The nuts and bolts of systems

Joseph Kasser
Temasek Defence Systems Institute, National University of Singapore
Block E1, #05-05, 1 Engineering Drive 2, Singapore 117576
Email tdskj@nus.edu.sg

Abstract-**This paper fills a gap in the systems engineering and project management education literature by providing examples of:**

1. The effect of desired and undesired emergent properties.
2. How the System Lifecycle (SLC) relates to the iterative problem-solving process.
3. The relationship between the "what's" and the "how's" of systems engineering.
4. How subsystem boundaries can change during system design when compensating for undesired emergent properties.
5. How the solution to one problem often creates a subsequent problem.
6. The effect of unanticipated problems on the schedule, usually in the form of the need to insert unplanned work into the schedule resulting in a delay to the project.

Keywords-emergent property; problem-solving; schedule; delay; boundaries; systems engineering; systems engineering education.

I. INTRODUCTION

This paper fills a gap in the education literature. Perceptions of systems engineering from different perspectives showed that the systems engineering literature often discusses the same concepts using different terminology and viewpoints and uses the same terminology for different concepts [1].This situation is similar to Humpty Dumpty telling Alice that, "*when he uses a word it means just what he chooses it to mean - neither more nor less"* [2] irrespective of how other people use the same word. Accordingly, after drawing the reader's attention to the situation, this paper begins by establishing a baseline for the meaning of specific terminology in Section II which defines the meanings of the words 'system', 'emergent properties', 'the System Lifecycle (SLC)' and 'milestone reviews' as used in this paper. Section III then relates the 'what's' and the 'how's' of systems engineering to the problem-solving process and the SLC. Section IV uses the hypothetical Widget system as an example to discuss:

1. Desired and undesired emergent properties.
2. How the solution to one problem gives rise to subsequent problems during the SLC and may require changing the boundaries of the subsystems after the initial architecture is approved.

Section V contains comments on the Widget case and Section VI summarises some of the lessons learned from the case.

II. DEFINITIONS

Perceptions of systems engineering from different perspectives showed that the systems engineering literature often discusses the same concepts using different terminology and viewpoints and uses the same terminology for different concepts [1]. To clarify the issues discussed in this paper, this section defines the following terms used in this paper:

1. A system
2. Emergent properties
3. The system lifecycle
4. Milestone reviews

A. Definition of a system

The example system discussed in this paper is based on the minimal definition of a system as, "*A set of different elements so connected or related as to perform a unique function not performed by the elements alone*" [3]. Accordingly, the characteristics of the system include:

1. At least two elements (subsystems).
2. Interactions between the elements.
3. Emergent properties that arise from the interactions between the elements.

B. Emergent properties

There seem to be three types of emergent properties [4]:

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

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

C. The System Lifecycle

The System Lifecycle (SLC) has been defined in many ways. This paper uses the grouping into nine different states of the SLC [5] defined in generic terms as:

A. The Needs Identification State.

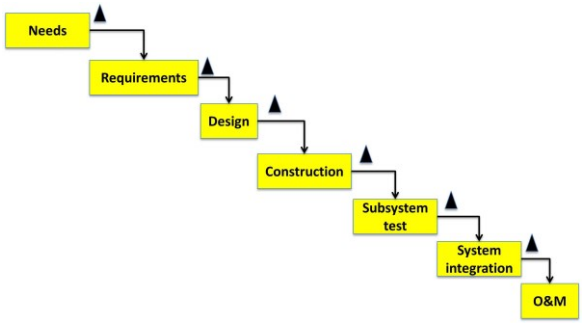

Fig. 1 The waterfall view of most of the SLC

B. The System Requirements State.

C. The System Design State which is split into:
   - The Preliminary System Design sub-state.
   - The Detailed System Design sub-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.

The output of each state in the SLC becomes the input to the subsequent state and is often shown in the waterfall [6] view of Fig. 1. 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) Any flaw other than replacing a defective item.
   b) An approved change that necessities rework.

The second type of exit is recognized and is often depicted in a chaotic view of the SLC which shows every state connected to every other state.

D. Milestone reviews

Each state in the SLC 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 are [1]:

- ***Start of project:*** formally starts the project and commencement of the Needs Identification State.
- ***Operations Concept Review (OCR):*** marks the termination of the Needs Identification State and commencement of the System Requirements State.
- ***Systems Requirements Review (SRR):*** marks the termination of the System Requirements State and commencement of the Preliminary Design sub-state of the System Design State.

	1	2	3	4	5	6	7
Needs							
Requirements							
Design							
Realization							
Integration							
Test							
O&M							

Fig. 2 Widget project: original schedule

- ***Preliminary Design Review (PDR):*** marks the termination of the Preliminary Design sub-state of the System Design State and commencement of the Detailed Design sub-state of the System Design State.
- ***Critical Design Review (CDR):*** marks the termination of the System Design State and commencement of the Subsystem Construction State.
- ***Test Readiness Review (TRR):*** marks the termination of the Subsystem Construction State and commencement of the Subsystem-Testing State.
- ***Integration Readiness Review (IRR):*** marks the termination of the Subsystem-Testing State and commencement of the System Integration State.
- ***Delivery Readiness Review (DRR):*** marks the termination of the System Integration and System Test States and commencement of the activities that deliver the system and lead to terminating the successful project.
- ***End of project:*** marks the formal termination of the project.

III. THE "WHAT'S" AND THE "HOW'S"

Perceived from the problem-solving and enabler systems engineering camps [1], each state in the SLC may be considered as starting with a problem and ending with a solution. Accordingly the solution output of any state becomes the problem input to the subsequent state. For example:

1. The matched set of specifications for the system and subsystems produced during the System Requirements State is both:
   - ***A solution*** to the problem of specifying a system that will meet the needs.
   - ***A problem*** to the designers/systems architects[1] because they now have to design a system that is compliant to the specifications.
2. The design or architecture produced at the end of the System Design State is both:
   - ***A solution*** to the problem of designing the system.
   - ***A problem*** for the Subsystem Construction State. This situation, shown in Fig. 3, is often referred to as the:

[1] In the System Design State.

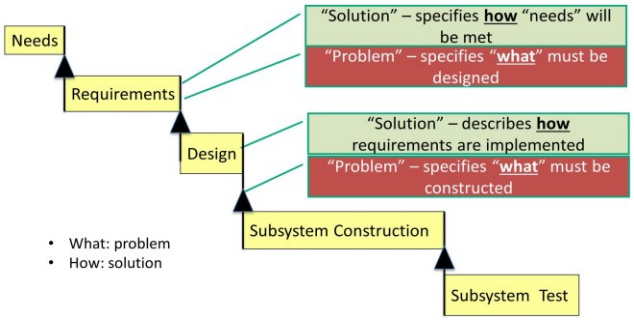

Fig. 3 The waterfall view – problem-solving perspective (partial) [1]

- ***"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.

IV. The Widget System

This section uses the Widget system as an example to discuss:

1. Desired and undesired emergent properties.
2. How the solution to one problem gives rise to a subsequent problem during the SLC and may require changing the boundaries of the subsystems after the initial architecture is approved.

Federated Aerospace is developing the Widget system according to the 'A' paradigm of systems engineering in which the SLC starts in the Needs Identification State [7]. The project is the first of its kind: there are no similar systems in existence. The Widget system comprises two subsystems, Part A and Part B. This paper abstracts out all aspects of the Widget system except for the approach to mechanically fastening the two subsystems to each other[2].

The original seven-month Widget project schedule shown in Fig. 2 was planned as a single pass through the waterfall assuming there would be no serious problems during the system development. Consider the Widget system as it passes through the sequential states of the SLC.

A. The Needs Identification State

In accordance with the 'A' paradigm[3], during the development of the Concept of Operations (CONOPS) of the system, the conceptual system architecture was defined as two subsystems, fastened together. The conceptual solution to the need to fasten the two subsystems together was to use a mechanical method.

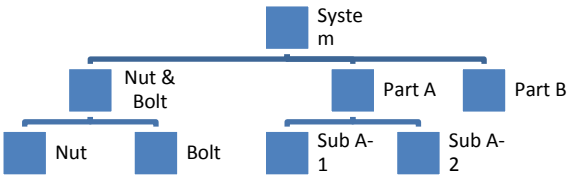

Fig. 4 Structural perspective; the three subsystems

B. The Requirements State

In accordance with the 'A' paradigm, the matching set of specifications for the Widget system and subsystems were developed from the CONOPS. A feasibility study on fastening methods identified a variety of suitable low-cost Commercial-Off-The-Shelf (COTS) fasteners at prices that were well-below the estimated costs of developing proprietary fasteners. The requirements for the fastening function were approved together with the rest of the specifications at the SRR. Note that even though the requirement limited the designer to the use of COTS, the requirements still specified the "what"; namely, "*The system shall use a COTS fastening function to fasten the two subsystems*"[4]. The "how", or the choice of which type of fastener, will be developed later during the System Design State.

C. The System Design State

When the System Design State began, the systems engineer framed the problem according to the problem formulation template [1] as follows:

- ***The undesirable situation:*** the need to fasten two subsystems together using a COTS fastener.
- ***The Feasible Conceptual Future Desirable Situation (FCFDS):*** the two subsystems fastened together using a COTS fastener.
- ***The problem:*** to decide on a specific type of COTS fastener.
- ***The solution:*** the specific type of COTS fastener to be determined (TBD) by the end of the state.

The activities in the System Design State take place in two sequential sub-states as follows.

1) *The Preliminary System Design sub-state.* During the Preliminary System Design sub-state the system engineer researched different types of COTS fastening products and identified the following: nuts and bolts, rivets, hooks and loops (Velcro®), and nails. The systems engineer presented a FCFDS using each product at the PDR together with their advantages and disadvantages.

[2] The simple function was chosen for this example to avoid getting bogged down in the details of the subsystem and subsequent real-world problems. Hence the use of months as time period is illustrative of the delays and is not intended to be realistic.

[3] In the 'A' paradigm, the preliminary conceptual system architecture is developed in the Needs Identification State and the requirements are later derived from the system architecture and CONOPS.

[4] The reason was stated as being the lowest cost option.

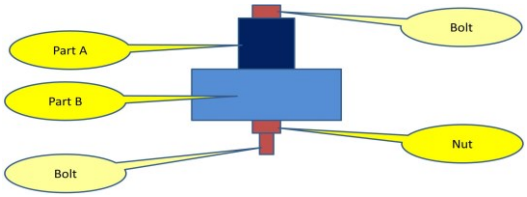

Fig. 5 The Structural perspective of the system

	1	2	3	4	5	6	7	8	9	10	11	12	13
Needs													
Requirements													
Design													
Realization													
Integration													
Test													
O&M													

Fig. 6 Widget project: revised schedule

2) *The Detailed System Design sub-state.* After some trade-off studies in the Detailed System Design sub-state, the selected solution accepted by consensus at the CDR was to use a nut and a bolt to fasten the two subsystems. The system architecture shown in Fig. 4 was split into three subsystems as:

1. The A sub system.
2. The B subsystem.
3. The Nut and Bolt subsystem.

Perceptions of a nut and bolt from the Holistic Thinking Perspectives (HTP) [8] include:

- ***The Structural perspective:*** a nut and a bolt constitute a very simple system in which the function of fastening two parts of the system together emerges from the combination of the nut and the bolt and the interaction between them. The A and B subsystems need to be modified to contain a hole sized to fit the bolt (or the bolt is sized to fit a hole). The bolt is passed through the mounting hole in the A and B subsystems; the nut is inserted into the bolt and tightened to a specified torque to fasten the components together as shown in Fig. 5.
- ***The Generic perspective:*** perceptions from the *Generic* perspective indicate that a nut and a bolt are generally used as a subsystem to fasten components together.
- ***The Continuum perspective:*** perceptions from the *Continuum* perspective predict that since the system is the first of its kind, unanticipated, unknown and unaccounted for factors at design time may emerge when the system has been constructed and have a negative or serendipitous effect on the system.
- ***The Quantitative perspective:*** perceptions from the *Quantitative* perspective indicate the numbers of nuts and bolts needed; the diameter of the bolt and the gauge of the screw thread necessary to carry the anticipated load[5]. This information becomes part of the requirement for the nuts and bolts[6] and is presented at the:
  - CDR for the Widget system.
  - SRR for the Nut and Bolt subsystem[7].

[5] The next largest standard COTS size would be used instead of creating the exact size needed.
[6] The location of the hole containing the nut and bolt and the necessary clearance on the surface of the A and B systems for the nut and bolt are also part of the specification.
[7] This is to illustrate that once the System Design State of the SDP has been completed at the CDR, each of the subsystems begin their own SDP.

D. The Realization States

The System Development Process (SDP) proceeded through the Subsystem Construction States, the Subsystem Test State and the System Integration State to the System Test State without experiencing any problems with the Nut and Bolt subsystem.

The A and B subsystems of the widget system were sufficiently complex to have their own systems engineers. It was the Widget system systems engineer's task to liaise with the systems engineers responsible for the A and B subsystems during the system realization states of the SLC to ensure that the subsystems contained the specified matching holes, and that the holes aligned as specified.

E. The System Test State

At this point in time, an undesirable property emerged giving rise to an undesirable situation; under some test conditions the system came apart[8]; namely the nut and bolt no longer had the capability to fasten the subsystems together.

An investigatory series of tests were carried out to determine under what conditions the nut and bolt came apart and it was determined that the system was fine as long as it was not subjected to vibration. However, once the system experienced a vibration >N m/s^2 the nut and bolt began to separate.

F. The second Needs Identification State

At this point, the problem impacted the development schedule and the SDP reverted to a second Needs Identification State as shown in the revised schedule in Fig. 6. The second iteration of the SDP began in Month 7 and the original O&M State was delayed to Month 13.

The systems engineer and mechanical engineer performed an analysis of the magnitude of expected vibration in each of the operational scenarios in the CONOPS. They determined that no anticipated mission was expected to produce vibration >1.5N m/s^2. This led to a new system requirement, "*the system shall NOT come apart*[9] *when experiencing continuous vibration of <1.5N m/s2 for up to 30 minutes"*[10].

G. The second System Design State

At the start of the second iteration of the System Design State the new design problem was then framed as:

[8] The test conditions had simulated the scenarios in the CONOPS where the system experienced different degrees of vibration.
[9] This is a poorly worded but understandable requirement. A well-written requirement would specify a measurable minimum value of the torque if any, still holding the nut and bolt together after 30 minutes of vibration.
[10] The 30 minute time limit came from the CONOPS.

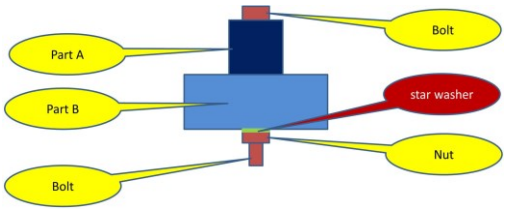

Fig. 7 Modified system with added star washer

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Needs															
Requirements															
Design															
Realization															
Integration															
Test															
O&M															

Fig. 8 Widget project: modified schedule with delays due to the unanticipated problems

- ***The undesirable situation:*** the system **comes apart** when experiencing vibration **greater than** N[11] m/s^2.
- ***The FCFDS:*** system **shall NOT come apart** when experiencing vibration **less than** 1.5N m/s^2 for up to 30 minutes.
- ***The problem:*** to create the FCFDS.
- ***The selected solution:*** TBD.

The designers examined a number of alternative ways of fixing the problem including revisiting the non-nut and bolt solutions. The selected way to compensate for effect of vibration was to add a star washer between the nut and the subsystem closest to the nut as shown in Fig. 7. The conceptual solution was prototyped, tested and shown to work and was accepted by consensus at the second iteration CDR.

H. The second iteration through the realization states

The system was constructed and tested and the solution was validated. The star washer stopped the nut and bolt from coming apart when experiencing vibration of <1.5N m/s^2 for up to 30 minutes.

I. The System Test State

The SLC then reverted to the end of the initial System Test State in Month 12 after the five-month schedule delay and cost escalation due to the unplanned activities in the additional states of the second iteration of realization states of the SLC.

An additional performance evaluation test was set up to determine the performance envelope[12] and determined that the prototype of the system as constructed:

1. Could experience vibration <1.5N m/s^2 for up to 88 minutes before it would start to come apart.
2. Could experience vibration <2N m/s^2 for up to 73 minutes before it would start to come apart.
3. Would start to come apart immediately it experienced vibration >3.14159N m/s^2.

For each nut and bolt, the system now had an extra component, the star washer. This gave rise to the next problem which was formulated as:

- ***The undesirable situation:*** the need to place the star washer in an existing subsystem or in a new subsystem.
- ***The FCFDS:*** the star washer is placed in an existing subsystem or in a new subsystem.
- ***The problem:*** create the FCFDS
- ***The solution:*** TBD.

Although the prototype had demonstrated that the star washer would meet the functional requirements, the design still needed to be validated for the non-functional and manufacturing requirements. Accordingly, at this point in time the SLC reverted back to the System Design State as shown in the revised schedule in Fig. 8 for an additional delay of two months.

J. The third System Design State

After due consideration of the alternatives, the systems engineer determined that the preferred solution was to include the star washer in the Nut and Bolt subsystem and rename the subsystem as the Fastening Subsystem. The solution was accepted at the CDR and the documentation was updated. The star washer became a part of the Fastening Subsystem as shown in Fig. 9. Since this design decision only impacted the documentation, there was no need for further subsystem realization states and the SLC returned to the System Test State in Month 14 as shown in Fig. 8.

V. Comments

The simple example in the Widget project has illustrated how:

1. The SDP was delayed by the activities in the second and third iterations as can be seen by comparing the original schedule in Fig. 2 with the actual schedule in Fig. 8. Accordingly, the Widget project's original optimistic success-oriented seven-month planned schedule turned into a 15 month project with corresponding cost escalations due to unforeseen problems in the system design.
2. First of a kind system development projects which correspond to Shenhar and Bonen's Type D projects with super high technological uncertainty [9] should use a schedule containing two or three passes through the waterfall rather than the single success-oriented approach commonly used. This concept may be generalized as 'the more complex the system, the more iterations of the SDP will be needed to realize the system'.
3. The original single pass waterfall iterated back to the Needs Identification State once the unfastening problem occurred. The systems engineering literature

[11] The value N represents the minimum amount of vibration.

[12] The components inside the subsystems were replaced by configuration controlled equivalent non-functional mass blanks for the duration of these performance tests so as not to damage the components.

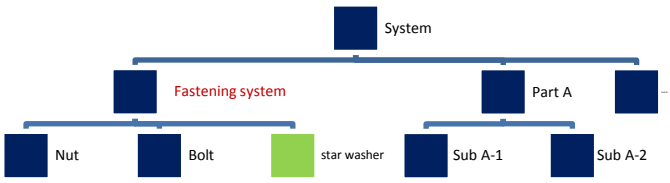

Fig. 9 Two subsystems with renamed fastening system

generally illustrates the iteration from the *Functional* perspective by drawing a line from one state to the other in the waterfall view shown in Fig. 1. This approach:

- Tends to gloss over the accompanying schedule delays since activities must be repeated and have to be inserted into the project timeline.
- Is an unfortunate side effect of treating systems engineering and project management as being independent when in fact they are interdependent.

4. The subsystem boundaries can change during the SDP. In this instance they did not, but a new component was added to the fastening subsystem.
5. Solutions gave rise to problems as the SDP progressed.
6. When an unanticipated undesirable emergent property is tackled, additional components may be included in the system to prevent or minimize the unanticipated undesirability if the unanticipated undesirable emergent property can't be prevented.
7. The System Engineering Management Plan (SEMP) should contain some slack time at the end of the System Test State after the tests have been completed and before the milestone review to allow for defects to be dealt with. Simple defects may be fixed at that time and not require iteration back to an earlier state of the SLC. If no defects show up, and there are no tasks to complete, then the milestone at the end of the state can be moved forward in time and the project becomes ahead of schedule.
8. The degree of iteration in the SDP should a problem arise depends on the nature of the problem.

The nut and bolt example problem replaced a complex problem for educational purposes to focus on the effect of the issues associated with a problem. In the real world, a problem this simple would not cause long schedule delays and would not require the iteration back to the earlier states of the SDP.

VI. Lessons Learned

Lessons learned included:

- System and subsystem boundaries may change during the SDP.
- Initially unknown emergent properties become known through experience.
- Once known, undesirable emergent properties are usually compensated for by additional functions in a component that may not seem to contribute to the mission of the system.
- Do not remove any function/component without planning some serious testing if you are not sure what purpose he component serves.
- The more complex the system, the more iterations of the SDP will be needed to realize the system [10].

VII. Summary

This paper fills a gap in the systems engineering and project management education literature by providing examples of:

1. The effect of desired and undesired emergent properties.
2. How the SLC relates to the iterative problem-solving process.
3. The relationship between the "what's" and the "how's" of systems engineering.
4. How subsystem boundaries can change during system design when compensating for undesired emergent properties.
5. How the solution to one problem often creates a subsequent problem.
6. The effect of unanticipated problems on the schedule, usually in the form of the need to insert unplanned work into the schedule resulting in a delay to the project.
7. Examples of the use of the Problem Formulation Template [1].
8. Some of the lessons learned from the Widget case.

REFERENCES

[1] J. E. Kasser, *Perceptions of Systems Engineering*: Createspace, 2015.
[2] L. Carroll, Through the Looking Glass, 1872.
[3] E. Rechtin, Systems Architecting, Creating & Building Complex Systems, Englewood Cliffs, NJ: Prentice-Hall, 1991.
[4] J. E. Kasser, and K. Palmer, "Reducing and Managing Complexity by Changing the Boundaries of the System," proceedings of the the Conference on Systems Engineering Research, Hoboken NJ, 2005.
[5] J. E. Kasser, "The Hitchins-Kasser-Massie (HKM) Framework for Systems Engineering," proceedings of the the 17th International Symposium of the INCOSE, San Diego, CA., 2007.
[6] W. W. Royce, "Managing the Development of Large Software Systems," proceedings of the IEEE WESCON, 1970.
[7] J. E. Kasser, "Getting the Right Requirements Right," proceedings of the the 22nd Annual International Symposium of the International Council on Systems Engineering, Rome, Italy, 2012.
[8] J. E. Kasser, Holistic Thinking: creating innovative solutions to complex problems, 2 ed.: Createspace Ltd., 2015.
[9] A. J. Shenhar, and Z. Bonen, "The New Taxonomy of Systems: Toward an Adaptive Systems Engineering Framework," IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans, vol. 27, no. 2, pp. 137 - 145, March 1997, 1997.
[10] J. E. Kasser, and Y.-Y. Zhao, "Simplifying Solving Complex Problems," proceedings of the the 11th International Conference on System of Systems Engineering, 2016.