussed below.

- Perpetual analyzer.
- Pragmatic performer.
- Strategic re-visioner.

Consider each of them.

14.2.1.2.1. *Confused fact finder*

A confused fact finder is a person who is characterised by the following:

- Looks for the "only" answer.
- Doesn't seem to "get it".
- Quotes inappropriately from textbooks.
- Provides illogical/contradictory arguments.
- Insists professor, the textbook, or other experts provide "correct" answers even to open-ended problems.

14.2.1.2.2. *Biased jumper*

A biased jumper is a person whose opinions are not influenced by facts. This person is characterised by the following:

- Jumps to conclusions.
- Does not recognise own biases; accuses others of being biased.
- Stacks up evidence for own position; ignores contradictory evidence.
- Uses arguments for own position.
- Uses arguments against others.
- Equates unsupported personal opinion with other forms of evidence.
- Acknowledges multiple viewpoints but cannot adequately address a problem from viewpoint other than their own.

14.2.1.2.3. *Perpetual analyzer*

A perpetual analyser is a person who can easily end up in "analysis paralysis". This person is characterised by the following:

- Does not reach or adequately defend a solution.
- Exhibits strong analysis skill, but appears to be "wishy-washy".
- Write papers that are too long and seem to ramble.
- Doesn't want to stop analysing.

14.2.1.2.4. *Pragmatic performer*

A pragmatic performer is a person who is characterised by the following:

- Objectively considers alternatives before reaching conclusions.

- Focuses on pragmatic solutions.
- Incorporates others in the decision process and/or implementation.
- Views task as finished when a solution /decision is reached.
- Gives insufficient attention to limitations, changing conditions, and strategic issues.
- Sometimes comes across as a "biased jumper", but reveals more complex thinking when prompted.

14.2.1.2.5. *Strategic revisioner*

A strategic revisioner is a person who is characterised by the following:

- Seeks continuous improvement/lifelong learning.
- More likely than others to think "out of the box".
- Anticipates change.
- Works toward construction knowledge over time.

14.2.1.3. Category 3: Individual traits

These are the traits providing the skills to communicate with, work with, lead and influence other people, ethics, integrity, etc. These traits include communications, personal relationships, team playing, influencing, negotiating, self-learning, establishing trust, managing, leading, emotional intelligence (Goleman, 1995), and more (Covey, 1989; Frank, 2010; ETA, 2010). These traits may be selected to suit the role of the systems engineer in the organisation and assessed in the way that the ETA industry standard competency models assess those traits (ETA, 2010). There is no need to reinvent an assessment approach.

14.2.2. The horizontal dimension

The horizontal dimension provides a way to assess the competence of a person in each broad area of the vertical dimension against the levels of increasing ability. The five types of systems engineer discussed in Section 10.9 form the horizontal axis.

Table 14.2 A Competency Model Maturity Framework (CMMF) for Systems Engineers

	Type I	Type II	Type III	Type IV	Type V
Category 1: Knowledge areas					
Systems engineering	Declarative	Procedural	Conditional	Conditional	Conditional
Problem domain	Declarative	Declarative	Conditional	Conditional	Conditional
Implementation domain	Declarative	Declarative	Conditional	Conditional	Conditional
Solution domain	Declarative	Declarative	Conditional	Conditional	Conditional
Category 2: Cognitive characteristics					
Systems Thinking					
Descriptive (8)	Declarative	Procedural	Conditional	Conditional	Conditional
Prescriptive (1)	No	No	Procedural	No	Conditional
Critical Thinking	Confused fact finder	Perpetual ana-lyser	Pragmatic performer	Pragmatic performer	Strategic re-visioner
Category 3: Individual traits (sample)					
Communications	Needed	Needed	Needed	Needed	Needed
Management	Not needed	Needed	Needed	Needed	Needed
Leadership	Not needed	Not needed	Needed	Needed	Needed
Others (specific to situation)	Organization specific	Organization specific	Organization specific	Organization specific	Organization specific

Table 14.3 Comparison of competency models

Assessment approach	**KSA's**	**INCOSE CSEP Exam**	**SECF**	**CEST**	**SECT**	**NASA 2010**	**JPL SEA**	**MITRE**	**NDIA**
Category 1: Knowledge									
Systems Engineering	Yes [1]	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Solution domain	Implied	No	No	No	Yes	Yes	Yes	Yes	No
Implementation domain [3]	Implied	No	Partial [4]	No	No	Partial [4]	Partial [4]	Partial [4]	No
Problem domain	Implied	No	No	Yes	Yes	Yes	Implied	Yes	Yes
Category 2: Cognitive skills	Yes	No	Yes	Yes	Yes	Implied	Yes	Yes	Yes
Category 3: Individual Traits	Yes	No	Unclear	Yes	Yes	Yes	Yes	Yes	Yes
Increasing levels of proficiency	No	No (Pass/Fail)	Partial [2]	No	Yes	Yes	No	Yes	No

14.2.3. The two-dimensional framework

A two-dimensional CMMF showing the assessment of the competency in increasing levels of competency (Type I to Type V) in the three categories discussed in Section 14.2.1 is summarised in Table 14.2. Assessment of knowledge, cognitive skills and individual traits is made in ways already practiced in the psychology domain and do not need to reinvented by systems engineers. Where knowledge is required at the conditional level, it includes procedural and declarative (Section 9.5). Similarly, where knowledge is required at the procedural level, it includes declarative knowledge.

14.2.4. Benchmarking the nine competency models

Each of the three categories contains some competencies that are common to all systems engineers and some competencies that apply to specific roles in specific domains in specific phases of the SLC in specific organisations. Each competency model thus contains information which can be allocated into these three categories and allows them to be subjectively compared or benchmarked as shown in Table 14.3[1]. Findings from this comparison, based on the published literature, include:

- The number of levels of proficiency differs between competency models.
- The definition of the ability for a level of proficiency differs between competency models.
- The lack of competencies in the implementation domain in all nine competency models examined. However, it is fair to say that some of the models do consider the culture in which the systems engineering is taking place.

---

[1] Notes in Table 14.3:

In several instances, the various ways in which the competency models describe the competencies made populating this table a subjective exercise.

The use of the word 'Yes' should be read with the understanding that each competency model identifies a different set of knowledge in each of the knowledge area rows in the table.

[1] Subjective approach, knowledge seems to be dependent on situation, no objective reference for validating characteristics.

[2] Lowest level is in a different dimension to remaining levels

[3] Systems engineering tools have been allocated to the systems engineering knowledge area rather than to implementation domain.

[4] Does contain knowledge about the culture of the organisation is which the systems engineering is taking place.

14.2.5. Using the CMMF as a competency model

In order for an organization to use the CMMF, the contents of each of the three categories must be determined and the CMMF populated. If the organization already has a competency model then the competencies need to be transferred from the organisations' competency model into the appropriate areas in the CMMF. If the organization does not have a competency model and wishes to develop one, then the CMMF allows standardization of groupings which helps to identify both errors of commission and errors of omission. However, before developing a competency model, a cost-benefit trade-off should be performed since the amount of effort will depend on the level of detail required. The development effort should be for a model that will be useful, not something that will keep the human resource department busy.

Candidates must qualify at the appropriate proficiency level in all three categories to be recognised as being competent at that competency level. While examination questions can require the respondent to use conditional knowledge, the successful application of conditional knowledge in the real world must be directly demonstrated by results documented in the form of KSAs supported by awards, letters and certificates of appreciation from third parties (e.g. employers, customers, etc.). The requirement for supporting documentation overcomes the current deficiency in the KSA. The assessment could thus be in two parts, one part by examination for the lower levels, the second by a portfolio demonstrating successful experience for the higher levels. Perceptions form the *Generic* perspective show that the portfolio model is already used by professional societies including:

- The INCOSE Expert Systems Engineering Professional (ESEP).
- The Certified Member of the Association for Learning Technology (ALT) (CMALT) awarded by the ALT
- Fellow of the Institution of Engineering and Technology (IET) (FIET) awarded by the IET in the UK.

Assessment of a candidate is simple in concept as follows.

- ***The cognitive skills and individual traits:*** Knowledge of the systems thinking perspectives is assessed as declarative, procedural and conditional as discussed in Section 9.5. Ways of assessing the degree of critical thinking as declarative, procedural and conditional have been described by Wolcott and Gray (Section 14.2.1.2) and are used in the CMMF. The appropriate individual traits are assessed as being 'needed' or 'not needed' at a specific level of ability.

- ***The systems engineering, implementation, problem and solution domain knowledge:*** The knowledge is also assessed as being declarative, procedural and conditional (Section 9.5). The question then arises as what is the knowledge to be? Consensus on the contents of a 'standard' SEBOK would be difficult to achieve across organisations and domains if it were to be based on the role of the systems engineer (SETR). That the knowledge competency is situational rather than generic does not stop the CMMF being populated by organisations needing competency assessments for <u>their</u> personnel working in their environment on <u>their</u> projects.

14.3. Using the principle of hierarchies to manage complexity

The principle of hierarchies in systems discussed in Section 7.2 is one of the ways humanity has managed complexity for most of its recorded history. Using the principle of hierarchies as a tool to manage complexity means:

- Keeping the systems and subsystems at the same appropriate level in the hierarchy of systems.
- Abstracting out or hiding the internal components of systems

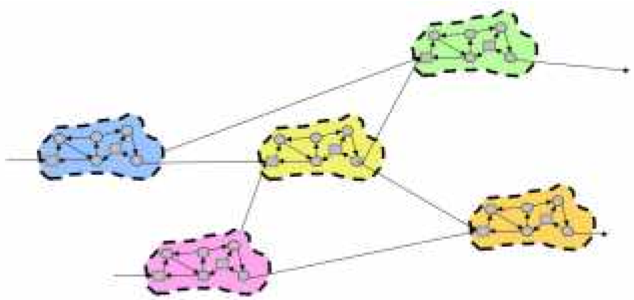


Figure 14.5 Situation partitioned into systems (complex view)

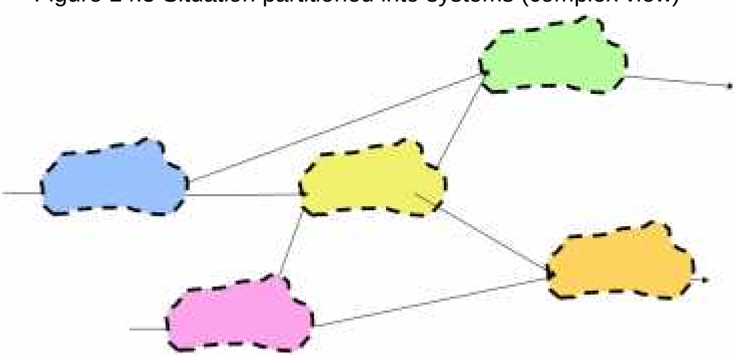


Figure 14.6 Situation partitioned into abstracted subsystems

and subsystems in any one view or figure. Maier and Rechtin recommend that the way to deal with high levels of complexity is to abstract the system at a high a level as possible and then progressively reduce the level of abstraction (Maier and Rechtin, 2000: page 6). Consequently, for example, systems engineers should never use drawings such as Figure 14.5 which creates artificial complexity (Section 9.27) by showing the internal components of a subsystem in the system level drawing and should always use the type of drawing shown in Figure 14.6 instead.

- Recognising the concept that one systems engineer's subsystem is another systems engineer's system. For example:
  - An Air Defence System (ADS) is a system as far as the ADS systems engineer is concerned but it is a subsystem within the containing or metasystem, the National Defence System.
  - A missile battery is a subsystem of the ADS, but is the system as far as the missile battery systems engineer is concerned.
  - A missile is a subsystem of the missile battery, but is the system as far as the missile systems engineer is concerned.
  - The radar is a subsystem of the missile battery, but is the system as far as the radar systems engineer is concerned.
- Recognising the concept that a situation[2] contains a number of systems. Each system may contain a number of subsystems. Each subsystem may be further elaborated into a number of components (subsystems of the subsystem). This concept is often shown in the traditional hierarchical structure such as in organisation charts, work breakdown structures and product breakdown structures.

14.4. The Hitchins-Kasser-Massie Framework

The Hitchins-Kasser-Massie Framework (HKMF) (Kasser, et al., 2001) shown in Figure 14.7[3] was developed when trying to determine the requirements for what should be taught in postgraduate systems engineering coursework at UniSA. The research attempted to develop a SEBOK based on the role of the systems engineers in the SETR paradigm in the different states of the SLC (Section 9.12) and in the different Hitchins' layer (Section 9.8). In its early days, the framework has:

- Provided one of the reasons why systems engineers can't agree on the nature of systems engineering discussed in Section 12.1.

[2] Sometimes also known as a meta-system or a containing system

[3] Revised into the format shown in the figure by Xuan Linh Tran.

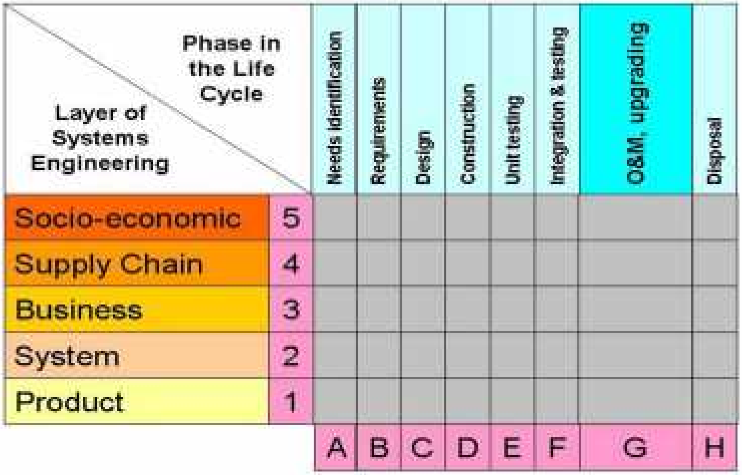


Figure 14.7 The HKMF for understanding systems engineering

- Identified the 'A' and 'B' paradigms in systems engineering discussed in Section 9.21.
- Identified that systems engineers operating in one layer use a different vocabulary to those operating in another layer. For example, the term "capability" has different meanings in Layers 1 and 3.
- The multitude of definitions of systems engineering some of which were quoted in Section 9.1 are based on internal perspectives from the different areas of the HKMF.
- Identified that systems engineers perform different functions in the different areas of the HKMF.
- Facilitated traceability of requirements; requirements on a system in Layer 2 can be traced back to the undesirable socio-economic situation in Layer 5.
- Shown that systems engineering, at least the INCOSE version, resides in Layer 2 while Operations Research resides in Layer 3, mainly in area 3G.
- Systems engineering (SETA) is performed in Layer 5 where it is known as Political Science.

The two dimensions of the framework plot the system or product layer of complexity and process (lifecycle) state on different axes where:

- ***The vertical or product axis*** is the five layers of systems engineering (Hitchins, 2000) discussed in Section 9.8.
- ***The horizontal or timeline axis*** is the eight states of the SLC

discussed in Section 9.12.

The idea for the HKMF came from the *Generic* perspective. Mendeleev created a framework, the Periodic Table of Elements, and populated it with the known elements, leaving gaps which represented unknown elements. in a similar manner, the HKMF forms a framework for studying activities in the workplace in the different layers and states of the SLC.

14.5. A problem formulation template

The following four-part problem formulation template based on the extended holistic problem-solving process can assist the problem-solving process.

1. ***The undesirable situation*** as perceived from the each of the descriptive HTPs.
2. ***The FCFDS*** as inferred from the descriptive HTPs (the *Scientific* perspective).
3. ***The problem,*** which is how to convert the FCFDS into reality (the *Scientific* perspective).
4. ***The solution*** which is something that remedies the undesirable situation and has to be interoperable with evolving adjacent systems over the operational life of the solution and adjacent systems (the *Scientific* perspective). In non-complex systems the solution is often the FCFDS. The solution is made of two interdependent parts:

   a) The SDP or transition process that converts the undesirable situation to a desirable situation.
   b) The solution system operating in the context of the FCFDS.

Placing the solution before the problem is based on the dictum of working back from the answer (Ackoff, 1999) and allows risk management to be incorporated into task planning instead being an add-on in the current systems engineering and project management paradigms. The risk management is achieved by ensuring that risks identified in a task are mitigated or prevented in earlier tasks in the project schedule.

14.5.1. Framing classroom exercises using the problem formulation template

This section provides an example of a generic problem formulation template for framing classroom exercises as follows:

- ***The undesirable situation*** is the need to successfully[4] complete

[4] Success is defined by the desired grade.

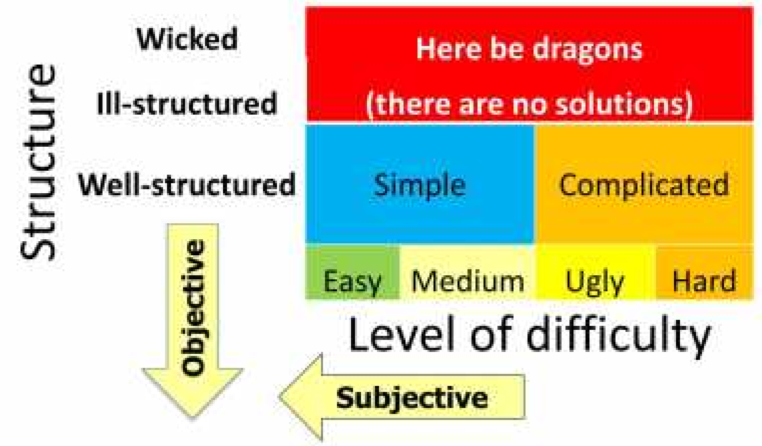


Figure 14.8 A problem classification framework

the exercise in a timely manner.

- ***The FCFDS*** is having successfully completed the exercise in a timely manner
- ***The problem*** is to figure out how to create and deliver a product that meets the requirements of the exercise.
- ***The solution*** is to create and deliver a product that meets the requirements of the exercise.

Students should be provided with the opportunity to practice using the template by framing the problem posed by the specific exercise or assignment by adapting the generic problem formulation template to their situation.

14.5.2. The benefits of using the problem formulation template

The benefits include being forced to think about the situation. In working out the steps of what to do to remedy the problem by providing the solution, students will be forced to plan their work. Accordingly, the template assists in building in best practices by building planning ahead into student projects. And of course it is just as suitable in the real world of systems engineering.

14.6. A problem classification framework

Problem-solving is at the heart of both systems engineering and project management and identifying the correct problem is one of the most important activities in systems engineering and project management. This framework for classifying problems (Kasser, 2012a) shown in Figure 14.8 is based on distinguishing between subjective and objective complexity (*Continuum* perspective) where the axes are:

- ***Level of difficulty:*** (subjective complexity) discussed in Section 10.12.
- ***Structure of the problem:*** (objective complexity) discussed in Section 7.6.

Different people may position the same problem in different places in the framework. This is because as knowledge is gained from research, education and experience a person can reclassify the subjective difficulty of a problem down the subjectivity continuum from 'hard' towards 'easy'. As discussed in Section 12.12.1, there are no solutions to ill-structured and Wicked problems; they must be converted to well-structured problems before the remedial part of the problem-solving process can begin.

14.7. Summary

This Chapter is a continuation of the *Scientific* perspective and contained a further selection of insights, suggestions, tools and frameworks for improving the practice of systems engineering which have been conceptualised, prototyped and found to be useful. They were:

- The TAWOO as a way of predicting technology availability.
- A CMMF for:
  - Comparing different competency models.
  - Assessing competency models for suitability for an organisation.
  - Use as a competency model.
- Using the principle of hierarchies to manage complexity.
- The HKMF.
- A problem formulation template.
- A problem classification framework.
- There are no solutions to ill-structured and Wicked problems. They have to be converted to well-structured problems.

--oo--

15. Seven principles for systems engineered solution systems

This Chapter[1] contributes to the improvement of systems engineering by suggesting an axiom that applies to the finished product, the system, irrespective of which systems engineering camp (Section 9.17) is producing the solution system. This approach avoids presenting principles specific to particular camps that may not be accepted by systems engineers in the other camps.

The reason for presenting this axiom is because systems engineering is presently demonstrating the characteristics of being in the emerging stages of a discipline as discussed in Section 12.18. As perceived from the *Temporal* perspective, a discipline generally matures when an overriding axiom is presented and accepted by the majority of practitioners.

Hitchins attributed the success of systems engineering in the NASA environment in the 1960's and 1970's to a set of eight principles (Hitchins, 2007: page 85). However, those principles applied in an environment where NASA's mission needs did not change very much during the SDP for each mission. Today's systems on the other hand, tend to be developed and exist in an environment where the needs change rapidly, sometimes even before the solution system is delivered.

This Chapter now introduces a set of updated principles for today's environment so that systems engineers working in different domains using various tools, techniques and methodologies, can meet the objective of systems engineering. The set of principles to the solution system they are realizing is:

1. There shall be a clear, singular objective or goal.
2. There shall be a CONOPS from start to finish of the mission describing the normal and contingency mission functions as well as the normal and contingency support functions performed by the solution system that remedies the problem.
3. The solution system shall be designed to perform the complete set of remedial mission and support functions for the operational life of the system.

[1] The chapter is a modified version of (Kasser and Hitchins, 2011; Kasser, 2013c: pages 427-437).

4. The solution system design may be partitioned into complementary, interacting subsystems.
5. Each subsystem is a system in its own right, and shall have its own clear CONOPS, derived from, and compatible with, the CONOPS for the whole.
6. Each subsystem may be developed independently and in parallel with the other subsystems provided that fit, form, function and interfaces are maintained throughout.
7. Upon successful integration of the subsystems, the whole solution system shall be subject to appropriate tests and trials, real and simulated, that expose it to extremes of environment and hazards such as might be experienced during the mission.

Consider each of these principles.

15.1. There shall be a clear, singular objective or goal

Principle 1: There shall be a clear, singular objective or goal.

The task of the systems engineer shall have a clear singular objective goal[2]. In the concept definition stage of a systems acquisition, this goal may be to identify the underlying problem or root cause of a situation, and to conceive one or more potential solutions. In the later states of the solution SDP[3] the goal is generally to realize a solution system that remedies the problem. For example, in the 1960's the NASA goal was to put a man on the moon and return him safely to earth by the end of the decade.

15.2. There shall be a clear CONOPS from start to finish of the mission …

Principle 2: There shall be a CONOPS from start to finish of the mission describing the normal and contingency mission functions as well as the normal and contingency support functions performed by the solution system that remedies the problem.

The CONOPS documents, or is a repository of, the information pertaining to the normal and contingency mission and support[4] performance of the overall solution system. One way of grouping the complete set of functions performed by any system is into the following classes:

---

[2] A well-structured problem.

[3] The notion that the solution system is generally a technological system that needs to be developed, and hence the name system development process, seems to be DoD inspired. Essentially, there need not be any (technological) development; instead, solution systems can be synthesized by bringing together existing systems to create a new unitary whole.

[4] The repeated use of "*normal and contingency mission and support*" is to emphasize the holistic approach.

- ***Mission:*** the functions which the system is designed to perform to provide a solution to the problem as and when required.
- ***Support:*** the functions the system needs to perform in order to be able to perform the mission as and when required. Support functions can further be grouped into (Hitchins, 2007: pages 128-129):
    - ***Resource management functions:*** the functions that acquire, store, distribute, convert and discard excess resources that are utilized in performing the mission.
    - ***Viability management functions:*** the functions that maintain and contribute to the survival of the system in storage, standby and in operation performing the mission.

Part of the CONOPS performs risk management by considering the consequences of failures of parts of the system to perform their mission and support functions and the contingency functions to be invoked in the event of these failures. The contingency functions may be in the process and consist of activities that will attempt to prevent the failure, or may be in the solution system in the form of viability functions.

The CONOPS is the foundation document[5] for both the solution system and the rest of the system realization activities since the remaining work in the SDP realizes the solution system by converting the mission and support functions described in the CONOPS into a real system. Application of this principle leads to a holistic system development approach ensuring that all pertinent mission and support functions, such as operational availability, logistics, human operations, threat neutralizations, etc. are included in the system up-front in an integrated holistic manner and not as a bolt-on after the fact.

A clear vision of the solution system anticipates, and consequently prevents, subsequent activities that try to clarify the original customer's problem represented by a set of poor requirements. The consequences of not having a CONOPS are shown in Figure 12.6[6]. I found this drawing in 1970 and it was old then. It has evolved somewhat in the intervening 40 years but the message it contains has not changed.

The CONOPS can also serve as a model of the solution and be incorporated in a simulation to allow various stakeholders to gain a better understanding of the problem space and determine if, and how well, the conceptual system being modelled could remedy the problem should that conceptual solution system be realized.

---

[5] The word 'document' is used herein to represent information, not a necessarily a paper document.

[6] While it is often used to depict the "systems engineering process", it really shows a lack of communications or common vision of what the customer wants by the stakeholders.

15.3. The solution system design …

Principle 3: The solution system shall be designed to perform the complete set of remedial mission and support functions for the operational life of the system.

The application of this principle produces a solution system that performs the mission and support functions described in the CONOPS over the complete lifecycle of the solution system. The solution system does not have to be technological or even a new acquisition. The solution system lies somewhere along a continuum that stretches from 'fully automatic technological' to 'manual with no technology'; and may be a modification of an existing system, a change to an existing process, tactics, doctrine, policy, or training or some combination. However, when applied to technological solution systems, this principle helps to ensure that the effects of component obsolescence, Diminishing Manufacturing Sources and Material Shortages (DMSMS), logistics, reliability, maintainability, the human element and other pertinent factors currently considered somewhat independently are considered interdependently in a holistic interdisciplinary manner from conception. Further, if the solution system is designed to perform in a hazardous or threatening context, then the solution system shall incorporate support functions to counter threats and to manage risks.

15.3.1. Coping with change is a design criterion

This principle takes into account changes in/to the need/problem at any point in the SDP. For example, in NASA's Apollo program, the need (and hence the requirements) did not change during the SDP, and the operational life of each iteration of the manned element of the system was short; measurable in days. Each Apollo Lunar Surface Experiments Package (ALSEP)[7] however had a much longer life span.

Other early successful systems engineering projects such as the transcontinental US television microwave relay system (Hall, 1962) were also not subject to changing needs. However, today's solution system creation and realization process must be able to cope with changes in the needs before the solution system is delivered, and the solution system itself needs be realized in such a manner that upgrades reflecting changing needs during the O&M State of its SLC can be incorporated without major perturbations.

---

[7] A set of scientific instruments deployed at the landing sites designed to operate for a year. Each ALSEP contained the same central station and a slightly different set of scientific instruments.

15.3.2. Cost is not an initial design criterion

According to this principle, the cost-effectiveness of the solution system is not a design criterion at least as far as the prototype or initial version is concerned. Once the prototype is shown to meet the needs, then costs may become an issue if the prototype is not affordable. Henry Ford wrote, "*Our policy is to reduce the price, extend the operations and improve the article. You will notice that the reduction of price comes first. We have never considered costs as fixed. Therefore we first reduce the price to a point where we believe more sales will result. Then we go ahead and try to make the price. We do not bother about the costs. The new price forces the costs down. The more usual way is to take the costs and then determine the price, and although that method may be scientific in the narrow sense, it is not scientific in the broad sense because what earthly use is it to know the cost if it tells you that you cannot manufacture at a price at which the article can be sold?*" (Ford and Crowther, 1922: page 146). It is a question of perspective and asking the right question. The usual non-holistic thinking question was "what does it cost to produce X?" From the *Continuum* perspective, the alternative (out-of-the-box) question that changes the problem was "how can X be produced for $Y?"

NASA's Apollo programme was more concerned with doing the job (meeting the goal of placing a man on the moon by the end of the 1960's) rather than doing it efficiently – money was not an issue in the initial design phase. When the systems engineer designs each of the solution system options, cost and schedule must not be an issue. Cost and schedule considerations may be used as selection criteria for choosing the desired solution system option after the solution system options have been designed as shown in blocks 4, 5 and 6 of Figure 6.3. In addition, systems engineers should be involved in any adjustments to the scope of the solution system realization project to fit the constraints of cost and schedule.

15.4. The solution system design partitioning …

Principle 4: The solution system design may be partitioned into complementary, interacting subsystems.

The solution that remedies the problem is the sum of the mission and support functions performed by the solution system and the functions performed in the realization process (Hall, 1989). Consider:

- Partitioning the product or solution system.
- Partitioning the production process.

15.4.1. Partitioning the product or solution system

The systems engineers design the solution system so that the desired functionality emerges from the complete design. For example, the performance of NASA's Apollo Moon Mission was emergent, coming as it did from the cooperation and coordination of the Saturn V launcher, the command module, the mission crew, the lunar excursion module, the telecommunications subsystem, mission control subsystem, etc. Performance is emergent because these various subsystems of the whole are of dissimilar nature, yet cooperate and coordinate their different functions and actions. So, you cannot point to any one subsystem and say, *'the performance was down to that one'*. All parts contributed, all cooperated and coordinated their actions.

15.4.2. Partitioning the production process

The systems engineers and project managers also architect the activities that will constitute the realization process as interdependent streams of activities between milestones (Section 9.8).

15.5. Each subsystem is a system in its own right …

Principle 5: Each subsystem is a system in its own right, and shall have its own clear CONOPS, derived from, and compatible with, the CONOPS for the whole.

This principle reflects the perception that systems exist within containing systems from the *Structural* perspective (Section 7.2). The principle has often been stated as, "*one person's system is another person's subsystem*". Hierarchies are fundamental to nature (Section 14.3).

As an example consider an allied naval convoy crossing the North Atlantic Ocean in 1942. The convoy is a system[8]. Each ship in the convoy can be considered as both a subsystem of the convoy, or as a system[9]. There was a CONOPS for the convoy. There were separate CONOPS for the naval escort ships and the merchant vessels describing the actions and interactions of these subsystems of the convoy in various scenarios.

15.6. Each subsystem may be developed independently and in parallel …

Principle 6: Each subsystem may be developed independently and in parallel with the other subsystems provided that fit, form, function and interfaces are maintained throughout.

---

[8] Some people might call it a System of Systems.

[9] Alternatively, the naval ships could be one subsystem and the merchant marine ships a second subsystem of the convoy. Each ship is then a subsystem within the naval or civilian subsystem of the convoy. If there are ships from the navies of more than one allied country in the convoy, then the ships of each country could constitute a subsystem within the naval subsystem. The choice of subsystem partitioning depends on the issues being considered.

Each subsystem, being a system, needs its own systems engineers who conceive, design and develop their system as an interacting part of the containing system. These system systems engineers face in two directions – upwards and outwards into the containing system, to ensure ongoing compatibility with the containing system and its CONOPS, including all of the other interacting subsystems at the same level in the hierarchy; and downwards, into the intra-acting sub-subsystems within their own system. The downward task of developing the subsystems (function) can be considered as engineering when the focus is on the system as an independent entity.

As discussed in Section 9.12, during the realization states of the SDP when the subsystems are being developed in parallel, the systems engineering activities are those that focus on the subsystem as a part of the complete system and ensure that fit, form and interfaces are maintained. If the SDP takes a long time, the effect of changes in the need on the subsystem realization has to be taken into account. Experience has shown that subsystem designs and development may be subject to "requirements creep". Consequently, it is necessary to have budgets for the whole system, as well as budgets for each of the subsystems - for instance the weight budget was important to Apollo, as was a failure rate budget. It would not have done for the failure rate for one subsystem - say the capsule - to go off the scale! Technical budgets have become known as 'Technical Performance Measures' (TPM). This is what conceiving, designing and developing the subsystem independently but within the context of the whole and the other interacting subsystems means.

15.7. Upon successful integration of the subsystems...

Principle 7: Upon successful integration of the subsystems, the whole solution system shall be subject to appropriate tests and trials, real and simulated, that expose it to extremes of environment and hazards such as might be experienced during the mission.

This principle minimizes situations in which solution systems are delivered that are not fit for purpose and do not provide a solution in the intended environment.

The consequences of not implementing this principle can be seen in the increasing stove-piping of the processes in the SDP and expansion of the various disciplines.

15.8. Discussion

Poor systems engineering has been blamed for system acquisition failures[10] (Section 4.1). An objective view might suggest that budget and

[10] Defined herein as cost and schedule overruns, cancellations and delivered systems that are not fit for purpose.

time overruns smack of either poor estimating of cost and schedules or understating the real estimates for reasons that appeared valid at the time. However, in all fairness, poor early state systems engineering does seem to have been a contributor to some of those failures resulting from producing solutions systems that do not remedy the need when deployed. Attempts in the later states of the SDP to mitigate the effects of poor systems engineering in the early states of the SDP have resulted in system development becoming increasingly technologically focused, excessively complicated and stove-piped into independent streams of activities including:

- Systems Engineering.
- Project Management.
- Lifecycle Costing or Total Cost of Ownership (TCO).
- Performance Based Logistics (PBL).
- Integrated Logistics Support (ILS).
- Maintenance Management.
- Supply Chain Management.
- Technical Training Management.
- Technical Data Management.
- Configuration Management (CM).
- Risk Management.
- Independent Verification and Validation (IV&V).
- Human Systems Integration.

These are but some examples of the independent streams of activities in the various specialties in the SDP. Not only is this stove-piping against the holistic concept of systems engineering, stove-piping produces overlapping activities, confusion, and unnecessary expense and also provides a breeding ground for turf wars in organizations. The documentation overhead is increasingly becoming expensive and documents that should be interdependent are independent being produced because of legislation rather than as a result of actual need. As of 1988, DoD systems engineering 'B' paradigm has added so many bolt-on's to compensate for having removed the front end of systems engineering that it has become expensive and unworkable (Costello, 1988). Reversion to the original holistic 'A' *Weltanschauung* (world view or paradigm) is long overdue since the Costello Report was published, the situation has worsened.

15.9. Summary

Systems engineering is presently demonstrating the characteristics of being in the emerging stages of a discipline. A discipline generally matures when an overriding axiom is presented and accepted by the majority of practitioners. This Chapter presented one such underpinning axiom for systems engineering. The principles within the axiom apply to the solution system, production of which is the common goal of all the camps within systems engineering. As a consequence, the axiom has the potential to improve systems engineering by uniting the disparate camps within systems engineering by allowing them to agree on the principles applying to the solution system which will then enable the practice of systems engineering to repeat the successes it achieved in the NASA environment in the 1960's and 1970's in all current and future application domains.

15.10. Conclusion

The principles presented in this Chapter apply to the solution system being systems engineered rather than to systems engineering. As such the axiom has the potential to unite the disparate camps within systems engineering by allowing them to agree on the principles. Applying these principles to the solution system will then enable the practice of systems engineering to repeat the successes it achieved in the NASA environment in the 1960's and 1970's in all current and future application domains.

--oo--

PART IV

Part IV contributes to the improvement of systems engineering by using the insights, inferences and conclusions from the *Scientific* perspective in Part III to suggest more tools and frameworks for improving systems engineering. Where the insight leads to a complex tool or concept, that complex tool or concept is presented in a separate chapter. As such:

- Chapter 16 improves systems engineering by introducing the Nine-System Model. Note the Nine-System Model is not a model of systems engineering, it is a framework and tool.
- Chapter 17 improves systems engineering by describing how to manage stakeholder expectations using a combination of the HTPs to identify the stakeholders, and the Nine-System Model to identify the stakeholders' areas of concern in the context of a Case Study.
- Chapter 18 improves systems engineering by filling a gap in the systems engineering literature by suggesting a process for creating a system to be used in the early states of the SDP to help to manage complexity at the time the system is created by optimizing the interfaces.
- Chapter 19 improves systems engineering by providing a way to measure technical progress and identify potential problems in near real-time so as to mitigate the problems before they occur.

--oo--

16. The Nine-System Model

The Nine-System Model (Kasser and Zhao, 2014):

- Is not a model of systems engineering. It is a framework for perceiving where the parts of systems engineering are performed and how they fit together as well as a tool for use by systems engineers.
- Provides a way to manage complexity when creating the system as discussed in Chapter 18.
- Is based on the problem-solving approach to systems engineering in accordance with IEEE 1220 which stated, *"the systems engineering process is a generic problem-solving process"* (IEEE 1220, 1998) Section 4.1).
- Maps into the extended holistic problem-solving process shown in Figure 6.5 annotated as shown in Figure 16.1.
- Manages complexity by abstracting out all information about the SOI that is not pertinent to the issue at hand (Kasser, et al., 2014).
- Is an application of the theory that complexity can be managed (but not reduced) by applying a set of rules for grouping/aggregation/synthesis.
- Is a self-similar framework model usable in any level of the hierarchy.

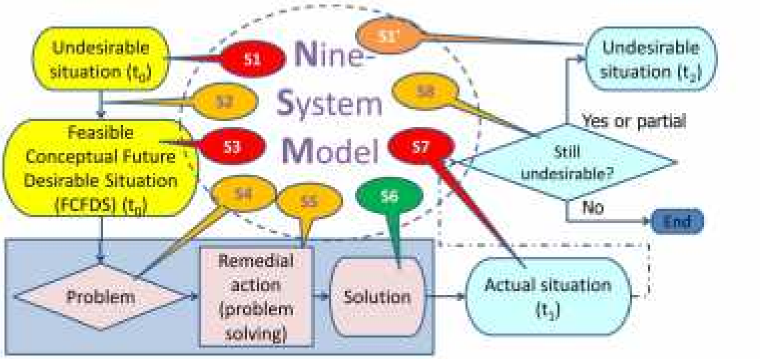


Figure 16.1 Mapping the nine systems to the extended holistic problem-solving process

- Encompasses aspects of the Seven Samurai (Martin, 2004), BPR, Checkland's SSM (Checkland and Scholes, 1990), Hitchins's approach to systems engineering (Hitchins, 2007) and the SIMILAR process (Bahill and Gissing, 1998).
- Incorporates much of the content of the MIL-STD-499 (MIL-STD-499A, 1974), EIA 632 (EIA 632, 1994) and IEEE 1220 (IEEE 1220, 1998) Standards as shown in Table 16.1.
- Incorporates the seven principles for systems engineered solution systems (Chapter 15).
- Provides a template incorporating built-in best practices that conform to the 'A' paradigm of systems engineering discussed in Section 9.21.1.
- Is a conceptual model since as the *Temporal* perspective shows, all the systems do not coexist at the same time.
- Comprises the following nine situations, processes and socio-technical systems in a clearly defined interdependent manner:

  S1. The undesirable or problematic ***situation.***
  S2. The ***process*** to create the FCFDS.
  S3. The FCFDS that remedies the undesirable ***situation.***
  S4. The ***process*** to plan the transition from the *undesirable or problematic situation* (S1) to the FCFDS (S3).
  S5. The ***process*** to perform the transition from the *undesirable or problematic situation* (S1) to the *FCFDS* (S3) by providing the *solution system* (S6) according to the plan developed in the *planning process* (S4). S5 could be the SDP or an acquisition process if a suitable COTS system is available.
  S6. The solution ***system*** that will operate within FCFDS[1].

---

[1] The adjacent and supporting systems are considered as either subsystems or adjacent systems of the *solution system* (S6). If they are:

- ***Subsystems:*** they are purview of the systems engineer of *solution system* (S6) in the same manner as any other subsystem and can be seen in the *Structural* and *Functional* perspectives of the *solution system* (S6).
- ***Adjacent systems:*** they show up in the *Big Picture* perspective of the *solution system* (S6); their operational interactions and interfaces are seen in the *Operational* perspective of the *solution system* (S6). However, since S6 and the adjacent systems are subsystems of the metasystem operating in S7, the specification of the nature of the adjacent systems are the purview of the system engineer of that metasystem in the same way as the specification of the nature of the subsystems of S6 is the purview of the system engineer of the *solution system* (S6).

S7. The actual or created ***situation***. S3 evolves into S7 during the time taken to perform S4 and S5.

S8. The process to determine that the realized solution system (S6 operating in the context of S7) remedies the evolved *undesirable situation.*

S9. The ***organization***(s) containing the processes and providing the resources for the operation and maintenance of the processes. S9 is also often known as the Enterprise. Each organisation can be perceived as comprising two major subsystems:

a. *The production (mission) subsystem* which produces the products from which the organisation makes its profits.
b. *The support subsystem* which provides support such as maintenance, purchasing, human resource supply, finance, etc. to the production subsystem.

Each of the nine systems must be viewed from each of the HTPs summarised in Section 2.2.2 as appropriate. The Nine-System model is not shown in a single figure, instead perceptions of the model from the following perspectives are provided:

- ***The Operational perspective*** shows how the nine systems map directly into the extended holistic problem-solving process shown in Figure 6.5 annotated in Figure 16.1 kicking off at time t_0. S1 is the undesirable situation. S2 is the process that produces an understanding of the *undesirable or problematic situation* (S1) and develops the FCFDS (F3). Once the FCFDS is approved, S4, the process that plans (creates) the realization process (S5) and solution system (S6) begins. S4 terminates at the SRR. The realization process (S5) realizes the solution system (S6). Once realized, the *solution system* (S6) is tested in operation in the *actual situation* existing at time t_1 (S7) to determine if it remedies the *undesirable situation*. However, since the solution realization process takes time, the *undesirable situation* may change from that at t_0 to a new *undesirable situation* existing at t_2. If the *undesirable situation* at t_2 is remedied, then the process ends; if not, the process iterates from the undesirable situation at t_2 and the actual situation (S7) becomes the new undesirable situation in the next iteration of the process (S1').
- ***The Functional perspective*** shown Figure 16.2 shows the relationships between the situations, systems and processes. The *process* to plan the transition from the *undesirable or problematic situation* (S1) to the FCFDS (S3) and the *process* to realize the transition

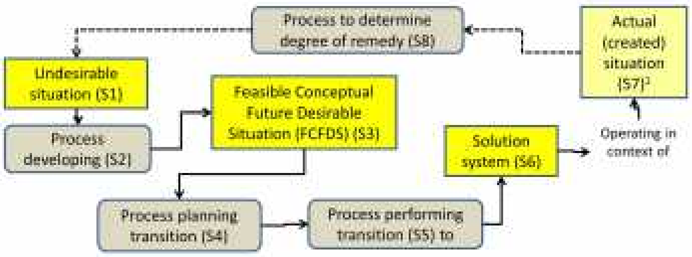


Figure 16.2 The Nine-System Model (*Functional* perspective)

from the *undesirable or problematic situation* (S1) to the *FCFDS* (S3), S4 and S5, constitute the two interdependent sequential systems engineering processes discussed in Section 12.16.

- ***The Structural perspective*** shown in Figure 16.4 shows the relationship between the process systems and the *solution system* and the organization(s) containing the process systems and *solution system*. For example, this perspective provides the:
  - Organisation charts in S9 for staffing the process systems (S2, S4, S5 and S8).
  - Product Breakdown Structure for the solution system (S6).
- ***The Temporal perspective*** shown in Figure 16.3 shows how the systems relate in time. The nine systems do not coexist at the same point in time; the relationship follows the problem solving process shown in Figure 16.1, kicking off at time t_0. S2 is the process that develops the FCFDS (F3). Once the FCFDS is approved, S4, the planning process to create the realization process (S5) and solution system (S6). S4 terminates at the SRR. The realization process (S5) realizes the solution system (S6). Once realized, the *solution system* (S6) is tested in operation in the *actual situation* existing at time t_1 (S7) to determine if it remedies the *undesirable situation*. However, since the solution realization process takes time, the *undesirable situation* may change from that at t_0 to a new *undesirable situation* existing at t_2. If the *undesirable situation* at t_2 is remedied, then the process ends; if not for reasons stated in

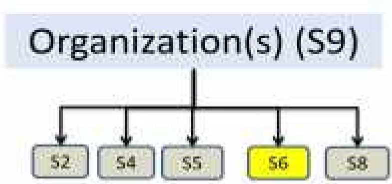


Figure 16.3 The Nine-System Model (*Structural* perspective)

Figure 16.4 The Nine-System Model (*Temporal* perspective)

Section 6.1.1, the process iterates from the undesirable situation at t_2 and the actual situation becomes the new *undesirable situation* (S1').

Consider each of the nine systems as follows:

16.1. System S1: The undesirable or problematic situation

The *undesirable or problematic situation* is a snapshot of the situation that exists at a point in time (t_0) consisting of one or more socio-technological systems working together. This system, known as the 'as-is' situation in BPR, provides the baseline when an entity with the appropriate authority initiates a project to remedy the *undesirable or problematic situation,* by developing something that will convert the undesirable situation to a *FCFDS* (S3). This situation is perceived from the multiple viewpoints of the perspectives on the perspectives perimeter (Section 2.2.2) rather than in one single graphic. For example:

1. ***The Big Picture perspective*** includes information about the adjacent systems.
2. ***The Operational perspective*** is a black box view which includes the operational interactions and interfaces between the situation and the adjacent systems.
3. ***The Functional perspective*** is a white box view which includes the interactions between the system and its adjacent systems that are functioning in the situation.
4. ***The Structural perspective*** is a white box view which includes the structure, information architecture, technology and physical nature of the systems in the situation.
5. ***The Temporal perspective*** includes a history of how the undesirable situation arose.
6. ***The Generic perspective*** includes information about the similarity of the situation to other situations.

7. ***The Continuum perspective*** includes information about pertinent differences between the situation and other situations.
8. ***The Quantitative perspective*** includes numerical information associated with the situation.
9. ***The Scientific perspective*** includes the conclusions inferred from the analysis of the information in the above eight descriptive perspectives about:
   a. The causes of the undesirable situation. If the stakeholders cannot agree on a single problem statement, they may be able to provide a consensus on the most acceptable *FCFDS* (S3).
   b. Ways to remedy the undesirable situation that could lead to the *FCFDS* which provide ideas relevant to S5.

16.2. System S2: The process to create the FCFDS

The concept development process to create the *FCFDS* (S3) is divided into the three streams of activities occurring between milestones discussed in Section 9.8 and contains the following sequential activities (perceptions from the *Functional* perspective):

1. Bounding the SOI and analysing the *undesirable situation* (S1) from the eight descriptive perspectives.
2. Conceiving a number of potential conceptual solution options in the form of FCFDS in accordance with the process shown in Figure 6.2. This activity is best performed as independent parallel tasks so that the conceptualisation of each *FCFDS* is not influenced by the conceptualisation of another *FCFDS*.
3. Identifying ideal solution selection criteria.
4. Performing the trade-off studies to determine the preferred *FCFDS*.
5. Producing the CONOPS that describes the *solution system* (S6) and the context and environment (*FCFDS)* in which the *solution system* will operate and how that operation is anticipated to occur.

This process (S2):

- Is performed in the context of, and uses resources provided by, the organization system (S9).
- Takes place in the Needs Identification State of the SDP (Section 13.2.2.1).
- Studies the *undesirable situation* (S1) and the *FCFDS* (S3) using the systems engineering mathematical and analytical tools of the 1960's (Section 7.3).
- Is divided into the three streams of activities occurring between

milestones discussed in Section 9.8.

The *Structural* perspective of the personnel working in S2 and other processes provides the organisation chart.

16.3. System S3: The FCFDS that remedies the undesirable situation

The *FCFDS* (S3):

- Is created at this time based on the principle of working back from the answer (Ackoff, 1999).
- Is the BPR 'to-be' situation.
- Is documented using the eight descriptive HTPs in an iterative manner using the descriptive perspectives in the same way as they are used in Chapters 4 to 11, the US Airborne Laser (ABL) (Section 20.6) and the Multi-Satellite Operations Control Center (MSOCC) Data Switching System (MCSS) Replacement Project (MCSSRP) (Section 17.1). This approach overcomes the defect in the current systems engineering paradigm in which the functional view precedes the physical view in theory[2] but cannot do so in practice (Halligan, 2014).
- Is the context and environment that will incorporate the *solution system* (S6) as conceptualized at time t_0 but actually deployed at time t_1 when it has evolved into S7.
- Is a hypothesis until validated once the *solution system* (S6) is operating in its context (S7) by the validation process (S8).
- Can be considered as S1 in which the:
    - Causes of the original *undesirable or problematic situation* have been eliminated.
    - Potential modifications and improvements to the original *undesirable or problematic situation* have been conceptualized.

16.4. System S4: The process to plan the transition from the undesirable or problematic situation to the FCFDS

The process to plan the transition from the 'as-is' *undesirable or problematic situation* (S1) to the 'to-be' created situation (S7) based on realizing the *FCFDS* (S3) is a set of activities that:

- Convert information in the CONOPS and *FCFDS* (S3) into a matched set of specifications for the *solution system* (S6), the subsystems of S6 and their infrastructure.
- Plan and create the process (S5), to realize and install the solu-

[2] In the 'B' paradigm

tion system (S6) in accordance with, "*The systems engineer creates a unique process for his or her particular development effort*" (Biemer and Sage, 2009: page 153).

- Produce the planning documents such as the Systems Engineering Plan (SEP), the Systems Engineering Management Plan (SEMP), and the Test and Evaluation Master Plan (TEMP).
- Is performed in the context of, and uses resources provided by, the organization system (S9).
- Is divided into the three streams of activities occurring within milestones discussed in Section 9.8.
- Take place during the Requirements State of the SDP (Section 13.2.2.2)
- Generally terminate with a SRR.

16.5. System S5: The process to perform the transition

The process to perform the transition from the *undesirable or problematic situation* (S1) to the to be created situation (S7) based on realizing the *FCFDS* (S3) by providing the *solution system* (S6) according to the plans (e.g. SEMP and TEMP) developed in the *planning process* (S4):

- Is often called the 'SEP' in the 'B' paradigm when the SDP is used to develop a new system. However, S5 can also be a COTS acquisition process or a combination of development and COTS acquisition.
- Takes place in the remaining states of the SDP (Section 13.2.2).
- Is divided into three streams of activities discussed in Section 9.8.
- May require several iterations when the requirements are dynamic and changing rapidly.
- May only require a single iteration when the requirements are stable.
- Must be able to cope with changes in the needs before the *solution system* (S6) is delivered.
- Is performed in the context of, and uses resources provided by, the organization system (S9).
- Is where the systems engineering tools of 2005 discussed in Section 7.3 are used.

16.6. System S6: The solution system that will operate within FCFDS

The solution system (S6):

- Does not have to be technological or even a new acquisition.

- Is first conceptualised during S2 as a part or subsystem of S3, and then realized into the reality that has evolved out of S3 during S5 to become a part or subsystem of S7.
- Is first partitioned into two major subsystems, the mission and support subsystems described in Section 15.2. The support systems for the solution system can be either subsystems or adjacent systems depending on the situation.
- Lies somewhere along a continuum that stretches from 'fully automatic technological' to 'manual with no technology'; and may be a modification of an existing system, a change to an existing process, tactics, doctrine, policy, or training or some combination.
- Needs to be realized in such a manner that upgrades reflecting changing needs during its operational state can be incorporated without major perturbations.
- Has to be interoperable with evolving adjacent systems in S7 during the operational life of S6 and the adjacent systems in S7.
- Must be viewed from at least the following perspectives:
    - ***Operational perspective*** which shows what the system does (scenarios) by describing the interactions with adjacent systems and the metasystem.
    - ***Functional perspective*** which show the internal mission and support functions.
    - ***Structural perspective*** which shows the information architecture, technology and physical components.
    - ***Quantitative perspective*** which shows the numbers associated with the functions, structure and other properties of the system (costs, reliability, etc.).
- Operates in the context of, and uses resources provided by, the *organization system* (S9).

16.7. System S7: The actual or created situation.

The *actual or created situation* (S7) exists once the *solution system* (S6) has been deployed. S7:

- Is the realization of the *FCFDS* (S3).
- Is the situation at the time solution system (S6) is realized (t1).
- Contains the *solution system* (S6) and adjacent systems operating interdependently.
- May only partially remedy the original *undesirable or problematic situation* (S1).
- May not remedy new undesirable aspects that show up during

time taken by *realization processes* (S2, S4 and S5).

- May contain unanticipated undesirable emergent properties from the *solution system* (S6) and its interactions with its adjacent systems for reasons discussed in Section 6.1.1.
- May be realized in partial remedies.

16.8. System S8: The process to determine that the realized solution remedies the evolved undesirable situation

This *validation process* (S8) sometimes known as Operational Test and Evaluation (OT&E) determines if the *solution system* (S6), operating in its context, remedies the new evolved undesirable situation at t_1. While this process if often thought of as the last stage of the SDP, when the SOI is viewed from the *Temporal* perspective, it can be seen that once the *solution system* (S6) is deployed and operational in the context of the *created situation* (S7), S8 evolves into the change control process that:

- Triggers a new iteration of the problem-solving process to modify/upgrade the *solution system* (S6). In this instance, S7 becomes the new undesirable situation (S1') at time t_2 as shown in Figure 6.5.
- May lead to the Disposal State of the SLC should the *solution system* (S6) no longer remedy the undesirable aspects of the evolved situation (S7).
- Is performed in the context of, and uses resources provided by, the organization system (S9).
- Is divided into the three streams of activities occurring within milestones discussed in Section 9.8.

16.9. System S9: The organization(s) containing the processes.

S9 is the *organization* or organizations containing the processes and providing the resources for the Operation and Maintenance (O&M) of the processes. S9 is also often known as the Enterprise which may be made up of more than one organization. However as they are instances of a single generic type of system, they can be treated as such. Each organization can itself be portioned into subsystems often known as departments and the Nine-System Model applies to each department in a self-similar manner. For example, consider the Human Resources (HR) department of the fictitious Federated Aerospace, which supports staffing the projects and other departments. From the perspectives of the HR department, the nine systems are:

S1. ***Undesirable situation:*** a lack of competent, motivated staff in projects and other departments.

Table 16.1 Focus of the Standards, MBSE, problem-solving, etc. and the nine systems

System	MIL-STD-499	EIA-632	IEEE 1220	ISO/IEC 15288	Hitchins (2007)	SIMILAR	MBSE	SSM	problem-solving
S1					X			X	X
S2					X				X
S3					X		X	X	X
S4	X		Partial	X	X				X
S5		X	X	X		X	X		X
S6		X	X			X	X		X
S7									X
S8						X			X
S9			Partial	X					

S2. ***Process to develop the FCFDS:*** one of the corporate personnel management processes that create and maintain the policies producing the FCFDS.
S3. ***FCFDS:*** projects fully staffed with competent personnel and retaining staff.
S4. ***Process to plan the transition to the FCFDS:*** the staff hiring and prevention of staff leaving processes.
S5. ***Process to perform the transition to the FCFDS:*** HR personnel management system (hiring, training, etc.).
S6. ***The solution system:*** the HR department personnel management system.
S7. ***The created situation:*** projects fully staffed with competent, motivated personnel and retaining staff.
S8. ***Process to verify:*** one of the corporate quality management processes.
S9. ***The organization:*** e.g., Federated Aerospace.

16.10. Clarifying the confusion of the different process description

This section shows how the Nine-System Mode relates to MIL-STD 499, EIA 632 and IEEE 1220 Standards, the SIMILAR process, Hitchins' version of systems engineering and the problem-solving process; they are all partial views of the nine systems as shown in Table 16.1.

16.10.1. MIL-STD-499

The purpose of the MIL-STD-499 (Systems Engineering Management) Standard (MIL-STD-499, 1969) was to provide a set of criteria for people writing plans. Its updated version MIL-STD 499A (Engineering Management) (MIL-STD-499A, 1974) was developed to assist Government and contractor personnel in defining (planning) the system engineering effort in support of Defense acquisition programs. These activities take place in S4.

16.10.2. EIA 632

EIA 632 (Processes for Engineering a System) defines five groups of processes for engineering a system (EIA 632, 1994), namely EIA 632 focuses on S5.

16.10.3. IEEE 1220

The focus of the IEEE 1220 (Standard for Application and Management of the SEP) is on the engineering activities necessary to guide product development (IEEE 1220, 1998) namely, IEEE 1220 focuses on S5 (to produce S6) with some coverage of S4 and the enterprise in which S4 and S5 are taking place (S9).

According to IEEE 1220, "*the systems engineering process is a generic problem-solving process*" (IEEE 1220, 1998) Section 4.1). The IEEE 1220 version of the SEP, based on the shortened problem-solving process; the section inside Figure 6.5 that starts with a problem and ends with a solution; has produced the 'B' paradigm of systems engineering. IEEE 1220's replacement of the words 'problem-solving process' by the words 'SEP' seems to have led to today's focus on process; specifically the development process. Had the Standard instead stated that 'systems engineers apply the generic problem-solving process', the focus of DoD-based and INCOSE- based systems engineering might have remained on the original focus (according to Jenkins) of managing complex problems (Jenkins, 1969).

16.10.4. The SIMILAR process

The SIMILAR process (Bahill and Gissing, 1998):

- Shown Figure 9.9 focuses on three aspects of systems engineering:
    a) Requirements definition.
    b) Architectural design.
    c) Testing and verification.
- Follows the 'B' paradigm of systems engineering (Section 9.21.2).

16.10.5. ISO /IEC 15288

ISO/IEC 15288 (System engineering – System life cycle processes) (Arnold, 2002):

- Focuses on the processes that span the conception of the idea through the retirement of a system within the context of the enterprise (S4 and S5 in the context of S9).

- Follows the 'B' paradigm (Section 9.21.2).

16.10.6. Hitchins' version of systems engineering

Hitchins' version of systems engineering shown in Figure 6.3:

- Follows the 'A' paradigm (Section 9.21.1).
- Is based on the problem-solving process but only ranges from identifying the problem to formulating the strategies and plans for realizing S6 namely, S1, S2, S3, and S4. As far as Hitchins is concerned, activities in S5 and S8 constitute engineering rather than systems engineering.

16.10.7. Dissolving paradoxes

The Nine-System Model dissolves the following paradoxes in the current systems engineering paradigm:

- ***The systems engineering tools paradox:*** discussed in Section 7.3 is dissolved as summarized in Section 12.19.4 by recognising that the focus of systems engineering changed in the DoD when the DoD moved early stage systems engineering out of systems engineering into CAIV and the early stage activities in S2 were to be performed by IPTs rather than by systems engineers (DOD IPPD, 1998; DOD 5000.2-R, 2002), pages 83-84)[3]. Thus, the systems engineering tools of the 1960's are used in S2 to apply to S1 and S3 while the systems engineering tools of 2005 are used in S5.
- ***The reductionist paradox:*** reductionism has been considered as poor practise in systems engineering, yet current system views are inherently reductionist since they exclude the metasystem. The paradox is dissolved by the Nine-System Model which considers the system (S6) and the metasystem (S7) as two of the nine systems. As such, most of the current system level drawings would not be acceptable in the nine-system paradigm since they tend to lack links to the metasystem and adjacent systems.
- ***The roles paradox:*** discussed in Sections 9.24.1 and 12.19.2 may be dissolved, by observing that systems engineering has evolved since 1969 when it was concerned mainly with S1 to S4 as observed by Jenkins and 1996 when the INCOSE version of

[3] Perhaps DoD noticed that the Needs Identification State activities were not being performed by systems engineers and allocated them to CAIV so someone would be doing those activities rather than removing the activities from systems engineering?

systems engineering was concerned mainly with S5 and S6 as observed by Sheard.

16.11. Examples of the Nine-System Model

Consider the following examples of the Nine-System Model to assist in gaining insight as to the capabilities of the model:

1. The NASA Apollo program discussed in Section 16.11.1.
2. An unmanned aerial vehicle discussed in Section 16.11.2.

16.11.1. NASA Apollo program

The NASA Apollo program was probably the most complex project ever tacked in human history up to and including the 1970s. Applying the Nine-System Model at the highest level of the hierarchy of systems that comprised the program, the nine systems were:

S1. ***Undesirable situation:*** the perception that the Soviet Union was ahead of the US in space.

S2. ***Process to develop the FCFDS:*** NASA's early statte systems engineering in association with public relations.

S3. ***FCFDS:*** the perception that the US was ahead of the Soviet Union in space.

S4. ***Process to plan transition to FCFDS:*** NASA's early state systems engineering.

S5. ***Process to realize transition to FCFDS*** took place in the Manned Space Flight development activities in NASA, the Defense Contract Administration Services (DCAS) and the private contractors.

S6. ***The system*** operating in the FCFDS at the highest level of the hierarchy can be considered as the following three subsystems (Kasser, 2013c: pages 225-226):

   a) ***The Earth subsystem*** containing the NASA manned spaceflight centres, NASA headquarters and the NASA Communications System (NASCOM).
   b) ***The Lunar subsystem*** which was empty before the first landing and then contained an increasing number of Apollo Lunar Surface Experiments Packages (ALSEP). Two astronauts were part of this subsystem while they were on the lunar surface.
   c) ***The interface subsystem*** which contained the spacecraft, astronauts (three while in transit, one when in lunar orbit) and the communications subsystems.

S7. ***The created situation:*** after Apollo 11 landed on the moon.

S8. ***Process to verify:*** Public opinion polls.
S9. ***Organizations:*** NASA, DCAS and its contractors.

16.11.2. An unmanned aerial vehicle

An Unmanned Aerial Vehicle (UAV) is a system that performs a variety of missions and provides an example of the model at an intermediate level in the hierarchy of systems. Applying the Nine-System Model applied to a military reconnaissance UAV the nine systems are:

S1. ***Undesirable situation:*** a need for accurate and timely information about something happening in a remote location.
S2. ***Process to develop the FCFDS:*** one of the early state system engineering activities.
S3. ***FCFDS:*** receipt of accurate and timely information about something happening in a remote location.
S4. ***Process to plan transition to FCFDS:*** one of the early state system engineering activities.
S5. ***Process to realize transition to FCFDS:*** the military acquisition process that would develop or purchase a UAV and supporting systems (ground control, data processing, etc.).
S6. ***The solution system:*** the UAV.
S7. ***The created situation:*** the UAV and adjacent systems operational and providing the accurate and timely information about something happening in a remote location.
S8. ***Process to verify:*** the OT&E process.
S9. ***Organizations:*** the contractor organisations in which the UAV is developed or purchased from and the military organisation in which the UAV is deployed.

16.12. Managing complexity via the application of the Nine-System Model at various levels in the system hierarchy

The Nine-System Model applies at every level in the hierarchy of systems as shown in Figure 16.5, Figure 16.6, Figure 16.7 and Figure 16.8 where:

- Figure 16.5 displays the lowest level of the hierarchy of this set of systems. This figure shows a radar system (S6) which will operate as a subsystem in the context of its metasystem (S7), the aircraft.
- Figure 16.6 shows the next level of the hierarchy, the aircraft (S6) which is a subsystem of the airfield (S7).
- Figure 16.7 shows an adjacent or sibling system to the aircraft; a hangar (S6) which is also a subsystem of the airfield (S7).
- Figure 16.8 shows the next level of the hierarchy, the airfield (S6) which is a subsystem of the ADS (S7). Note that these hierar-

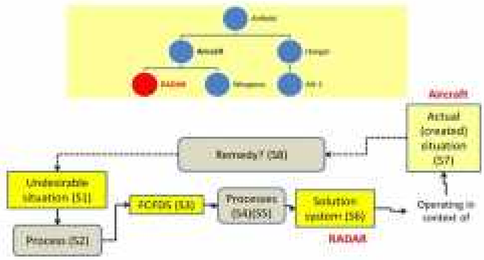


Figure 16.5 Radar as a subsystem of an aircraft

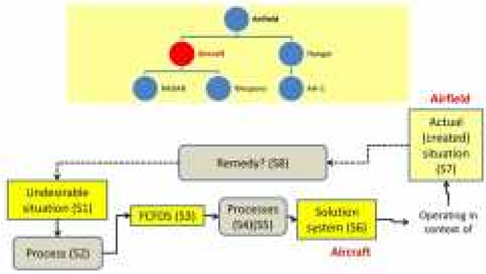


Figure 16.6 Aircraft as a subsystem of an airfield

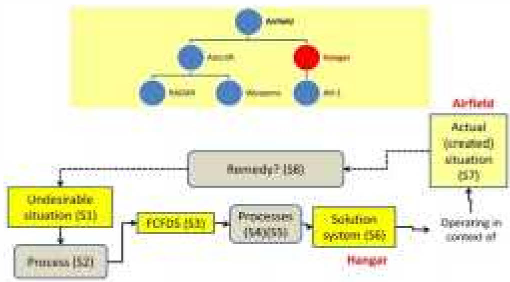


Figure 16.7 Hangar as a subsystem of an airfield

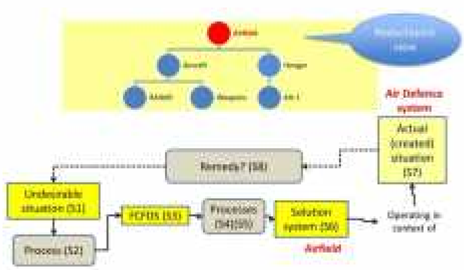


Figure 16.8 Airfield as a subsystem of the ADS

chical views are reductionist if used on a stand-alone (single view) basis as pointed out in Figure 16.8 because the hierarchical view does not show the metasystem (S7).

- In any of the four figures:
  - Each system has its own nine systems.
  - Each system is described by its eight descriptive HTPs.
  - S6 and its adjacent systems are subsystems of S7.
  - All information not pertinent to the points being made in the discussion, e.g. the organizations (S9), has been abstracted out.
- Each systems engineer working on S6 only needs to be concerned with their subsystems, S6 and S7 and manage the rest of the complexity in the following manner:
  - ***Subsystems:*** the internal components are purview of the subsystems engineer in the same manner as any other subsystem. However, factors that affect S6 are within the purview of the S6 systems engineer, hence the need for coordination.
  - ***Adjacent systems:*** they show up in the *Big Picture* perspective of S6; their operational interactions and interfaces are seen in the *Operational* perspective of S6. However, since S6 and the adjacent systems are subsystems of the metasystem operating in S7, the specification of the nature of the adjacent systems are the purview of the system engineer of that metasystem operating in S7, in the same way as the specification of the nature of the subsystems of S6 is the purview of the system engineer of S6.

The partitioning of information in the perspectives associated with each system chunks the information to mask the complexity and allows it to be managed. The descriptive perspectives provide templates for describing each of the nine systems. For example:

- Horizontal views:
  - Show appropriate support systems as adjacent systems in the *Big Picture* and *Operational* and perspectives.
  - All systems at the same level in the hierarchy will have the same metasystem and a slightly different list of adjacent systems.
- Vertical views:
  - Show appropriate mission and support systems as subsystems in the *Functional* and *Structural* perspectives.
  - Provide traceability from system to subsystem in the hierarchy.

These templates could be built into future systems engineering tools and provide similar functionality to that provided by today's requirements management tools such as identifying missing links, etc.

16.13. Benefits of the Nine-System Model

The benefits of the Nine-System Model include, it:

- Is founded on a theory based on aspects of problem solving and system engineering.
- Links into the existing problem-solving and process paradigms.
- Builds Best Practices into systems engineering.
- Discourages the current reductionist and isolationist views of a system by means of the built-in metasystem (S7).
- Encourages operational testing of the solution system (S6) in context of the created situation (S7) in S8 (OT&E).
- Abstracts out complexity and consequently opposes today's tendency to make things more complex.
- Contains clear boundaries and lines of demarcation between the nine systems.
- Shows that DT&E takes place as one of the streams of work in S5 and OT&E takes place in S8. Hence by definition, adoption of the Nine-System Model incorporates those activities as Best Practice.
- Includes aspects that tend to be ignored in the current systems engineering paradigm, such as:
  - Planning the realization process (S4)[4].
  - The concept that the top-level system is something else's subsystem.

16.14. Summary

This Chapter introduced the Nine-System Model to improve systems engineering. Note the Nine-System Model is not a model of systems engineering, it is a framework and tool.

--oo--

[4] Which is omitted from many systems engineering courses and taught in project management courses.

17. Simplifying Managing Stakeholder Expectations using the Nine-System Model and the Holistic Thinking Perspectives

This Chapter discusses how to manage stakeholder expectations using a combination of the HTPs to identify the stakeholders, and the Nine-System Model discussed in Chapter 16 to identify the stakeholders' areas of concern in the context of the Needs Identification State of the SDP activities in the Multi-Satellite Operations Control Center (MSOCC) Data Switching System (MCSS) Replacement Project (MCSSRP) (Kasser and Mirchandani, 2005). The Chapter:

- Summarizes stakeholder management in the literature.
- Summarizes the pertinent information about the MCSSRP from the HTPs to provide the situational example.
- Shows how the HTPs can be used to identify the stakeholders.
- Shows how the Nine-System Model can be used to identify the areas of concern of each stakeholder, and abstract out non-pertinent areas of concern.
- Discusses identifying the complete set of stakeholders and their areas of concern in the context of the MCSSRP.

17.1. The MSOCC data switching system replacement project

The MCSSRP (Kasser and Mirchandani, 2005) provides the context. In the MSOCC situation:

- ***The undesirable situation*** is the perception that the MSOCC will not be able to cope with its anticipated future switching requirements coupled with some undesirable aspects of the current switching system that need to be eliminated.
- ***The FCFDS*** is an MSOCC that is able to cope with its anticipated future switching requirements.
- ***The solution*** is an upgraded higher performance switch operating within the context of the FCFDS.
- ***The problem*** is how to manage stakeholder expectations to gain consensus on a plan to transition from the undesirable situation to the FCFDS.

Perceive the pertinent information about the MSOCC and its stakeholders from the HTPs as follows.

17.1.1. Big Picture perspective

In 1989, the NASA Goddard Space Flight Center (GSFC) MSOCC was facing the problem of replacing the data switch that routed signals from multiple Low Earth Orbit (LEO) satellites to data processing computers. At that time, the MSOCC was the major interface between the LEO data streams from the global satellite tracking network and the Telemetry Tracking and Control system at NASA's GSFC. There was minimal data capture and storage functionality in the ground stations and the NASA Communications Network (NASCOM).

17.1.2. Operational perspective

The MSOCC received and forwarded data in several scenarios documents in the CONOPS. The data streams from the LEO satellites contained data telemetered from onboard experiments and instruments. These data were supplied to Principal Investigators (PI) who would be very upset if they lost scientific data during the time period that the data switch was in transition. It was thus not acceptable to close down the MSOCC during the replacement of the NASCOM switch by the MCSS.

17.1.3. Functional perspective

The MSOCC used a switching system known as the NASCOM switch to route serial asynchronous digital data between NASCOM and the computer equipment within MSOCC and external facilities.

17.1.4. The Structural perspective

Perceptions from the *Structural* perspective of the MSOCC identified the architecture shown in Figure 17.1[1]. The NASCOM Switch shown as a single entity in Figure 17.1, really consisted of a number of subsystems including three separate switches controlled by a central Data Operations Control System (DOCS). The first switch connected some of the MSOCC equipment to the NASCOM lines and the second the remainder. The third switch handled connections between the Mission Planning Terminal (MPT), the Command Management Facility (CMF), the Deep Space Network (DSN), NASCOM and the Attached Shuttle Payload Center (ASPC). Each switch also contained a patch panel to allow the NASCOM lines to be manually tested, patched to another circuit, or looped back to NASCOM or to MSOCC equipment. To complicate the situation:

---

[1] In this situation, since the functions are mapped into the physical units, the same figure can be used to represent both perspectives.

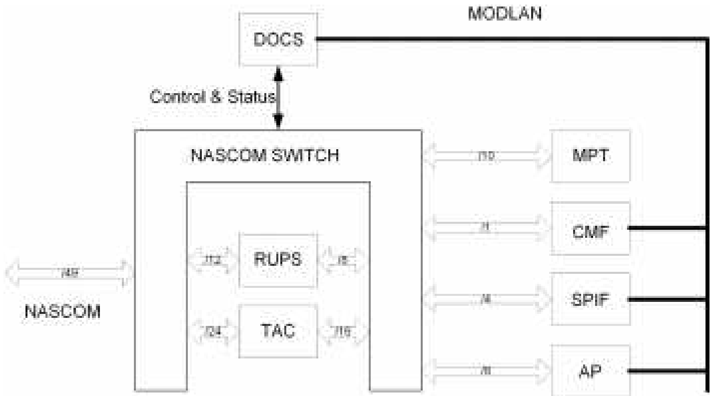


Figure 17.1 The MSOCC

- The MSOCC forward link equipment sourcing uplink data to the LEO spacecraft did not generate the Send Timing (ST) signals (synchronizing pulses) to accompany the data. As a result, the ST for this data was generated by a timing signal generator called a Clock Buffer located in each switch.
- The NASCOM switch could not be removed during the replacement switch integration phase due to insufficient space in the MSOCC to hold both the NASCOM switch and the MCSS.
- The MSOCC was supported by two somewhat overlapping contracts, the Systems Engineering and Services (SEAS) contract and the Network Maintenance and Operations Support (NMOS) contract.

17.1.5. Quantitative Perspective

The three switches were identical, each having a capacity of 62 full duplex 1.544 MHz serial asynchronous RS-422A digital data ports. The switches had been custom-designed for the MSOCC and were not commercially available. Crossovers were used to connect Switch numbers 1 and 2. Switch number 3 was independent of the other two. As a result of using ports for crossovers, only 112 duplex connections could be made through the first two switches.

The system could be taken out of service for pre-scheduled periods of up to 20 minutes at a time.

17.1.6. Temporal perspective

Each of the three NASCOM switches had been added to the MSOCC over time in an incremental upgrade manner as the requirements for additional communications ports exceeded the number of ports available at the time the upgrade took place.

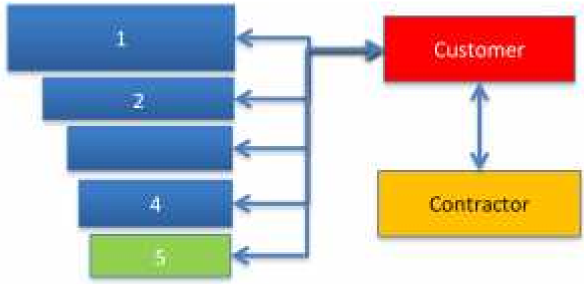


Figure 17.2 Contractual interface

As a result of deficiencies perceived from the *Quantitative* perspective the need for a replacement for the three switches was recognized and the MCSSRP initiated. The new switch system was to be named the MCSS.

17.1.7. Continuum perspective

Perceptions from the *Continuum* perspective identified a number of differences including:

- ***Differences in the stakeholder interests***. Different stakeholders have different areas of concern. As such, not every stakeholder is interested in all the aspects of the MCSS replacement project.
- ***Differences between stakeholders and customers***. While the stakeholders may levy requirements on the MCSS, the customer[2] is the entity that funds the realization of those requirements. Consequently, the customer makes the decision to accept or reject requirements levelled by the stakeholders.
- ***Differences between the stakeholder communications and control interfaces***. The communications interface passes information about stakeholder cares, concerns and needs between the contractor and MSOCC personnel. The control or contractual information first flows from the stakeholders to the customer and then to the contractor as shown in Figure 17.2. In this instance, the figure also provides information from the *Quantitative* perspective by using the size of the box to roughly represent the importance/influence of the stakeholder; information which can be used to prioritize the impact of the stakeholder needs on the project's decisions by adjusting the weighting on the decisions accordingly.
- ***Difference between "no loss of data" and "no downtime"***

[2] The customer was the NASA GSFC Associate Technical Representative (ATR) known as the Contracting Officer's Technical Representation (COTR) in other agencies.

during the transition. Recognition of this difference allows for the switching system to be taken off-line for short periods of time with due prior notice.

17.1.8. Generic Perspective

Perceptions from the *Generic* perspective indicate that the process to address the stakeholders' areas of concern and convert stakeholder's requests to requirements[3] is an instance of the change management process in an upgrade situation. In the change management process, requests for changes are made because something is undesirable due to the system:

1. Not doing what it should be doing, because:
   - Something is broken.
   - Something does not have capability any more (it is overloaded).
2. Not doing something it could be doing.
3. Doing something, but not as well as it could be doing it.
4. Doing something it should not be doing.

The *Functional* perspective of the change management process shown in Figure 17.3 consists of the following activities:

1. Convert the stakeholder area of concern into one or more requirement(s)/change request(s).
2. Assign a unique identification (ID) number to the requirement(s)/change request(s).
3. Prioritize the requirement request(s) with respect to the other requirements/change requests.
4. Determine if a contradiction exists between the requirement(s)/change request(s) and existing accepted requirements/changes.
5. Perform an impact assessment which must:
   - Estimate the cost/schedule to implement the requirement(s)/change request(s)[4].

[3] The term 'request for requirement' is used because the stakeholder's requests must not become requirements until the customer has agreed to accept the request and fund the realization of the request.

[4] In this pre-SRR situation, there is no need to determine the cost and schedule for every requirement. Applying the quantitative perspective in the form of the Pareto principle, it can be perceived that the cost and schedule impact only needs to be determined for the most expensive and longest time to realize requests (Hari, et al., 2008).

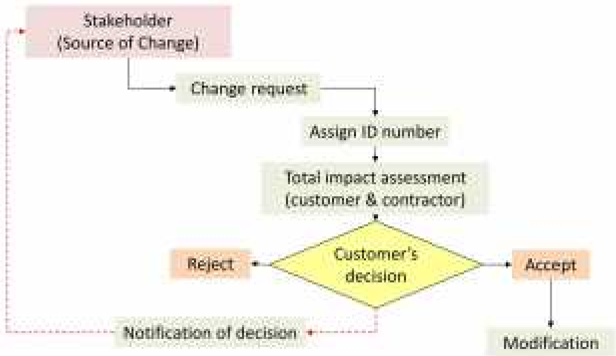


Figure 17.3 Functional view of the generic change management process

- Determine the cost/schedule drivers: the factors that are responsible for the greatest part of the cost/schedule implementing the requirement(s)/change requests(s).
- Perform a sensitivity analysis on the cost/schedule drivers.
- Determine if the high cost/schedule drivers are really necessary and how much negotiating the requirement(s)/change request(s) with stakeholders can make modifications to the high cost/schedule drivers based on the results of the sensitivity analysis.

6. Make the customer's decision to accept, accept with modifications, or reject the request.
7. Notify the stakeholder of the decision.
8. Document the decision(s) in the requirement/change repository to provide a history in case the same requirement(s)/change request(s) are received at some future time.
9. If the requirement(s)/change request(s) is accepted, allocate the implementation to a specific future version of the system, modifying the documentation appropriately.

17.1.9. Scientific perspective

After examining the situation from the eight descriptive HTPs, the conclusion was that the problem of how to transition the MSOCC from the undesirable situation (S1) to the FCFDS (S3) could be split into the following two well-structured problems, each having unique and shared stakeholders:

1. ***Determine the requirements for the MCSS:*** a well-structured non-complex problem since the CONOPS for S3 will be an up-

graded version of the existing CONOPS for S1; as is common in an upgrade situation (*Generic* perspective).

2. ***Convert the stakeholder plurality of opinions*** on the transition from the existing NASCOM switch to the replacement switch to a consensus on an approach. This was a well-structured complex problem with a prime directive of "no loss of satellite data" during the transition.

The problematic or uncertain situation (S1) posed a well-structured problem, in which:

1. There were only seven pertinent systems since S2 had been completed, and the activities were taking place in S4.
2. The CONOPS in the FCFDS (S3) was almost identical to that in the original undesirable situation (S1):
   - This is standard in an upgrade situation (*Generic* perspective).
   - The requirements for the MCSS (S6) were based on the anticipated number of input data streams and data processing equipment in the FCFDS. A quick check of several potential switch vendors identified COTS switches that could meet the MCSS requirements for the numbers of inputs and outputs at a price that was well within the budget. This risk management activity removed the uncertainty associated with S6.
   - The uncertainty was restricted to the transition process (S5).
   - The remaining complexity was abstracted out and the MCSSRP just needed to focus on gaining a consensus on the transition process (S5).

17.2. Stakeholder management in the literature

Given the problem of managing the stakeholder expectations in the MCSSRP, the first activity was to research the literature to determine how other projects managed their stakeholders. The literature published on the Internet is full of "helpful advice" on how to manage stakeholders with comments such as:

- "*Stakeholder management is the process of managing the expectation of anyone that has an interest in a project or will be effected by its deliverables or outputs*" (Project Smart, 2013).
- Stakeholders are entities that can level requirements on the system.
- Stakeholders will include project sponsors, team members, etc.
- Involve stakeholders early in the project to get their support. However, the literature does not state that some of the stake-

Figure 17.4 Stakeholder circles (Recklies, 2001)

holders have tacit knowledge that you will need throughout the project life cycle.

- Identify stakeholders by looking at the formal and informal relationships envisioning the stakeholder environment as a set of concentric circles as shown in Figure 17.4. The inner circles stand for the most important stakeholders who have the highest influence (Recklies, 2001). While the figure identifies categories of stakeholders, it is not that helpful in determining which of them has a stake in a specific project.
- Provides the traditional view of stakeholders as shown in Figure 17.5. While the figure identifies the stakeholders and shows that there is a relationship between the stakeholders, the figure does not provide any information about the nature of the relationships, nor how to manage them.

In general, the literature is helpful but incomplete.

17.3. Managing stakeholder expectations

Managing stakeholder concerns can be considered as a process containing the following activities:

1. Identifying the stakeholders.
2. Identifying the areas of concern of each stakeholder.
3. Addressing the areas of concern of each stakeholder.
4. Converting stakeholder concerns to requirements.
5. Informing the stakeholders how their areas of concern were considered.

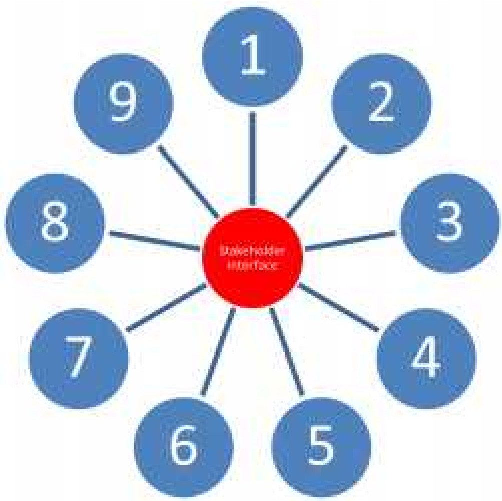


Figure 17.5 Traditional view of stakeholders

6. Gaining stakeholder consensus on the outcome.
7. Maintaining stakeholder consensus.

Perceiving the situation from the HTPs identified the stakeholders and the process to manage stakeholder concerns, when turning them into requirement-requests, but did not identify the stakeholder's areas of concerns.

17.3.1. Identifying the stakeholders

The stakeholders can be identified from the information in the *Big Picture*, *Operational*, *Functional* and *Structural* perspectives of each of the nine systems in the Nine-System Model of the MSOCC. The external perspectives, the *Big Picture* and *Operational* perspectives identify the external stakeholders, while from the internal perspective, the *Functional* and *Structural* perspectives identify the internal stakeholders. The identified stakeholders were:

- ***MSOCC operators:*** identified from the *Functional* perspective.
- ***NASA managers:*** identified from the *Big Picture* perspective.
- ***NASA facilities personnel:*** identified from the *Structural* perspective.
- ***SEAS and NMOS managers:*** identified from the *Operational* perspective.
- ***Hardware and software developers and testers:*** identified from the *Functional* perspective.

Table 17.1 Representation of some of the stakeholder interests

Stakeholder	S1	S2	S3	S4	S5	S6	S7	S8	S9
Dr Principle Investigator							O		
Oswald Operator	X		X		X	X	X		
Ollie Operator	X		X		X	X	X		
Danny Developer		X			X	X	X		
Debora Developer					X				
Development manager		X		X	X		X		X
Tammy Tester	X		X						
Thomas Tester								X	

- ***NASCOM personnel:*** identified from the *Operational* perspective.
- ***Experiment PIs:*** identified from the *Big Picture* perspective.

17.3.2. Identifying stakeholders' areas of concern

Perceptions from the *Scientific* perspective in Section 17.1.9 reduced the area of concern to two of the nine systems; the *MCSS* (S6) and the *transition process* (S5). However, the pre-SRR activities are taking place in S4, and these are the activities that create the *transition process* (S5) and the *MCSS* (S6). Consequently, the stakeholders with the information pertinent to the MCSS upgrade are those with an interest in the *undesirable situation* (S1), the *FCFDS* (S3), and the situation in which the MCSS will operate (S7) as well as the *transition process* (S5) and the *MCSS* (S6). This finding simplified stakeholder management because S2, S4, and S9 could be abstracted out as not being of any major concern (at least during the initial phase).

The areas of concern of each of the stakeholders can be matched to one or more of the nine systems using the assumption that the stakeholder will only be concerned about the aspect of the MCSS upgrade in the system in which they are located. This assumption can be validated during discussions with the stakeholder.

When sorted by the areas of stakeholder concern, a table can be drawn up such as the example presented in Table 17.1. S2 and S4 are shaded in the Table because S2 is history, having been completed when the FCFDS (S3) was created and these pre-SRR activities are taking place in S4. The X's and O's in the table show which of the nine systems is associated with the specific stakeholders. For example, using fictitious names:

- The developers are concerned with the processes (S5) and the solution system (S6) developed by those processes. Deborah Developer, as an example, will only be working in S5 which limits her area of concern to S5.

- The operators are concerned with the undesirable situation (S1), the transition process (S5), the MCSS (S6) and the upgraded MSOCC (S7).
- The testers are concerned with the testing aspects of the project, and upon discussions, we determined that Tammy Tester has a stake in S1 and S3 while Thomas Tester is only concerned with the final acceptance test (S8).
- The development (process) managers are concerned with the management aspects of the processes (S2, S4, S5 and S8).
- Dr Principle Investigator is only concerned with the MCSS upgrade project if he fails to receive his data, hence the 'O' in his column in the Table.

17.3.3. Addressing the areas of concern of each stakeholder

Perceptions from the *Generic* perspective indicated that the process to address the areas of concern and convert stakeholder's requests to requirements[5] is an instance of the generic change management process. Part of the Nine-System Model S4 carries out these activities with all of the pertinent stakeholders as discussed herein. These activities first necessitated arranging a number of meetings with the different stakeholders at their offices at the GSFC. To save time, the discussions covered stakeholder concerns about both of the problems identified in Section 17.1.8. The meetings:

- Were short, taking less than an hour to minimize the impact on the stakeholder's schedule.
- Began with an overview of the methodology being used in the task.
- Discussed the stakeholder's needs and concerns.
- Summarized the concerns, if appropriate, as applying to:
    - The MCSS (S6).
    - Conceptual approaches and selection criteria for the transition from the NASCOM switch to the MCSS (S5).

17.3.4. Converting stakeholder concerns to requirements

As part of the discussion about stakeholder concerns and needs, stakeholders were asked to provide two categories of requirement requests based on their needs; mandatory and "wishes". The "wish" category was

[5] The term 'request for requirement' is used because the stakeholder's requests must not become requirements until the customer has agreed to accept the request and fund the realization of the request.

one where if a decision had to be made to implement a mandatory requirement, and a "wish" could be implemented with little or no extra cost, the "wish" would be taken into account. During the discussion with the stakeholders, the key questions asked were:

- What is good about the current system?
- What is bad about the current system?
- What would you change, and why?

When the responses from the different stakeholders to the questions were compared, we found that some of the answers were complementary and some were contradictory. As each requirement request was identified it was:

- Assigned a unique identification (ID) number.
- Prioritized with respect to the other requirement requests.
- Examined to determine if a contradiction existed between the requirements request and requirement requests from other stakeholders. In the rare instances where there was a contradiction, we met with the stakeholders concerned, discussed and resolved the contradictions.
- Tagged with acceptance criteria. These criteria were obtained by asking the stakeholders "how will you know when the requirement is met?" This question avoids ambiguous requirements. The response to the question clarifies the need and provides the acceptance criteria that will be used in developing the acceptance tests.
- Inserted into the draft MCSS requirements SRD without performing the impact assessment since this was an initial version of the document (the MCSS was replacing the NASCOM switch and so had a new set of requirements although many were inherited from the NASCOM switch) rather than a change to an existing system.

Once the customer accepted the requirement request it became a requirement and all three attributes, the requirement, the corresponding acceptance criteria and the stakeholder identification which provides traceability to the source, were stored in the requirements database. The stakeholder information is to be used when the need for additional information to resolve issues concerning the design, testing or modification of the parts of the system whose purpose is to meet the requirement arise.

17.3.4.1. The MCSS

Once the draft MCSS SRD was complete, we determined that nearly all the requirements requests[6] for the MCSS (S6):

1. Were based on the CONOPS of the MCSS (S6) operating in the MSOCC (S7) switching the anticipated future LEO satellite data streams in a manner that was compatible with the existing control system in the DOCS, coupled with improvements suggested by the stakeholders to overcome irritations and deficiencies in the use of the existing NASCOM switch.
2. Could be met by COTS switches with a price that was well within the budget. All COTS switches could meet the data throughput needs; the deficiencies were in the command and control functionality. When this was pointed out to the stakeholders and customer, after some negotiation, the stakeholders agreed to limit their requirement requests to the functionality provided by the COTS switches so as to remain within the budget. This determination meant that since the COTS switch would be purchased, there was no need to perform the impact assessment to determine the effect on cost and schedule of each requirement request which reduced the duration and cost of the MCSSRP.

17.3.4.2. The transition plan (S5)

The process to develop the transition plan (S5) conformed to that shown in Figure 6.3. Recognizing that something would have to move temporarily to allow parts of the NASCOM switch and the MCSS to be installed simultaneously in the MSOCC, the conceptual candidate transition approaches identified 11 different MSOCC systems as candidates for temporary removal.

We recognized that the prime directive of "no loss of data" did not equate to "no down time" (*Continuum* perspective). There were short periods of time when no data were being received and these times could be determined in advance. Thus each candidate conceptual transition approach could incorporate some down time when data sources and sinks were being rerouted from the NASCOM switch to the replacement MCSS. We met with the stakeholders again at their convenience and discussed the advantages and disadvantages of each conceptual candidate transition approach and their other concerns. These issues became the selection criteria for the recommended transition approach.

At this point in time, somewhere in the MCSSRP S4, we:

---

[6] Since the initial set was to be presented at the SRR for consensus on acceptance, the set constituted requirements requests rather than requirements until accepted at SRR.

- Knew who the stakeholders were from the perspectives of the MSOCC.
- Knew their areas of concern from their system within the Nine-System Model, and confirmed by discussion.
- Had identified 11 candidate transition approaches and their advantages and disadvantages through discussion with the stakeholders.
- Had identified 8 transition approach selection criteria by discussion with the stakeholders.

We then identified the appropriate decision-making tools to use and selected to use the two-part approach in which we would identify the relative importance (i.e. which was more important than the other on a scale of 1-8, with 1 being the most important) and absolute importance (how important each was in itself on a scale of 1-10) of the transition approach selection criteria.

We then formally surveyed the stakeholders as to their preferences. Since the preferences of the stakeholders in the system, being a plurality, had different impacts, we identified a weighting scheme for prioritizing the preferences of the stakeholders[7]. The survey requesting that the evaluation criteria be ranked by the respondent, both in the order of relative importance and standalone importance, was sent to the MSOCC operations, maintenance and engineering personnel.

17.3.5. Informing the stakeholders how their areas of concern were addressed.

Once the areas of concern had been identified and their concerns translated to requirement requests. The two sets of short meetings with the stakeholders allowed us to discuss their concerns and in a few instances how their concerns contradicted other stakeholders' concerns and more importantly, why their concern was noted but not acted upon.

Where the stakeholders' requirements requests for MCSS command and control functions contradicted other requirements requests, we met with the stakeholders, discussed and resolved the contradictions well before the SRR. From the *Generic* perspective this is a standard negotiating technique where the persons involved in the negotiations do not meet directly but pass their concerns through a middleman or negotiator.

[7] We assigned a higher weighting to the stakeholder closest to the system. For example, the operators concerns received a higher weighting than the managers. Although we stated that the survey results had been weighted we never actually provided the weighting scheme, nor were we asked for it.

Informal meetings to report on stakeholder concerns should be held between the formal milestone reviews.

17.3.6. Gaining stakeholder consensus on the outcome

Consensus was gained in the informal meetings, so when the SRR was held at GSFC and covered both the requirements for the *MCSS* (S6) and the transition plan (S5), all requirement requests were accepted and became requirements without a single Review Item Discrepancy (RID)[8].

17.3.7. Maintaining stakeholder consensus

The traditional formal SDP meetings in the form of milestone reviews such as the SDR, TRR and DRR provide opportunities for demonstrating consensus that the stakeholder concerns have been addressed and the *system being developed* (S6) *operating in its context* (S7) will remedy known undesirable aspects of the situation that will exist at the time the *system* (S6) is to be deployed.

The same approach using informal and formal meetings should be used in the later phases of the SDP following the SRR between the formal milestones[9] to:

- Update stakeholders as to the status of the way their concerns are being addressed.
- Manage changes in the stakeholder concerns as they evolve during the SDP.

17.4. Managing indirect stakeholders

While the literature provides lists of potential stakeholders it is not very helpful in identifying whose concerns need to be managed. The perspectives and the Nine-System Model can be used to identify stakeholders using perceptions from the *Structural* and *Temporal* perspectives as discussed herein.

Section 17.3 discussed managing direct stakeholder expectations. Indirect stakeholders can be managed using perspectives from perspectives as follows.

17.4.1. The **Structural** *perspective*

Perceptions from the *Structural* perspective, identified the systems of interest using the principle of hierarchies and the direct and indirect stakeholders as follows:

---

[8] Which were informed was unprecedented. Perceptions from the *Continuum* perspective indicate that either we did a good job, or nobody cared.

[9] It is usually cheaper to prevent a RID than deal with one.

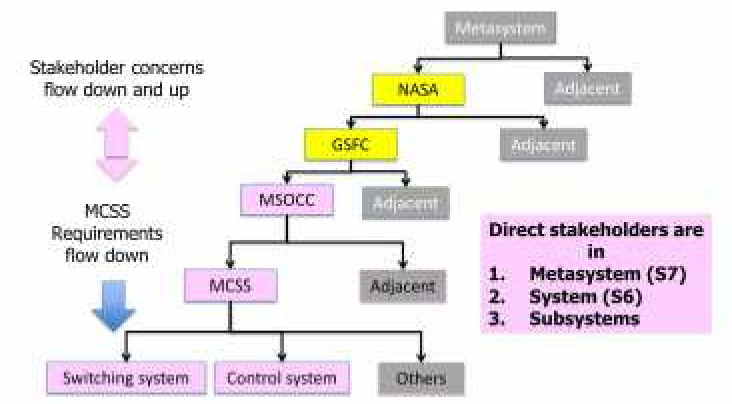


Figure 17.6 Direct and indirect stakeholders

- The MCSS (S6) and MSOCC (S7) prior to S4 as indirect stakeholders.
- The MCSS (S6) and MSOCC (S7) during S4 as direct stakeholders.
- The MSOCC (S7) metasystem as direct stakeholders.
- The MSOCC is S6 in the GSFC (S7) so the GSFC contains indirect stakeholders.
- The GSFC is S6 in NASA (S7) so NASA contains more indirect stakeholders.
- And so on up the levels in the hierarchy of systems as appropriate.

In a different situation, you could now:

- Use the HTPs to examine each S6 and S7 at each level of the hierarchy to identify potential stakeholders in the same manner as the identification of the internal and external MCSS stakeholders.
- Create a table similar to Table 17.1 and use the same approach discussed in the rest of Section 17.3.4.

However, the *Generic* perspective indicates that this should have already been done in the different levels of the hierarchy of systems.

17.4.2. The **Generic** *perspective*

From the *Generic* perspective, just as the MCSS system level requirements flow down into the switch, control and other subsystems of the MCSS, the stakeholder concerns flow up and down into the MSOCC and MCSS as shown in Figure 17.6. This is because the concerns of the external stakeholders should have been addressed at their metasystem or subsystem level, and any applicable concerns should have been passed on as concerns from the stakeholders at the MCSS and MSOCC levels in the system hierarchy. However, since this is an assumption, risk management was performed by inviting the indirect stakeholders as well as the direct stakeholders to attend or be represented at the SRR and subsequent formal milestone reviews to verify that their concerns have been addressed in a satisfactory manner.

17.4.3. The Temporal perspective

Perceptions from the *Temporal* perspective considers Figure 17.4 as a representation of a short list of potential stakeholders extracted from an unspecific longer list but without any additional information as to the state in the SDP in which the stakeholders may have a stake. As a project passes though the different states of the SDP, from conception to termination, the stakeholders change; stakeholders from the previous state fall away, new stakeholders appear, and some of the previous stakeholders sometimes remain.

Stakeholder concerns from the previous states of the SDP must be addressed even if the stakeholders cease to have an active interest in the SDP because a failure to do so will probably result in new stakeholders having the same concerns or as the SDP transitions from S1 to S7, the concerned stakeholders in S1 become concerned stakeholders in S7.

17.5. Comments on managing stakeholder expectations

The ultimate goal in managing stakeholders is to satisfy all stakeholders' expectations. However, in practice, generally, all stakeholders' expectations cannot be completely fulfilled. Thus, the goal in managing stakeholders often ends in a form of negotiated agreement with the stakeholders. That is to say, the difficulty in managing stakeholders is not about how to meet all the stakeholders' requests, but help all the stakeholders gain maximal satisfaction at the same time. Achieving stakeholder satisfaction is a continual activity for the entire SDP. Even though the example discussed the case as sequential activities, several iterations of the process may take place.

Achieving one stakeholder's satisfaction doesn't always mean that another stakeholder has to sacrifice. In general stakeholders have different concerns and a final win-win agreement can often be achieved after several rounds of discussion or negotiations.

17.6. Summary

The problems of stakeholder management and requirements elicitation and elucidation are complex and sometimes the roles, responsibilities and areas of concern of the stakeholders seem difficult to identify and integrate. This Chapter:

- Introduced the concept of direct and indirect stakeholders in addition to internal and external stakeholders.
- Addressed those issues and described a systemic and systematic way of simplifying stakeholder management and requirements elicitation and elucidation in a situational example using the:
  - HTPs to identify the stakeholders.
  - Nine-System Model to sort stakeholders and identify their areas of concern in order to translate their expectations into system requirements using the MCSSRP as an Experiential Case Study example.

--oo--

18. Guidelines for creating a system

This Chapter[1] improves systems engineering by filling a gap in the systems engineering literature by suggesting a process for creating a system to be used in the early states of the SDP to help to manage complexity at the time the system is created by optimizing the interfaces. The process follows Maier and Rechtin's recommendation that the way to deal with high levels of complexity is to abstract the system at a high a level as possible and then progressively reduce the level of abstraction (Maier and Rechtin, 2000: page 6) and with reference to the Nine-System Model discussed in Chapter 16 contains the following activities:

1. Examine the undesirable situation (S1) from several different perspectives.
2. Develop an understanding of the situation (S1).
3. Create the FCFDS containing the SOI (S3).
4. Use the principle of hierarchies to abstract out the complexity.
5. Abstract out the parts of the situation (S1 and S3) that are not pertinent to the problem.
6. Partition the FCFDS (S3) into the SOI (S6) and adjacent systems.
7. Optimize the interfaces.
8. Partition the SOI into subsystems.

Note:

- The activities should be performed in an iterative sequential parallel manner not in a sequential manner.
- The FCFDS (S3) will evolve to the actual or created situation (S7) during the time taken to plan the SDP (S4) as well as the time taken to perform the SDP (S5).

Consider each of these activities as follows.

18.1. Examine the undesirable situation from several perspectives

Traditional systems enquiry creates dynamic views of the behaviour of a SOI using tools such as causal loops (Senge, 1990), system dynamics (Clark, 1998; Wolstenholme, 1990), queuing theory, linear programming

[1] The chapter is a modified version of (Kasser, 2015).

and other tools used in Operations Research. Other approaches include building models or applying sets of equations suitable to the class of situation. However, while modelling the behaviour of a SOI does provide a wealth of information, using this single behavioural perspective does not provide a full understanding of the SOI and may even lead to a misunderstanding, identification of the wrong cause of the undesirability and a definition of the wrong problem, as discussed in Section 2.2.1. Thus use of these traditional systems thinking tools must be considered as only a part of the process of examining the situation to gain an understanding of the situation since one needs to go beyond systems thinking and employ perceptions from the *Generic* and *Continuum* perspectives to identify the right problems and some acceptable solutions.

18.2. Develop an understanding of the situation

After examining the situation from the eight descriptive perspectives, the systems engineer should develop an understanding of the situation. For example:

- The entities involved in the situation should have been identified. These entities include those directly involved and the indirect stakeholders such as those in the example of using the Nine-System Model to manage stakeholder expectations in Chapter 17.
- The behaviour of the SOI can be understood from the information obtained from the relationships in the *Operational* and *Functional* perspectives. This information is often used to build a behavioural model.
- The undesirable aspects tend to show up in the *Structural*, *Operational* and *Functional* perspectives and should have been identified by discussions with the stakeholder and by analysis.
- The cause or causes of the undesirability and a conceptual approach to remedying the undesirability should then have been inferred (*Scientific* perspective).

18.3. Create the FCFDS

The FCFDS (S3) is a modified existing situation (S1). Even in in situations where the stakeholders cannot agree on the causes of the undesirability, they should be able to agree on the nature of the undesirability and a situation in which the desirability is no longer present. As such, the initial version of the FCFDS is the existing situation with the undesirability removed, and often with suggested improvements added. The FCFDS will contain a number of elements coupled together as shown in Figure 18.1.

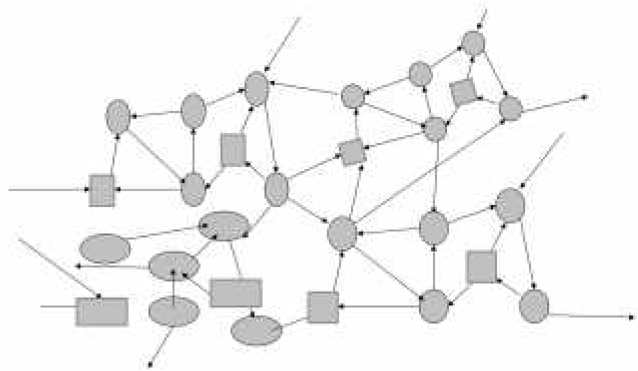


Figure 18.1 The FCFDS

18.4. Use the principle of hierarchies to abstract out the complexity

The principle of hierarchies in systems discussed in Section 7.2 is one of the ways humanity has managed complexity for most of its recorded history. It includes:

- Keeping the systems and subsystems at the same level in the hierarchy of systems.
- Abstracting out or hiding the internal components of systems and subsystems. For example, Maier and Rechtin recommend that the way to deal with high levels of complexity is to abstract the system at a high a level as possible and then progressively reduce the level of abstraction (Maier and Rechtin, 2000: page 6).
- The concept that one systems engineer's subsystem is another systems engineer's SOI as discussed in Section 16.12.

A situation is a system which contains a number of systems. Each system in turn may contain a number of subsystems. Each subsystem may be further elaborated into a number of components (subsystems of the subsystem). This concept is often shown in the traditional hierarchical structure such as in organisation charts, work breakdown structures and product breakdown structures.

18.5. Abstract out the parts of the situation that are not pertinent to the problem

Dealing with issues in any specific situation will probably only need a subset of the information perceived from the different HTPs. For example, consider the problem of docking a resupply vehicle to the International Space Station (ISS). Each is a complex system in itself, yet when solving the problem of docking a resupply vehicle to the ISS, all the underlying complexity that is not relevant to the docking problem is abstracted out. Thus, we construct a closed system view to simplify the problem by abstracting out (filtering out) everything other than information pertinent to the:

- Relative positions of the spacecraft.
- Relative velocity of the spacecraft.
- Relative orientation in X-, Y- and Z-axes of rotation.

Instead of a single system view, there are a number of views, each of them dealing with some aspect of the SOI. So, it is the systems engineer's role to determine which elements are pertinent to the problem and abstract out the remainder[2]. Consider further examples of a rock, a camera and a human being.

18.5.1. A rock

A rock is a very simple system made up of chemical molecules[3]. The system boundary is drawn at the surface. While determining the nature of the rock, various views can be used including:

- ***Sight:*** one looks at its colours.
- ***Taste:*** taste might give us some information about the chemicals in the rock.
- ***Weight/mass:*** might tell us something about its composition.
- ***Touch:*** the surface texture might be of interest.
- ***Chemical analysis:*** the components might be of interest.
- ***Radiation:*** could tell us something.

Each view provides information that the others do not, helping to build up a complete understanding of the nature of the rock. Which view we use depends on what issue we are dealing with.

18.5.2. A camera

Perceive a camera. When we consider the device that takes the photograph, we draw the system boundary around the camera. However, when we consider the act of taking the photograph the boundary is redrawn to include the photographer. When considering transporting the camera the boundary is drawn to include the transportation elements including the carrying case. Developing one representation that includes all the ele-

---

[2] When dealing with existing systems or systems that have already been realized in other places, this information will be generally be available using the *Generic* perspective. When dealing with unprecedented systems, good systems engineers will immerse themselves in the situation to identify which elements are important, the underlying assumptions that may cause problems, etc.

[3] If it contains more than one element and the properties of the rock are due to the elements and their interactions.

ments for photographing and transportation and then requiring the elements under consideration for a specific situation to be abstracted out of the representation, creates unnecessary complexity. The three separate simpler views, abstracted out of the real world are simpler for understanding the various aspects of the use of a camera in photography.

18.5.3. A human being

Some areas of the real world can only be fully understood by:

- Examining the internal components of the system.
- Observing it in action in its environment.

Consider a human being, a biological system. To learn about the interaction between internal subsystems we may have to observe the sample in action in specific situations and either observe or infer the interaction. To learn about the internal subsystems we have to dissect a sample of the system. However, once dissected, an individual sample cannot usually be restored to full functionality. However we have learnt something about the class of systems it represents which can be applied to other instances (human beings); the assumption being that the internal components of human beings are almost identical.

18.6. Partition the FCFDS into the SOI and adjacent systems

It is the act of drawing the system boundary that creates the system (Beer, 1994; Churchman, 1979: page 91). When the undesirable situation already contains a SOI, such as in an upgrade or replacement situation, then the existing SOI tends to be the starting point for creating a new SOI. However, the systems engineer should not assume that the boundaries of the existing and new (replacement) SOIs are identical and keep in mind that the boundaries of the SOI may need to change to remedy the undesirable situation as described below.

The entities in the FCFDS should be aggregated into the SOI and adjacent systems by some common denominator such as function, mission or physical commonality according to the rules for performing the aggregation described below.

18.6.1. Rules for performing the aggregation

The FCFDS is partitioned into the SOI and adjacent systems using the following rules for performing aggregation:

1. ***Keep number of subsystems at any level to less than 7±2*** in accordance with Miller's rule to facilitate human understanding of the SOI (Miller, 1956).
2. ***Configure each subsystem for the maximum degree of homeostasis***. This rule which is widely used in human systems as

well as in technological systems provides risk management and interface simplification since a subsystem configured according to this rule:

- Ensures that the subsystem can continue to operate if the command and control link is lost.
- Often requires a simple interface that passes relatively low-speed high-level commands and status information rather than high-speed real-time control commands.

3. ***Maximize the cohesion of the individual subsystems and minimize the coupling between subsystems*** (Ward and Mellor, 1985).

There are various types of cohesion and coupling.

18.6.1.1. Continuum of coupling

When perceiving coupling and cohesion from the *Continuum* perspective, the degree of coupling and cohesion can be seen as lying on a continuum as follows:

- ***Independent:*** the end of the continuum where the elements are not coupled at all.
- ***Interdependent:*** the middle sections of the continuum where the coupling of the elements ranges from loosely-coupled to tightly-coupled.
- ***Inseparable:*** the other end of the continuum where the elements are so tightly coupled that they cannot be separated.

18.6.1.2. Relating or joining the elements together

Cohesion and coupling also define how the elements relate or join together, where:

- ***Cohesion:*** the term used with respect to the view seen from an internal perspective looking at a single system or subsystem.
- ***Coupling:*** the term used with respect to the view seen from an external perspective looking at more than a single subsystem or subsystem.

Sommerville provided the following list of types of cohesion in the software domain (Sommerville, 1998):

- ***Coincidental***: the elements have no relationship.
- ***Logical:*** the elements are performing similar functions.
- ***Temporal:*** the elements that are activated at a single (the same) time.

- ***Procedural:*** the elements that make up a single control sequence.
- ***Communicational:*** the elements that operate on the same input data or produce the same output data.
- ***Sequential:*** the output from one element in the component serves as input for some other element.
- ***Functional:*** each element is necessary for the execution of a single higher level function.

Other types of coupling from the software domain include:

- ***Content coupling (high):*** one element modifies or relies on the internal workings of another element, e.g. accessing local data of another element.
- ***Common coupling:*** two elements share the same global data, e.g. a global variable.
- ***External coupling:*** two elements share an externally imposed data format, communication protocol, or device interface.
- ***Control coupling:*** one element controls the logic of another, by passing it information on what to do.
- ***Stamp coupling (data-structured coupling):*** the elements share a composite data structure and use only a part of it, possibly a different part, e.g. passing a whole record to a function which only needs one field.
- ***Data coupling:*** the elements share data, e.g., through parameters.
- ***Message coupling (low):*** the elements are not dependent on each other; instead they use a public interface to exchange parameter-less messages.
- ***No coupling:*** the elements do not communicate with one another.

In the physical realm, one can add other forms of coupling including:

- ***Mechanical coupling:*** the elements are coupled together by mechanical means, e.g. rivets, nuts and bolts, nails, joints, glue, welds, hook and loop fasteners, etc.
- ***Gravitic coupling:*** the elements are coupled together by gravity, e.g. one element rests on top of another. This type of coupling is common on planetary surfaces.
- ***Magnetic coupling:*** the elements are coupled together by magnetic means, e.g. intruder alarms, magnetic locks and items on refrigerator doors.

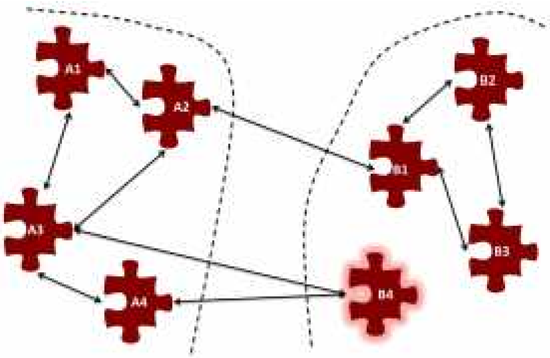


Figure 18.2 Cohesion and coupling

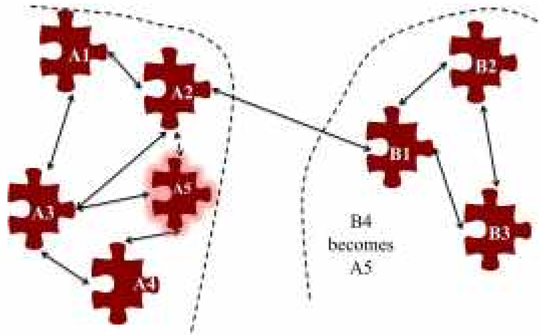


Figure 18.3 Better coupling and cohesion

- ***Electrostatic coupling:*** the elements are coupled together by electrostatic charges.

Each type of coupling has advantages and disadvantages. The role of the systems engineer is to examine the different ways components can be aggregated into subsystems and use a design approach that maximizes cohesion and minimizes coupling which contributes to optimizing the interaction between the interfaces of the subsystems. A useful tool to perform this activity is the N^2 chart (Lano, 1977) or the Design Structure Matrix (Eppinger and Browning, 2012).

18.6.1.3. Variations on a theme

However, in practice, maximizing cohesion and minimizing coupling is not always the rule in systems engineering. Consider the two subsystems 'A' and 'B' shown in Figure 18.2. There are three interfaces between the two subsystems. Note that element B4 in subsystem 'B' does not have any connection with the remaining elements in subsystem 'B'. From the software perspective the coupling is coincidental and if the rules are followed, element B4 should be moved to subsystem 'A' to become element

A5 and reduce the number of interfaces to a single interface as shown in Figure 18.3. The systems engineering rules are slightly different and depend on the situation. For example:

- If subsystem 'A' is the flight subsystem of an aerial reconnaissance system and subsystem 'B' is the ground control subsystem, then B4 may be located in the ground subsystem because it may consume too much power or be too heavy to fly. In such a situation, it is the role of the systems engineer to monitor the rate of change in technology maturity to determine that if in the future, should the system be upgraded, element B4 is a candidate for replacement with a different technology that would allow it to be moved to subsystem A4.
- Element B4 could also represent a function performed by the operator in the ground subsystem in the initial release of the operational software. This approach allows for an incremental software delivery approach where the function is intended to be migrated to the flight subsystem in subsequent software upgrades.

18.7. Optimize the interfaces

Optimizing complex systems represents a challenge for reasons that include:

- There will usually be different viewpoints on what should be optimized.
- Traditional approaches to complex systems development either ignore the issue or optimize subsystems.
- The system optimization paradox discussed in Section 9.24.1.

The system optimization paradox can be dissolved as discussed in Section 12.26.3. System optimization at any level optimizes the interactions between the subsystems at that system level within the constraints imposed by the systems engineer of the metasystem, via:

- The "*the proper allocation of the system requirements to the subsystems*" (Wymore, 1997).
- The rules for performing the aggregation discussed in Section 18.6.1.

This section now examines the following range of systems from a perspective in which the subsystem boundaries are redrawn to show that the SOI can be considered as having been optimised for the interactions between the subsystems:

1. Your sex life.
2. Weapons systems.

3. Logistics systems.
4. The Apollo Program.
5. Resupplying the MIR space station.
6. The human cardiovascular system.
7. A distance-learning classroom.
8. The Library.
9. Forming the INCOSE Australia chapter.

18.7.1. Your sex life

Optimizing your sex life raises several issues mostly not addressed. For example, in this situation, is each of the participants a system on their own, or are they subsystems of a greater whole? Traditional subsystem optimization approaches would result in an optimization of either the male experience or the female experience[4], while a holistic approach to optimization would seek to optimize the mutual experience by applying holistic thinking to the problem.

In such a situation, you could seek to understand the situation using perceptions from the eight descriptive HTPs on the perspectives perimeter discussed in Section 2.2.2 as a starting point. This is an iterative research situation in the manner of Hall's methodology for systems engineering (Hall, 1962) where you have to research the application domain to gain an understanding of the situation but is generally shown as a sequential process such as the one in Figure 6.1. The only practical difference is that the product of the Scientific Method is a supported hypothesis, while the product produced here is a CONOPS. Consider the process steps.

1. Observe.
2. Research.
3. Understand the situation.
4. Formulate the hypothesis for the solution.
5. Test the solution performance to verify it meets the need.

18.7.1.1. Observe

You first seek perceive the situation starting from the eight descriptive HTPs on the perspectives perimeter shown in Figure 2.4 and then using Active Brainstorming with cognizant stakeholder personnel to generate ideas. In this instance questions from the *Functional*, *Operational* and *Generic* perspectives would be good starting points. Section 6.2.2 in Volume 1 provides a list of starter questions (Kasser, 2013c).

[4] Assuming heterosexual activity in keeping with the traditional view.

18.7.1.2. Research

You then perform some research by immersing yourself in the situation or by means of a literature review or by holding discussions with domain experts to clarify issues or answer questions that came up during the Active Brainstorming sessions. You might also undertake some prototyping experiments to clarify aspects of the situation. The results of the prototyping experiments would be analysed and further research undertaken if necessary. The research findings might determine that some of the factors are subjective and depend on the person (the subsystem), the time and place (the environment), a function of age, length of relationship or other factors. In such a situation you would list these factors as solution selection criteria and determine ways to identity and weight these factors. It should be noted that this step is often overlooked, and when it is overlooked, tends to result in the formulation of the wrong problem statement.

18.7.1.3. Understand the situation

The next step is to gain an understanding of the situation as discussed in Section 18.2.

18.7.1.4. Formulate the hypothesis for the solution

The next step, assuming a linear sequence, is to formulate the problem statement in the form of a hypothesis (the *Scientific* perspective). "*A problem well stated is a problem half solved*" (Dewey, 1933). If the problem can be stated as a function, then the solution system is one that provides the needed functionality (Hall, 1989) which can be described in a CONOPS. The first version of a CONOPS constitutes a hypothesis for the operation of the solution system in its FCFDS. In this instance, you would determine the factors that make your sex life enjoyable and what signals need to be exchanged between you and your partner[5] on all interfaces (tactile, audible, visual, etc.) at all times. You (and your partner, if available) would develop a CONOPS containing scenarios for the mission and support[6] functions performed in different aspects of your sex life.

18.7.1.5. Test the hypothesis performance to verify it meets the need

The linear sequence approach teaches that once the hypothesis for the functionality of the solution has been developed in the form of the CONOPS, the hypothesis would be tested against solution selection cri-

[5] Before, during and after the actual sex act.

[6] In this instance, the support functions might be concerned with creating the appropriate environment, and ensuring that appropriate consumable supplies are available as and when needed (logistics).

teria. In reality, this is not a linear process; it is a continual process of observation, brainstorming, research and hypothesis formulation and in-process hypothesis formulation and testing as shown in Figure 6.1 so that when completed, the CONOPS represents the SOI operating in a FCFDS.

18.7.1.6. Comments

Traditional subsystem optimization would tend to result in an optimization of either the male experience or the female experience. The traditional approach might begin by considering one of the parties and optimizing the system to provide maximum pleasure for that party. The holistic approach on the other hand considers both parties as parts of a larger system and optimizes the interactions at the interface for maximum pleasure to both parties. In a really complex system, there may be a number of interfaces such that the individual interfaces may be grouped into a third high-level subsystem. Notice that there may be different subsystem boundaries in the traditional and holistic approaches as shown in the examples that follow.

18.7.2. Weapons systems

Weapons systems are initially designed to perform specific missions. The general goal of a weapons system is to deliver the required amount of something, usually, but not necessarily, explosive ordnance, to the target in a timely manner. The 'required' amount depends on the mission. For example, tanks were originally designed as part of a system that would enable troops to pass safely through territory swept by hostile machine gun fire, specifically the trenches in World War I. From the holistic thinking perspective, let the battlefield be the system and the allied forces and enemy forces be the two major subsystems (friend and foe), then the tank can be considered as an element of the interface between the friend and foe subsystems. The subsystem partitioning is reasonably traditional.

With hindsight, what actually happened can be mapped into the process discussed in Section 18.7.1 as if holistic thinking had been employed.

- ***The undesirable situation*** is the inability to break through the enemy front line trenches (swept by machine gun fire which, according to lessons learned from experience, precluded the traditional infantry or cavalry charge from performing the function) so that infantry and cavalry could then be used in their traditional manner to route the enemy after a breakthrough.
- ***The FCFDS*** is a break through into the enemy front line trenches by the application of yet-to-be-developed technology.
- ***The solution*** was unknown at the time the problem was formu-

lated.

- ***The problem*** was to provide a solution to create the FCFDS.

Various scenarios would have been conceptualized and rejected. Research would have been carried out to see if there was anything appropriate that could be employed. Concepts such as shields (hand-held or motorized) and land ships would have been prototyped and various types of tanks evolved together with the tactics for their use. In fact, the lack of holistic thinking meant that the tank was not effectively integrated into the British forces until the Battle of Amiens which began on 8 August 1918. This was the battle that led to the end of the First World War. However, by then the Germans had learned to deal with tanks. Consequently, 72% of the Allied Tank Corps was destroyed in the first days of the battle, 41.4% of all British tanks had been destroyed by the 64th day and on 5 November there were only eight tanks left in the British tank corps. Luckily, the tank was not the deciding factor in ending the war. The holistic approach might have produced a better system (integration of tanks, infantry and doctrine) and fewer casualties.

Other weapons systems subsystems partitions include 'gun-bullet-target' where the system is optimized to cause maximum damage to the target at the other end of the bullet interface.

18.7.3. Logistics systems

Once Total Cost of Ownership (TCO) and Life Cycle Costing (LCC) were taken into account at system design time, logistic systems were generally designed to support the mission and deliver optimal support to the operational system.

In the holistic view, it is the interface (subsystem) between the mission and support subsystems that keeps the mission functions operational. In many situations, once the CONOPS for the mission and support functions has been developed, the system is optimized for maximum operational availability of the operational subsystem. The trade-offs to optimize the operational availability of the mission system at design time deal with reliability, failure rates, failure modes and failure consequences, Mean Time To Repair (MTTR), etc.

18.7.4. The Apollo Program

The Apollo program was a major systems engineering success. From the *Structural* perspective, consider the Apollo program as the system containing three top-level physical subsystems, (1) the earth, (2) the lunar and (3) the interface system between the earth and lunar subsystems, where:

- ***The earth subsystem*** contained the (NASA manned spacecraft centers and headquarters.

- ***The lunar subsystem*** was empty before the first landing and then contained an increasing number of Apollo Lunar Surface Experiments Packages (ALSEP)[7], the set of scientific instruments deployed by the astronauts at each of the landing sites. Two astronauts were part of this subsystem while on they were on the lunar surface.
- ***The interface subsystem*** contained the spacecraft, the astronauts (three while in transit, one when in lunar orbit) and the NASA Communications Network (NASCOM) communications subsystems.

From this perspective, the Apollo program seems to have been optimized to transfer men and ALSEPs between the earth and the moon in the most efficient manner within the constraints of the then available technology. This resulted in a manually intensive complex understandable earth subsystem. Unfortunately this subsystem arrangement was perpetuated into the post Apollo era for various reasons resulting in a minimally reusable overly expensive space transportation system commonly known as the Space Shuttle.

In addition note how the subsystem boundaries changed during the mission. The astronauts moved between the interface subsystem and the lunar subsystem. The Lunar Lander was originally a part of the interface subsystem and then became a part of the lunar subsystem when it was left behind after the return ascent.

18.7.5. Resupplying the MIR space station

MIR was a Soviet/Russian space station in Low Earth Orbit (LEO) from 1986 to 2001. When faced with the problem of resupplying MIR, the subsystem boundaries remained the MIR, the earth and the interface subsystem. The system was optimized for the delivery of personnel and cargo to MIR, personnel being delivered by manned vehicles and cargo mainly by unmanned autonomous vehicles. Simple, readily understandable and effective!

18.7.6. The human cardiovascular system

The human cardiovascular system delivers oxygen to the muscles in the human body. Here the system can be represented by the lung subsystem which oxygenates the blood, the muscles subsystem, and the heart and blood vessels which comprise the bulk of the interface to the between the lungs and the muscles subsystems.

[7] Each flight transferred an ALSEP from the earth to the lunar subsystems.

18.7.7. A distance-learning classroom

The distance-learning classroom at Missouri University of Science and Technology (MS&T) for SysEng 412 Complex Engineering Systems Project Management in the Fall 2010 semester was a complex system. The traditional non-holistic view might have organised the subsystems as:

- A face-to-face classroom at MS&T equipped with the appropriate synchronous technology for including distant students in the learning process.
- The students in the face-to-face classroom[8].
- A synchronous distance-learning classroom using the Webex platform.
- An asynchronous distance-learning classroom using the Blackboard 9 platform.
- The distance mode students in the USA.
- The instructor in Singapore.
- The email system for asynchronous communications.
- The real-time support staff at MS&T. Note, support was available online during each weekly session and offline in non-real time with a timely response.

On the other hand, the holistic perspective partitions the system into two subsystems and an interface system. The subsystems are the:

- Instructor.
- Students.
- Interface subsystem consisting of the classrooms and other facilities.

The system was designed to optimize the learning experience based on the needs of postgraduate employed students studying in their spare time (Kasser, et al., 2008). The design of this iteration of SysEng 412 included a mixture of lectures, readings and problem-based learning activities using both synchronous and asynchronous activities. When the semester began, the study materials were loaded into Blackboard for asynchronous downloading prior to the weekly Webex synchronous session. The lecture was given synchronously; the students worked together synchronously and asynchronously and made a synchronous presentation in the weekly Webex synchronous sessions. However, a week or so after the semester began an anomaly showed up in the synchronous lectures. The instructor's Webex audio suffered from distortion that made it unintelligible at times according to some but not all students. Upon enquiring

[8] There weren't any in this instance.

about the situation, the support staff acknowledged that this was a recurring problem when the instructor was located outside the US.

The interface system was quickly redesigned to keep the learning experience optimal. Subsequent lectures were pre-recorded as MP3 voice quality bandwidth audio files and uploaded to the Blackboard area for the specific session together with the lecture slides. The students downloaded the lecture audio files together with the lecture slides and listened to the lecture asynchronously prior to the Webex synchronous session. The redesigned lecture faced a delivery domain problem due to the differences between synchronous and asynchronous lectures. The major one being that the students could not ask questions in an interactive synchronous manner. This drawback was overcome using domain knowledge in the following manner.

- The instructor would cue the students to change slides in the pre-recorded lectures using wording such as "and on the next slide". Additionally, every now and again during the talk, the instructor would mention the slide number as a synchronization signal. At the appropriate points in the lecture where the instructor would pause and ask for questions, an 'any questions slide' was inserted into the lecture slides. The questions were posed asynchronously and a comment was added to each question that answers would be provided in the interactive session.
- The asynchronous lecture was reformatted to allow for multiple threads so that later content did not depend on a previous discussion in the same session.
- During the interactive synchronous session, the instructor paged through the lecture slides summarizing the lecture, sometimes adding additional information and always stopping at the appropriate places for questions and comments.

The students soon caught on to the idea and the end result was a shortened synchronous session which allowed the students to spend more time on the problem-based learning activities (even more optimal). Indeed the system was flexible enough so that on one occasion when the instructor was travelling to a conference at the exact time the synchronous session was due to take place, the pre-warned students were able to prepare and upload asynchronous presentations to Blackboard and the whole session took place asynchronously (presentations and post presentation dialogue (questions and comments)) in Blackboard.

18.7.8. The Library

The library-patron system provides desired information (books, journals, and publications) sourced in, or obtained by, the library subsystem to patron subsystem. Libraries have been optimizing the interface delivery for years finding newer and better ways to provide patrons with the desired information. Librarians just call this providing better service.

18.7.9. Forming the INCOSE Australia chapter

After the Memorandum of Understanding (MOU) between the INCOSE and the Systems Engineering Society of Australia (SESA) expired in 2004, the members of SESA attending its annual general meeting voted that SESA not become a chapter of INCOSE and remain an independent organization. This left an undesirable situation in which there was a desire and support for a chapter of INCOSE in Australia, while at the same time the overwhelming majority of Australian systems engineers wanted a single professional organisation for systems engineers in Australia and feelings were running high on the issue. The innovative solution which came from the *Generic* perspective was to constitute a chapter of INCOSE in Australia, INCOSE-Australia as *a special interest group within SESA*. This solution:

- Avoided a "civil war" within the systems engineering profession in Australia.
- Meant that nobody could join INCOSE-Australia without being a member of both INCOSE and SESA.
- Allowed those SESA members who desired INCOSE services and products to obtain them without having to join two professional societies;
- Allowed those systems engineers that did not desire the INCOSE products and services to be part of SESA.

In this situation, Australia has a single systems engineering professional society within the Institute of Engineers Australia (IEAust) constituency namely SESA. However, as far as INCOSE is concerned there are two systems engineering professional societies in Australia. In the traditional view, the two societies may be viewed as subsystems of the systems engineering community in Australia (the system). The innovative solution was made possible by considering SESA as containing the following non-traditional three functional subsystems:

- INCOSE Australia which constituted the members of SESA who were also members of INCOSE.
- The remaining non-INCOSE membership of SESA.
- The SESA Headquarters which received the dues payment from INCOSE.

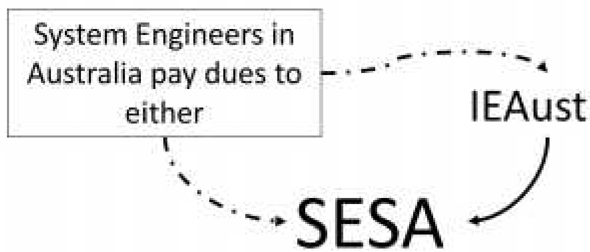


Figure 18.4 SESA dues payment process

The system was optimized for minimal interface activity on the interfaces between the subsystems by a simple modification or addition to the existing system. The original dues paying process shown in Figure 18.4 allowed systems engineers in Australia to pay their dues directly to SESA or via the IEAust. The modified dues paying process is shown in Figure 18.5. The modification allowed Australian systems engineers who wished to be part of INCOSE to pay their membership dues to INCOSE directly just like any other regular INCOSE member anywhere else in the world. INCOSE then bulk refunded a portion of the dues to INCOSE Australia but made the payment directly to SESA; the refunded portion covering the membership dues for SESA. The single individual dues payment to INCOSE provided membership of both organizations. In addition, INCOSE Australia did not need a bank account, as INCOSE Australia incurred no costs since all professional systems engineering society activities in Australia were SESA activities by definition. The only information that needed to be exchanged at the interface between the INCOSE Australia and SESA was the list of INCOSE Australia members that was passed to SESA for the purpose of providing mailing labels

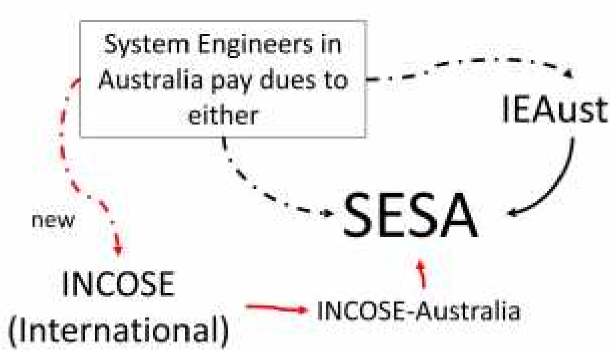


Figure 18.5 Modified SESA dues payment process

for the quarterly SESA newsletter. As a serendipitous benefit, SESA had the advantage of autonomy from INCOSE and did not have to conform to any INCOSE rules and regulations[9].

The elected officers of INCOSE Australia had little to do on behalf INCOSE Australia other than remembering to hold the required annual general meeting[10]; in particular there was nothing for the treasurer to do.

18.7.10. Discussion

The holistic approach to optimizing a system may be defined as an approach that *optimises* the system for *the interactions between the subsystems* at design time, rather than an approach that optimizes the subsystems after the subsystem boundaries have been determined. This approach is self-similar and should apply to any level in the system hierarchy thus dissolving the paradox/problem discussed by Machol and Miles. In each of the examples discussed in Section 18.7, even though the systems are complex, understanding the system functionality is reasonably straightforward. This is because the functionality of each subsystem can be understood, as can the interactions at the interface.

In some of the examples the subsystem boundaries were traditional, in others they were non-traditional. The tank development can be mapped into the holistic approach but the development wasn't holistic and the results were less than optimal. The objective was achieved but the price in loss of lives and materiel was higher than it could and should have been. The holistic approach to designing a system is a slightly different approach from that currently employed. It is a structured hierarchical approach to design and analysis. The functional allocation of the CONOPS is mapped into two major physical subsystems and an interface (subsystem) between them. The interfaces between the functional subsystems are then optimized.

Domain knowledge in the problem, solution and implementation domains (Section 9.11) is a critical element in the holistic approach to optimizing complex systems. The systems engineer uses the domain knowledge to visualize a conceptual two subsystems and optimized interface implementation of the CONOPS.

It was an analysis of the holistic approach to improving your sex life that provided the insight to create the two subsystems and optimal interface approach to optimizing complex systems. Use of the approach

---

[9] The downside was that the very small number of SESA members who were members of IEAust and also wanted to be members of INCOSE had to pay dues to both IEAust and INCOSE, which is what they had to do before the modification.

[10] All professional systems engineering society activities with Australia are SESA activities by definition.

should also provide a serendipitous indirect benefit: not worrying about how to understand and optimize complex systems should reduce stress and consequently also improve your sex life.

18.8. Partition the SOI into subsystems

Once the FCFDS has been partitioned into its subsystems, the SOI and adjacent systems, by the metasystem systems engineer, the SOI systems engineer then partitions the SOI into subsystems using the same process for creating a system, namely by going back to Section 18.1 and working on the SOI. This is in accordance with the concept that one systems engineer's subsystem is another systems engineer's system in the hierarchy of systems.

The internal subsystem partitioning within each adjacent system are the province of the particular adjacent system systems engineer just like the internal details of the SOI are the province of the SOI systems engineer. Note:

- In some cases the system boundaries may need to change over time, such as when an organization is reorganized and as discussed in cohesion and coupling above.
- The metasystem systems engineer may occasionally override the SOI subsystem partitioning to meet metasystem requirements as discussed in Section 18.6.1.3.
- If subsystems are moved between systems such as in the Apollo program (Section 18.7.4), the changes may be considered as different iterations of the O&M State of the generic extended SLC (Section 13.4), or as different systems.

18.9. The recursive perspective

As may be noted from Section 18.8, the process for creating systems is recursive. The first time through the process, the SOI is the entire undesirable situation (S1) which is partitioned into the SOI (S6) and adjacent systems. The second time through the process, S1 is the solution system, the SOI, and the undesirable situation is the need to partition the SOI into its subsystems. The adjacent systems are the province of their own systems engineers. And so on down the system-subsystem hierarchy

18.10. The contribution of the HTPs to the system requirements

If the SOI is going to be created, then the SDP (S5) includes the production of a matched set of specifications for the SOI (S6) and each of its subsystems. In general, the:

- ***Big picture*** perspective contributes to the interface requirements.

- ***Operational*** perspective contributes to the performance requirements.
- ***Functional*** perspective contributes to the functional requirements.
- ***Structural*** perspective contributes to the technology, physical and 'ility' requirements, e.g., reliability, maintainability, survivability, etc.
- ***Generic*** perspective contributes requirements than can be inherited from that class of system.
- ***Continuum*** perspective contributes to identifying differences between the SOI and similar systems that affect the requirements. For example, some of the requirements may not be as stringent, or may be more stringent than those of a particular similar system.
- ***Temporal*** perspective contributes to the requirements for adoption of new technology, managing obsolescence and flexibility to adapt to future situations.
- ***Quantitative*** perspective provides the numbers and tolerances for the requirements.

18.11. Summary

This Chapter:

- Improved systems engineering by filling a gap in the systems engineering literature by suggesting a process for creating systems to be used in the early states of the SDP to help to manage complexity at the time the system is created by optimizing the interfaces.
- Described the S2 process in the Nine-System Model discussed in Chapter 16.
- Described the contribution of the HTPs to the systems requirements.

--oo--

19. A way to measure technical progress and identify potential problems in near real-time

In both current paradigms of the current SDP the cost and schedule tracking tools can measure the amount of:

1. Money that has actually been spent.
2. Work that has actually has been done.

However, the cost and schedule tracking tools cannot:

1. Answer the question "How much of my project has been completed?"
2. Identify potential technical problems in near real-time so as to mitigate them.

This Chapter improves systems engineering by providing a way to measure technical progress and identify potential problems in near real-time so as to be able to mitigate the problems before they occur.

While simplistic approaches of tracking the realization activities of all the requirements or features such as Feature Driven Development (FDD) (Palmer and Felsing, 2002) can inform about the state of the realization activities, they cannot be used to estimate the degree of completion since each requirement or feature has a different level of complexity and takes a different amount of effort to realize. The need is for a measurement approach that can:

- Roll up the detailed information into a summary that can be displayed in one or two charts.
- Readily relate to the existing cost and schedule information.

The Categorized Requirements in Process (CRIP) approach (Kasser, 1999) presented in the chapter meets that need by looking at the change in the state of a summary of the realization activities which convert requirements into systems during the SDP from several perspectives. The summary information is presented in a table known as a CRIP Chart which:

- Cover the entire SDP.
- Use a technique similar to FDD charts.
- Provide summaries suitable for management.

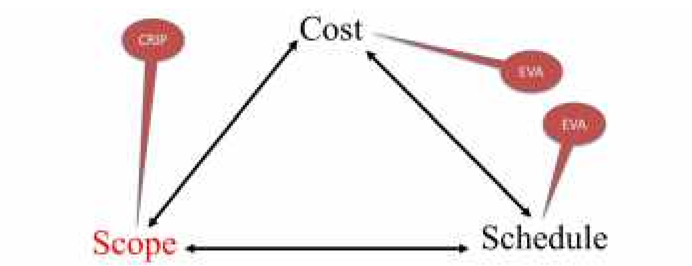


Figure 19.1 The relationship between CRIP charts and EVA

- Indicate variances between plan and progress but not the causes of the variance. It is up to management to ask for explanations of the variances.
- Are based on the Work Package (WP) approach to planning a project, hence have to be integrated into the SDP at the planning stage of a project.
- Are designed to be used in association with EVA budget and schedule information as shown in Figure 19.1.

19.1. The five-step CRIP approach

The five-step CRIP approach is:

1. Identify categories for the requirements
2. Quantify each category into ranges.
3. Categorize the requirements.
4. Place each requirement into a range.
5. Monitor the differences in the state of each of the requirements at the SDP formal and informal reporting milestones

The last step is the key element in the CRIP approach because it is a dynamic measure of change rather than a static value. Consider each of the steps.

19.1.1. Identify Categories for the requirements

Categories are identified for each of the requirements based on information in the Work Packages (WP). Typical categories are:

- ***Priority*** of the requirement.
- ***Complexity*** of the requirement.
- ***Estimated cost*** to implement the requirement.
- ***Risk*** - probability of occurrence, severity if it occurs, etc.

Every requirement shall be placed in each category.

19.1.2. Quantify each category into ranges

Each category is then quantified into no more than ten ranges. Thus, for:

- ***Priority:*** requirements may be allocated priorities between one and ten.
- ***Complexity:*** requirements may be allocated estimated complexities between "A" and "J".
- ***Estimated cost to implement:*** requirements may be allocated estimated costs to implement values between "A" and "J".
- ***Risk:*** requirements may be awarded a value between one and five.

The ranges are relative, not absolute. Any of the several quantitative techniques for sorting items into relative ranges may be used. The buyer/customer and supplier/contractor determine the range limits in each category.

A requirement may be moved into a different range as more is learned about its effect on the development or the relative importance of the need changes during the SDP. Thus, if the priority of a specific requirement or the estimated cost to implement changes between SDP reporting milestones changes, the requirement may be moved from one range to another. However, the rules for setting the range limits, and the range limits must not change during the SDP.

19.1.3. Place each requirement into a range

Each requirement is then placed into one range slot for each category. If all the requirements end up in the same range slot, such as all of them having the highest priority, the range limits should be re-examined to spread the requirements across the full set of range slots.

The information used to place the requirements into the ranges for the categories comes from the Work Packages (WP) in the Project Plan (PP).

19.1.4. States of implementation

Each requirement shall be in one, and only one, of five CRIP states at any time during the SDP. These CRIP states of implementation of each requirement during the project are:

1. ***Identified:*** A requirement has been identified, documented and approved.
2. ***In-process:*** The supplier has begun the development activities to realize the requirement.
3. ***Completed:*** The supplier has completed development activities on the requirement.

Table 19.1 An unpopulated CRIP Chart

	Identified			In process			Completed			In test			Accepted		
Range	P	E	A	P	E	A	P	E	A	P	E	A	P	E	A
1															
2															
3															
4															
5															
6															
7															
8															
9															
10															

'P' links to 'Work to be scheduled for next period'
'E' links to 'Work Scheduled at last milestone for period'
'A' links to 'Work Performed in last period'

4. ***In test:*** The supplier has started to test compliance to the requirement.
5. ***Accepted:*** The buyer has accepted delivery of the part of the system (a Build) containing the implementation of the requirement.

The summaries of the number of requirements in each state are reported at project milestones.

19.1.5. Populating and using the CRIP Chart

An unpopulated CRIP Chart is shown in Table 19.1 where:

- ***The vertical axis*** of the chart is split into the ten ranges in the category.
- ***The horizontal axis*** of the chart is split into five columns representing the CRIP states of a project.

Each CRIP state contains three cells; planned 'P', expected 'E' and actual 'A', where:

- ***[P] Planned for next reporting period:*** The number of requirements planned to be in the CRIP state before the following reporting milestone.
- ***[E] Expected:*** The number of requirements expected to be in the CRIP state, based on the number planned in the previous reporting milestone. This is a copy of the 'P' value in the CRIP Chart for the previous milestone.
- ***[A] Actual:*** The number of requirements actually in the CRIP state.

For the first milestone-reporting period, the values for "expected" 'E' are derived from the PP for the time period. The "actual" value 'A' is

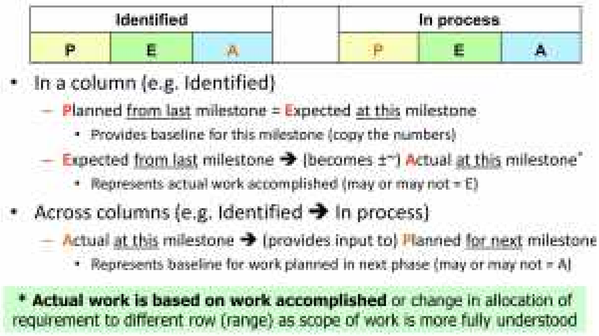


Figure 19.2 Movement between CRIP states

the number measured at the end of the reporting period, and the "planned for next reporting period" value 'P' is a number derived from the PP and the work done during the current reporting period.

As of the first milestone following the start of a project, the numbers in the 'P' column of a CRIP state of the chart at one milestone are always copied into the 'E' column of the same CRIP state in chart for the next milestone. The 'A' and 'P' values reflect the reality. As work progresses the numbers flow across the CRIP states from "Identified" to "Accepted" as shown in Figure 19.2.

At each reporting milestone, the *changes* in the *CRIP state* of each of the requirements *between* the milestones are monitored. The numbers of each of the requirements in each of the categories are presented in tabular format in a CRIP Chart at reporting milestones (major reviews or monthly progress meetings). Colours can be used to draw attention to the state of a cell in the table. For example the colours can be allocated[1] such that:

- ***Violet:*** shows realization activities for requirements in that range is well ahead of estimates.
- ***Blue:*** shows realization activities for requirements in that range is ahead of estimates.
- ***Green:*** shows realization activities for requirements in that range is close to estimates.
- ***Yellow:*** shows realization activities for requirements in that range is slightly below estimates.
- ***Red:*** shows realization activities for requirements in that range is well under the estimates.

[1] The quantitative numbers for the ranges would be agreed upon between the stakeholders and specified in the contract prior to the commencement of the project and not changed during the SDP.

One CRIP Chart can show that a problem might exist. Any time there is a deviation between 'E' and 'A' in a CRIP State, the situation needs to be investigated. A comparison of the summaries from different reporting milestones can identify progress and show that problems may exist. On its own however, the chart cannot identify the actual problem.

The CRIP Charts when viewed over several reporting periods can identify other types of "situations". While the CRIP Chart can be used as a stand-alone chart, it should really be used together with EVA budget and schedule information. For example, if there is a change in the number of:

- ***Identified requirements and there is no change in the budget or schedule:*** there is going to be a problem. Thus, if the number of requirements goes up and the budget does not, the risk of failure increases because more work will have to be done without a change in the allocation of funds. If the number of requirements goes down, and the budget does not, there is a financial problem[2].
- ***Requirements being worked on, and there is no change in the number being tested:*** there is a potential supplier management or technical problem if this situation is at a major milestone review.
- ***Requirements being tested, and there is no change in the number accepted:*** there may be a problem with the supplier's process or a large number of defects have been found and are being reworked.
- ***Identified requirements at each reporting milestone:*** the project is suffering from requirements creep if the number is increasing. This situation may reflect controlled changes due to the change in the customer's need, or uncontrolled changes.

Since projects tend to delay formal milestones until the planned work is completed, the CRIP Charts are more useful in the monthly or other periodic meetings between the formal major milestones.

19.1.6. Advantages of the CRIP Approach

The advantages of the CRIP approach include:

- Can be used in both the 'A' and 'B' paradigms (Section 9.21).
- Links all work done on a project to the customer's requirements.
- May be used at any level of system decomposition.
- Provides a simple way to show progress or the lack of it, at any

[2] Unless it is in the context of a fixed price contract; in which case it shows additional profit.

reporting milestone. Just compare the 'E' and 'A' numbers and ask for an explanation of the variances.

- Provides a window into the project for top management (buyer and supplier) to monitor progress.
- Can indicate if lower priority requirements are being realized before higher priority requirements if priority is a category.
- Identifies the probability of some management and technical problems as they occur, allowing proactive risk containment techniques.
- May be built into requirements management, and other computerized project and development management tools.
- May be incorporated into the progress reporting requirements in system development contracts. Falsifying entries in the CRIP Chart to show false progress then constitutes fraud.
- Requires a process. Some organisations don't have one, so they will have to develop one to use CRIP Charts.
- Requires Configuration Management (CM) which tends to be poorly implemented in many organisations. The use of CRIP Charts will force good CM.

19.1.7. Disadvantages of the CRIP Approach

The CRIP Chart approach has the following disadvantages, it:

- Is a different way of viewing project progress.
- Requires categorization of the requirements.
- Requires sorting of the requirements into ranges in each category.
- Requires prioritisation of requirements if priority is used as a category, which it should be.

19.2. Examples CRIP Charts in different types of projects

The subsequent sections show how CRIP Charts can indicate the technical progress of a project and identify potential problems using the following stereotype examples:

1. An ideal project.
2. A project with requirements creep.
3. A challenged project.
4. A make up your mind project.

The projects are all identical until completion of SRR.

Since projects tend to delay formal milestones until the planned work is completed, the CRIP Charts are more useful in the monthly or other

Table 19.2 CRIP Chart at RFP time (Cost Category)

	Identified			In process			Completed			In test			Accepted		
Range	P	E	A	P	E	A	P	E	A	P	E	A	P	E	A
1	0		86	0											
2	0		73	0											
3	0		23	0											
4	0		34	0											
5	0		26	0											
6	0		15	0											
7	0		8	0											
8	0		7	0											
9	0		5	0											
10	0		2	0											
Totals	0		279	0											

periodic meetings between the formal major milestones. However, for the example of these stereotype projects, the CRIP Charts are provided for the formal milestones since their names are widely known.

One of Federated Aerospace's (FA) projects provides examples of the use of CRIP Charts as follows. Federated Aerospace organized a proposal team to bid on a RFP issued by a Government agency. Upon receipt of the RFP the project team identified 279 requirements in the document. The proposal team estimated the costs to realize those requirements as part of the proposal effort. Once the costs were estimated, the proposal team defined ten ranges of costs and allocated each requirement into the appropriate range. The CRIP Chart at the completion of the proposal shown in Table 19.2 indicates that:

- The RFP contained 279 requirements.
- The requirements have been grouped into 10 cost ranges. There are 86 requirements in Range 1, 73 in Range 2, 23 in Range 3 and so on.
- No further work is planned at this time as shown by the zero values assigned to the 'P' columns of the 'Identified' and the 'In process' States[3].

FA's proposal was accepted, and the Government awarded a contract to FA for the project. This example provides typical CRIP Charts for the following major milestones: SRR, PDR, CDR, TRR, IRR and DRR.

The FA development stream of activities in the project started by confirming that all the requirements in the RFP:

- Were understood by the FA project team.

[3] This would change if Federated Aerospace wins the contract award.

Table 19.3 CRIP Chart at the start of the project

	Identified			In process			Completed			In test			Accepted		
Range	P	E	A	P	E	A	P	E	A	P	E	A	P	E	A
1	0	86	0												
2	0	73	0												
3	0	23	0												
4	0	34	0												
5	0	26	0												
6	0	15	0												
7	0	8	0												
8	0	7	0												
9	0	5	0												
10	0	2	0												
Totals	0	279	0												

- Had not changed since the RFP had been issued.
- Were complete; there were no additional or deleted requirements which would change the scope and cost of the contract.
- Were tagged with acceptance criteria[4].

The CRIP Chart at the start of the ideal project shown in Table 19.3, based on the information in the RFP CRIP Chart in Table 19.2 indicated that:

- There would be no planned change between the number of requirements identified at SRR and those identified in the RFP since:
  - The numbers in each row of the 'E' column in Table 19.3[5] match those in the corresponding rows of the 'A' column in Table 19.2.
  - The number in each row in the 'P' column in the 'Identified' State of in Table 19.3 has been set to zero since there are no planned additional requirements.

Note, as of the first milestone following the start of a project, the numbers in the 'P' column of a State of the CRIP Chart at one milestone are always copied into the 'E' column of the same State in the CRIP Chart for the next milestone as shown in Table 19.2.

The CRIP Chart at SRR shown in Table 19.4 indicates that the project has deviated from the baseline plan since:

- There are differences between the expected numbers and the ac-

---

[4] Not used in CRIP Charts, but needed elsewhere.

[5] This is the only time in a project when 'A' column numbers are copied from one CRIP State in a CRIP Chart at one reporting period to the CRIP Chart of the following CRIP State.

Table 19.4 The CRIP Chart at SRR

	Identified			In process			Completed			In test			Accepted		
Range	P	E	A	P	E	A	P	E	A	P	E	A	P	E	A
1	0	86	81	81											
2	0	73	78	78											
3	0	23	35	35											
4	0	34	30	30											
5	0	26	26	26											
6	0	15	20	20											
7	0	8	8	8											
8	0	7	7	7											
9	0	5	5	5											
10	0	2	2	2											
Totals	0	279	292	292											

tual numbers of identified requirements since there are differences between the numbers in several rows of the 'A' column and the 'E' column of the 'Identified' State. For example, in Range 1 an 'E' number of 86 became an 'A' of 81 and in Range 2 an 'E' number of 73 became an 'A' of 78. Changes can also be seen in Ranges 4 and 6[6].

- The total number of identified requirements has increased from 279 to 292[7].
- The project development team plans to work on all the requirements to put them into a development following the conclusion of the SRR since the numbers from the 'A' column in the 'Identified' State have been copied into the 'P' column of the 'In process' State.
- The project does not plan to identify any new requirements between SRR and PDR since all the rows in the 'P' column of the 'Identified' State have been reset to zero.

The stereotype projects diverge after SRR. Consider how the CRIP Charts provide early identification of the technical progress or lack of progress (an indication of a potential problem) in the milestone reviews of the stereotype projects using the ideal project as a reference.

---

[6] Upon investigation it was found that the changes in the number of requirements are due to the clarifications that occurred during the requirements elicitation and elucidation process, a typical project happening.

[7] The cost and schedule was renegotiated as a result and is reflected in the updated cost and schedule summaries also presented in the SRR (not included herein).

Table 19.5 The ideal project CRIP Chart at PDR

	Identified			In process			Completed			In test			Accepted		
Range	P	E	A	P	E	A	P	E	A	P	E	A	P	E	A
1	0	0	0	0	81	81	0								
2	0	0	0	0	78	78	0								
3	0	0	0	0	35	35	0								
4	0	0	0	0	30	30	0								
5	0	0	0	0	26	26	0								
6	0	0	0	0	20	20	0								
7	0	0	0	0	8	8	0								
8	0	0	0	0	7	7	0								
9	0	0	0	0	5	5	0								
10	0	0	0	0	2	2	0								
Totals	0	0	0	0	292	292	0								

19.3. The ideal project

The ideal project is the one in which everything happens according to the plan and there are no changes in the requirements during the SDP, such as in a short duration project or an educational example.

19.3.1. The ideal project CRIP Chart at PDR

The ideal project CRIP Chart at PDR shown in Table 19.5 indicates that the project is proceeding according to plan since:

- No additional requirements were levied on the system as indicated by the zero value in all the rows in 'A' column in the 'Identified' State.
- The System Design State activities (Section 9.12.3) commenced as expected since the 'A' numbers in the 'In process' State match the 'E' numbers.
- The project does not plan to complete the development of any requirements by CDR since there is a zero value in all of the rows in the 'P' column in the 'Completed'. This is because the CDR will be held before the end of the 'In process' CRIP State.
- The project does not plan to identify any new requirements between PDR and CDR since all the rows in the 'P' column of the 'Identified' CRIP State remain at zero.

19.3.2. The ideal project CRIP Chart at CDR

The ideal project CRIP Chart at CDR shown in Table 19.6 indicates that the project is still proceeding according to plan since:

- No additional requirements were levied on the system as indicated by the zero value in all the rows in the 'A' column of the 'Identified' State.

Table 19.6 The ideal project CRIP Chart at CDR

	Identified			In process			Completed			In test			Accepted		
Range	P	E	A	P	E	A	P	E	A	P	E	A	P	E	A
1	0	0	0	0	81	81	81								
2	0	0	0	0	78	78	78								
3	0	0	0	0	35	35	35								
4	0	0	0	0	30	30	30								
5	0	0	0	0	26	26	26								
6	0	0	0	0	20	20	20								
7	0	0	0	0	8	8	8								
8	0	0	0	0	7	7	7								
9	0	0	0	0	5	5	5								
10	0	0	0	0	2	2	2								
Totals	0	0	0	0	292	292	292								

- The plan is for all the development activities to be completed by TRR since all the numbers in the 'A' column of the 'In-process' State have been copied into the 'P' column of the 'Completed' State.
- The project does not plan to identify any new requirements between CDR and TRR since all the rows in the 'P' column of the 'Identified' State remain at zero.

19.3.3. The ideal project CRIP Chart at TRR

The ideal project CRIP Chart at TRR shown in Table 19.7 indicates that the project is still proceeding according to plan since:

- No additional requirements were levied on the system as indicated by the zero value in all the rows in the 'A' column of the 'Identified' State.
- The development activities for the system have been completed

Table 19.7 The ideal project CRIP Chart at TRR

	Identified			In process			Completed			In test			Accepted		
Range	P	E	A	P	E	A	P	E	A	P	E	A	P	E	A
1	0	0	0	0	0	0	0	81	81	81					
2	0	0	0	0	0	0	0	78	78	78					
3	0	0	0	0	0	0	0	35	35	35					
4	0	0	0	0	0	0	0	30	30	30					
5	0	0	0	0	0	0	0	26	26	26					
6	0	0	0	0	0	0	0	20	20	20					
7	0	0	0	0	0	0	0	8	8	8					
8	0	0	0	0	0	0	0	7	7	7					
9	0	0	0	0	0	0	0	5	5	5					
10	0	0	0	0	0	0	0	2	2	2					
Totals	0	0	0	0	0	0	0	292	292	292					

Table 19.8 The ideal project CRIP Chart at IRR

	Identified			In process			Completed			In test			Accepted		
Range	P	E	A	P	E	A	P	E	A	P	E	A	P	E	A
1	0	0	0	0	0	0	0	0	0	0	81	81	81		
2	0	0	0	0	0	0	0	0	0	0	78	78	78		
3	0	0	0	0	0	0	0	0	0	0	35	35	35		
4	0	0	0	0	0	0	0	0	0	0	30	30	30		
5	0	0	0	0	0	0	0	0	0	0	26	26	26		
6	0	0	0	0	0	0	0	0	0	0	20	20	20		
7	0	0	0	0	0	0	0	0	0	0	8	8	8		
8	0	0	0	0	0	0	0	0	0	0	7	7	7		
9	0	0	0	0	0	0	0	0	0	0	5	5	5		
10	0	0	0	0	0	0	0	0	0	0	2	2	2		
Totals	0	0	0	0	0	0	0	0	0	0	292	292	292		

since the numbers in the 'A' column of the 'Completed' State match those in the 'E' column.

- Testing of all the requirements is expected to begin immediately following TRR since the numbers in the 'A' column of the 'Completed' State have been copied into the 'P' column of the 'In test' State.
- The project does not plan to identify any new requirements between TRR and the following milestone since all the rows in the 'P' column of the 'Identified' State remain at zero.

19.3.4. The ideal project CRIP Chart at IRR

The ideal project CRIP Chart at IRR, shown in Table 19.8 indicates that the project is still proceeding according to plan since:

- No additional requirements were levied on the system as indicated by the zero value in all the rows in the 'A' column of the 'Identified' State.
- Testing has begun to verify that the system meets all the requirements since all the numbers in the 'A' column of the 'In test' State match those in the 'E' column.
- The project is planning to integrate the system for successful acceptance by the customer before the DRR since the values in the 'A' column of the 'In test' State have been copied into the 'P' column of the 'Accepted' State
- The project does not plan to identify any new requirements between IRR and the following milestone since all the rows in the 'P' column of the 'Identified' State remain at zero.

Table 19.9 The ideal project CRIP Chart at DRR

	Identified			In process			Completed			In test			Accepted		
Range	P	E	A	P	E	A	P	E	A	P	E	A	P	E	A
1	0	0	0	0	0	0	0	0	0	0	0	0	0	81	81
2	0	0	0	0	0	0	0	0	0	0	0	0	0	78	78
3	0	0	0	0	0	0	0	0	0	0	0	0	0	35	35
4	0	0	0	0	0	0	0	0	0	0	0	0	0	30	30
5	0	0	0	0	0	0	0	0	0	0	0	0	0	26	26
6	0	0	0	0	0	0	0	0	0	0	0	0	0	20	20
7	0	0	0	0	0	0	0	0	0	0	0	0	0	8	8
8	0	0	0	0	0	0	0	0	0	0	0	0	0	7	7
9	0	0	0	0	0	0	0	0	0	0	0	0	0	5	5
10	0	0	0	0	0	0	0	0	0	0	0	0	0	2	2
Totals	0	0	0	0	0	0	0	0	0	0	0	0	0	292	292

19.3.5. The ideal project CRIP Chart at DRR

The ideal project CRIP Chart at DRR shown in Table 19.9 indicates that the project is still proceeding according to plan since:

- No additional requirements were levied on the system as indicated by the zero value in all the rows in the 'A' column of the 'Identified' State.
- The integrated system has been accepted by the customer as having met all its requirements as indicated by the match between each of the values of the rows in the 'A' column and the corresponding rows in the 'E' column in the 'Acceptance' State.

19.4. A project with requirements creep

This section shows how the CRIP Charts can indicate that a project has requirements creep. Assume that the project has completed the SRR shown in Table 19.4 and that the changes in the number of requirements identified occur during the System Design State (Section 9.12.3) and the Subsystem Construction States (Section 9.12.4) between SRR and TRR.

The project has chosen to hold the milestone reviews as scheduled even though the work on the additional requirements may be out of phase with the original requirements. This is fine when the additional requirements can be realised without impacting the original planned work such as when the additional requirements are for additional functionality which can be provided independently and integrated into the system as a separate plug in.

If the project had chosen a two-step iterative generic extended SDP (Figure 13.7), each SDP should have had its own set of EVA and CRIP Charts.

Table 19.10 The CRIP Chart for a project with requirements creep at PDR

	Identified			In process			Completed			In test			Accepted		
Range	P	E	A	P	E	A	P	E	A	P	E	A	P	E	A
1	0	0	20	0	81	101	0								
2	0	0	0	0	78	78	0								
3	0	0	0	0	35	35	0								
4	0	0	0	0	30	30	0								
5	0	0	2	0	26	28	0								
6	0	0	0	0	20	20	0								
7	0	0	4	0	8	12	0								
8	3	0	0	0	7	7	0								
9	0	0	0	0	5	5	0								
10	0	0	0	0	2	2	0								
Totals	3	0	26	0	292	318	0								

19.4.1. The CRIP Chart for a project with requirements creep at PDR

Twenty-six unexpected additional requirements were identified between SRR and PDR resulting in the total number of requirements increasing from 292 to 318. The CRIP Chart for the project with requirements creep at PDR shown in Table 19.10 indicates:

- Twenty of the unexpected requirements were identified in Range 1 shown by the 20 in the 'A' column of the 'Identified' State.
- Development activities have begun on these 20 requirements since the value of 81 in the 'A' column of the 'In process' State has become 101, namely the original 81 and the additional 20.
- Two of the unexpected requirements were identified in Range 5 shown by the 2 in the 'A' column of the 'Identified' State.
- Development activities have begun on these 2 requirements since the value of 26 in the 'A' column of the 'In process' State has become 28, namely the original 26 and the additional 2.
- Four of the unexpected requirements were identified in Range 7 shown by the 4 in the 'A' column of the 'Identified' State.
- Development activities have begun on these 4 requirements since the value of the 'A' column of the 'In process' State has become 12, namely the original 8 and the additional 4.
- Three of the additional requirements in Range 8 are expected to be identified after PDR as indicated by the 3 in row 8 of the 'P' column of the 'Identified' State.
- The project does not plan to identify any new requirements in any of the other ranges between PDR and CDR since all the rows in the 'P' column of the 'Identified' State in those ranges

Table 19.11 The CRIP Chart a project with requirements creep at CDR

	Identified			In process			Completed			In test			Accepted		
Range	P	E	A	P	E	A	P	E	A	P	E	A	P	E	A
1	0	0	0	0	101	101	101								
2	0	0	0	0	78	78	78								
3	0	0	0	0	35	35	35								
4	0	0	0	0	30	30	30								
5	0	0	0	0	28	28	28								
6	0	0	0	0	20	20	20								
7	0	0	0	0	12	12	12								
8	0	3	2	2	7	7	9								
9	0	0	0	0	5	5	5								
10	0	0	0	0	2	2	2								
Totals	0	3	2	2	318	318	320								

remain at zero.

19.4.2. The CRIP Chart for a project with requirements creep at CDR

Two additional requirements were identified between PDR and CDR resulting in the total number of requirements increasing from 318 to 320. The CRIP Chart for the project with requirements creep at CDR shown in Table 19.11 indicates:

- No additional requirements were levied on the system as indicated by the zero value in all the rows in the 'A' column of the 'Identified' State.
- Only two of the three requirements in the 'E' column of Range 8 were actually identified as indicated by the 2 in row 8 of the 'A' column of the 'Identified' State[8].
- The project plans to start and complete the development activities on these additional 2 requirements in Range 8 as indicated by the '2' in the 'P' column of the 'In process' State and the 9 (7+2) in the 'P' column of the 'Completed' State in row 8.
- The development activity on the remaining requirements is progressing according to plan as shown by the entries in the 'In process' and 'Completed' States.
- The project does not plan to identify any new requirements between PDR and CDR since all the rows in the 'P' column of the 'Identified' State remain at zero.

[8] It is possible that a change request was made for the third requirement and the request was rejected for some reason. The CRIP Chart just indicates the change; the CRIP Chart does not provide reasons for the change.

Table 19.12 The CRIP Chart for a project with requirements creep at TRR

	Identified			In process			Completed			In test			Accepted		
Range	P	E	A	P	E	A	P	E	A	P	E	A	P	E	A
1	0	0	0	0	0	0	0	101	101	101					
2	0	0	0	0	0	0	0	78	78	78					
3	0	0	0	0	0	0	0	35	35	35					
4	0	0	4	4	0	0	0	30	30	30					
5	0	0	6	0	0	6	6	28	28	34					
6	0	0	0	0	0	0	0	20	20	20					
7	0	0	0	0	0	0	0	12	12	12					
8	0	0	0	0	2	2	0	9	9	9					
9	0	0	0	0	0	0	0	5	5	5					
10	0	0	0	0	0	0	0	2	2	2					
Totals	0	0	10	4	2	8	6	320	320	326					

19.4.3. The CRIP Chart for a project with requirements creep at TRR

Ten unexpected additional requirements were identified between CDR and TRR resulting in the total number of requirements increasing from 320 to 330. The CRIP Chart for the project with requirements creep at TRR shown in Table 19.12 indicates:

- Four of the unexpected requirements were identified in Range 4 shown by the 4 in row 4 of the 'A' column of the 'Identified' State.
- The project plans to start the development activities on the additional 4 requirements in Range 4 as indicated by the '4' in row 4 of the 'P' column of the 'In process' State.
- Six of the unexpected requirements were identified in Range 5 shown by the 6 in row 5 of the 'A' column of the 'Identified' State.
- Development activities actually began on these 6 requirements as shown by the 6 in row 5 of the 'A' column in the 'In process' State.
- The project plans to complete the development activities on these 6 requirements as shown by the 6 in row 5 of the 'P' column of the 'Completed' State.
- Development work began on two of the additional requirements in Range 8 as shown by the match between the numbers in row 8 of the 'E' and 'A' columns of the 'In process' State of Table

Table 19.13 The CRIP Chart for a project with requirements creep at IRR

	Identified			In process			Completed			In test			Accepted		
Range	P	E	A	P	E	A	P	E	A	P	E	A	P	E	A
1	0	0	0	0	0	0	0	0	0	0	101	101	101		
2	0	0	0	0	0	0	0	0	0	0	78	78	78		
3	0	0	0	0	0	0	0	0	0	0	35	35	35		
4	0	0	0	0	4	4	4	0	0	0	30	30	30		
5	0	0	0	0	0	0	0	6	6	6	34	28	34		
6	0	0	0	0	0	0	0	0	0	0	20	20	20		
7	0	0	0	0	0	0	0	0	0	0	12	12	12		
8	0	0	0	0	0	0	0	0	0	2	9	7	9		
9	0	0	0	0	0	0	0	0	0	0	5	5	5		
10	0	0	0	0	0	0	0	0	0	0	2	2	2		
Totals	0	0	0	0	4	4	4	6	6	6	326	318	326		

19.12[9].

- The development work for the two additional requirements in Range 8 is not expected to be completed by the subsequent milestone since there is a zero in row 8 of the 'P' column of the 'Completed' State.
- Development work on the remaining requirements is progressing according to plan and the project is expected to commence testing as shown by the matches between the entries in the 'A' columns of the 'Completed' State and the 'P' column of the 'In test' State.
- The project does not plan to identify any new requirements between TRR and the following milestone since all the rows in the 'P' column of the 'Identified' State remain at zero.

19.4.4. The CRIP Chart for a project with requirements creep at IRR

For a change, no unexpected additional requirements were identified between CDR and IRR resulting in no change in the total number of requirements. The CRIP Chart for the project with requirements creep at IRR shown in Table 19.13 indicates:

- No additional requirements were identified since the values of all rows of the 'A' column of the 'Identified' State are zero.
- Development on the requirements in Range 4 proceeded according to plan by the match between numbers in each of the rows

[9] The reason for only starting work on two of the three could be that one was rejected for some reason, or that the System Design State for meeting that requirement was deferred. The CRIP Chart just indicates the variance without providing a reason.

in the 'E' and 'A' columns of the 'In process' State of Table 19.13. The number 4 in the 'P' column of the 'Completed' indicates that the project development activities are planned to have been completed by the following milestone.

- Development on the requirements in Range 5 proceeded according to plan since the since the number 6 was copied from the 'P' column of the 'Completed' State in Table 19.12 to the 'E' and 'A' columns of the 'Completed' State of Table 19.13. The number 6 in the 'P' column of the 'In test' State' indicates that the testing activities are planned to have begun by the following milestone.
- Something has stopped the development of activities of the two requirements in row 8 as shown by the zeros in the 'E' and the 'A' columns of the 'In process' State and the zero in the 'P' column of the 'Completed' State. However, this can only be seen when the two CRIP Charts are compared directly. The CRIP Chart does not provide a reason for the stoppage; it only provides the information that a stoppage has occurred.
- Nearly all the requirements that were planned to enter the 'In test' State have done so because most of the numbers in the 'P' column of the 'In test' State in Table 19.12 have been copied to the 'E' and 'A' columns in Table 19.13.
- There are some problems in starting to test the requirements in rows 5 and 8 since the numbers in the 'E' columns do not match those in the 'A' columns.
- The project plans to catch up on these requirements as since there is a 6 in the 'P' column of row 5 and a 2 in the 'P' column of row 8 in the 'In test' State.
- The project plans for the testing of all the requirements in the 'In test' State to be successfully completed and accepted by the next milestone as indicated by the match between the numbers in the 'A' column of the 'In test' State and the 'P' column of the 'Accepted' State.
- The project plans to overcome the delays in testing the requirements in rows 5 and 8 by the next milestone as indicated by the match between the numbers in the 'E' column of the 'In test' State and the 'P' column of the 'Accepted' State.

19.4.5. The CRIP Chart for a project with requirements creep at DRR

The CRIP Chart for the project with requirements creep at DRR shown in Table 19.14 indicates:

Table 19.14 The CRIP Chart for a project with requirements creep at DRR

	Identified			In process			Completed			In test			Accepted		
Range	P	E	A	P	E	A	P	E	A	P	E	A	P	E	A
1	0	0	0	0	0	0	0	0	0	0	0	0	0	101	101
2	0	0	0	0	0	0	0	0	0	0	0	0	0	78	78
3	0	0	0	0	0	0	0	0	0	0	0	0	0	35	35
4	0	0	0	0	0	0	0	4	4	0	0	4	0	30	34
5	0	0	0	0	0	0	0	0	0	0	6	6	0	34	34
6	0	0	0	0	0	0	0	0	0	0	0	0	0	20	20
7	0	0	0	0	0	0	0	0	0	0	0	0	0	12	12
8	0	0	0	0	0	0	0	0	0	0	2	2	2	9	7
9	0	0	0	0	0	0	0	0	0	0	0	0	0	5	5
10	0	0	0	0	0	0	0	0	0	0	0	0	0	2	2
Totals	0	0	0	0	0	0	0	4	4	0	8	12	2	326	328

- No additional requirements were identified.
- Development activities on the requirements in row 4 have been completed as planned, as indicated by the number 4 in the 'P' column of the 'Completed' State in Table 19.13 being copied into the 'E' and 'A' columns of Table 19.14. Moreover, the testing activities on those requirements have not only begun as indicated by the 4 in the 'A' column of the 'In test' State, they have been completed and accepted by the customer as indicated by the 34 in the 'A' column of the 'Accepted' State. This is 4 more than the expected 30 in the 'E' column of the State.
- The customer accepted all the requirements that had been tested except for two in row 8 as indicated by numbers in the 'E' and 'A' columns of the 'Accepted' State.
- The project expected that two requirements in row 8 would go into testing and they did, as indicated by the numbers 2 in the 'E' and 'A' columns of the 'In test' State.

19.5. The challenged project

Consider the challenged project which is the same as the ideal project until the 'In test' State begins. Accordingly, the CRIP Charts for the challenged project at SRR, PDR and CDR are the same as those for the ideal project shown in Table 19.4, Table 19.5 and Table 19.6 respectively. The project diverges from the ideal project after CDR so discrepancies can be seen when the TRR is held on the originally scheduled date.

19.5.1. The CRIP Chart for the challenged project at TRR

The CRIP Chart for the challenged project at TRR shown in Table 19.15 indicates:

Table 19.15 The CRIP Chart for the challenged project at TRR

	Identified			In process			Completed			In test			Accepted		
Range	P	E	A	P	E	A	P	E	A	P	E	A	P	E	A
1	0	0	0	0	0	0	40	81	41	81					
2	0	0	0	0	0	0	40	78	38	78					
3	0	0	0	0	0	0	5	35	30	35					
4	0	0	0	0	0	0	10	30	20	30					
5	0	0	0	0	0	0	10	26	16	26					
6	0	0	0	0	0	0	0	20	20	20					
7	0	0	0	0	0	0	6	8	2	8					
8	0	0	0	0	0	0	4	7	3	7					
9	0	0	0	0	0	0	1	5	4	5					
10	0	0	0	0	0	0	1	2	1	2					
Totals	0	0	0	0	0	0	117	292	175	292					

- No additional requirements were identified.
- Development activities in all requirement ranges except Range 6 have not been completed since the 'E' and 'A' values in row 6 of the 'Completed' State do not match.
- The project plans to catch up on the development activities as shown by the numbers in the 'P' column of the 'Completed' State.
- The project is optimistic about commencing testing following the TRR as evidenced by the difference between numbers in the 'P' column of the 'In test' State and the numbers in the corresponding rows of the 'A' column of the 'Completed' State. The customer definitely needs to find out the reason for the optimism.

19.5.2. The CRIP Chart for the challenged project at IRR

The CRIP Chart for the challenged project at IRR shown in Table 19.16 indicates:

- No additional requirements were identified.
- The project should not have transitioned into the Subsystem Testing State of the SDP (Section 9.12.5) because of the difference between the numbers in the 'E' and 'A' columns in the 'In test' State.
- The project plans to catch up as shown by the numbers in the 'P' column of the 'In test' State.
- The project is still optimistic about completing the testing before DRR because the 'P' numbers in the 'Accepted' State match the 'E' numbers instead of the 'A' numbers in the 'In test State'. The customer definitely needs to determine the reasons for the opti-

Table 19.16 The CRIP Chart for the challenged project at IRR

	Identified			In process			Completed			In test			Accepted		
Range	P	E	A	P	E	A	P	E	A	P	E	A	P	E	A
1	0	0	0	0	0	0	0	0	0	40	81	21	81		
2	0	0	0	0	0	0	0	0	0	30	78	48	78		
3	0	0	0	0	0	0	0	0	0	5	35	30	35		
4	0	0	0	0	0	0	0	0	0	0	30	22	30		
5	0	0	0	0	0	0	0	0	0	14	26	12	26		
6	0	0	0	0	0	0	0	0	0	0	20	20	20		
7	0	0	0	0	0	0	0	0	0	7	8	1	8		
8	0	0	0	0	0	0	0	0	0	6	7	1	7		
9	0	0	0	0	0	0	0	0	0	4	5	1	5		
10	0	0	0	0	0	0	0	0	0	0	2	2	2		
Totals	0	0	0	0	0	0	0	0	0	114	292	158	292		

mism.

19.6. The make up your mind project

Consider the typical make up your mind project which is the same as the ideal project until SRR and diverges between SRR and PDR because the customer keeps changing their mind.

19.6.1. The CRIP for the typical makeup your mind project at SRR

The CRIP Chart at SRR is that same as that for the ideal project shown in Table 19.4.

19.6.2. The CRIP Chart for the typical make up your mind project at PDR

Fifty-six unexpected additional requirements were identified between SRR and PDR resulting in the total number of requirements increasing from 292 to 348. The CRIP Chart for the project at PDR shown in Table 19.17 indicates:

- Twenty of the unexpected additional requirements were identified in Range 1 as indicated by the number 20 in the 'A' column of the 'Identified' State in row 1.
- Development activities have commenced on these requirements as shown by the difference in the numbers in the 'E' and 'A' columns in row 1 of the 'In process' State (20+80=101).
- Ten of the unexpected additional requirements were identified in Range 2.
- Development activities have commenced on these requirements as shown by the difference in the numbers in the 'E' and 'A' columns in row 2 of the 'In process' State.
- Fourteen of the unexpected additional requirements were identi-

Table 19.17 The CRIP Chart for a make up your mind project at PDR

	Identified			In process			Completed			In test			Accepted		
Range	P	E	A	P	E	A	P	E	A	P	E	A	P	E	A
1	0	0	20	0	81	101	0								
2	0	0	10	0	78	88	0								
3	0	0	14	0	35	49	0								
4	5	0	0	0	30	30	0								
5	9	0	2	0	26	28	0								
6	12	0	0	0	20	0	0								
7	0	0	4	0	8	12	0								
8	3	0	0	0	7	7	0								
9	0	0	6	0	5	11	0								
10	2	0	0	0	2	0	0								
Totals	31	0	56	0	292	326	0								

fied in Range 3.

- The project plans to identify five additional requirements in Range 4 following PDR[10].
- Two unexpected requirements were identified in Range 5 and the project plans to identify nine additional requirements following PDR.
- The project plans to identify 12 additional requirements in Range 6 following PDR.
- Four of the unexpected additional requirements were identified in Range 7.
- Development activities have commenced on these requirements.
- The project plans to identify 3 additional requirements in Range 8 following PDR.
- Six of the unexpected additional requirements were identified in Range 9.
- Development activities have commenced on these requirements.
- The project plans to identify 2 additional requirements in Range 10 following PDR.
- Development activities are proceeding according to plan except for the requirements in Ranges 6 and 10.
- No development activities have started on the requirements in Range 6 as shown by the 20 in the 'E' column and the zero in the 'A' column of the 'In process' State in row 6.
- Development activities have proceeded as planned on the requirements in Range 7 since the values in the 'E' column and 'A' column of the 'In process' State in row 7 are the same.

[10] The change requests have been submitted.

- No development activities have started on the requirements in Range 10.
- The project does not plan to complete the development activities in process before the following milestone because every row in the 'P' column of the 'In process' and 'Completed States have been set to zero. This could be because the project has been cancelled or for some other reason.

19.7. Comments

CRIP Charts can be used in both the 'A' and 'B' paradigms since they trace work back to requirements. However, although written up for requirements, they can also be used for Use Cases, scenarios, Technical Performance Measures (TPM) and any other technical measurement that can be tracked across the SDP.

19.8. Summary

This Chapter improved systems engineering by introducing CRIP Charts which provide a way to:

- Measure technical progress.
- Identify potential problems in near real-time.

The CRIP Charts introduced in this chapter can be used in both the 'A' and 'B' paradigms since they trace work back to requirements. However, although written up for requirements, they can also be used for Use Cases, scenarios, TPMs and any other technical measurement that can be tracked across the SDP.

--oo--

PART V

Improving systems engineering through the use of Case Studies

Part V of the book discusses improving systems engineering through the use of Case Studies. While Case Studies have been shown to be a very effective way of learning (Mauffette-Leenders, et al., 2007) there are few good Case Studies in systems engineering. The purpose of a teaching case study is to establish a framework for discussion and debate among students (Yin, 2009: page 4). There are three types of Case Studies, Research, Experiential and Role Playing. The basic difference between the Experiential Case Study and the Research Case Study is as follows:

- ***In the Experiential Case Study***, the event has happened, the outcomes are known and the event is being documented.
- ***In the Research Case Study*** the experiment is being designed, hence the outcomes may not be known (for sure) ahead of time.

Consider each type of Case Study and its use by practitioners and in the classroom.

- ***Research:*** documents what happened in an experiment. Case Studies are used extensively in research in practice-oriented fields (Yin, 1989). They are generally designed to answer a specific research question in the form of "how" and "why" as opposed to "who", "what", "where", "who much" and "how many" (Yin, 1989). The major difference between a Case Study and an experiment is that a Case Study does not require control over behavioural events (Yin, 1989).

  There are few if any Research Case Studies in systems engineering.

- ***Experiential:*** a modification of the Research type which documents what happened in a project or situation allowing the students to discuss the events and perhaps what should have been done differently. The students read the material and then analyse and comment on the situation. In a variation of the documented Experiential Case Study, the students write their own Case Study based on their experience, or from information gleaned from the literature such as in the US Airborne Laser (ABL) Test Bed (ABLT) program in Section 20.6.

  Most of the systems engineering Experiential Case Studies in the literature, generally focus on the 'what' and the 'how' and so do not allow students to develop an understanding of 'why' the various decisions were made; the insight into the reason for the decisions. This lack of the 'why' is an undesirable situation because the focus of the case study should be on answering the 'how' and 'why' questions (Yin, 2009) presumably since students will potentially be in similar situations in their futures and need to understand the reasons behind the decisions in the Case Studies so they can make their own informed decisions using other people's experiences.

  While practitioner-written Experiential Case Study conference papers have the potential to be excellent sources of information for postgraduate students, the academic quality of the published papers is often less than desired. This is because these papers:

  - Often document a discovery made after strenuous efforts, most of which would have not been necessary had the authors reviewed the literature or at least the proceedings of earlier conferences. As an example, LaPlue et al. discussed the development of a methodology for specifying requirements that describe the behaviour of a system and its interaction with its environment (LaPlue, et al., 1995). In fact they reinvented the environmental and behavioural models of the Ward and Mellor software development methodology (Ward and Mellor, 1985)[1]. This situation also escalated their project costs since the methodology existed and could have been used instead of being reinvented had a literature search been conducted before the task began.
  - Often mix facts and paper author's opinions together and sometimes it is difficult to distinguish between them.

[1] More recent examples except for MBSE discussed in Section 12.21 were deliberately not cited.

- Often make it difficult to locate specific types of information such as lessons learned.

  Chapter 20 introduces a proposed template for Experiential Case Studies for use by practitioners in participation in conference and symposia as perceived from the *Operational* perspective (Section 5.1) and for use by students in the classroom to overcome this deficiency and provides an example.

- ***Roll Playing:*** in which the student takes the role of a decision-maker and uses the material in the case as the basis for making decisions with incomplete information. Role-Playing Case Studies can take the form of paper documents, and computer simulations often in the form of games. Chapter 21 provides such a context in the form of the Engaporean Air Defence System (ADS) upgrade.

  Role-Playing Case Studies:

  - Are becoming important in systems engineering because the bulk of systems engineering in HKMF Layer 2 (Section 14.4) is moving away from "top-down" development of brand new systems to the "middle-out" development of systems that have to be interoperable with existing systems (Long and Scott, 2011: page 14).
  - May also be used as the context of an Experiential Case Study since a fictitious setting minimizes the emotional aspects incurrent in discussing real-world scenarios since there is no blame to be allocated and even if there is, the fictitious sponsoring organization can be at fault.

Part V contains:

- Chapter 20 which introduces a template to improve the quality of practitioner written Experiential Case Studies to format practitioner papers as a way to link their experiences into the literature in a systemic and systematic manner to provide information to assist students studying, and researchers improving, the practice of systems engineering.
- Chapter 21 which contributes to improving systems engineering by introducing a multi-purpose Case Study to provide:
  - A framework for Role-Playing Case Studies in classes on systems engineering and engineering (project) management written in such a manner so as to provide additional examples of the tools, templates and frameworks described in Parts III and IV, including:
    - An example of the generic multi-iteration SDP.

  - An example of the use an Experiential Case Study.
  - More examples of the problem formulation template.
- Yet another example of the use of the HTPs to organize information in a systemic and systematic manner.

--oo--

20. Improving the quality of Experiential Case Studies

This Chapter introduces a template to improve the quality of practitioner written Experiential Case Studies[1] to format practitioner papers as a way to link their experiences into the literature in a systemic and systematic manner to provide information to assist students studying, and researchers improving, the practice of systems engineering. Examples of the use of the template are included.

Perceptions from the *Generic* perspective indicate that there is little difference between a postgraduate term paper and a conference/symposium paper[2]. Consequently, this template should apply to both types of papers.

McNamara provides the following five-step process for developing a Case Study (McNamara, 1999):

1. Gather data about the case.
2. Organise the data to highlight the focus of the study.
3. Develop the narrative.
4. Validate the narrative.
5. Compare the study with appropriate others to identify areas of improvement.

The majority of practitioner-written papers in the INCOSE Experiential Case Study genre in which the practitioner tells a story concentrate on Step 3 and provide a conclusion. These papers are not true Case Studies in the sense discussed by McNamara and Yin but are close enough for the purpose of providing an educational resource to improve systems engineering. This is means that the papers should be:

- Organised in a systemic and systematic manner to facilitate the writing and retrieval of information.
- Grounded in the literature.
- Indicate how the lesson(s) learned from the story may also apply

[1] Prototyped starting at SETE 2004 and following SETE conferences
[2] At least in my classes!

in other situations.

20.1. Templates for Papers

Templates can help provide better papers and presentations. For example:

- ***The UMUC experience.*** This situation is similar to the responses to assignments in a postgraduate course the Graduate School of Management and Technology at University of Maryland University College[3] (UMUC). However, when a template was introduced in one course, albeit for another purpose (Kasser and Williams, 1998), a serendipitous result was that the quality of the assignments also improved. In that instance, the students were asked to write term papers describing their personal experiences in projects that were in trouble. The papers were to adhere to the following template:
  1. A description of a scenario based on personal experience.
  2. An analysis of the scenario.
  3. A list of the reasons the project succeeded or ran into trouble.
  4. A list of, and comments on, the lessons learned from the analysis.
  5. A section identifying a better way with 20/20 hindsight.
  6. A list of a number of situational indicators extracted from the scenario that can be used to identify a project in trouble or a successful project while the project is in progress.

- ***Templates for documents.*** Templates for the contents of documents to assist in locating the information stored in a document are not a new idea. For example:
  - The US Government Request for Proposals (RFP) uses a template so that the same type of information is always located in the same section of the RFP. For example, Section M of the RFP always contains the evaluation criteria for the proposals.
  - MIL-STD 490 specified which type of requirements to group into which section of a requirement specification.

- ***The SETE 2004 Experience.*** Having observed that the template both improved the quality of submitted assignments and helped UMUC postgraduate students to upgrade their assign-

[3] These students were employed in the workforce and were working towards their degree in the evening. Their employment positions ranged from programmers to project managers. Some also had up to 20 years of experience in their respective fields.

ments to conference papers, it was felt that a similar template for Experiential Case Study papers would improve presentations in the SETE 2004 conference. Such a template was provided to prospective practitioner authors upon acceptance of their abstracts, but its use was optional. While no quantitative measurements were made, anecdotal evidence suggested that those authors who used the template produced better presentations than those who didn't.

Using the template in the classroom is convenient as:

1. Students are assigned to locate a situation for the Case Study. This can be from experience, the literature or an existing Case Study such as those on the US Air Force Institute of Technology (AFIT) web site.
2. The students then collect the information about the situation from various sources.
3. The students review the information, sort and store it in the eight descriptive HTPs as per the rules discussed in Section 3.2 and cite the original source documents as shown in the numerous examples in this book, and discussed in Section 20.5.
4. The students document their insights, inferences and suggestions in the *Scientific* perspective.

The ease of locating information and the separating of facts from opinions also facilitates grading the student assignments.

20.2. The process for producing the Experiential Case Study paper

A suggested process for producing Experiential Case Study papers based on McNamara's five-step process discussed above is as follows:

1. Decide on the point(s) to be made in the paper by creating the abstract.
2. Determine what data needs to be collected to reinforce the point(s) to be made.
3. Locate and evaluate a similar document and learn from it (what to do and what not to do).
4. Collect the data.
5. Sort the data using the rules for documenting real-world situations in the HTPs discussed in Section 3.2.
6. Research the literature to determine if the points have been made before and set up the citations to the material to be referenced.

7. Write up the story[4].
8. Analyse the story and document the analysis.
9. Write up the lessons learned.
10. Summarise the Case Study.
11. Develop and document the conclusions.
12. Circulate draft paper for comment.
13. Request and receive comments.
14. Evaluate and incorporate comments.
15. Submit paper for publication.

This process is not proposed as being a sequential series of activities however Step 1 should come first. Steps 12 to 14 should be done iteratively until the submission deadline date at which point in time the paper is to be submitted.

20.3. The structure of the template

The purpose of the template is to try to improve the quality of practitioner presentations and papers in the Experiential Case Study genre by providing a template (prototyped at SETE 2004) to format practitioner papers as a way to link their experiences into the literature to provide data to assist researchers improving the practice of systems engineering. The format of the template is as follows:

1. Abstract or overview.
2. Introductory road map
3. The story in which the data is sorted by the eight descriptive HTPs.
4. The analysis and commentary; inferences and insights from the *Scientific* perspective
5. Lessons learned.
6. Summary.
7. Conclusion.
8. References.
9. Glossary of acronyms and corporate terms.

Consider the content and rationale for each section.

[4] Using the Airborne Laser Test Bed program discussed in Section 20.6 and the MCSSRP discussed in Section 17.1 as examples. Other examples of documenting information in the HTPs including the Royal Air Force (RAF) Battle of Britain Air Defence System (RAFBADS) can be found in Chapter 6 of Volume 1 of the series (Kasser, 2013c: pages 145-176).

20.3.1. Abstract or overview

An abstract is an overview of the document written to entice the reader into reading the whole document. A typical abstract should contain the following three parts:

1. The undesirable situation which triggered the work described in the paper.
2. An outline of the story or idea or proposal.
3. The outcome, results or resulting benefits.

The process for creating the draft abstract is to use bullets or dot points in the form of an outline list to create a draft abstract. The draft abstract list should only be converted to the prose version of the abstract once the paper is complete. This is a time saving technique because the content of the paper may change as the paper is being written and if the full abstract is written in prose at the beginning of the process, the time spent composing that prose will be wasted since the abstract will have to be rewritten at the end of the process.

The following provides an example of a draft and fleshed-out abstract containing all three parts using the **[]** signs[5] to separate the parts. The abstract is for a paper that discusses improving the way difficult concepts in systems engineering are taught.

The draft abstract is:

```
1  Relationships difficult to explain
2  Used modified FRAT (Kasser, 2013c: page 157)
   in class with positive results
3  Paper uses FRAT for LuZ
     Lessons learned
```

The fleshed out abstract is:

Abstract. [In teaching systems engineering the relationship between functions, physical decomposition and requirements during the process of defining, designing and developing the system, has been difficult to get across to the students.**][**While trying to improve the learning process, an explanation of the relationship between functions, physical decomposition and requirements during the process of defining, designing and developing the system based on a modification of the Functions Requirements Answers and Test (FRAT) views of a system (Mar, 1994) was tried on undergraduate students at the University of South Australia in 2006-2007 with positive results(Kasser, et al., 2007).**][**This paper uses the

5 Don't use the [] in your paper.

adapted FRAT as a frame in which to describe the relationship between functions, physical decomposition and requirements using as an example the definition, design and development of the control and electronics part of the LuZ solar electrical power generating system (SEGS-1) in 1981-1983 (Kasser, 1984). The paper also provides some lessons learned from the project.**]**

The following is an example of a draft and fleshed-out abstract of a paper on thinking about systems thinking. This abstract however only contains the first and last parts.

The draft abstract is:

```
1. Thinking about thinking
     Emerging paradigm
     Need for multiple perspectives
3.   Proposes a set of perspectives
     Use RAFBADS as example
     The two flaws
    Observations on the state of systems
    engineering
```

The fleshed out abstract is:

Abstract. [This is a paper on thinking about thinking. Systems engineering is an emerging discipline in the area of defining and solving problems in the manner of (Wymore, 1993). The emerging paradigm for problem solving is “systems thinking”. Both systems engineering and systems thinking have recognized the need to view a system from more than one perspective.**][**This paper proposes a set of perspectives for applying systems thinking in systems engineering and then defines a systems thinking perspective set of views for a system, the use of which will provide one way of aligning systems thinking to systems engineering. The paper then provides an example of applying the set of perspectives to the Royal Air Force Battle of Britain Air Defence System (RAFBADS) and shows that not only does the set of perspectives provide a way to model the system; it also picked up two potentially fatal flaws in the system. The paper then concludes with some observations on the state of systems engineering from a number of the perspectives.**]**

Both of these examples communicate what their papers are about. Readers who are not interested in the topic will skip the paper, not wasting time reading it and finding it of little value. Readers who are interested in the topic will continue to read the paper. Hopefully the contents will hold their interest.

Opinions are split as to whether abstracts should contain citations to source documents. My preference is to insert them unless the style guide for the publication precludes including the citations. My general rule is when in doubt, avoid plagiarism and cite!

20.3.2. The introductory road map

This section of the paper provides an overview of each section of the paper telling the reader what to expect in each section, the contribution of each section to the paper if appropriate and explaining the flow of the paper and the rationale for the flow. Chapter 1 of this book provides the road map for this book and serves as an example.

20.3.3. The story

This section contains the information sorted by descriptive perspective as demonstrated in the example of storing perceptions of systems engineering in Chapters 4 to 11, in the Airborne Laser Test Bed program discussed in Section 20.6 and the MCSSRP discussed in Section 17.1. If perceptions from one or more of the perspectives are not pertinent, then they should be temporarily included and *removed only after the conclusions have been developed*. This is because the information initially thought not to be pertinent may generate a key question during the analysis part of writing the paper.

The story starts with the *Big Picture* perspective to provide the background or context and is often followed by the *Temporal* perspective to indicate how the situation arose. Section 3.1 discussed which type of information to place in each descriptive perspective.

When telling the story, there is a difference between academic and non-academic communications in papers and presentations. In non-academic speech, we tend to say something like "there are a number of factors" then talk about the first factor. After that, we might say, "the next factor is" or "another factor is" and talk about it, then continue in the same manner with "another factor". Academic oral and written communications are different and orderly. For example in academic oral and written communications:

- We should start with "there are a number of factors" or better still, state exactly how many if they are known. Then, list them and then discuss each of them turn in the same way as in the examples in this book.
- If the number of items in a list is known, then use the word "are" as in "the factors are …".
- If the number of items in a list is unknown, or a subset is being described, then use the word "including" or "includes", as in

"the factors include two on …".

20.3.4. Analysis and commentary

This section:

- Contains the *Scientific* perspective; the insights and inferences from the analysis, and the reflections on why the success (or failure) happened. This section should be mandatory for acceptance of the paper for the conference.
- Is generally lacking in most practitioner-written experiential papers. Yet, this is the most important section containing the results of the literature review which provides the support for the author's comments and ideas and suggestions for improving systems engineering.
- Shall as a minimum:
  - Comment on the story citing references to the literature (e.g. INCOSE handbook, journal articles, conference and symposia proceedings, text books, etc.) and state how the events described support or refute the cited references.
  - Point out differences between the situation in the Case Study and the context in the literature to explain the reasons for the project's (subject of the Case Study) success or failure.
- Demonstrate the application of critical thinking to create the argument supporting the inferences and conclusions.

20.3.4.1. Creating or analysing arguments

One way of using critical thinking to create or analyse an argument is to use the following process adapted from Tittle as appropriate (Tittle, 2011: page 17)[6]:

1. Determine the point of the argument (claim/opinion /conclusion).
2. Identify the reasons and the evidence.
3. Articulate all unstated premises and connections in the reasoning (assumptions)[7].
4. Define the terms used in the argument.
5. Clarify all imprecise language (*Quantitative* perspective).
6. Differentiate between facts and opinions.

[6] This process, based can be used to examine an argument or to create one (*Continuum* perspective)

[7] Or at least as many as you can

7. Eliminate or replace "loaded" language and other manipulations[8].
8. Assess the reasoning/evidence:
   - If deductive, check for truth (factual), acceptability and validity.
   - If inductive, check for truth (factual), acceptability, relevance and sufficiency.
9. Determine ways to strengthen the argument[9] by:
   - Providing and incorporating additional reasons and/or evidence.
   - Anticipating objections and providing adequate responses to the objections.
10. Determine ways to weaken the argument [10]by:
    - Considering and assessing counterexamples, counterevidence and counterarguments.
    - Determining if the argument should be modified or rejected because of the counterarguments.
11. If appropriate, identify and provide any additional information required before the argument could be accepted or rejected.

20.3.4.2. The literature review

The questions that drive the literature review come from the *Temporal*, *Generic* and *Continuum* perspectives and include:

1. Has anybody faced the same situation in the past?
2. What worked then?
3. What didn't work then?
4. What is different about that situation and the current one?

The assumption behind this section is that during the course of writing up the Experiential Case Study paper according to the template, the practitioner will perform the literature review after the event documented in the story as finished. Findings from the literature review may indicate that some of the effort put in to the event documented in the story could have been saved if the literature review had been performed prior to commencing the task. Accordingly, the next time a task is begun, the

[8] Words which have emotional significance or contain implied judgments.
[9] And do so
[10] If writing the paper and ways are found, then strengthen the argument to remove those weaknesses.

practitioner might perform the literature review at the start of the activity, hence preventing some unnecessary work.

20.3.5. Lessons learned

This section is a summary section where the lessons learned are itemised and briefly discussed. References to the literature are appropriate where the literature discusses identical or similar lessons and situations.

20.3.6. Summary

This is the standard summary of the paper which summarises the content.

20.3.7. Conclusion

This section contains the conclusion(s) from the Case Study.

20.3.8. References

This section contains the list of references sorted by author and year, unless the conference paper preparation template requires a different order. In that case the conference instructions take precedent. Chapter 24 provides an example of a list of references.

Perceptions from the *Continuum* perspective indicate that there is a subtle difference between a list of references and a bibliography as follows:

- ***A list of references*** only contains the information sources used in the production of the paper and citied in the paper.
- ***A bibliography*** may be a list of references or it may be a list of the information sources consulted in the production of the paper but not actually cited.

  Since the term bibliography is ambiguous, it is preferable to use the heading 'References' for the list of references.

20.3.9. Glossary of acronyms and corporate terms

While the use of corporate jargon is undesirable, it is often used in a Case Study. This section contains a table which spells out the acronyms. Creating the glossary of acronyms:

- After the paper is complete makes sure that all acronyms are spelled out in full the first time they are used.
- Is simply reading the paper or other kind of document from start to finish, locating the acronym and ensuring that it is spelled out in full the first time it is used. The acronyms may be stored in a spreadsheet during this process and then imported into the document as a table with hidden cell borders; the process

used in creating this book.

Locating the text in a table where the cells and the table have hidden borders facilitates the alignment of acronyms and meanings. See Chapter 23 for an example.

20.4. Plagiarism and leveraging on other people's work

Plagiarism is using someone else's work and passing it off as if it was your own. You should incorporate someone else's work in your own to build on what has been created before, BUT do it in the right way. If you plagiarise you might get away with it for a while, but sooner or later you will be found out, and then your reputation will be destroyed forever. You will never be able to restore it. However, it is so easy to prevent that from happening. **All you have to do to avoid plagiarism is** to *give credit where it is due* citing the source in the appropriate format discussed in Section 20.5.

When using material from various sources, apply the following rules.

- Don't use too much from a single source. If you are unsure, then you are probably using too much.
- Don't use figures and drawings from other sources without attribution both in the text and in the caption of the figure. Request permission if you include the figure or drawing from a publication for profit. Clip art and government documents, which are in the public domain, are excluded from this rule.
- Use of figures and drawings from copyrighted sources is generally permissible in a one-time educational presentation, but not in the hand-outs.
- Do not post your content [which contains figures and drawings from other sources] on the Internet; that constitutes 'publishing'.
- If the material is available under Creative Commons licensing, then conform to the license.

Check these rules with your legal department.

20.5. Citing sources or incorporating references

Citing or referencing other people's work and then building on their work gives your publication credibility as well as showing that you are conversant with the literature. Citing sources can be done in various ways and tend to be publication specific. That is different publishers have different styles. The most common ways of citing sources are:

- (Author, date). This is the style used in this book and there are numerous examples. There are varieties of this style where the author and date may be separated by a space character or a punctuation mark such as a comma, or the author and date may be enclosed in square brackets.

- Numbered in square brackets as in [1], [2], [3] etc. Note that the order of numbering also varies, in some instances the order is by appearance in your document and in other instances the order can be chronological by publication date, or in alphabetical by last name of author order.
- Placing the references in footnotes.

If citing a source from a book, add the page number to facilitate looking up the reference. This allows readers of your work to check the source and lets you find it again after you have forgotten where in the source book the concept was mentioned.

When citing Internet web sites, the citation should include the Uniform Resource Locator (URL) and the date of access, since unlike the static printed page, contents of web pages are dynamic and can be changed by their owners. This means that something seen on a web page today may not be there in a week; hence the need to provide the access date. You should incorporate the citation using the appropriate style and format. You might also want to capture and archive a copy of the web page for future personal reference in case you need to refer back to the original text once it has been removed from the web site.

Citations can be to primary sources or to secondary sources. The following sections contain examples of citations to primary and secondary sources with explanations of the purpose of the citations.

20.5.1. Citing primary sources

Primary sources are those that you have seen. The following are examples of citing primary sources:

- An example of citing other people supporting a statement made by the writer.

  In a paper discussing the differences between systems engineering and project management in the literature, the author writes, *"Depending on their perspective, authors have written that the activities performed in producing the ancient pyramids, the canals and railways of the 19th century and other systems of the past are those embodied in systems engineering (Kasser, 1996a) or project management (George, 1972)".*

- An instance of citing a source to support a statement.

  In the same paper as before, the writer makes the following statement, "*For example, the activities in the 1930's that led to the creation of the Air Defence System used by the Royal Air Force in the Battle of*

Britain have been called systems engineering with hindsight (Haskins, 2006)".

- An example where the citations are used to support the "*there have been many*" part of the quotation.

  In the same paper as before, the writer makes the following statement, *"There have been many discussions in the literature about the overlapping of, and differences in, the roles of systems engineering, Operations Research, systems architecting, and project management, e.g. (Brekka, et al., 1994; Roe, 1995; Kasser, 1996a; Sheard, 1996; Mooz and Forsberg, 1997; Friedman, 2006)".*

- An example of a citation used as a lead into quoting the source in the words that follow the '...'.

  In the same paper as before, the writer makes the following statement, "*Mooz and Forsberg wrote that systems engineering and project management should be integrated "(Mooz and Forsberg, 1997). They state that...*". Note the double use of the author's names. The first time the text mentions that the authors wrote something, the second use provides the citation. The text should read clearly as if the citations were invisible footnotes. So while it may look desirable to use the form *"(Mooz and Forsberg, 2009) wrote that systems engineering and project management should be integrated"* and avoid double mention of the authors (in the author date format), you should resist the temptation. The effect of doing this when the citation format is numeric is discussed in Section 20.5.4.

- An example where the author names and cites a source in the first part of the quotation and then adds a conclusion in the last sentence. The citation at the end of a sentence also makes it clear which part of the paragraph is cited and which is not.

  In a paper on education the author wrote, "*As van Peppen and van der Ploeg wrote "typically, an educational program is carefully designed, giving attention to the individual elements of the curriculum, the learning environment, and their interdependencies" (van Peppen and van der Ploeg, 2000). A curriculum design (a specific sequence of knowledge-base and skill-building courses) specifies the criteria for course design (a specific combination of learning objectives, course materials, teaching methods, and tests), as well as the staffing of teaching faculty, course scheduling, and teaching facilities. Thus designing a curriculum is an example of systems engineering of both the product and the process hence the title of this chapter".*

- Another example where the author cites a source in the first part of the quotation and then adds a conclusion in the last sentence.

In the same paper as before, the author makes the following statement, "*Students could even fail to complete the post-class assignment and still pass the course (albeit with a minimum passing grade) (Kasser, et al., 2005). The students were learning to do systems engineering by numbers!*"

- An example where the author begins with a cited quotation which is then followed by a conclusion.

In the same paper as before, the author makes the following statement, "'*Effective systems engineering calls for careful coordination of process, people and tools'. Such coordination cannot be learned from books' (Hall, 1962: page v), these needs levelled requirements on the pedagogy to add something to the book learning*".

20.5.2. Citing secondary sources

Secondary sources are sources that are cited by a primary source. You should never cite a secondary source as a primary source, namely pretend that you have seen the primary source. Sometimes the secondary source quotes the earlier document out of context or makes an error. Using the secondary source format absolves you from an error made by the primary source, and shows respect to both sources. This form of respect goes back at least 2,000 years and can be found in many places in the Jewish Talmud in the form of citations such as, "*Rabbi Judah said in the name of Rabbi Zechariah that...*". Secondary sources can be cited in the form *"text being cited (Kasser, 2006) as cited by (Hari, 2008)*". In this instance, (Kasser, 2006) is the secondary source and (Hari, 2008) is the primary source.

20.5.3. Paraphrasing

Citations also need to be used when paraphrasing source materials. For example, in one paper on education a concept (intellectual property) from a source in the literature was paraphrased to support the work being documented in the paper. The original text in Vélez and Sevillano stated, *"In a digital hardware design course, students should work similarly to digital hardware engineers in a company (Vélez and Sevillano, 2007)".* The statement was incorporated in the paper to support the work being documented. As incorporated in a paper, the concept was rewritten as, *"The immersion course format was developed to allow the students to perform systems engineering in the classroom in a systems engineering environment. This concept is supported by Vélez and Sevillano who stated that students in a digital hardware design course should do the same type of work as digital hardware engineers perform in a company (Vélez and Sevillano, 2007)".*

20.5.4. In line citations

The following example illustrates why the citation should not be used as part of the text. In the (author, date) format the text is written as. "*Mooz and Forsberg wrote that systems engineering and project management should be integrated*" (Mooz and Forsberg, 1997).

In the numbered format the text is, "*Mooz and Forsberg wrote that systems engineering and project management should be integrated*" [3].

If the citation had been incorporated in the text, using the author-date format, the text would have read as, "(*Mooz and Forsberg, 1997) wrote that systems engineering and project management should be integrated*" which makes the source clear, but if the citation had been incorporated using the numbered format the text would have read, *"[3] wrote that systems engineering and project management should be integrated*" which does not provide the author information unless you refer to the list of references at the end of the text. Remember the goal of writing a document is to make it easy for the reader to follow the flow of concepts and understand what you are trying to communicate.

20.5.5. Citation management software

You should also use a software tool for managing the styles of citations and references: the tool is a great timesaver. Consider the following text written using a style in which the two citations are shown as (author, date).

```
"As a consequence, demand for skilled, knowledgeable, Systems Engineers in government, industry, and academia is increasing around the world (Arnold, 2006). However, in general, systems engineering seems to be poorly practiced (Kasser, 2007)."
```

In the following version of the text, the same citations are numbered in the brackets style.

```
"As a consequence, demand for skilled, knowledgeable, Systems Engineers in government, industry, and academia is increasing around the world [4]. However, in general, systems engineering seems to be poorly practiced [52]."
```

If sections of text have to be included in different documents with different requirements for citation styles then retyping citations wastes a lot of time. There are software tools such as EndNote and RefWorks that help you collect, store, and manage reference information. The tools allow you to insert citations into documents as fields and can change the format of the citations and the list of references of an entire document with a few mouse clicks.

20.6. Example Case Study: The Airborne Laser Test Bed program

This section provides an example of the story section of an Experiential Case Study; the US Airborne Laser (ABL) Test Bed (ABLT) program.

20.6.1. Introduction

Once upon a time[11] the US recognized that:

- The probability of a hostile nation developing an ICBM was very high.
- An ICBM carrying ten-megaton range fusion warheads could inflict trillions of dollars in damage as well as a possible return of the nation to the Stone Age.

The US embarked on a program to develop a Theater Missile Defense (TMD) family of systems. One of these systems was the ABLT program, an advanced platform for the DoD Directed Energy Research Program that ran from 1996 to 2012. Perceive the ABLT from the different perspectives on the perspective perimeter.

20.6.2. Big Picture perspective

Perceptions from the *Big Picture* perspective include the context in which the system is used. Ballistic missile weapons pose a threat that is difficult to defend against. The first use of a ballistic missile weapon was the V-2 rockets used by Nazi Germany against London during World War II when the missiles were undetectable. There was no possible way to detect, let alone intercept, the incoming missiles in real time at that time, so the defence technique developed by the British, was (1) to attempt to destroy the launch sites and (2) to provide disinformation that the missiles had overshot their target in an effort to make the Nazis shorten the range so the V-2s would land in the countryside south of London. Since then, the concept of real-time defence against ballistic missiles in-flight has focused on intercepting the incoming missile during the three phases of its flight[12].

1. ***The launch or boost phase.*** The best time to destroy the missile since the missile is relatively slow moving and the debris will fall on enemy territory close to the launch site.
2. ***The in-flight or ballistic phase.*** Debris will fall on countries that may or may not be involved in the conflict.

[11] I could not resist starting the story this way.

[12] Other methods such as destroying the launch sites are out of the scope of this story.

3. ***The descent or terminal phase.*** The worst time to destroy the missile since the debris may fall on friendly territory. However, Patriot missile systems were deployed in Kuwait by US forces during Operation Iraqi Freedom in 2010 against descending SCUD missiles with some degree of success.

Conceptually, destroying a rocket during its launch phase can be achieved in a number of ways. The ABLT program was designed to test the concept of achieving that destruction using an airborne directed energy weapon in the form of an ABL system. The ABL system would be networked to the adjacent TMD family of systems that would also provide sensor information about launches and command authorization to destroy what would be perceived as a first strike launch by a potentially hostile nation. For example, information about a launch could also be received from an orbital satellite, Airborne Early Warning and Control *(*AEW&C*)* system or from an UAV (Kopp, 2012).

20.6.3. Temporal perspective

Perceptions from the *Temporal* perspective cover the history leading up to the ABLT program and the events in the program.

Chronologically, precursors to the project were:

- ***1973:*** the USAF demonstrated the feasibility of the concept of destroying a missile at a distance using a directed energy weapon by shooting down a winged drone at their Sandia Optical Range, New Mexico, using a carbon dioxide Gas Dynamic Laser (GDL) and a gimballed telescope (Kopp, 2012).
- ***1976:*** the USAF launched their Airborne Laser Lab (ALL) program. The aim of this effort was to construct a technology demonstrator, carried on a modified NKC-135 Stratotanker, which could successfully track and destroy airborne targets (Kopp, 2012).
- ***1992:*** the USAF planned the ABL as a technology development project to be managed to high readiness levels by a Science and Technology (S&T) organization. The project was started as an advanced technology transition demonstration to design, fabricate, and test a single demonstrator weapon system and was to take eight years to complete. The pacing technologies were to be matured to a high level - equivalent to TRL[13] 6 or 7 before being included in a product development program. Requirements had not been fixed. In other words, the planned approach was that adopted by successful projects (GAO, 1999).

[13] See Section 10.11.2.

- ***1993:*** The concept of destroying a missile at a distance using a directed energy weapon was demonstrated during two separate tests at White Sands Missile Range (WSMR), New Mexico, in October 1993. A one-megawatt Mid-Infrared Advanced Chemical Laser (MIRACL) was used to destroy a number of pressurized tanks which simulated SCUDs at the High Energy Laser System Test Facility. In each test the MIRACL and its associated optics were used to rapidly target and destroy several fuel tanks which were sized and pressurized differently. The tests demonstrated a laser's ability to destroy a Theater Ballistic Missile (TBM) as well as the capability to retarget quickly in a multiple-launch situation (Wirsing, 1997).
- ***1996:*** the USAF abandoned this approach.

The ABLT program's significant events were:

- ***1996:*** the USAF decided to launch ABL as a weapon system development program, not because technologies were sufficiently mature but because of funding and sponsorship concerns. At this time, the two key technologies were at TRLs 3 and 4. According to the retired manager of the S&T project, a product development program was deemed necessary to make the technology development effort appear real to the users and not a scientific curiosity (GAO, 1999). The USAF awarded a product definition risk reduction contract to Boeing, TRW and Lockheed-Martin. The Boeing team were to deliver two prototypes. The plan was to follow up the success of the contract by purchasing five operational aircraft.
- ***1997:*** invention of the Chemical Oxygen Iodine Laser (COIL) by the USAF Weapons Laboratory.
- ***2000:*** the ABL development program faced a number of technical challenges (DOT&E, 2000) including:
    - Development of an autonomous surveillance system onboard the ABL that would provide timely, accurate missile targeting information required to meet stressing ABL engagement timelines.
    - The contractor's ability to build COIL flight modules that would provide adequate power for the operational system and would be sufficiently low weight to fit within the 747-400 aircraft platform capabilities.
    - Development and demonstration of a laser beam compensation and tracking system that would meet stringent pointing and tracking requirements for engaging ballistic missiles.

  - Demonstration of a fully capable Battle Management/Command, Control, Communications, and Computers & Intelligence (BMC4I) system that would interact in real-time with other TMD systems for cross-cueing and fire control.
  - Ability of the contractor to successfully integrate all of the above systems into the finite weight and volume limitations of the 747-400 aircraft.
  - The ABL's ability to meet the reliability and maintainability requirements without excessive contractor support.
  - Limitations and vulnerabilities of the planned ABL lethality mechanisms against all threat missiles, and the potential effects and responses to predicted enemy countermeasures.

- ***2001:*** the ABL development program was converted into an ABL acquisition program and transferred to the Missile Defense Agency (MDA). Boeing became the integrating and platform development contractor for two prototype ABL systems; a learning prototype, and an operational prototype.
- ***2006:*** due to delays and major technical problems, the ABL program was relegated to a technology demonstration status while the planned five-aircraft purchase by the USAF was put on hold.
- ***2009:*** the ABL acquisition program was eight years behind schedule and $4 billion over cost. Moreover, the program's proposed operational role was highly questionable because of significant affordability and technology problems. This led to the acquisition program being shifted back to a Research and Development (R&D) effort during a major Defense budget reduction and the acquisition of the second ABLT aircraft was cancelled. A GAO report stated, "*None of the ABL's seven critical technologies were fully mature. Program officials assessed one of the ABL's seven critical technologies – managing the high-power beam – as fully mature, but the technology had not yet been demonstrated in a flight environment. The remaining six technologies – the six-module laser, missile tracking, atmospheric compensation, transmissive optics, optical coatings, and jitter control were assessed as nearing maturity*" (GAO, 2010).
- ***2010 and 2011:*** the ABLT was able to prove that the concept of destroying unprotected missiles in their boost phases at a distance using a high power directed energy weapon was feasible by shooting down a number of targets, however the concept was not operational in that ABLT configuration.
- ***2012:*** the ABLT was flown to Davis-Monthan Air Force Base on February 14, put in storage, and retired from active service.

20.6.4. Operational perspective

Perceptions from the *Operational* perspective include the scenarios in which the system is used. The ABL was conceived as being rapidly deployable and adding a boost phase layer to the TMD Family of Systems. It was to be positioned behind the forward line of friendly troops and moved closer towards enemy airspace as local air superiority was attained.

The USAF proposed a seven-aircraft fleet, and envisioned that five aircraft would be deployed to support two 24-hour combat air patrols in a theatre. Pairs of aircraft would fly patrols over friendly territory close to the borders of a potentially hostile nation scanning the horizon for a rocket launch. Once a launch is detected, the ABL tracks the missile, illuminating the missile with a tracking laser beam while onboard computers lock onto the target. After acquiring target lock, a high power laser fires a three- to five-second burst of directed energy destroying the missile over the launch area. However as the laser beam is distorted by atmospheric turbulence caused by fluctuations in air temperature[14], the focus of the beam must be adjusted in real time to compensate for the fluctuations.

20.6.5. Structural perspective

Perceptions from the *Structural* perspective include physical components that make up the system and their architecture. The ABL weapon system consisted of the following systems:

1. ***The weapons platform:*** the YAL-1, a modified Boeing 747-400F (freighter). The modifications caused changes to the aerodynamic profile of the aircraft.
2. ***Six COIL modules:*** based on an improved version of the COIL invented in 1977. The COIL's fuel consists of hydrogen peroxide, potassium hydroxide, chlorine gas and water.
3. ***The turret ball*** on the nose of the 747-400F is used to point the 1.6 metre primary laser mirror produced by Corning Glass and Contraves.
4. ***The Beam Control System (BCS):*** ensures that the laser's power can be effectively delivered to the target by compensating for atmospheric distortion. The BCS comprises the wave front sensor and control system for beam distortion control, the systems for beam jitter control, beam alignment and beam 'walk' control, calibration hardware, and target acquisition and tracking equipment.

[14] The same phenomenon that causes stars to twinkle.

5. ***The Track Illuminator Laser (TIL):*** consists of three low-powered Kilowatt-class Ytterbium – Doped: Yttrium Aluminium Garnet (Yb:YAG) solid-state diode-pumped lasers developed by Raytheon and Northrop Grumman Space Technology.
6. ***The Beam Illuminator Laser (BIL):*** measures atmospheric disturbances providing the information to the BCS also consists of three low powered Kilowatt-class solid state diode-pumped lasers developed by Raytheon and Northrop Grumman Space Technology.
7. ***The Battle Management/Command, Control, Communications, and Computers & Intelligence (BMC4I)*** system manages the weapon system and its operator console and also the supporting communications.

20.6.6. Functional perspective

Perceptions from the *Functional* perspective the set of mission and support functions discussed below.

- ***Mission functions:*** The mission functions include:
  - Scanning the horizon looking for missile launches.
  - Acquiring a target missile in its boost stage.
  - Detecting, tracking and prioritizing the target.
  - Directing the high-energy laser onto the target missile for a long enough time period to damage the missile so that it self-destructs or falls back to earth.
  - Report target events and status of the ABL systems.
- ***Support functions:*** The support functions include:
  - Managing the health and status of ABL operations.
  - Monitoring/engaging warning and self-protection measures.
  - Maintaining theatre situational awareness.

20.6.7. Continuum perspective

Perceptions from the *Continuum* perspective include information about the functions that the system cannot perform. For example, the ABL cannot distinguish between the boost phases of a rocket launching a space satellite and the launch of an attacking missile in a first strike sneak attack. Destroying a space satellite launch would have undesirable political consequences as well as probably being considered as an act of war.

20.6.8. Quantitative perspective

Perceptions from the *Quantitative* perspective include perceptions about the numbers associated with the functions performed by the system, the specifications on the hardware and software, costs and other quantitative data including:

- ***Operational range:*** the ABL was expected to achieve effective range of at the most 400 Km (GAO, 2004).
- ***Missile destruction time:*** the high power laser needs to fire a three- to five-second burst of directed energy.
- ***Fuel needed:*** one of the design aims of the ABL system was to carry enough laser fuel to destroy twenty to forty missiles during a single 12 to 18 hour sortie (Kopp, 2012).
- ***Each COIL*** operates at an infrared wavelength of 1.315 microns and vents toxic materials in operation.
- ***The turret ball*** has a ±120 degree field of regard in azimuth.
- ***The BCS:*** a deformable mirror with 341 actuators which update the shape of the mirror 1,000 times a second. This means that the time required to measure the atmospheric distortion, perform the calculations and control the mirror actuators is less than 1/1000 sec.
- ***Number of ABLs needed:*** patrolling an operational theatre 24 hours a day 7 days a week would require seven aircraft. The US Defense concept of being able to support two simultaneous theatres would need a fleet of 14 aircraft.
- ***Costs of the system:*** the cost of a single operational aircraft was estimated as $1.5 Billion. The support costs could run an additional $100 million cost per each aircraft per year. Multiply these numbers by 14 for the cost of an operational system.
- ***Technology Readiness Level (TRL):*** the DoD established nine levels of TRLs discussed in Section 10.11.2

20.6.9. Generic perspective

Perceptions from the *Generic* perspective identify similarities between the system and others of the same type. For example, projects such as the ABL acquisition program can be characterised in several ways by:

- ***Management style:*** such as type of project, e.g. research, development, etc.
- ***Technology:*** such as missiles, networked projects, technology uncertainty, etc.

20.6.10. Scientific perspective

This is where the person writing the Case Study inserts insights, inferences, comments analyses, lessons learned and conclusions separating the facts in the descriptive HTPs from the opinions and hypotheses which are placed in this section. So with respect to the ABL case study, insights and inferences from the *Scientific* perspective, might include:

1. The need for, and suggestions for, upgrading the TRL.
2. The dual technology development approach in the early stage of the acquisition.
3. Changing the project cycle from a single waterfall to a series of waterfalls at project planning time.
4. The ABL program management style.
5. Was the ABL a spoof program?
6. How much of systems engineering is overly complex and complicated or even wrong because people have used techniques[15] do not want to admit they don't understand?

Consider each of these issues. Note however, where an insight or inference has already been described in this book, it is referenced rather than repeated in this section.

20.6.10.1. The need for upgrading the TRL

The Case Study writer might point out that deficiencies in the TRL have already been identified (Section 10.11.1). The Case Study writer might then add a discussion as follows.

Sauser et al. proposed a replacement System Readiness Level (SRL) incorporating the current concept of the TRL scale with the addition of an Integration Readiness Level (IRL) to dynamically calculate a SRL index (Sauser, et al., 2006). Their SRL index approach is both complex and complicated. In addition, Kujawski argued that the SRL index approach is also fundamentally flawed (Kujawski, 2010). An alternative simpler approach for upgrading the TRL is by looking at the rate of change of technology maturity instead of the single static TRL and then comparing the anticipated rate of change with the actual rate of change in the manner of EVA is discussed in Section 14.1. Here the Case Study writer would discuss the material now in Section 14.1.

20.6.10.2. The dual technology development approach in the early stage of the acquisition

The Case Study writer might recommend considering adapting another approach in the R&D phase: a dual technology research approach to prototype development similar to the approach already used by the DOD in competitive prototype development procurements such as the Joint Strike Fighter (JSF)[16] where the research phase contractors Boeing and

---

[15] They just follow the process.

[16] While the JSF was a predictable failing project in its post research phase (Kasser, 2001), the research phase did produce two viable alternatives.

Lockheed-Martin produced two viable radically different approaches. So instead of focusing on a single solution such as in the ABL, the research phase would develop more than one approach based on different technologies until some time when one or more of the proposed solutions reach a point where it can be shown to be practical. The decision as to which technology to use in the actual system would then be made.

20.6.10.3. Changing the project cycle from a single waterfall to a series of waterfalls at project planning time

The Case Study writer might point out that the traditional SDP is based on a single pass through the waterfall. Given that most weapons systems development are Shenhar and Bonen's Type B and C systems (Shenhar and Bonen, 1997), Shenhar and Bonen recommend that the traditional timeline needs to be changed from one pass through the waterfall to at least two passes for Type B projects and three for Type C projects as discussed in Section 12.14.

20.6.10.4. The ABL program management style

The Case Study writer might point out that the ABL program could be characterised as a Type D project according to Shenhar and Bonen's four-level scale of technological uncertainty discussed in Section 9.16. As such the single pass through the waterfall was not the appropriate methodology to use as discussed in Section 20.6.10.3.

20.6.10.5. Was the ABL a spoof program?

The Case Study writer might point out that perceptions from the *Quantitative* perspective indicated that the ABL was expected to achieve effective range of at the most 400 Km. All a potential adversary had to do was locate their launch sites more than 400 Km from their frontier to defeat the ABL.

The Case Study writer might point out that given the vast amount of non-classified information freely accessible on the ABL and its precursors, the possibility arises that the research program was one that the real experts knew could never produce an operational airborne system, but looked so promising that a potential adversary would be persuaded to fund development work on a similar program instead of spending their resources on programs that could succeed to the detriment of the US. As the GAO wrote in 1996, according to the retired manager of the S&T project, a product development program was deemed necessary to make the technology development effort appear real to the users and not a scientific curiosity (GAO, 1999).

The Case Study writer might conclude with a statement which raises a question, such as, "*one wonders if the ABL began as such a spoof project during* *the Cold War, but with that aspect of the project being classified, it was overlooked when key personnel transferred out, and the program then undertook a life of its own*" (Kasser and Sen, 2013).

20.6.10.6. How much of systems engineering is overly complex and complicated or even wrong?

The Case Study writer might point out that this was a program that should have been challenged as discussed in Section 20.6.10.5 but does not seem to have been. Jenkin's roles of a systems engineer include the role of challenging the status quo (Jenkins, 1969: page 164). However, few if any systems engineers actually do challenge the status quo[17]. Perhaps they are too busy doing systems engineering to think about what they are doing. Or are they? Perhaps do not want to admit they don't understand what they are doing and are just following the process, exhibiting Type II behaviour (Section 10.9). Examples of challenging aspects of systems engineering include:

- ***Is systems engineering a myth or a reality?*** In the course of researching that question pointed out seven myths of, and eight defects in, systems engineering (Kasser, 2010b; 2007; 1996; 2013b).
- ***The DODAF useful?*** Perceptions from the *Operational* perspective indicate that the DODAF was designed to be used to *"provide correct and timely information to decision makers involved in future acquisitions of communications equipment"* (DoDAF, 2004). It was not designed to describe systems. When perceived from the *Quantitative* perspective:
  - 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).
- Kujawski challenged the SRL (Sauser, et al., 2006) pointing out that it is fundamentally flawed (Kujawski, 2010).

These issues could be, and should be, published for discussion in:

[17] And even if they do, they have trouble getting published because reviewers reject their manuscripts because they disagree with the author.

- Conferences to explore aspects of the issues and improve and advance systems engineering.
- Postgraduate classes to help students develop an understanding of the 'why' as well as the 'what' and the 'how' of systems engineering.

Other inferences can be made and depend on the purpose of the Case Study.

20.6.11. Summary

The ABL program Case Study has provided an example of an Experiential Case Study using the HTPs as a template.

20.7. Summary

This Chapter described a previous experience of the use of a template for postgraduate student assignments and has proposed a template to improve the quality of student and practitioner Experiential Case Study papers to provide practitioners with a way to link their experiences into the literature to provide data to assist students learning about systems engineering and researchers improving the practice of systems engineering. The ABLT program was written up as an Experiential Case Study to provide an example of the template in action.

--oo--

21. A Role-Playing Case Study: the Engaporean Air Defence System upgrade

This Chapter introduces the Engaporean Air Defence System (ADS) upgrade Case Study (Kasser, 2013a) to provide:

- A framework for Role-Playing Case Studies in classes on systems engineering and engineering (project) management written in such a manner so as to provide additional examples of the tools, templates and frameworks described in Parts III and IV, including:
  - An example of the generic extended SDP shown in Figure 13.7.
  - An example of the use an Experiential Case Study as discussed in Section 21.15.2.
  - More examples of the problem formulation template discussed in Section 14.5.
  - Yet another example of the use of the HTPs to organize information in a systemic and systematic manner.

The Engaporean scenario was first chosen in 2007 as a Role-Playing Case Study in a systems engineering class in which the students were to develop an integrated transportation system for the nation of Engaporia (Kasser, et al., 2008). Their roles were to perform some of the systems engineering in each state of the SDP, sequentially, developing a CONOPS, requirements, designs, test plans, and transition plans. The scenario was chosen because the bulk of systems engineering in HKMF Layer 2 (Section 14.4) is moving away from "top-down" development of brand new systems to the "middle-out" development of systems that have to be interoperable with existing systems (Long and Scott, 2011: page 14).

21.1. Introduction

Once upon a time, in 2003, the (fictitious) nation of Engaporia (Kasser, 2009) discovered a large quantity of off-shore oil reserves and the government at the time felt that its then current air-defence capability might not have had the capability to protect itself from its belligerent northern

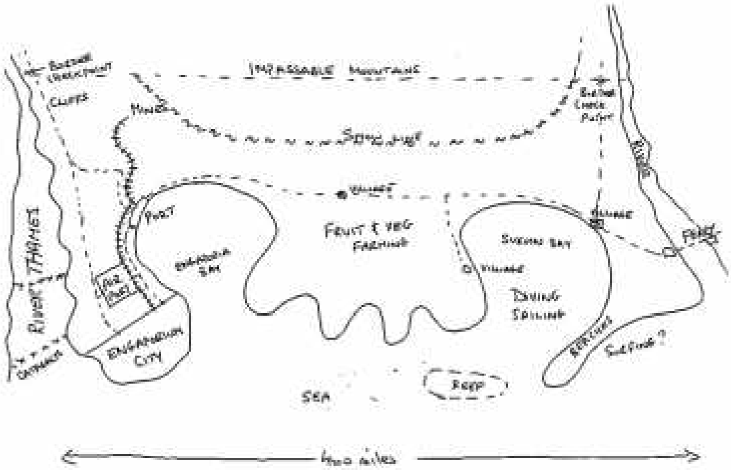


Figure 21.1 Engaporia

neighbour. Perceive the Engaporean ADS upgrade project from the HTPs on the perspectives perimeter.

21.2. Big Picture perspective

Perceptions from the *Big Picture* perspective provide the context for the Case Study. Engaporia is an old British colony, with a stable democratic government, and a small population. It is a non-aligned, mostly ignored member of the United Nations. It is located between sea and mountains as shown in the map in Figure 21.1. Other details are:

- A Mediterranean climate; the coastal plain having warm summers and mild winters.
- A mining and farming economy.
- A strategic port location, the Royal Navy used it as a naval base.
- The population is concentrated in Engaporium city.
- The government has recognised that the population is growing to the point where there will be a serious unemployment problem in near future.
- Impassable mountains to north which are snow covered in winter.
- A disputed border with the northern belligerent neighbour.
- Non-navigable (into the hinterland) rivers to east and west, although there is a ferry across the western river boundary.
- Friendly borders with eastern and western neighbours.

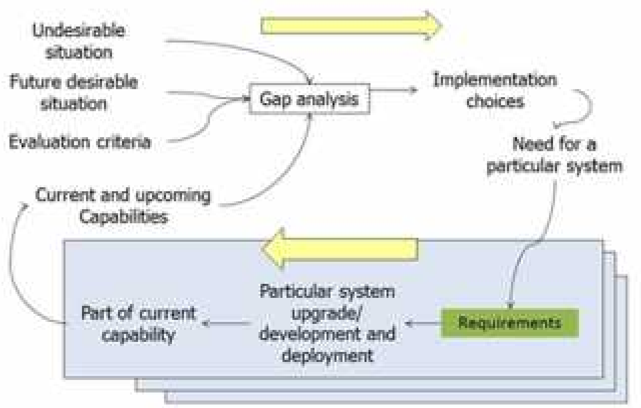


Figure 21.2 DSTD context loop

The government tasked the Engaporean Capability Development Agency (ECDA) to deal with the issue.

21.3. Operational perspective

Perceptions from the *Operational* perspective indicate that the ECDA employs the extended holistic problem-solving process to managing problems and solutions shown in Figure 6.5.

21.4. Functional perspective

Perceptions from the *Functional* perspective provide details of the processes involved in the upgrade[1].

21.5. Structural perspective

Perceptions from the *Structural* perspective include:

- A list of the physical facilities and equipment in the then current ADS including airfields, missile sites, communications facilities.
- In alphabetical order, the organisations involved in the project are:
    - Federated Aerospace (FA); the prime contractor.
    - The Defence Systems and Technology Department (DSTD); the Engaporean government agency tasked with maintaining national security.

[1] A *Functional* perspective of HEADS itself would provide a view of the internal functions performed by HEADS. The system under study is the upgrade process; hence the *Functional* perspective is that of the upgrade process.

- The Engaporean Capability Development Agency (ECDA). ECDA manages the state of the entire Engaporean Defence System in the context shown in Figure 21.2.
- The Engaporean Defence Force (EDF).
- The subcontractors to FA.

These organisations contain the main stakeholders in the project.

21.6. Generic Perspective

Perceptions from the *Generic* perspective indicated that the situation was similar to, and so lessons could be learnt from, the Royal Air Force Battle of Britain Air Defence System (RAFBADS) used in World War II (Bungay, 2000).

21.7. Continuum Perspective

Perceptions from the *Continuum* perspective identified differences between the current situation and the RAFBADS particularly in that while RAFBADS Active Counter Measures (ACM) were only performed by manned fighter aircraft, Engaporia had the option of using Surface-To-Air-Missiles (SAM) as well as manned and unmanned aircraft. In addition, the rest of the technology potentially available for use in Engaporia has had 60 years of modernization resulting in greatly expanded functionality.

21.8. Temporal perspective

Perceptions from the *Temporal* perspective provide a view of the timeline of the story told in sequence from past to present as described herein. This timeline provides the reference or framework for variations and "what-if" discussions in the classroom. See Section 20.6.3 for an example of the timeline in the ABL program for an example that documents events leading up to the situation and during the situation. This example, documents the states of the Engaporean Air Defence System (ADS) upgrade SDP.

21.9. The early state systems engineering

The early state systems engineering iterated though the SDP as described in this section. The ECDA framed their problem as:

- ***The undesirable situation*** was the uncertainty if the then current ADS needed upgrading.
- ***The FCFDS*** was knowing for sure if the ADS needed upgrading or if not.
- ***The problem*** was:

    - Determining if the then current ADS needed upgrading, and if so,
    - Initiating a project to perform the upgrade.
- ***The Solution*** was the FCFDS.

The ECDA's way of remedying the problem was to assign the task to the Defence Systems and Technology Department (DSTD). DSTD performed a classified feasibility study to determine if the then current ADS needed upgrading, and if so, to estimate the scope, costs and development schedule for the upgrade. The feasibility study iterated through the activities in the early states of the SDP as follows:

- ***The Needs Identification State*** produced:
    - A summary of the need for defence against the known and estimated threats posed by the unfriendly northern neighbour.
    - A number of scenarios of what threats the upgraded air-defence system would have to counter (*Operational* perspective).
    - A gap analysis between the capabilities needed to counter anticipated threats and the then-current operational and upcoming capability taking into account the dates in which the upcoming capability would become operational.
    - A report that stated that while parts of the then current ADS were state-of-the-art, in general, the ADS did need upgrading.
- ***The Requirements State*** produced the requirements for the additional upcoming capability to be acquired or developed.
- ***The System Design State*** showed that: there were at least three viable affordable alternative ways to provide the necessary upgrade.

The ECDA reviewed and accepted the results of the feasibility study and funded a DSTD project to initiate the SDP which would develop a new Holistic Engaporean Air Defence System (HEADS), an HKMF Layer 3 system (Section 14.4), whose mission was defined as detecting and destroying enemy aircraft penetrating Engaporean airspace[2] preferably before they caused any damage to the Engaporean infrastructure.

The DSTD framed the problem as:

- ***The undesirable situation*** was the need for upgrading the Engaporean air-defence capability from the then current capability to that provided by HEADS without reducing or interfering

[2] Engaporia wanted to show that HEADS was purely defensive.

with the operation of the air-defence system in case a threat occurred before HEADS was fully operational. From the *Generic* perspective this is similar to the "no loss of data prime directive" in the MCSSRP discussed in Section 17.3.4.2, but of course, a lot more serious.

- ***The FCFDS*** was an upgraded operation ADS.
- ***The problem*** was to conceptualise a number of solutions and select acceptable ones for further development.
- ***The Solution*** was unknown at the start of the project.

The DSTD followed the 'A' paradigm (Section 9.21.1) began the project in the Needs Identification State of the SDP with the creation of a preliminary CONOPS for each the following candidate solutions:

1. Lighter than Air Missile Platforms (LAMP).
2. Long range surface to air missiles.
3. Manned fighter interceptors similar to that used in the RAFBADS.
4. Short range surface to air interception (anti-aircraft guns, missiles).
5. A combination of the above.

Each solution was conceptualized as a system containing normal and contingency mission and support functions where:

- ***Mission functions:*** consisting of the:
  a) The command, control, communications, computers, intelligence, surveillance, and reconnaissance (C4ISR) functions to detect enemy aircraft.
  b) The ACM functions that would then destroy the enemy aircraft.

- ***Support functions:*** consisting of the necessary functions that would keep HEADS fully operational at all times.

The detection and C4ISR functions performed for all candidate solutions were almost identical. The support functions differed for each candidate due to the nature of the ACM.

Since there was no point in considering a non-feasible candidate, each candidate CONOPS was accompanied by a feasibility study, drilling down into the proposed system to show that there was at least one viable feasible way of physically realizing each candidate. The LAMP option was a conceptual mix of tethered barrage balloons World War II style with aerial missile platforms supported by Helium filled and hot-air balloons that would remain aloft for long periods of time. However, the accompanying feasibility study determined that while the concept was

Table 21.1 Section criteria for conceptual options

	Criteria	Importance
1	Technology transfer to domestic industry	1.00
2	Development schedule – preference given to Stated implementation	1.00
3	Non-interference with the operational system at any stage in the upgrade process	1.00
4	Local industry involvement	0.95
5	Damage tolerance in action and due to possible pre-attack sabotage	0.95
6	Flexibility for local area defence	0.90
7	Reuse or incorporation of existing capability however obsolescence needed to be taken into account	0.80
8	Interoperable with existing system or subsystems	0.75
9	Lifecycle cost	0.75
10	Self-supportability and maintainability to avoid dependence on foreign contractors once deployed	0.60
11	Complexity – the lower the complexity, the higher the evaluation	0.45

innovative, it was not feasible at the time, and so the option was dropped before wasting resources investigating too many details.

The solution selection criteria for evaluating the remaining candidate options were developed from all of the HTPs after discussions with the stakeholders. The criteria and their importance, on a scale of 0 to 1 where 1 was most important, are shown in Table 21.1. The DSTD HEADS project team identified their stakeholders using the Nine-System Model as descried in the MCSSRP in Section 17.3 as:

- The Engaporean Defence Force (EDF).
- The potential realization contractors.
- The local civil government representatives.

The DSTD HEADS project team then held many separate meetings with the various stakeholders. These meetings:

- Developed an understanding of the impact of air attacks on the military and civilian infrastructure.
- Allowed the stakeholders to buy-in on the project.
- Had the added benefit of providing the project team with addi-

tional (mostly undocumented and tacit) knowledge in the problem, solution and implementation domains (Section 9.11). When Multi-attribute Variable Analysis (MVA) was applied in the decision-making process, the decision favoured the combination option. It also soon became clear during the numerous discussions with the stakeholders[3] that the Subsystem Construction, Subsystem Testing and System Integration and System Testing States would have to be in four iterations of the generic extended SDP shown in Figure 13.7 and the CONOPS and SEP were adjusted accordingly. At the completion of the Needs Identification State, the results were presented to the stakeholders in an OCR, in which the following were summarized[4]:

- In the product or system domain:
    - Each of the scenarios.
    - The solution selection criteria and their importance.
    - The trade-offs.
    - The selected optimal solution.
    - The technical feasibility.
- In the process domain:
    - The cost and schedule feasibility.
    - The acquisition and development strategy.
    - The type of contract, and the reason for the choice, for upgrading the ADS.

At the end of the OCR, the decision to proceed to initiate the ADS upgrade was unanimous, so the ECDA authorized the project to proceed.

21.10. The pre-tender state of the acquisition

The DSTD HEADS project team iterated through the early states of the SDP[5]: to produce:

- A detailed CONOPS for the HEADS covering the normal and anticipated contingency mission and support functions performed by the future operational ADS in the context of its environment (adjacent systems) which became the FCFDS (*Opera-*

[3] Who also included potential realization contractors.

[4] The stakeholders were fully cognizant of the facts and the reasons underlying the various choices because of the numerous meetings held before the OCR. Consequently, the purpose of the OCR was to summarize the situation and document the consensus to proceed to the next state of the project.

[5] Note that the HEADS project team focused on the transition process as well as the HEADS.

tional perspective). In the BPR environment this is known as the 'to be' model or view.

- A summary of the then-current ADS covering the normal and anticipated contingency mission and support functions based on the inputs to the earlier feasibility study. In the BPR environment this is known as the 'as is' model or view.
- A summary of the gap between the then current situation and the FCFDS (between the 'as is' and 'to be' views).
- A Systems Engineering Plan (SEP) containing a detailed CONOPS in the process domain, expanding the acquisition and development strategy presented at the OCR into a four-stage iterative realization process to implement the strategy to bridge the gap.
- More detailed cost and schedule estimates (*Quantitative* perspective).
- A Request for Proposal (RFP) for a local Engaporean prime contractor using domestic and foreign-owned subcontractors in a Multiple-Award-Task-Ordered (MATO) contract scenario, since the strategy was to acquire a system and contribute to building local technological capability to the maximum possible extent.

The problem was to create each of the physical subsystems in such a manner that when all iterations were subsequently integrated, HEADS would perform the mission and support functions according to its specifications without adversely affecting the operation of the air-defence system during the transition period. The contents of the SEP included showing:

- What current capability would be integrated into HEADS in each stage.
- When that integration would take place.
- How HEADS would be realized in an iterative manner
- The type of development contracts to be used in the acquisition.
- Where the government-contractor interfaces could be.
- What types of resources would be needed.

The basic realization strategy which was to use the generic extended SDP based on the Cataract Methodology (Kasser, 2002b) summarized in Figure 13.7 in which:

- ***Iteration 0*** would create the HEADS architecture, set up the management and engineering processes and disseminate the detailed transition plan.
- ***Iteration 1*** would incorporate some elements of the then-

current air-defence system into skeleton HEADS architecture.

- ***Iteration 2*** would put flesh into the skeleton with the priority of bridging any gaps.
- ***Iteraction3*** would complete the HEADS.

The documents were studied by the ECDA and approved. The HEADS project then received the go ahead to move forward and issue a RFP.

21.11. The proposal state

The RFP was issued and four responses were received where each response represented a candidate solution. The responses were evaluated using the same selection criteria as shown in Table 21.1 to select the winning tender. The contract award was made to a consortium led by Federated Aerospace (FA)[6].

21.12. SDP Iteration 0

This section summarizes the activities in the initial iteration of the HEADS upgrade project

21.12.1. The Needs Identification State

The engineering work in The Needs Identification State (Section 9.12.1) focused on the *Big Picture, Functional* and *Operational* perspectives of the HEADS. During the Needs Identification State:

- FA competed the task of creating a preliminary HEADS conceptual architecture based on the CONOPS incorporating appropriate existing EDF physical elements to its subcontractors[7].
- FA selected two subcontractors to produce independent architectures.

21.12.2. The Requirements State

During the Requirements State (Section 9.12.2) FA used perceptions from the *Functional* and *Operational* perspectives to create a matched set of specifications for the system and the subsystems. Produced documents included:

- The Project Plan (PP).

---

[6] FA was basically an interface between the government and the consortium. It subcontracted all the work to a consortium of subcontractors both foreign-based and local. Each major task was tendered to the consortium which consisted of both large and small businesses and the local FA division. In addition, FA was known to ECDA and DSTD because they were already developing an integrated transportation system for the nation and their performance had been satisfactory.

[7] The mixture of functional and physical was an imposed real-world constraint.

- A matched set of specifications for the system and its top-level subsystems based on the optimal architectural solution, namely the System Requirements Document (SRD) and the Subsystem Requirements Documents (*Operational* and *Quantitative* perspectives).
- The Systems Engineering Management Plan (SEMP).
- The Test and Evaluation Master Plan (TEMP).
- The risk and opportunity management plan identifying process and product risk and opportunities as well as preliminary risk and opportunity management concepts.
- The logistics support plan.
- The rest of the documentation defined in the contract.

The SRR presentations included summaries of these documents. Consensus to proceed was given and the ECDA authorized the project to continue to the System Design State.

21.12.3. The System Design State

The work in the System Design State (Section 9.12.3) focused on the *Functional* and *Structural* perspectives of the HEADS. The work was split between the Preliminary and Detailed System Design sub-states.

21.12.3.1. The Preliminary System Design sub-state

The work was performed in two subcontracted tasks:

- FA competed the task to create the conceptual architectures and selected two subcontractors to produce independent designs.
- FA competed the task to identify the solution selection criteria among its subcontractors, precluding the two who were developing the conceptual architectures from tendering for this task. The initial set of solution selection criteria was inherited from those in Table 21.1.

The DSTD HEADS project team together with FA and the subcontractors evaluated the solutions and created an optimal solution by combining aspects from the two independent solutions. The work was performed jointly because the domain knowledge needed for the reuse of existing capability resided in the EDF members of the DSTD project team rather than in FA personnel.

The DSTD HEADS project team:

- Performed the feasibility study to ensure that the selected conceptual architectural solution was feasible (affordable and deliverable within the time constraints).
- Monitored the situation with respect to the unfriendly neighbour

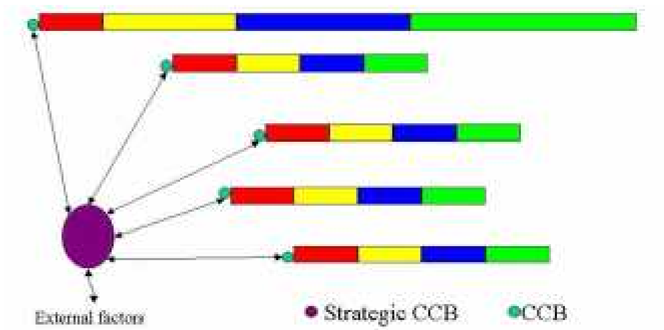


Figure 21.3 HEADS subsystem evolution and control

in the north to determine if any changes were necessary to the HEADS CONOPS (*Big Picture* perspective). As it happened, none were necessary.

In consultation with the stakeholders the documents FA[8] prepared included:

- A feasibility study report.
- Two independent mostly conceptual HEADS architecture designs incorporating appropriate existing EDF physical elements using perceptions from the *Functional* and *Structural* perspectives.
- The selection criteria for selecting the preliminary and detailed designs.
- Updated versions of previously produced documents (i.e. CONOPS, PP, SEMP, TEMP etc.). The detailed PP:
  - Described how each subsystem would evolve through the iterations and be managed including how its Change Control Board (CCB) would operate.
  - Defined a HEADS CCB that would be the strategic CCB for the entire project (Kasser, 2002a) as illustrated in Figure 21.3. Figure 21.3 shows a number of subsystems evolving though iterations (represented by different colours over time as in Figure 4.2), where changes within a subsystem are controlled by the subsystem CCB while system-wide changes are controlled by the Strategic CCB.

The DSTD HEADS project team together with FA used perceptions from the *Structural*, *Functional*, *Continuum* and *Generic* perspectives[9] to eval-

[8] From this point on, when FA is mentioned, the task was competed and a subcontractor selected to perform the task. Subcontractor selection criteria varied depending on the degree of knowledge to be transferred into Engaporia from the foreign subcontractor.

uate the physical architecture solutions and create an optimal physical architecture by combining aspects from the two independent solutions.

The DSTD HEADS project team:

- Performed an independent feasibility study on the optimal preliminary physical solution which showed that the optimal preliminary solution was feasible.
- Updated the Project Plan (PP) to take into account minor changes in the schedule as a result of findings from the feasibility study.

The work performed during the Preliminary System Design sub-state was summarized at the PDR; consensus to proceed was achieved and ECDA authorized the project to continue to the Detailed System Design sub-state.

21.12.3.2. The Detailed System Design sub-state

During the Detailed System Design sub-state, documents FA created included:

- Updated versions of previously produced documents (i.e. CONOPS, PP, SEMP, TEMP etc.).
- Two independent detailed physical HEADS architecture designs based on variations of the optimal preliminary design (*Functional* and *Structural* perspectives). Each architecture design contained a different mixture of SAM, fighter aircraft squadron, and anti-aircraft gun subsystems together with their appropriate supporting subsystems.
- The draft system and subsystem test plans for verifying that each Build of HEADS would be compliant to requirements pertaining to the build.

The DSTD HEADS project team together with FA evaluated the detailed physical architecture solutions and created an optimal detailed physical architecture by combining aspects from the two independent solutions.

The DSTD HEADS project team:

- Performed an independent feasibility study on the optimal preliminary physical solution.
- Updated the PP information to take into account minor changes

[9] The *Structural* perspective provided the architecture. Perceptions from the *Generic* and *Continuum* perspectives provided the concepts for evaluating the response of the architecture to failures and other abnormal modes of operation, namely the degree of robustness.

in the schedule as a result of findings from the feasibility study.

- Monitored the situation with respect to the unfriendly neighbour in the north to determine if any changes were necessary to the HEADS CONOPS. As it happened, a few did show up but they were minor and were accommodated within the scope of the proposed system via the Engineering Change Process (ECP).

The work during the Detailed System Design sub-state was summarized at the CDR, consensus to proceed was achieved and ECDA authorized the project to continue to the Subsystem Construction State.

21.12.4. The Subsystem Construction State

During the Subsystem Construction State (Section 9.12.4) each physical subsystem became a HKMF Layer 2 (Section 14.4) project in itself, and went through its own SDP with its own milestone reviews. HEADS systems engineers coordinated with the physical subsystem system engineers to manage unanticipated undesirable emergent properties and other factors that impacted the system. The engineering work in the Subsystem Construction State was split between the DSTD HEADS project team and FA as follows:

- FA performed the work pertaining to constructing the solution system including the building and procurement tasks because some components were to be built and some were to be purchased from local and foreign vendors.
- The DSTD HEADS project team monitored the situation with respect to the unfriendly neighbour in the north to determine if any changes were necessary in the HEADS.

21.12.5. The Subsystem Testing State

During the Subsystem Testing State (Section 9.12.5) as each subsystem was constructed to the Iteration 0 specifications, it was validated as a stand-alone subsystem. Where and when elements of adjacent subsystems were not available they were simulated by documented calibrated test equipment. The documented calibrated test equipment was then incorporated into the system for use later in the SDP. Once an iteration of a subsystem was approved as being validated, it was turned over to the HEADS systems engineers for integration into HEADS at an appropriate time in an appropriate manner so as to not impact the operation of the then-current ADS.

21.12.6. The Systems Integration and System Testing States

During the Systems Integration and System Testing States (Section 9.12.6) Iteration 0 provided both product and process capability as follows:

- ***Product:*** Iteration 0 laid out the HEADS architecture with enough communications capability to confirm that the concept was feasible.
- ***Process:*** DSTD and FA used Iteration 0 to set up and validate the multi-contractor management, engineering and change control processes.

21.12.7. The O&M State

Iteration 0 went into its O&M State (Section 9.12.7) when the architecture was validated. As new capability was developed, and integrated, the system was upgraded to the pre-planned Iteration 1, Iteration 2 and so on using transition plans evolving from the PP as shown in Figure 13.7.

21.13. The subsequent iterations

From a theoretical textbook perspective, the SDP for the subsequent iterations followed the traditional waterfall sequence conforming to the generic extended SLC shown in Figure 13.7. The activities in each state of the SDP for an iteration revised the products produced for the previous iteration and updated the processes, took into account the intelligence provided on the threat posed by the northern neighbour, and created the appropriate versions for the iteration. The change control system was designed to manage change by being able to assess the impact of a proposed change on each subsystem, the adjacent subsystem and HEADS as a whole. In particular:

- ***The undesirable situation*** for the iteration was the state of HEADS at the time the Iteration began.
- ***The FCFDS*** for the iteration was expressed in a CONOPS using the functionality originally allocated to the iteration together with all approved changes since that time.
- ***The problem*** was to realize the FCFDS.
- ***The solution*** to achieving the FCFDS was to use one waterfall cataract to realize the iteration such that:
  a) ***The Requirements State*** of the iteration focused on the interface requirements between the subsystems and the specifications for the purchase of COTS subsystems.
  b) ***The System Design State*** of the iteration focused on the system architecture.

c) ***The Subsystem Construction and Subsystem Test States*** of the iteration focused on the individual subsystems.
d) ***The System Integration and System Test States*** of the iteration integrated the subsystems of the system into the upgraded subsystem. Once each upgraded subsystem had been validated it was integrated into the HEADS in a manner that minimally impacted the operation of the system.

21.14. Opportunities for class exercises

The context provides an opportunity for many different types of class exercises.

From a practical perspective, things did not go according to plan. Iteration zero had laid out the framework for both the operational HEADS and the realization process using the Cataract Methodology for subsequent iteration as shown in Figure 13.7. However it soon became apparent that the iterations for the subsystems were stretching out and were in danger of losing synchronization with each other. Design and construction of airfields, missile sites, communications facilities all had different problems; some equipment was ready early, some was ready late. Vendors made changes in COTS that had to be investigated. Requests for change came from internal and external stakeholders. Each change request had to be investigated and accepted or rejected. Accepted change requests were allocated to the appropriate iteration depending on the urgency and nature of the impact of the change on the process or the system.

After a while, there was no clear distinction between the various iterations, because as time went by, while the communications links of the C4ISR subsystem were generally implemented according to the schedule, civil construction and delivery of equipment from overseas tended to be late. The procurement officers in the subsystems were sometimes able to compensate and even order supplies from alternative vendors who could deliver ahead of schedule. Project managers and senior systems engineers used Program (Project) Evaluation and Review Technique (PERT), Stoplight/Traffic Light Charts, EVA and Categorized Requirements in Process (CRIP) Charts (Chapter 19) to keep track of dependencies, identify and compensate for risks and take advantage of opportunities provided by early deliveries.

Somewhere in the middle of the third iteration, intelligence reports were received that the unfriendly neighbour was in the process of acquiring a number of surface-to-surface missiles with sufficient range to reach any ground location in Engaporia. The impact of that intelligence was such that a fifth iteration had to be added at additional cost to provide the capability needed to deal with that threat.

21.15. Classroom uses of the ADS upgrade

This Section discusses some of the classroom uses of the ADS upgrade as an Experiential Case Study and as a Role-Playing Case Study.

21.15.1. Use as an Experiential Case Study

This Case Study with its description of the events when used as an Experiential Case Study with the description of the events in sections 21.9 to 21.14 provides a broad overview with plenty of scope for extension, discussion and exercises. Here are some examples of assignments a systems engineering class:

1. Discuss the "why's", namely the reasons for the "what's" discussed in this case.
2. Discuss the SDP described in this case and map it into those discussed in the systems engineering literature.
3. Identify and display the changes from functional to physical, or "what's" and "how's" through the SDP.
4. Discuss the differences between the SDP and the "system engineering process".
5. Discuss the impact the fifth iteration on the project.
6. Identify the roles of systems engineers and project managers and discuss where and why they overlap.
7. Discuss aspects of survivability and robustness of the HEADS.
8. Discuss aspects of risks and opportunities in the HEADS SDP.

21.15.2. Use as a Role-Playing Case Study

As a framework, the reference scenario is that of a successful project and can be used in classes on systems engineering and engineering (project) management.

21.15.2.1. Systems engineering classes

As the students learn about the SDP, they can sequentially perform systems engineering to produce examples of the documentation associated with each state of the SDP. Where things, such as the events discussed in Section 21.14, can go wrong provides plenty of scope for teachers to develop scenarios in the same context to reinforce specific learning points. Here are some examples to be developed for a systems engineering class:

1. Design the conceptual alternatives including the LAMP approach.
2. Reverse engineer the importance of the solution selection criteria shown in Table 21.1 to identify the contents of the appropriate

Engaporean government policies to show the things the government is concerned about and the things it is not? One example is the importance of technology transfer to local industry.

3. Develop the CONOPS for the FCFDS.
4. Define the architecture for HEADS.
5. Develop the DODAF for the HEADS.
6. Develop aspects of survivability and robustness of the HEADS.
7. Develop aspects of risks and opportunities in the HEADS SDP.
8. Role-play the CCB and assess the impact of a change request supplied by the instructor on the HEADS.
9. Identify and prioritise the stakeholders in the different systems and subsystems, Chapter 17 may be used as a guide.

21.15.2.2. Engineering (project) management

The HEADS framework is also useable in a class on project management. One example might be as follows:

The class exercise is set in the fictitious FA which has won the contract to implement the HEADS. The class plan is that the students create the SEMP in the first half of the class, building it up sequentially as they learn about project management. The SEMPs are presented in the management portion of an SRR. The students then manage the project though each state of the SDP in the second part of the class[10].

Each team will perform project management on the same project to allow students to compare their project management with that of other teams. To make the simulation interesting, there will be instructor provided differences between the teams, and a number of different unforeseen events will occur in the second half of the class.

The scope of effort is determined by the number of people in the team and the number of hours students are expected to invest in the class. Within this constraint, presentations are expected to contain a representative sample of information showing that the knowledge acquired during a session has indeed been applied to the HEADS.

The example class is organized in 14 sessions. Starting in Session 3[11], student teams will prepare and present sections of a PP for the HEADS in the following session. The weekly presentation will be the section of the PP based on knowledge learnt in the previous session as specified by the exercise in that session together with any upgraded/corrected sections from the previous session. The weekly presentations from Session 3

[10] The author has successfully used this role-playing scenario, but based on the MCSSRP (Section 17.1) in his classes on project management at NUS. While the students initially complained that it was a lot of work, they also recognized how much they had learned.

[11] Out of 14 sessions

to Session 7 should be considered practice for the SEMP part of the SRR presentation to be presented in Session 8 in which a summary of the SEMP is presented in its entirety[12]. Feedback comments and ideas from other team presentations that would presentations should be incorporated into subsequent presentations. Space precludes further discussion of this and other exercises. However instructor guides for the use of this Case Study may be available as discussed in Section 22.3.

Starting in Session 10 in the second part of the semester, each session represents a major milestone, where:

- Session 10 is the PDR.
- Session 11 in the CDR.
- Session 12 is the TRR.
- Session 13 is the IRR.

Student teams revise the costs and schedules presented at the SRR as a result of one of a number of events, chosen by lot, occurring between the previous milestone and the session milestone[13]. Each presentation covers the period of activity between the previous milestone review and the one being made. During that period of activity one of the following events occurred (as determined by the drawing of a numbered slip during class and provided to team leaders). Note, not all numbers may be drawn each session.

1. Expected resources were not available, project was delayed by two time periods[14].
2. Critical component delivery was late by one time period.
3. Critical component delivery was early by one time period.
4. The Chief Systems Engineer resigned.
5. Company won a major contract for new and exciting project, 50% of managerial staff applied for transfer to new project.
6. Company won a major contract for new and exciting project, 50% of all technical staff applied for transfer to new project.
7. Company won a major contract for new and exciting project, 50% of all technical and managerial staff applied for transfer to new project.

[12] Within the time constraints

[13] Each team will experience a different event.

[14] The students convert the time periods in the event to days, weeks or months depending on their original SEMP presentation. Since the students do not have the detailed knowledge for making estimates, the instructor is looking for consistency and relative amounts.

8. No major glitches, project proceeds according to plan. Note: any team that gets this number in a session is disqualified from receiving this number in a subsequent session.
9. Engineering was implemented smartly reducing the critical path by 20%.
10. Last milestone review presentation generated 20 Review Item Discrepancies (RID).
11. Customer informed you that remaining project schedule is to be speeded up (reduced in time) by 25%.
12. Expected resources were available; project was early by one time period.
13. Customer's budget has been reduced by 25% for rest of project.
14. Customer's budget has been reduced by 35% from previous milestone to this milestone.
15. Another company project ended; you had your choice of up to three additional junior personnel.
16. Customer changed requirements to increase number of inputs to the switch by 50%.
17. Two junior personnel quit.
18. Project manager was severely injured in automobile accident and was on medical leave for ten time periods.
19. Poor engineering resulted in delay of five time periods in task requiring most time.
20. Poor engineering resulted in delay of five time periods in most costly task.
21. Innovative engineering reduced project costs by 10%.
22. Customer cancelled the project (only applicable to Session 13).
23. Vendor/manufacturer of the most critical component went bankrupt and cannot deliver (only applicable after Session 12).

In all instances, students are instructed as follows:

24. Presentation is 10 minutes (max).
    a) Show event number, event and assumed impact of event, in the introduction to the presentation.
    b) Show previous milestone as a baseline and differences due to event.
    c) Do not repeat presentation from last time.
    d) Document and present assumptions and appropriate supporting rationale.
    e) Show lessons learned in exercise during presentation. Lessons learned must come from your experience, not as quoted knowledge from book. For example, 'a plan is important'

is knowledge while 'a plan helped us manage our time because …' can be a lesson learned.

f) Present the process the team went through to perform the exercise.
g) Prefix your presentation file with the session number (e.g. Session 1, presentations shall start with 01, Session 12 presentations with 12, etc.).

25. Assumptions must be realistic and consistent with information presented in prior sessions.
26. Some event mitigation techniques may not be reused by another team when the same event shows up in a later session; instructor will notify students at time of presentation.
27. Do not repeat statements made in previous presentations (your teams' or other teams'), refer to the statements instead.
28. You may have to backfill something (provide information that wasn't there before) into your presentation to cope with events.
29. A time period will depend on the team project's way of measuring time (e.g. weeks, months, etc.)
30. The team can choose (and state in the presentation) when the even took place during the reporting period, i.e. start, sometime in the middle, end.

21.16. Summary

This Chapter contributed to improving systems engineering by introducing the multi-purpose Engaporean Air Defence System (ADS) upgrade Case Study (Kasser, 2013a) to provide:

- A framework for Role-Playing Case Studies in classes on systems engineering and engineering (project) management written in such a manner so as to provide additional examples of the tools, templates and frameworks described in Parts III and IV, including:
    - An example of the generic multi-iteration SDP.
    - An example of the use of an Experiential Case.
    - More examples of the problem formulation template.
    - Yet another example of the use of the HTPs to organize information in a systemic and systematic manner.

--oo--

PART VI

22. Summary and conclusions

This chapter contains the:

- Chapter summaries and a list of the key points in each chapter of the book in Section 22.1.
- Answers to the questions in Chapter 1 in Section 22.2.
- Afterword and location of educational resources in Section 22.3.

22.1. Chapter summaries and key points

Chapter 1 introduced the book.

Chapter 2 summarised some aspects of systems thinking discussed in Volume 1 of this series. The key points were:

- Thinking systemically and systematically.
- Systems thinking provides understanding.
- Holistic thinking goes beyond systems thinking to provide insight as to causes of undesirability and solutions that may remedy the undesirable situations.
- The blind spots when perceiving a situation from a single perspective.
- The need for perceiving situations from multiple perspectives.
- One set of standard perspectives, the HTPs on the perspectives perimeter.
- There are other perspectives on the perspectives perimeter not discussed in this Chapter.

Chapter 3:

- Introduced the layout of the book.
- Described the methodology used to store the perceptions of systems engineering accumulated from 20 years of research in a systemic and systematic manner.
- Provided a set of rules for storing the perceptions of situations for use in the workplace and in Case Studies

The key points in Chapter 3 were:

- Using the HTPs as a document template to facilitate storing and locating information.

- Separation of facts from opinions, insights, inferences and conclusions.
- How to store information in the HTPs.
- Details on how and why specific information is located in specific chapters in this book.

Chapter 4 contained perceptions of systems engineering from the *Big Picture* perspective. The key points were:

- Systems engineering covers a broad spectrum of activities.
- Systems engineering is performed in the context of three streams of activities between reporting milestones.
- Systems engineering succeeds and fails in the real world.

Chapter 5 contained perceptions of systems engineering from the *Operational* perspective. The key points were systems engineers:

- Performing systems engineering to provide value.
- Performing a wide range of different activities in projects in the workplace.
- Continuing their education and training and also mentoring, educating and training junior personnel via journals, books and conferences.
- Using holistic thinking as a way of life or at least some of them.

Chapter 6 contained perceptions of systems engineering from the *Functional* perspective. The key points were:

- Some systems engineers think; most follow the problem-solving process thinking through the problem, conceiving solutions and selecting the most acceptable solutions; some systems engineers just follow processes without thinking.
- Systems engineers remedy problems.
- The difference between the short problem-solving process and the extended holistic problem-solving process.
- The time delays in realizing solutions.

Chapter 7 contained perceptions of systems engineering from the *Structural* perspective. The key points were:

- Systems engineering meets one set of requirements for being a discipline.
- The principle of hierarchies.
- The tools paradox.
- The standards for systems engineering are not standards for systems engineering.
- Some characteristics of a systems engineer.

- The structure of the problem.

Chapter 8 contained perceptions of the similarities between systems engineering and other disciplines from the *Generic* perspective. The key points were:

- The similarity between the SEP, the decision-making process and the problem-solving process.
- The use of models is not unique to systems engineering.
- There are different styles of system engineering in the same way as there are different styles of management.

Chapter 9 contained perceptions of differences within systems engineering and between systems engineering and other disciplines from the *Continuum* perspective. The key points were:

- There are many different definitions of the word "system".
- There are many different definitions of "systems engineering".
- There are many different definitions of the term "requirement".
- The different meaning of the word "problem".
- The difference between problem formulators and problem solvers.
- The different layers of systems engineering.
- The differences between the problem, solution and implementation domains.
- The different states in the SLC.
- The different camps in systems engineering.
- The different roles of the systems engineer.
- The overlap between systems engineering, project management and other disciplines.
- The 'A' and 'B' paradigms.
- The paradoxes and dichotomies in systems engineering.
- Distinguishing between objective and subjective complexity.

Chapter 10 contained perceptions of systems engineering from the *Quantitative* perspective. The key points were:

- Research has shown there is value in systems engineering.
- While requirements are considered an essential part of systems engineering, there is no metric for measuring the goodness of a requirement.
- There are three types of emergent properties.
- There are ways of measuring and improving systems engineering.
- The TRL.

- The four levels of difficulty of a problem.

Chapter 11 contained perceptions of systems engineering from the *Temporal* perspective. The key points were:

- Systems engineering as a discipline has only existed since the middle of the 20th century.
- The evolution of the role of the systems engineer.
- The evolution of requirements engineering.
- The use of models in systems engineering is not a new concept.
- The introduction of, and increase in the degree of micromanagement in, the standards for systems engineering.

Chapter 12 contained inferences and insights on systems engineering from the *Scientific* perspective. The key points were:

- Systems engineering is more than just applying process standards.
- Some reasons why systems engineers cannot agree on the nature of systems engineering.
- There are three types of systems engineering, pure, applied and domain.
- The implementation domain needs to be considered.
- The devolution of systems engineering.
- The need to focus on people as well as process.
- The 'B' paradigm is inherently flawed.
- Five reasons for the failure of systems engineering.
- One reason for the success of systems engineering.
- Dealing with problems.
- The need to change the SDP from a single waterfall to a series of cataracts at process design time.
- While there is a consensus that systems engineering is a discipline there does not seem to be consensus as to what type of discipline.
- The process, roles, emergent properties, tools, and optimisation, paradoxes were resolved.
- MBSE is:
  - Much ado about nothing new.
  - Is an attempt to return to the 'A' paradigm.
- The different ways of assessing systems engineering competency are specific to the originators and not really suitable for general use.
- Systems engineering can be improved by adjusting the terminol-

ogy.

- The challenge of systems optimisation.
- Stop using the terms ‘open systems’ and ‘closed systems’ and use the terms ‘black box” or ‘operational view’ and ‘white box’ or ‘functional/structural view’ as appropriate instead.
- Detailed design decisions shall:
  - Be made on a Just in Time (JIT) basis.
  - Maximize the “don’t care’s”.
- Three of the myths of systems engineering are:
  - There is a single SEP.
  - There are standards for systems engineering.
  - Systems of Systems are a different class of problem and need new tools and techniques.

Chapter 13 suggested improvements to systems engineering based on insights and inferences from perceptions of the SLC. Key insights included:

- The SLC as a State Machine
- The “what’s” and the “how’s” of system engineering match the problem-remedy model.
- The way each state is described via the problem formulation template.
- The generic extended SLC.

Chapter 14 is a continuation of the *Scientific* perspective and contained a selection of tools and frameworks for improving the practice of systems engineering which have been conceptualised, prototyped and found to be useful. They were:

- The TAWOO as a way of predicting technology availability.
- A CMMF for:
  - Comparing different competency models.
  - Assessing competency models for suitability for an organisation.
  - Use as a competency model.
- Using the principle of hierarchies to manage complexity.
- The HKMF.
- A problem formulation template.
- A problem classification framework.
- There are no solutions to ill-structured and Wicked problems. They have to be converted to well-structured problems.

Chapter 15 presented an underpinning axiom for systems engineering. The principles within the axiom apply to the solution system, production of which is the common goal of all the camps within systems engineering. As a consequence, the axiom has the potential to improve systems engineering by uniting the disparate camps within systems engineering by allowing them to agree on the principles applying to the solution system which will then enable the practice of systems engineering to repeat the successes it achieved in the NASA environment in the 1960's and 1970's in all current and future application domains.

Chapter 16 introduced the Nine-System Model to improve systems engineering. Note the Nine-System Model is not a model of systems engineering, it is a framework and tool.

Chapter 17:

- Improved systems engineering by addressing the problems of stakeholder management and requirements elicitation and elucidation which are complex and sometimes the roles, responsibilities and areas of concern of the stakeholders seem difficult to identify and integrate.
- Described a systemic and systematic way of simplifying stakeholder management and requirements elicitation and elucidation in a situational example using the:
  - HTPs to identify the stakeholders.
  - Nine-System Model to sort stakeholders and identify their areas of concern in order to translate their expectations into system requirements in the context of an experiential Case Study example.
- Introduced the concept of direct and indirect stakeholders in addition to internal and external stakeholders.

Chapter 18:

- Improved systems engineering by filling a gap in the systems engineering literature by suggesting a process for creating a system to be used in the early states of the SDP to help to manage complexity at the time the system is created by optimizing the interfaces.
- Described the S2 process in the Nine-System Model discussed in Chapter 16.
- Described the contribution of the HTPs to the systems requirements.

Chapter 19 improved systems engineering by introducing CRIP Charts which provided a way to:

- Measure technical progress.
- Identify potential problems in near real-time so as to be able to mitigate the problems before they occur.

The CRIP Charts introduced in this Chapter can be used in both the 'A' and 'B' paradigms since they trace work back to requirements. However, although written up for requirements, they can also be used for Use Cases, scenarios, Technical Performance Measures (TPM) and any other technical measurement that can be tracked across the SDP.

Chapter 20 introduced a template to improve the quality of practitioner written Experiential Case Studies to format the practitioner papers as a way to link their experiences into the literature in a systemic and systematic manner to provide information to assist students studying, and researchers improving, the practice of systems engineering. Key points included:

- Described a previous experience of the use of a template for postgraduate student assignments.
- Introduced a template to improve the quality of student and practitioner Experiential Case Study papers to provide practitioners with a way to link their experiences into the literature to provide data to assist students learning about systems engineering and researchers improving the practice of systems engineering.
- Provided an Experiential Case Study to provide an example of the template in action.

Chapter 21 contributed to improving systems engineering by introducing a multi-purpose Case Study to provide:

- A framework for Role-Playing Case Studies in classes on systems engineering and engineering (project) management written in such a manner so as to provide additional examples of the tools, templates and frameworks described in Parts III and IV, including:
  - An example of the generic multi-iteration SDP.
  - An example of the use an Experiential Case Study.
  - More examples of the problem formulation template.
  - Yet another example of the use of the HTPs to organize information in a systemic and systematic manner.

22.2. The answers to questions in Chapter 1

The answers to the questions in Chapter 1 are as follows.

22.2.1. What is systems engineering?

The answer depends on whom you ask, because:

- ***Process camp*** systems engineers (Section 9.17.2) will tell you that systems engineering is a process, and probably quote from the INCOSE Systems Engineering Handbook (Haskins, 2011) and/or ISO /IEC 15288.
- ***Problem-solving camp*** systems engineers (Section 9.17.3) will tell you that systems engineering is solving complex problems and providing the best solution available given the constraints at the time.
- ***Meta-discipline*** camp systems engineers (Section 9.17.4) will tell you that systems engineering incorporates the other disciplines and will add that systems engineering needs to widen its span to take over the other disciplines.
- ***Systems thinking*** camp systems engineers (Section 9.17.5) will tell you that systems engineering is the application of systems thinking.
- ***Domain systems*** camp systems engineers (Section 9.17.7) will tell you that systems engineering is what they do to their particular system.
- ***Enabler camp*** systems engineers (Section 9.17.8) will tell you that systems engineering can be, and is, used in all disciplines for tackling certain types of complex and non-complex problems.

After about 45 years of practicing systems engineering and 20 years of researching systems engineering, my answers to the question are:

- Perceptions from the *Operational* perspective indicate that systems engineering is a systemic and systematic way of converting a complex or non-complex undesirable situation into a desirable situation. Each camp of systems engineering (Section 9.17) does this in a different manner with different degrees of success.
- Perceptions from the *Generic* perspective indicate that systems engineering is an enabling discipline. Just as mathematics can be considered as an enabling discipline because it provides tools to other disciplines to solve mathematical problems, systems engineering is an enabling discipline that provides the conceptual tools for other disciplines to use to provide a remedy to complex problems.

My recommendation is systems engineers in the enabler camp differentiate themselves from systems engineers in the other camps and start calling themselves *solution engineers*, and describe what they do as *solution engineering*.

The perceptions from the *Operational* and *Generic* perspectives can be combined to define *solution engineering* as an enabling discipline. Just as mathematics can be considered as an enabling discipline because it provides tools to other disciplines to solve mathematical problems, *solution engineering* is an enabling discipline that provides the conceptual tools for other disciplines to use to provide a remedy to complex problems.

Solution engineering does this by going beyond systems thinking and applying holistic thinking to understand the situation and then conceptualize and use a systemic and systematic approach to realise the solution system, which when operating in its context will remedy the causes of the original, evolved and emergent undesirability.

22.2.2. Why are there different opinions on the nature of systems engineering?

The answer is the different options stem from reasons that include:

- Systems engineers perceiving different aspects of systems engineering from single viewpoints from the different areas of the HKMF (Section 14.4).
- Systems engineers perceiving systems engineering from single viewpoints from their camp (Section 9.17)
- The different allocations of systems engineering and non-systems engineering activities to the role of the systems engineer (SETR) (Section 9.18).

22.2.3. Why does systems engineering succeed at times?

The answer is discussed in section 12.11.

22.2.4. Why does systems engineering fail at other times?

The answer is discussed in section 12.10.

22.2.5. Why does systems engineering seem to overlap project management and problem-solving?

The answer is discussed in Section 12.7.

22.2.6. Why do the textbooks about systems engineering cover such different topics?

The answer is the differences in the content of textbooks on systems engineering discussed in Section 9.22 was because each textbook focused on a different mix of pure systems engineering, applied systems engineering and domain systems engineering (Section 12.2).

22.2.7. What do systems engineers actually do in the workplace?

The answer is, they do what their supervisor tells them to do (SETR); generally a mixture of system engineering and non-systems engineering activities (SETA).

22.2.8. Is systems engineering an undergraduate course or a post graduate course?

This is a closed question that requires an either-or answer. In terms of the knowledge, the holistic thinking approach rephrases the question from the *Functional* and *Operational* perspectives as, "*What knowledge do systems engineers need to know to perform their duties in an effective manner?*" While the answer to this question depends on the systems engineering camp and area in the HKMF (Section 14.4) in which they are working, the information is out there and can be found. Some research in to the content of syllabi of undergraduate and postgraduate course will then identify where the topics are being taught and provide an answer to the original question.

However, it is likely that once the knowledge topics are identified and documented in a SEBOK, the different campers and supporters of the different roles will argue about the inclusion of various topics in the SEBOK.

The holistic thinking approach rephrased the problem in the same way that it rephrased the problem of determining the maturity of technology (Section 14.1.4).

Now studies in education have shown that in order to learn something, the learner has to have an anchor point for the new knowledge. This raises a second question as, "*Does an undergraduate have the anchor points to enable retention of the knowledge taught in a class on systems engineering?*" The answer is, *"it depends"*.

22.2.9. Which come first, functions or requirements?

The answer is, it depends. In the:

- 'A' paradigm, functions generally come first being developed in CONOPS in the Needs Identification State of the SLC discussed in Section 9.12.1.
- 'B' paradigm, requirements generally come first followed by functions since the CONOPS is developed from the requirements as discussed in Section 9.12.2.

In both paradigms, the design process for subsystem components creates a functional design which may be followed by requirements for the physical designs should the subsystem be sufficiently complex.

22.2.10. Why is there no standard definition of a system?

The answer is the many definitions of a system including those listed in Section 9.1 are formulations of problem statements by the persons who wrote the definitions. They defined their system to suit their problem. However, within all the definitions there is a consensus that the minimum requirement[1] for something to be a system is that:

1. It has to consist of more than one part.
2. There has to be some interaction between the parts.
3. The function performed by the system can only be performed by the combination of the parts and the interaction between the parts.

Disputes about whether something is or is not a system can be dissolved if each party in the dispute note that the opposing argument is based on a different definition of a system.

22.3. Afterword, educational resources

This book is designed for use by practitioners, students and educators. Accordingly, the original figures used in this book can be found for use in the classroom under Creative Commons licensing on the author's educational resources web page at
http://therightrequirement.com/Resources.

The insights, inferences, tools and frameworks developed from the *Scientific* perspective presented in this book have:

- Been used successfully.
- The potential to move the practice of systems engineering forward. The question is, "is the movement in the right direction"? The journey continues, and time will tell.

--oo--

[1] Some systems engineers insist that the system has to meet all the parts of their definition; hence the disagreement as to what constitutes a system.

23. Glossary of terms and acronyms

Since the following acronyms are used in more than one section in this book, this table is provided to enable the reader to quickly identify an unremembered acronym.

ABL	Airborne Laser
ABLT	ABL Test Bed
ACM	Active countermeasures
ADS	Air Defence System
AHP	Analytical Hierarchical Process
ANSI	American National Standards Institute
APCOSEC	Asia-Pacific Council on Systems Engineering Conference
BCS	Beam Control System
BPR	Business Process Reengineering
BPR	Business Process Reengineering
C3	Communications, Command, and Control
C4ISR	Command, control, communications, computers, intelligence, surveillance, and reconnaissance
C4ISREW	C4ISR Electronic Warfare
CAIV	Cost as an Independent Variable
CASE	Computer Aided Software Engineering
CCB	Configuration Change Board
CDR	Critical Design Review
CESE	Computer Enhanced Systems Engineering
CEST	Capacity for Engineering Systems Thinking
CM	Configuration Management
CMM	Capability Maturity Model
CMMF	Competency Maturity Model Framework
CMMI	CMM- Integrated
COBOL	Common Business Oriented Language
COIL	Chemical Oxygen Iodine Laser
CONOPS	Concept of Operations

COTS	Commercial-Off-The-Shelf
CSEP	Certified Systems Engineer Professional
CSER	Conference on Systems Engineering Research
DCAS	Defense Contract Administration Services
DERA	Defence Evaluation and Research Agency (UK)
DMSMS	Diminishing Manufacturing Sources And Material Shortages
DoD	Department of Defense [US]
DODAF	DoD Architecture Framework
DRR	Delivery Readiness Review
DSTD	Defence Systems and Technology Department
DT&E	Development Test and Evaluation
dTRL	dynamic TRL
ECDA	Engaporean Capability Development Agency
ECP	Engineering change process
EDF	Engaporean Defence Force
EIA	Electronic Industries Alliance
EVA	Earned Value Analysis
FASE	Federated Aerospace of Engaporia
FCFDS	Feasible conceptual future desired situation
FLASH	Force-Level Australian Defence Force Systems Harbinger
FLSE	Force Level Systems Engineering
FORTRAN	FORmula TRANslation
FRAT	Functions Requirements Answers and Test
GAO	Government Accounting Office (US)
GDRC	Global Development Research Center
GIGO	Garbage-In-Garbage-Out
GSFC	Goddard Space Flight Center
HEADS	Holistic Engaporean Air Defence System
HKMF	Hitchins- Kasser- Massie Framework
HR	Operational Test and Evaluation
HTP	Holistic Thinking Perspectives
ICAM	Integrated Computer Aided Manufacturing
ICBM	Inter-Continental Ballistic Missile
ID	Identification
IEAust	Institute of Engineers Australia

IDEF	ICAM Definition
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
IIE	Integrated Information Environment
ILS	Integrated Logistics Support
INCOSE	International Council on Systems Engineering
IPPD	Integrated Product and Process Development
IPT	Integrated Product Teams
IRR	Integration Readiness Review
ISO	International Standards Organisation
ISS	International Space Station
IV&V	Independent Verification and Validation
JAD	Joint Applications Development
JAXA	Japan Aerospace Exploration Agency
JIT	Just in Time
JPL	Jet Propulsion Laboratory
JSF	Joint Strike Fighter
KSA	Knowledge, Skills, and Abilities
LAMP	Lighter Than Air Missile Platforms
LCC	Life Cycle Costing
LEO	Low Earth Orbit
LOC	Local Controller
MATO	Multiple-Award-Task-Ordered
MBSE	Model Based Systems Engineering
MBTF	Mean Time between Failures
MBWA	Management by Walking Around
MCSS	MSOCC data switching system
MIL-STD	Military-Standard
MISE	Master of Industrial and Systems Engineering
MOU	Memorandum of Understanding
MSOCC	Multi-Satellite Operations Control Center
MTTR	Mean Time to Repair
MVA	Multi-attribute variable analysis
NASA	National Aeronautical and Space Administration
NASCOM	NASA Communications Network
NDIA	National Defense Industrial Association

NGT	Nominal Group Technique
NMOS	Network Maintenance and Operations Support
O&M	Operations and Maintenance
OCD	Operations Concept Document
OCH	Operations Concept Harbinger
OCR	Operations Concept Review
OPM	Office of Personnel Management
OPM	Object-Process Methodology
OS	Operational system (in Whole System Model)
OT&E	Operational Test and Evaluation
OVAE	Office of Vocational and Adult Education
PBL	Performance Based Logistics
PDR	Preliminary Design Review
PERT	Program (Project) Evaluation and Review Technique
PI	Principal Investigator
PP	Project Plan
QA	Quality Assurance
QC	Quality Control
QFD	Quality Function Deployment
QRD	Quality Requirements Definition
RAF	Royal Air Force
RAFBADS	RAF Battle of Britain Air Defence System
RFP	Request for Proposal
RFT	Request for Tender
RID	Review Item Discrepancy
RTM	Requirements Traceability Matrix
ROI	Return on Investment
S&T	Science and Technology
SAGE	SemiAutomatic Ground Environment
SAM	Surface-to-air missile
SCADC	Standard Central Air Data Computer
SDLC	System Development Lifecycle
SDP	System Development Process
SEAS	Systems Engineering and Services
SEBOK	Systems Engineering Body of Knowledge
SECAM	Systems Engineering Capability Assessment Model

SECF	Systems Engineering Competencies Framework
SECT	Systems Engineering Competency Taxonomy
SEEC	Systems Engineering and Evaluation Centre
SEGS	Solar Electrical Power Generating System
SEMP	Systems Engineering Management Plan
SEP	Systems Engineering Process
SERC	Systems Engineering Research Center
SESA	Systems Engineering Society of Australia
SETA	Systems Engineering - The Activity
SETE	Systems Engineering Test and Evaluation
SETR	Systems Engineering - The Role
SIMILAR	State, Investigate, Model, Integrate, Launch, Assess and Re-evaluate
SLC	System Life Cycle
SOI	System of Interest
SPRDE	Systems Planning, Research, Development, and Engineering
SRD	System Requirements Document
SRR	System Requirements Review
SSM	Soft Systems Methodology
SysML	Systems Modeling Language
T&E	Test and Evaluation
TAWOO	Technology Availability Window of Opportunity
TCO	Total Cost of Ownership
TEMP	Test and Evaluation Master Plan
TMD	Theater Missile Defense
TQM	Total Quality Management
TRL	Technology Readiness Level
TRR	Test Readiness Review
TTM	Time To Market
UAV	Unmanned Aerial Vehicle
UK	United Kingdom (of Great Britain)
UML	Unified Modeling Language
UMUC	University of Maryland University College
UniSA	University of South Australia
US	United States of America
USAF	US Air Force

WP	Work Package

-oo--

24. References

Ackoff, R. L., *Redesigning the Future: A Systems Approach to Societal Problems*, John Wiley & Sons, New York, 1974.

---., "The Future of Operational Research is Past," *Critical Systems Thinking Directed Readings*, R. L. Flood and M. C. Jackson (Editors), 1991.

---, *Ackoff's Best. His Classic Writings on Management*, John Wiley & Sons, Inc, New York, 1999.

Ackoff, R. L. and Addison, H. J., *A Little Book of f-Laws 13 common sins of management*, Triarchy Press Limited, Axminster, 2006.

Ackoff, R. L., Addison, H. J. and Andrew, C., *Systems Thinking for Curious Manaers*, Triachy Press Ltd., Axminster, Devon, UK., 2010.

Adcock, R. D., "Tailoring Systems Engineering Lifecycle Processes to meet the challenges of Project and Programme applications," *The 15th International Symposium of the International Council on Systems Engineering (INCOSE)*, Rochester, NY., 2005.

Alexander, I. F. and Stevens, S., *Writing Better Requirements*, Addison-Wesley, 2002.

Alexander, J. E. and Bailey, J. M., *Systems Engineering Mathematics*, Prentice-Hall, Inc., Englewood Cliff, NJ., 1962.

Allen, M., *Smart thinking: skills for critical understanding and writing*, Oxford University Press, 2004.

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.

Anderson, K. and Dibb, P., "Strategic Guidelines for enabling research and development to support Australian Defence, paragraph 121," 1996.

Anderson, L. W., Krathwohl, D. R., Airasian, P. W., Cruikshank, K. A., Mayer, R. E., Pintrich, P. R., Raths, J. and Wittrock, M. C. (Editors), *A Taxonomy for Learning, Teaching, and Assessing: A Revision of Bloom's Taxonomy of Educational Objectives*, Allyn and Bacon, 2000.

Angel, C. D. and Froelich, J., Six Sigma: What Went Wrong?, 2008, http://www.destinationcrm.com/Articles/Columns-

Departments/The-Tipping-Point/Six-Sigma-What-Went-Wrong-51394.aspx, accessed on 26 April 2009.

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

Archer, L. B., *Systematic Method for Designers*, Council of Industrial Design, London, 1965.

Armstrong, J. R., *How Maturity Modeling Saved My Softball Team*, proceedings of the 8th INCOSE International Symposium, Vancouver, BC., 1998.

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

---, "ISO 15288 Systems engineering — System life cycle processes," International Standards Organisation, 2002.

Athey, T. H., *Systematic Systems Approach. An Integrated Method for Solving Systems Problems*, Prentice-Hall, Englewood Cliffs, NJ., 1982.

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

Avison, D. and Fitzgerald, G., *Information Systems Development: Methodologies, Techniques and Tools*, McGraw-Hill Education (UK), Maidenhead, 2003.

Bahill, A. T. and Dean, F. F., *The Requirements Discovery Process*, proceedings of the 7th International Symposium of the INCOSE, Los Angeles, CA., 1997.

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 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.

Bate, R., Kuhn, D., Wells, C., Armitage, J., Clark, G., Cusick, K., Garcia, S., Hanna, M., Jones, R., Malpass, P., Minnich, I., Pierson, H., Powell, T. and Reichner, A., A Systems Engineering Capability Maturity Model Version 1.1, Software Engineering Institute, Carnegie Mellon University, SECMM-95-01, CMU/SEI-95-MM-003, , 1995.

Beasley, R. and Partridge, R., *The Three Ts of Systems Engineering - Trading, Tailoring and Thinking*, proceedings of the 21st Annual Symposium of the INCOSE, Denver, CO., 2011.

Beer, S., *The Heart of Enterprise*, John Wiley & Sons, Stafford Beer Classic Edition, Chichester, 1994.

Benjamin, D., "Technology Readiness Level: An Alternative Risk Mitigation Technique," *Project Management in Practice: The 2006 Project*

Risk and Cost Management Conference, Boston University Metropolitan College, Boston, MA, 2006.

Bentley, C., *PRINCE 2 A Practical Handbook*, Oxford: Butterworth Heinemann, 1997.

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

Biggs, J., *Teaching for Quality Learning in University*, Society for Research into Higher Education and Open University Press, 1999.

Bissell, C., *A great disappearing act: the electronic analogue computer*, proceedings of IEEE Conference on the History of Electronics, Bletchley Park, UK, 2004.

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

Boardman, J. and Sauser, B., *Systems Thinking coping with 21st century problems*, CRC Press, Boca Raton, FL., 2008.

Boehm, B., *A Spiral Model of Software Development and Enhancement*, IEEE Computer (1988), no. May.

---, *Integrating Software Engineering and Systems Engineering*, Systems Engineering 1 (1994), no. 1.

Bottomly, P. C., Brook, P., Morris, P. W. and Stevens, R., *A Study of the Relationship of Systems Engineering to Project Management*, proceedings of Fourth Annual Symposium of the INCOSE-UK, 1998.

Brecka, J., *Sabotage! Survey Finds Internal Resistance to Quality Initiatives*, Quality Progress (1994).

Brill, J. H., *Systems Engineering ---A Retrospective View*, Systems Engineering 1 (1998), no. 4, 258-266.

Brodie, E. J., *Software Engineering An Object-Oriented Perspective*, John Wiley & Sons, Inc., 2001.

Brooks, K., Career Success Starts With a "T", Psychology Today, 2012, https://www.psychologytoday.com/blog/career-transitions/201204/career-success-starts-t, accessed on 19 June 2015.

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.

Buede, D. M., *The Engineering Design of Systems*, John Wiley & Sons, Inc., 2000.

Bungay, S., *The Most Dangerous Enemy*, Aurum Press, London, England, 2000.

Buren, J. V. and Cook, D. A., *Experiences in the Adoption of Requirements Engineering Technologies*, Crosstalk - The Journal of Defense Software Engineering (1998), no. December.

CareerOneStop, Using Competency Models, 2011, http://www.careeronestop.org/competencymodel/learncm.aspx, accessed on 11 November 2011.

Chamberlain, R. G. and Shishko, R., *Fundamentals of Systems Engineering at NASA*, proceedings of INCOSE/ASEM Proceedings, 1991.

CHAOS, The Chaos study, The Standish Group, 1995, http://www.standishgroup.com/chaos.html, accessed on March 19, 1998.

Chapanis, A., "Human Engineering," *Operations Research and Systems Engineering*, C. D. Flagle, W. H. Huggins and R. H. Roy (Editors), Johns Hopkins Press, Baltimore, 1960.

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

Checkland, P., *Systems Thinking, Systems Practice*, vol. Chichester, John Wiley & Sons, 1991.

Checkland, P. and Holwell, S., *Information, Systems and Information Systems: making sense of the field*, vol. Chichester, John Wiley & Sons, 1998.

Checkland, P. and Scholes, J., *Soft Systems Methodology in Action*, John Wiley & Sons, 1990.

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

Churchman, C. W., *The Systems Approach and its Enemies*, Basic Books, Inc., New York, 1979.

Clark, R., *System Dynamics and Modeling*, Operations Research Society of America, 1998.

Clausing, D. and Cohen, L., *Recent Developments in QFD in the United States*, proceedings of Institution of Mechanical Engineering Conference, Coventry, England, 1994.

CMM, *Carnegie Mellon University, The Capability Maturity Model: Guidelines for Improving The Software Process*, Addison-Wesley, 1995.

Colwell, B., *Complexity in Design*, IEEE Computer 38 (2005), no. 10, 10-12.

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

Cook, S. C., Kasser, J. E. and Asenstorfer, J., *A Frame-Based Approach to Requirements Engineering*, proceedings of 11th International Symposium of the INCOSE, Melbourne, Australia, 2001.

Cook, S. C., Kasser, J. E. and Ferris, T. L. J., *Elements of a Framework for the Engineering of Complex Systems*, proceedings of the 9th ANZSYS Conference, Melbourne, 2003.

Costello, R. B., Bolstering Defense Industrial Competitiveness: Preserving Our Heritage, the Industrial Base Securing Our Future, Office of the Undersecretary of Defence (Acquisition), 1988.

Covey, S. R., *The Seven Habits of Highly Effective People*, Simon & Schuster, 1989.

Crépin, M., El-Khoury, B. and Kenley, C. R., "It's All Rocket Science: On the Equivalence of Development Timelines for Aerospace and Nuclear Technologies," *the 22nd Annual International Symposium of the International Council on Systems Engineering*, Rome, Italy, 2012.

Crosby, P. B., *Quality is Free*, McGraw-Hill, 1979.

Cundiff, D., "Manufacturing Readiness Levels (MRL)," 2003.

Daniels, J., Bahill, T. and Botta, R., "A Hybrid Requirements Capture Process," *the 15th annual International Symposium of the INCOSE*, Rochester, NY., 2005.

Darke, P. and Shanks, G. G., *User Viewpoint Modelling: understanding and representing user viewpoints during requirements definition*, Information Systems Journal 7 (1997), no. 3, 213-219.

DAU, Defense Acquisition University (DAU). SPRDE-SE/PSE competency model, 4/14/10 version, 2010, https://acc.dau.mil/CommunityBrowser.aspx?id=315691&lang=en-US, accessed on

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

Demarco, T., *The Deadline: A Novel about Project Management*, Dorset House Publishing, New York, 1997.

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

Deming, W. E., *The New Economics for Industry, Government, Education*, MIT Center for Advanced Engineering Study, 1993.

Denzler, D. W. R. and Kasser, J. E., *Designing Budget Tolerant Systems*, proceedings of the First International Symposium on Reducing the Cost of Spacecraft Ground Systems and Operations Rutherford Appleton Laboratory, England, 1995.

Denzler, D. W. R. and Mackey, W., *An Operations Concept Development Methodology Using a Graphic Process Flow Technique*, proceedings of 4th Annual International Symposium of the National Council of Systems Engineering (NCOSE), San Jose, CA, 1994.

DERA, DERA Systems Engineering Practices Reference Model, DERA/LS(SEC-FH)/PROJ/018/G01, Defence Evaluation and Research Agency (DERA), 1997.

---, *Systems Engineering has a Promising Future*, DERA News (1998), no. January.

Descartes, R., *A Discourse on Method, trans. E. S. Haldane and G. R. T. Ross, Part V*, Washington Square Press, New York, 1637, 1965.

Dewey, J., *How We Think*, D.C. Heath and Company, New York, 1933.

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, DoD Guide to Integrated Product and Process Development (Version 1.0),, DOD, 1996.

---, "DoD Integrated Product and Process Development Handbook," O. o. t. U. o. D. A. a. Technology) (Editor), Department of Defence, Washinton DC, 1998.

---, The DoDAF Architecture Framework Version 2.02, US DOD, 2010, http://dodcio.defense.gov/dodaf20.aspx, accessed on 27 December 2012.

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.

Dolan, T., *Best Practices In Process Improvement*, Quality Progress (2003), no. August 2003, 23-28.

Dorfman, M. and Thayer, R. H., *System and Software Requirements Engineering*, IEEE Computer Society Press, 1990.

DOT&E, Director, Operational Test & Evaluation (DOT&E) FY'99 Annual Report submitted to Congress February 2000, 2000.

Dowling, T. and Pardoe, T., Timpa - Technology Insertion Metrics Volume 1., QinetiQ, 2005.

Dreyfus, H. and Dreyfus, S., *Mind over machine: The power of human intuition and expertise in the era of the computer*, Free Press, New York, 1986.

Drucker, P. F., *Management: Tasks, Responsibilities, Practices*, Harper & Roe, New York, 1973.

DSMC, *Systems Engineering Management Guide*, vol. May 1996, Defence Systems Management College, 1996.

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

EIA 731, "System Engineering Capability Model, EIA/IS 731," Electronics Industry Association, Washington DC, 1998.

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

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

---, *Essentials of project and systems engineering management*, Wiley, 2008.

El-Khoury, B., "Analytic framework for TRL-based cost and schedule models," *Engineering Systems Division*, vol. Master of Science in Technology and Policy, Massachusetts Institute of Technology, 2012, p. 163.

Elm, J. P., Goldenson, D. R., Emam, K. E., Donatelli, N. and Neisa, A., A Survey of Systems Engineering Effectiveness - Initial Results (with detailed survey response data), CMU/SEI-2008-SR-034, Software Engineering Institute, Carnegie Mellon University, 2008.

ElMaraghy, W., ElMaraghy, H., Tomiyama, T. and Monostori, L., *Complexity in engineering design and manufacturing*, CIRP Annals - Manufacturing Technology 61 (2012), no. 2, 793-814.

Emes, M., Smith, A. and Cowper, D., *Confronting an identity crisis - How to brand systems engineering*, Systems Engineering 8 (2005), no. 2, 164-186.

Endres, A. and Rombach, D., *A Handbook of Software and Systems Engineering*, Pearson Education Ltd., 2003.

Ennis, M. R., Competency Models: A Review of the Literature and The Role of the Employment and Training Administration (ETA), U. S. Department of Labor, 2008, http://www.careeronestop.org/competencymodel/info_documents/OPDRLiteratureReview.pdf, accessed on 8 September 2011.

Eppinger, S. D. and Browning, T. R., *Design Structure Matrix Methods and Applications*, The MIT Press, Cambridge, MA., 2012.

ETA, General Competency Model Framework, The Employment and Training Administration, 2010, http://www.careeronestop.org/competencymodel/pyramid.aspx, accessed on 9 September 2011.

Evans, D. C., Systems Engineering Lessons Learned (SELL), 1989, http://evansopticalengineering.com/Page00/sysengll.htm, accessed on 6 September 2011.

Evans, R. P., *Engineering of Computer-based Systems (ECBS) Three New Methodologies--Three New Paradigms*, proceedings of the 6th International Symposium of the INCOSE, Boston, MA., 1996.

Fanmuy, G., *Best practices for drawing up a requirements baseline - P192*, proceedings of the 14th International Symposium of the INCOSE, Toulouse, France, 2004.

Faulconbridge, R. I. and Ryan, M. J., *Managing Complex Technical Projects: A Systems Engineering Approach*, Artech House, Boston, 2003.

Fayol, H., *General and Industrial Management*, Sir Isaac Pitman and Sons, Ltd., London, 1949.

Fielden, G. B. R., ('The Fielden Report') Engineering Design, Her Majesty's Stationary Office, 1963.

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.

Flood, R. L. and Jackson, M. C., *Creative Problem Solving*, Wiley, 1991.

FM_770-78, Field Manual: System Engineering, Headquarters, Department of the Army, 1979.

Ford, H. and Crowther, S., *My Life and Work*, Doubleday Page & Company, New York, 1922.

Ford, W., Learning and teaching math, 2010, http://mathmaine.wordpress.com/2010/01/09/problems-fall-into-

four-categories/, accessed on 8 April 2015.
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.
Frank, M., *Characteristics of Engineering Systems Thinking: A 3-D Approach for Curriculum Content*, IEEE Transaction on Systems, Man, and Cybernetics, Part C 32 (2002), no. 3, 203-214.
---, *Knowledge, Abilities, Cognitive Characteristics and Behavioral Competences of Engineers with High Capacity for Engineering Systems Thinking (CEST)*, The INCOSE Journal of Systems Engineering 9 (2006), no. 2, 91-103.
---, *Assessing the interest for systems engineering positions and the Capacity for Engineering Systems Thinking (CEST)*, Systems Engineering 13 (2010), no. 2.
Frank, M. and Waks, S., *Engineering systems thinking: A multifunctional definition*, Systemic Practice and Action Research 14 (2001), no. 3, 361-379.
Friedenthal, S., Griego, R. and Sampson, M., *INCOSE Model Based Systems Engineering (MBSE) Initiative*, proceedings of the 17th International Symposium of the INCOSE, San Diego, 2007.
Gabb, A., *Front-end Operational Concepts - Starting from the Top*, proceedings of The 11th Annual Symposium of the INCOSE, Melbourne Australia, 2001.
Gabb, A., Caple, G., Haines, D., Lamont, D., Davies, P., Hall, A., Van Gaasbeek, J., Eppig, S., Jones, D. and Vietinghoff, B., *Requirements Categorization*, proceedings of the 11th Annual Symposium of the INCOSE, Melbourne Australia, 2001.
Gabbar, H. A., Shimada, Y. and Sukuzi, K., *Object Interface System Using a Plan Object-Oriented Model*, Systems Engineering 4 (2001), no. 3.
GAO, BEST PRACTICES Better Management of Technology Development Can Improve Weapon System Outcomes, GAO/NSIAD-99-162, United States General Accounting Office, 1999.
---, Uncertainties Remain Concerning the Airborne Laser's Cost and Military Utility, GAO-04-643R, 2004.
---, DEFENSE ACQUISITIONS Major Weapon Systems Continue to Experience Cost and Schedule Problems under DOD's Revised Policy, GAO, 2006.
---, Defense Acquisitions, Assessments of Selected Weapon Programs, GAO-10-388SP, United States General Accounting Office, 2010.
Gardiner, G., *Concurrent and Systems Engineering same thing, different name, or are they both just new product introduction?*, IEE Engineering Management Journal (1996), no. February.

Gavito, V., Verma, D., Dominick, P., Pennotti, M., Giffin, R., Barrese, T. and et al., Technical leadership development program, Systems Engineering Research Center (SERC), Stevens Institute of Technology, Hoboken, NJ., 2010.

GDRC, The Problem Solving Process, 2009, http://www.gdrc.org/decision/problem-solve.html, accessed on 11 Jan 2009.

Gelbwaks, N. L., *AFSCM 375-5 as a Methodology for System Engineering*, Systems Science and Cybernetics, IEEE Transactions on 3 (1967), no. 1, 6-10.

Gelosh, D., *Development and Validation of a Systems Engineering Competency Model*, proceedings of 11th NDIA Systems Engineering Conference, San Diego, CA, 2008.

George, C. S., Jr., *The History of Management Thought*, Prentice-Hall Inc., 1972.

Gharajedaghi, J., *System Thinking: Managing Chaos and Complexity*, Butterworth-Heinemann, Boston, 1999.

Gibbons, P. J., *A Case Study in the Application of UML and OOA/D in an Information Management Program*, proceedings of The 11th Annual Symposium of the INCOSE, Melbourne Australia, 2001.

Giganti, J. J., Larson, J. V., Richard, J. P. and Weber, J., "12 Lunar Surface Gravimeter Experiment," *APOLLO 17 Preliminary Science Report* E. M. Jones and K. Glover (Editors), NASA, 1971.

Glass, R. L., *Building Quality Software*, Prentice Hall, Englewood Cliffs, NJ., 1992.

Godfrey, S., "What is CMMI?," *Systems Engineering Lecture Series*, Goddard Space Flight Center, Greenbelt, MD, 2004.

Goldberg, R. and Assaraf, S., *System Engineering Process Improvement using the CMMI in Large Space Programs*, proceedings of 13thAnnual NDIA System Engineering Conference 25-28 Oct., San Diego CA, 2010.

Goldsmith, R. F., *Discovering Real Business Requirements for Software Project Success*, Artech House Inc., Boston, MA, 2004.

Goleman, D., *Emotional Intelligence*, 1995.

---, Leadership that gets results, Academic OneFile, 2000, accessed on 2.

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

Gooding, R. Z., *Systems Engineering: A problem solving approach to improving program performance*, Evaluation and Program Planning 3 (1980), 95-103.

Gordon, G. et al., *A Contingency Model for the Design of Problem Solving Research Pro-gram*, Milbank Memorial Fund Quarterly (1974), 184-220.

GradSchools.com, Systems Engineering Graduate Programs, 2013, http://www.gradschools.com/search-programs/systems-engineering, accessed on 31 October 2013.

Grobshtein, Y. and Dori, D., "Generating SysML Views from an OPM Model: Design and Evaluation " *the 19th Annual International Symposium of the International Council of Systems Engineering* Chicago, IL., 2010.

Guo, J., "Incorporating Multidisciplinary Design Optimization into Spacecraft Systems Engineering," *8th Annual Conference on Systems Engineering Research*, Hoboken, NJ., 2010.

Hall, A. D., *A Methodology for Systems Engineering*, D. Van Nostrand Company Inc., Princeton, NJ, 1962.

---, *Metasystems Methodology. a new synthesis and unification*, Pergamon Press, Oxford, England., 1989.

Halligan, R., "A Journey Through Systems Engineering in an Hour and a Quarter," INCOSE Singapore Chapter, 2014.

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

Hari, A., Kasser, J. E. and Weiss, M. P., *How lessons learnt from creating requirements for complex systems using QFD led to the evolution of a process for creating quality requirements for complex systems*, Systems Engineering: The Journal of the International Council on Systems Engineering 10 (2007), no. 1, 45-63.

Hari, A., Shoval, S. and Kasser, J. E., *Conceptual Design to Cost: A new systems engineering tool*, proceedings of the 18th International Symposium of the INCOSE, Utrecht, Holland, 2008.

Hari, A., Weiss, M. and Zonnenshain, A., *ICDM - An Integrated Methodology for the Conceptual Design of New Systems*, proceedings of System Engineering Test and Evaluation Conference SETE 2004, Adelaide, Australia, 2004.

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

---, *Was W. Edwards Deming wrong?*, Measuring Business Excellence 4 (2000), no. 3, 35 - 41.

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

--- (Editor), *Systems Engineering Handbook: A Guide for Life Cycle Processes and Activities, Version 3.1. Revised by K. Forsberg and M. Krueger*, The International Council on Systems Engineering, San Diego, CA., 2006b.

--- (Editor), *Systems Engineering Handbook: A Guide for Life Cycle Processes and Activities, Version 3.2.1. Revised by K. Forsberg and M. Krueger*, The International Council on Systems Engineering, San Diego, CA., 2011.

Hately, D. J. and Pirbhai, I. A., *Strategies for Real-time systems Specification*, Dorset House Publishing, New York, NY, 1987.

Hauser, J. R. and Clausing, D., *The House of Quality*, Harvard Business Review May-June (1988), 63-65.

Hiremath, M., *Systems Engineering in Acquisition Strategy: Change Needed,*, INCOSE Insight 11 (2008), no. 5, 32-33.

Hitchins, D. K., *Putting Systems to Work*, John Wiley & Sons, Chichester, England, 1992.

---, *Systems Engineering…In Search of the Elusive Optimum*, proceedings of Fourth Annual Symposium of the INCOSE-UK, 1998.

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

---, *Advanced Systems Thinking, Engineering and Management*, Artech House, 2003.

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

---, Basic Models for System Thinking, 2013, http://www.hitchins.net/systems/systems-models/, accessed on 20 September 2013.

Holt, J., *UML for Systems Engineering: watching the wheels*, The Institute of Electrical Engineers, 2001.

Honour, E. C., "Systems engineering return on investment," *Defence and Systems Institute, School of Electrical and Information Engineering*, vol. PhD, University of South Australia, Adelaide, 2013, p. 239.

---, "Clarifications of ROI," 2015.

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.

Hooks, I., *Writing Good Requirements*, proceedings of Proceedings of the 3rd NCOSE International Symposium, 1993.

Howard, R., "The SCADC project," J. E. Kasser (Editor), Adelaide, Australia, 2001.

Hudson, S., Systems Engineering Competencies Framework, INCOSE UK, 2006.

Hughes, T. P., *Rescuing Prometheus*, Random House Inc., New York, 1998.

Hull, M. E. C., Jackson, K. and Dick, A. J. J., *Requirements Engineering*, Springer, 2002.

IEEE 610, "IEEE Standard Glossary of Software Engineering Terminology," IEEE, 1990.

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

IEEE 1320, "IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), IEEE Std 1320.2-1998," New york, NY, 1998.

INCOSE-CAWG, "Systems Engineering Capability Assessment Model,

CAWG-1996-01-1.50," *International Council On Systems Engineering Annual Symposium*, Seattle WA, 1996.
INCOSE, What is systems engineering?, 2000, , (http://www.incose.org/whatis.html, accessed on 9th November 2000.
---, Systems Engineering Vision 2020, INCOSE-TP-2004-004-02 (Version 2.03), INCOSE, 2007.
---, Certification Examination, 2008, http://www.incose.org/educationcareers/certification/examination.aspx, accessed on 19 February 2012.
INCOSE UK, Systems Engineering Competencies Framework, INCOSE UK, 2010.
Jackson, M. C., *Systems thinking: Creative Holism for Managers*, John Wiley & Sons Inc., Chicheste, UK, 2003.
Jackson, M. C. and Keys, P., *Towards a System of Systems Methodologies*, Journal of the Operations Research Society 35 (1984), no. 6, 473-486.
Jacobs, S., *Introducing Measurable Quality Requirements: A Case Study*, proceedings of the IEEE International Symposium on Requirements Engineering, Limerick, Ireland, 1999.
Jansma, P. A. T. and Jones, R. M., *Advancing the Practice of Systems Engineering at JPL*, proceedings of IEEE Aerospace Conference, 2006.
Jarke, M., *Meta Models for Requirements Engineering*, proceedings of Tenth Knowledge Acquisition for Knowledge-Based Systems Workshop, Banf, Alberta, Canada, 1996.
JAXA, Basics of Systems Engineering (draft) , Version 1B, Japan Aerospace Exploration Agency, 2007.
Jenkins, G. M., "The Systems Approach," *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.
Johnson, E. A., "The Executive, the Organisation and Operations Research," *Operations Research for Management, Volume 1.*, J. F. McCloskey and F. N. Trefethen (Editors), The Johns Hopkins Press, Baltimore, 1954.
Johnson, S. B., *Three Approaches to Big Technology: Operations Research, Systems Engineering, and Project Management*, Technology and Culture (1997), 891-919.
Jonassen, D. H., *Instructional design model for well-structured and ill-structured problem-solving learning outcomes*, Educational Technology: Research and Development 45 (1997), no. 1, 65-95.
Juran, J. M., *Juran on Planning for Quality*, The Free Press, 1988.

Kasser, J. E., *A Distributed Control System for a Solar Collector Field*, proceedings of Energy 84, Tel Aviv, Israel, 1984.

---, *Applying Total Quality Management to Systems Engineering*, Artech House, Boston, 1995.

---, *Systems Engineering: Myth or Reality*, proceedings of The 6th International Symposium of the INCOSE, Boston, MA., 1996.

---, *Using Organizational Engineering to Build Defect Free Systems, On Schedule and Within Budget*, proceedings of PICMET, Portland OR., 1999.

---, *A Framework for Requirements Engineering in a Digital Integrated Environment (FREDIE)*, proceedings of the Systems Engineering, Test and Evaluation Conference (SETE), Brisbane, Australia, 2000.

---, *Writing Requirements for Flexible Systems*, proceedings of INCOSE-UK Spring Symposium, 2001.

---, "The acquisition of a System of Systems is just a simple multi-phased parallel-processing paradigm," *The International Engineering Management Conference*, Cambridge, UK, 2002a.

---, "The Cataract Methodology for Systems and Software Acquisition," *SETE 2002*, Sydney Australia, 2002b.

---, *Configuration Management: The silver bullet for cost and schedule control*, proceedings of The IEEE International Engineering Management Conference (IEMC 2002), Cambridge, UK, 2002c.

---, *A Prototype Tool for Improving the Wording of Requirements*, proceedings of the 12th International Symposium of the INCOSE, Las Vegas, NV, 2002d.

---, *Researching the Properties of Object-Oriented Requirements*, proceedings of the Conference on Systems Engineering Research (CSER), Los Angeles, CA, 2004.

---, *Eight deadly defects in systems engineering and how to fix them*, proceedings of the 17th International Symposium of the INCOSE, San Diego, CA, 2007.

---, "Systems Approach to Engineering Management: Study Guide," National University of Singapore, 2009.

---, *Holistic Thinking and How It Can Produce Innovative Solutions to Difficult Problems*, proceedings of the 7th bi-annual European Systems Engineering Conference (EuSEC), Stockholm, Sweden, 2010a.

---, *Seven systems engineering myths and the corresponding realities*, proceedings of the Systems Engineering Test and Evaluation Conference, Adelaide, Australia, 2010b.

---, "Complex solutions for complex problems," *Third International Engineering Systems Symposium (CESUN)*, Delft, Holland, 2012a.

---, "Getting the Right Requirements Right," *the 22nd Annual International Symposium of the International Council on Systems Engineering*, Rome, Italy, 2012b.

---, "The Engaporean Air-Defence Upgrade: a framework for a case study development project," *the 23rd Annual International Symposium of the International Council on Systems Engineering*, Philadelphia, PA., 2013a.

---, *A Framework for Understanding Systems Engineering (2nd Edition)*, CreateSpace Ltd., 2013b.

---, *Holistic Thinking: creating innovative solutions to complex problems*, Createspace Ltd., 2013c.

---, "Guidelines for creating systems," *25th Annual Conference of the International Council on Systems Engineering*, Seattle, WA, 2015.

Kasser, J. E. and Arnold, E., "Are postgraduate courses in systems engineering teaching the right things?," *the 24th International Symposium of the International Council on Systems Engineering (INCOSE)*, Las Vegas, 2014.

Kasser, J. E. and Cook, S. C., *A Program of Research into Systems Engineering*, proceedings of the 14th International Symposium of the INCOSE, Toulouse, France, 2004.

Kasser, J. E., Cook, S. C. and Cropley, D., A Framework for Postgraduate Development of DSTO Professional Officers, University of South Australia, 2001.

Kasser, J. E., Cook, S. C., Scott, W., Clothier, J. and Chen, P., *Introducing a Next Generation Computer Enhanced Systems Engineering Tool: The Operations Concept Harbinger*, proceedings of SETE 2002, Sydney Australia, 2002.

Kasser, J. E. and Hitchins, D. K., A framework for a systems engineering body of knowledge, 0.6, Report to the Fellows Committee, International Symposium of the International Council on Systems Engineering, 2009.

Kasser, J. E. and Hitchins, D. K., "Unifying systems engineering: Seven principles for systems engineered solution systems," *the 21st International Symposium of the INCOSE*, Denver, CO., 2011.

---, "Yes systems engineering, you are a discipline," *the 22nd Annual International Symposium of the International Council on Systems Engineering*, Rome, Italy, 2012.

---, *Clarifying the Relationships between Systems Engineering, Project Management, Engineering and Problem Solving*, proceedings of Asia-Pacific Council on Systems Engineering Conference (APCOSEC), Yokohama, Japan, 2013.

Kasser, J. E., Hitchins, D. K., Frank, M. and Zhao, Y. Y., *A framework for benchmarking competency assessment models*, The Journal of the International Council on Systems Engineering (INCOSE) 16 (2013), no. 1.

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

Kasser, J. E., John, P., Tipping, K. and Yeoh, L. W., *Systems engineering a 21st century introductory course on systems engineering: the Seraswati Project*, proceedings of the 2nd Asia Pacific Conference on Systems Engineering, Yokohama, Japan, 2008.

Kasser, J. E., Kaffle, S. and Saha, P., *Applying FRAT to improve systems engineering courseware: Project Review*, proceedings of SEEC Research Group, SESA-South Australia and INCOSE-Australia joint meeting, Adelaide, 2007.

Kasser, J. E. and Mackley, T., *Applying systems thinking and aligning it to systems engineering*, proceedings of the 18th INCOSE International Symposium, Utrecht, Holland, 2008.

Kasser, J. E. and Massie, A., *A Framework for a Systems Engineering Body of Knowledge*, proceedings of 11th International Symposium of the INCOSE, INCOSE, Melbourne, Australia, 2001.

Kasser, J. E. and Mirchandani, C. J., *The MSOCC Data Switch Replacement: A Case Study in Elicitating and Elucidating Requirements*, proceedings of the 15th International Symposium of the International Council on Systems Engineering (INCOSE), Rochester, NY, 2005.

Kasser, J. E. and Palmer, K., *Reducing and Managing Complexity by Changing the Boundaries of the System*, proceedings of the Conference on Systems Engineering Research, Hoboken NJ, 2005.

Kasser, J. E., Scott, W., Tran, X.-L. and Nesterov, S., *A Proposed Research Programme for Determining a Metric for a Good Requirement*, proceedings of Conference on Systems Engineering Research, Los Angeles, CA., 2006.

Kasser, J. E. and Sen, S., "The United States Airborne Laser Test Bed program: a case study," *the 2013 Systems Engineering and Test and Evaluation Conference (SETE 2013)*, Canberra, Australia, 2013.

Kasser, J. E. and Shoshany, S., *Systems Engineers are from Mars, Software Engineers are from Venus*, proceedings of the Thirteenth International Conference on Software & Systems Engineering & their Applications, Paris, 2000.

Kasser, J. E., Sitnikova, E., Tran, X.-L. and Yates, G., *Optimising the Content and Delivery of Postgraduate Education in Engineering Management for Government and Industry*, proceedings of the International Engineering Management Conference (IEMC), St. John's, Newfoundland, Canada, 2005.

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

Kasser, J. E. and Williams, V. R., *What Do You Mean You Can't Tell Me If My Project Is in Trouble?*, proceedings of First European Conference on

Software Metrics (FESMA 98), Antwerp, Belgium, 1998.

Kasser, J. E. and Zhao, Y.-Y., *Managing Complexity via the Nine Systems in Systems Engineering*, proceedings of the 24th International Symposium of the International Council on Systems Engineering (INCOSE), Las Vegas, NV, 2014.

Kasser, J. E., Zhao, Y.-Y. and Mirchandani, C. J., *Simplifying Managing Stakeholder Expectations using the Nine-System Model and the Holistic Thinking Perspectives*, proceedings of the 24th International Symposium of the International Council on Systems Engineering (INCOSE), Las Vegas, NV, 2014.

Katz, D. R., Sarkani, S., Mazzuchi, T. and Conrow, E. H., "The Relationship of Technology and Design Maturity to DoD Weapon System Cost Change and Schedule Change During Engineering and Manufacturing Development," *Systems Engineering*, vol. 0, Wiley Periodicals, Inc., 2014.

Kemp, D. and Elphick, J., "The One-eyed Systems Engineer – Pirate or Prophet? An analysis of the effectiveness of different systems engineering leadership styles," *the 22nd Annual International Symposium of the International Council on Systems Engineering*, Rome, Italy, 2012.

Kezsbom, D. S., Schilling, D. L. and Edward, K. A., *Dynamic Project Management. A practical guide for managers and engineers*, John Wiley & Sons, New York, 1989.

Khisty, C. J. and Mohammadi, J., *Fundamentals of Systems Engineering with economics, probability and statistics*, Prentice-Hall Inc., Upper Saddle River, NJ, 2001.

Kipling, J. R., "The Elephant's Child," *The Just So Stories*, The Country Life Press, Garden City, NY, 1912.

Kline, S. J., *Conceptual Foundations for Multidisciplinary Thinking*, Stanford University Press, Stanford, 1995.

Koestler, A., *JANUS A Summing Up*, Random House, New York, 1978.

Kopp, C., High Energy Laser Directed Energy Weapons, APA-TR-2008-0501 updated April 2012, Air Power Australia, 2012.

Kossiakoff, A. and Sweet, W. N., *Systems Engineering: Principles and practice*, John Wiley & Sons Inc., 2003.

Kossmann, M., Odeh, M., Gillies, A. and Ingamells, C., *'Tour d'horizon' in Requirements Engineering - Areas left for exploration*, proceedings of the 17th International Symposium of the INCOSE, San Diego, CA, 2007.

Kotonya, G. and Summerville, I., *Requirements Engineering processes and techniques*, John Wiley & Sons, Chichester, 2000.

Kujawski, E., "The trouble with the System Readiness Level (SRL) index for managing the acquisition of defense systems," *13th Annual Systems Engineering Conference*, National Defense Industrial Association

(NDIA), San Diego, 2010.

Kursawa, A., "Shichinin No Samurai," 1954.

Lagakos, V., Kaisar, E. I. and Austin, M., *Object Modeling for the Management of Narrow Passageways in Transportation Systems*, proceedings of The 11th Annual Symposium of the INCOSE, Melbourne Australia, 2001.

Lake, J. G., *Axioms for Systems Engineering*, Systems Engineering. The Journal of the National Council on Systems Engineering 1 (1994), no. 1, 17-28.

Lano, R., The N² Chart, 1977.

LaPlue, L., Garcia, R. and Rhodes, R., *A Formal Method for Rigorous Requirements Definition*, proceedings of Fifth Annual Symposium of the NCOSE, St. Louis, MO, 1995.

Lasfer, K. and Pyster, A., *Growth of Systems-Centric Systems Engineering Graduate Programs in the United States and the Role of Their Non-Tenure-Track Faculty*, Systems Engineering 16 (2013), no. 2, 125-133.

Leveson, N., *The Role of Software in Spacecraft Accidents*, AIAA Journal of Spacecraft and Rockets 41 (2004), no. 4.

Lewis, R. O., *Independent Verification & Validation*, John Wiley & Sons, 1992.

Long, D., "Faster, Better, Cheaper – The Fallacy of MBSE?," *the 2013 Systems Engineering and Test and Evaluation Conference (SETE 2013)*, Canberra, Australia, 2013.

Long, D. and Scott, Z., *A Primer for Model-Based Systems Engineering*, Vitech Corporation, 2011.

Lui, P. C., *An Example of Large Scale Systems Engineering*, proceedings of Keynote presentation at the 1st Asia-Pacific Systems Engineering Conference, Singapore, 2007.

Luzatto, M. C., *The Way of God*, Feldheim Publishers, 1999, New York and Jerusalem, Israel, circa 1735.

M'Pherson, P., *Systems engineering: A proposed definition*, IEE Proc 133 (1986), no. September.

Machol, R. E. and Miles Jnr, R. F., "The Engineering of Large Scale Systems," *Systems Concepts*, R. F. Miles Jnr (Editor), John Wiley & Son, Inc., 1973, pp. 33-50.

Mackley, T., "The Role of Lifecycle Systems in the Through-life Engineering of System Solutions," *the 18th INCOSE International Symposium*, Utrecht, Holland, 2008.

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

Maimonides, M., *The Guide for the Perplexed translated by M Friedlander*, Dover Publications, circa 1200.

Mankins, J. C., "Technology Readiness Levels," Advanced Concepts Office, Office of Space Access and Technology, NASA, 1995.

---, *Approaches to Strategic Reseach and Technology (R&T) Analysis and Road Mapping*, Acta Astronautica 51 (2002), no. 1-9, 3-21.

Mar, B. W., *Systems Engineering Basics*, Systems Engineering: The Journal of INCOSE 1 (1994), no. 1, 7-15.

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

---, *The Seven Samurai of Systems Engineering: Dealing with the Complexity of 7 Interrelated Systems*, proceedings of 14th Annual Symposium of the International Council on Systems Engineering, Toulouse, France, 2004.

Martin, N. G., *Work Practice in Research: A Case Study*, proceedings of the 14th International Symposium of the INCOSE, Toulouse, France, 2005.

Maslow, A. H., *The Psychology of Science*, Harper and Row, 1966.

Mason, G. A., Sherwood, W. B., McGrath, W. E., Silva, F. G., Rutter, C. K. and Spence, J. P., *Application of System Engineering Principles in the Development of the Advanced Photo System*, proceedings of the 9th INCOSE International Symposium, Brighton, England, 1999.

Matchett, E. and Briggs, A. H., "Practical Design Based on Method," *The Design Method*, S. Gregory (Editor), Butterworths, London, 1966.

Mauffette-Leenders, L. A., Erskine, J. A. and Leenders, M. R., *Learning with Cases*, Ivey Publishing, 2007.

MBSE, Model-Based Systems Engineering FAQ, The Model-Based Systems Engineering Forum, 2011, http://www.modelbasedsystemsengineering.com/mbse-faq/, accessed on 18 July 2013.

McConnell, G. R., *Emergence : A Partial History of Systems Thinking*, proceedings of the 12th international symposiunm of the INCOSE, Las Vegas, NV., 2002.

McNamara, C., Basics of Developing Case Studies, 1999, http://www.managementhelp.org/evaluatn/casestdy.htm, accessed on 12 October 2005.

Memory Jogger, "Memory Jogger," GOAL/QPC, Mathuen, MA, 1985.

Menzes, T., Easterbrook, S., Nuseibeh, B. and Waugh, S., *An empirical investigation of multiple viewpoints reasoning in requirements engineering*, proceedings of IEEE International Symposium on Requirements Engineering, Limerick, Ireland, 1999.

Metzger, L. S. and Bender, L. R., *MITRE Systems Engineering (SE) Competency Model Version 1.13E*, The MITRE Corporation, 2007.

Meystel, A., Albus, J., Messina, E. and Leedom, D., "Performance Measures for Intelligent Systems: Measures of Technology Readiness," *PERMIS '03 White Paper*, 2003.

MIL-HDBK-61A, *Military Handbook: Configuration Management Guidance*,

Department of Defense, 2001.
MIL-STD-490A, *Specification Practices*, United States Department of Defense, 1985.
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, *Mil-STD-499B Systems Engineering*, United States Department of Defense, 1992.
---, Draft MIL-STD-499B Systems Engineering, United States Department of Defense, 1993.
MIL-STD-2167A, *Defense System Software*, United States Department of Defense, 1998.
Miller, G., *The Magical Number Seven, Plus or Minus Two: Some Limits on Our Capacity for Processing Information.*, The Psychological Review 63 (1956), 81-97.
Moorehouse, D. J., *Detailed Definitions and Guidance for Application of Technology Readiness Levels*, Journal of Aircraft 39 (2001), no. 1, 190-192.
Mooz, H., Pyster, A., Gemert, D. V. and Wartenberg, M., "Integrating Systems Engineering with Program and Project Management," *17th Annual International Symposium of the International Council on Systems Engineering*, San Diego, CA, 2007.
Morse, P. M. and Kimball, G. E., *Methods of Operations Research*, The Technology Press of Massachusetts Institute of Technology and John Wiley & Sons, New York, 1951.
NASA, *NASA Systems Engineering Handbook, draft, September 1992*, 1992.
---, NASA's systems engineering competencies, 2010, http://www.nasa.gov/pdf/303747main_Systems_Engineering_Competencies.pdf, accessed on 29 August 2011.
NDIA E&T, December 8, 2010 Meeting: Education & Training Committee Report, NDIA, 2010.
Needham, J., *Integrative levels: a revaluation of the idea of progress*, Clarendon Press, Oxford, 1937.
Nolte, W. L., "TRL Calculator," *AFRL at Assessing Technology Readiness and Development Seminar*, 2005.
O'Connor, J. and McDermott, I., *The Art of Systems Thinking*, Thorsons, San Francisco, 1997.
O'Toole, P., *Do's and Don'ts of Process Improvement*, proceedings of 2005 U.S. Census Bureau SEPG Conference, Suitland. MD, 2004.
OPM, Policies and instructions, U.S. Office of Personnel Management, 2009, http://www.opm.gov/qualifications/policy/index.asp, accessed on 29 October 2009.

OVAE, Problem-Solving Process, Office of Vocational and Adult Education (OVAE), US Department of Education, 2005, http://www.c-pal.net/course/module3/pdf/Week3_Lesson21.pdf, accessed on 11 Jan 2009.

Palmer, S. R. and Felsing, J. M., *A Practical Guide to Feature - Driven Development*, Prentice Hall, 2002.

Paul, A. S. and Owunwanne, C., "The Systems Project: Life Cycle Development/Management of as Many as Four Interrelated Systems," *16th Annual Symposium of the International Council on Systems Engineering*, Orlando, FL., 2006.

Paul, R. and Elder, L., *Critical Thinking: learn the tools the best thinkers use - concise ed.*, Pearson Prentice Hall, 2006.

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.

Powell, R. A. and Buede, D., *Innovative Systems Engineering: A Creative System Development Approach*, proceedings of the 16th International Symposium of the International Council on Systems Engineering (INCOSE), Orlando, FL., 2006.

Prieto-Díaz, R. (Editor), *Classification of Reusable Models published in Software Reusability, 1987*, ACM Press, 1987.

Project Smart, Stakeholder Management managing expectations, 2013, http://www.projectsmart.co.uk/stakeholder-management.html, accessed on 31 October 2013.

Recklies, D., Stakeholder Management, 2001, http://www.themanager.org/Resources/Stakeholder%20Management.htm, accessed on 31 October 2013.

Rhodes, D. H., *Systems Engineering on the Dark Side of the Moon*, proceedings of the 12th International Symposium of the INCOSE, Las Vegas, NV., 2002.

Richmond, B., *Systems thinking: critical thinking skills for the 1990s and beyond*, System Dynamics Review 9 (1993), no. 2, 113-133.

Rittel, H. W. and Webber, M. M., *Dilemmas in a General Theory of Planning*, Policy Sciences 4 (1973), 155-169.

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

Roe, C. L., *The Role of the Project Manager in Systems Engineering*, proceedings of The 5th Annual International Symposium of the NCOSE, 1995.

Rook, P., *Controlling software projects*, Software Engineering Journal 1

(1986), no. 1, 7-16.

Roy, R. H., "The Development and Future of Operations Research and Systems Engineering," *Operations Research and Systems Engineering*, C. D. Flagle, W. H. Huggins and R. H. Roy (Editors), Johns Hopkins Press, Baltimore, 1960.

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

Ruggiero, V. V., *The Art of Thinking. A guide to critical and creative thought*, Pearson, Education, Inc. , 2012.

Russo, J. E. and Schoemaker, P. H., *Decision Traps*, Simon and Schuster, New York, 1989.

Saaty, T., *Decision Making for Leaders*, RWS Publications, Pittsburgh, PA, 1990.

Saaty, T. L. and Alexander, J. M., *Thinking with Models*, Pergamon Press, Oxford, UK, 1981.

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

Sauser, B., Verma, D., Ramirez-Marquez, J. and Gove, R., *From TRL to SRL: The Concept of Systems Readiness Levels*, proceedings of Conference on Systems Engineering Research, Los Angeles, CA., 2006.

Savage, S. L., *The flaw of averages*, John Wiley and Sons, Inc., 2009.

Schach, S., *Object-Oriented and Classical Software Engineering*, McGraw Hill, 2002.

Schön, D. A., *The Reflective Practitioner*, Ashgate, 1991.

Scott, W., Kasser, J. E. and Tran, X.-L., *Improving the Structure and Content of the Requirement Statement*, proceedings of The 16th International Symposium of the International Council on Systems Engineering (INCOSE), Orlando, FL., 2006.

Selby, R. W., *Enabling Measurement-Driven System Development by Analyzing Testing Strategy Tradeoffs*, proceedings of The 16th International Symposium of the INCOSE, Orlando, FL., 2006.

Senge, P. M., *The Fifth Discipline: The Art & Practice of the Learning Organization*, Doubleday, New York, 1990.

Serrat, O., "The Five Whys Technique," *Knowledge Solutions*, Asian Development Bank, 2009.

Sheard, S. A., *Twelve Systems Engineering Roles*, proceedings of The 6th Annual International Symposium of the NCOSE, Boston, MA., 1996.

Shenhar, A. J. and Bonen, Z., *The New Taxonomy of Systems: Toward an Adaptive Systems Engineering Framework*, IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans 27 (1997), no. 2, 137 - 145.

Sherwood, D., *Seeing the Forest for the Trees. A Manager's Guide to Applying Systems Thinking*, Nicholas Brealey Publishing, London, 2002.

Shishkio, R., Ebbeler, D. H. and Fox, G., *NASA Technology Assessment Using Real Options Valuation.*, Systems Engineering 7 (2003), no. 1, 1-12.

Shoshani, S., Effective use of MBSE, Israel Institute of Technology, 2010.

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

---, "On Systems Architects and Systems Architecting: some thoughts on explaining and improving the art and science of systems architecting," *the 19th International Symposium of the International Council on Systems Engineering*, Singapore, 2009.

Smith, J. D., "An Alternative to Technology Readiness Levels for Non-Developmental Item (Ndi) Software," *38th Hawaii International Conference on System Sciences*, Hawaii, 2005.

Sommerville, I., *Software Engineering*, Addison-Wesley, 1998.

Spencer, H., "First principles," *A System of Synthetic Philosophy*, Williams and Norgate (Editor), vol. 1, London, 1862.

Squires, A., "Qualifying Exam: Systems Thinking and K12 Education," Stevens Institute of Technology, Hoboken, NJ., 2007.

Squires, A., Wade, J., Dominick, P. and Gelosh, D., *Building a competency taxonomy to guide experience acceleration of lead program systems engineers*, proceedings of the 9th conference on systems engineering research (CSER 2011), Los Angeles, CA, 2011.

Suh, N. P., *Complexity in Engineering.*, Annals of CIRP 54 (2005), no. 2, 46-63.

Sutcliffe, A., Galliers, J. and Minocha, S., *Human Errors and System Requirements*, proceedings of IEEE International Symposium on Requirements Engineering, Limerick, Ireland, 1999.

Swarz, R. S. and DeRosa, J. K., *A Framework for Enterprise Systems Engineering Processes*, proceedings of International Conference on Computer Science, Engineering and Applications (ICSSEA), 2006.

Thomas, J. A., *Energy-Environment-Economic System of Systems*, proceedings of 5th Asia-Pacific Conference on Systems Engineering, Seoul, Korea, 2011.

Tittle, P., *Critical Thinking: An appeal to reason*, Routledge, 2011.

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

Tomiyama, T., D'Amelio, V., Urbanic, J. and ElMaraghy, W., *Complexity of Multi-Disciplinary Design*, Annals of the CIRP 56 (2007), no. 1, 185-188.

Tran, X.-L., Huynh, T. V., Ferris, T. and Shoval, S., *Research on a Framework for Systems Engineering Tools for Australian Small and Medium*

Enterprises, proceedings of the 18th INCOSE International Symposium, Utrecht, Holland, 2008.

UIA, Integrative knowledge project: levels of organization, Union of International Associations, 2002, http://www.uia.org/uialists/kon/c0841.htm, accessed on 28th May 2002.

UML, OMG Unified Modeling Language Specification Version 1.3, OMG, 1999.

---, Unified Modeling Language: Superstructure, OMG, 2005.

Valerdi, R. and Kohl, R. J., "An Approach to Technology Risk Management," *Engineering Systems Division Symposium*, Cambridge, MA., 2004.

Van Gaasbeek, J. R. and Martin, J. N., *Getting to Requirements: The W5H Challenge*, proceedings of The 11th Annual Symposium of the INCOSE, Melbourne Australia, 2001.

Ward, P. T. and Mellor, S. J., *Structured Development for Real-Time Systems*, Yourdon Press, 1985.

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

Watts, J. G. and Mar, B. W., *Important Skills and Knowledge to Include in Corporate Systems Engineering Training Programs*, proceedings of The 7th Annual International Symposium of the INCOSE, Los Angeles, CA, 1997.

Webster, Merriam-Webster Online Dictionary, 2004, http://www.webster.com, accessed on 12 January 2004.

---, Merriam-Webster Online Dictionary, 2013, http://www.merriam-webster.com/dictionary/operations%20research, accessed on 6 August 2013.

Weiss, S. I., *Product and Systems Development*, Wiley, Hoboken, NJ., 2013.

Welby, S., *DoD Systems Engineering Update*, proceedings of 36th Air Armament Symposium, 2010.

Westerman, H. R., *Systems Engineering Principles and Practice*, Artech House, Boston, 2001.

Williams, T. J., *Systems Engineering for the Process Industries*, McGraw-Hill, 1961.

Wilson, T. D., *Philosophical foundations and research relevance: issues for information research (Keynote address)*, proceedings of Fourth International Conference on Conceptions of Library and Information Science: Emerging Frameworks and Method, University of Washington, Seattle, USA, 2002.

Wilson, W. E., *Concepts of Engineering System Design*, McGraw-Hill Book Company, 1965.

Wirsing, G. W., "The Airborne Laser & the Future of Theater Missile

Defense," *The Research Department*, Air Command and Staff College, 1997.

Wolstenholme, E. F., *System Enquiry a system dynamics approach*, John Wiley & Sons, 1990.

Wood, J. and Silver, D., *Joint Application Development*, John Wiley and Sons Inc., 1995.

Woolfolk, A. E., "Chapter 7 Cognitive views of learning," *Educational Psychology*, Allyn and Bacon, Boston, 1998, pp. 244-283.

Wu, W. and Yoshikawa, H., *Study on Developing a Computerized Model of Human Cognitive Behaviors in Monitoring and Diagnosing Plant Transients*, proceedings of 1998 IEEE International Conference on Systems, Man, and Cybernetics, 1998.

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

---, *Model-Based Systems Engineering*, Systems Engineering: The Journal of INCOSE 1 (1994), no. 1, 83-92.

---, Subsystem Optimization Implies System Suboptimization: Not!, 1997, http://www.sie.arizona.edu/sysengr/wymore/optimal.html, accessed on 30 September 2013.

Wynne, M. W. and Schaeffer, M. D., *Revitalization of Systems Engineering in DoD*, Defense AT&L (2005), 14-17.

Yen, D. H., The Blind Men and the Elephant, 2008, http://www.noogenesis.com/pineapple/blind_men_elephant.html, accessed on 26 October 2010.

Yin, R. K., *CASE STUDY RESEARCH Design and Methods*, vol. 5, SAGE Publications, Newbury Park, CA, 1989.

Yin, R. K., *Case Study research design and methods*, SAGE Publications Inc., Thousand Oaks, CA., 2009.

Zonnenshain, A., "Comments on draft manuscript," J. E. Kasser (Editor), 2015.

--oo--

25. Index