REDUCING ACQUISITION COST BY MINIMIZING THE REQUIREMENTS SOLUTION SPACE

Bradford Logan
The George Washington University, Email: bloga001@hotmail.com

ABSTRACT

The purpose of this paper is to explore the need to develop a methodology for reducing acquisition costs by minimizing the requirements' solution space. The requirements solution space is the result of two or more solutions, resulting from the interpretation of system's requirement. This can be problematic if solution spaces created by the requirements do not: 1) fall within the systems' prescribed solution space, 2) or if the solution spaces from other neighbouring system requirements oppose one another. If either of these situations is realized, there will likely be negative consequences relating to costs, schedule and quality.

The author briefly explores critical requirements and the relations to their respective constraints. In doing so, the concept of assigning a degree of freedom (DoF) nomenclature to these critical requirements is introduced, which helps relates the DoF to complexity. This nomenclature will prove instrumental in providing the initial assessment to the number of potential solutions existing per system requirement. The paper will also establish a relation between the requirements DoF, complexity and cost.

Keywords: Requirements, Degree of Freedom, Complex Systems, System Boundary, Feasible Solution Space

INTRODUCTION

A need exist for developing a new methodology for generating, verifying and validating requirements. Numerous works have been published by academics and practitioners over the decades, yet the requirement process can be challenging. Katrina, et al (2014) states three major challenges currently exist when generating requirements for complex systems due to "… (1) role of the system observer, (2) nature of system requirements in complex situations, and (3) influence of the system environment. Authors have asserted that the expectation of unambiguous, consistent, complete, understandable, verifiable, traceable, and modifiable requirements is not consistent with complex situations. In contrast, complex situations are an emerging design reality for requirements engineering processes, marked by high levels of ambiguity, uncertainty, and emergence." Furthermore Katrina et al, (2014) states, "…dealing with requirements for complex situations requires a change in paradigm. The elicitation of requirements for simple and technically driven systems is appropriately accomplished by proven methods. In contrast, the elicitation of requirements in complex situations (e.g., integrated multiple critical infrastructures, system-of-systems, etc.) requires more holistic thinking and can be enhanced by grounding in systems theory."

As essential as requirements are to the system engineering process, failures due to requirements continue occur. According to Byun et. al (2014) "… many projects fail with the improper results of Requirement Engineering (RE). One of the results is an inconsistency of requirements. Causing unnecessary efforts, it can interfere with the good performance of RE."

The concept of writing requirements undoubtedly can be traced at least as far back as the Egyptians. Yet in many instances, despite the advances we've made, poor requirement leading to failure in systems still occur (Bayhill and Henderson, 2005, Byun et. al, 2014).

Mobasher and Cleland-Huang, (2011) states, "The importance of the requirements engineering task is illustrated by several studies, which have shown that requirements-related issues, such as poorly specified requirements (Leffingwell 1997) and incomplete and changing requirements and specifications, are the root cause of many failed projects. To address these problems, researchers and practitioners have developed processes for identifying relevant stakeholders; discovering their needs, wants, and desires for the system; prioritizing and negotiating requirements; and specifying them in understandable, measurable, and testable ways (Robertson and Robertson 1999)."

The objective of this paper is to view requirements from a unique perspective, in an attempt to produce a new approach to the requirements process. As stated earlier, requirement generation is an essential part of the system engineering process. Neglecting to place proper emphasis on requirements can result in higher costs if not discovered and corrected early, likely increasing the further along a program advances. Blanchard & Fabrycky (2006) writes, "The lack of defining an early "baseline" has resulted in greater individual design efforts downstream. Many of these are not well integrated with other design activities, causing costly modifications later on." Ideally efforts should be made to improve the requirements engineering process in an effort to lower cost. Specifically the paper focuses on developing 'unambiguous requirement statements' and its correlation to 'successful systems.' By viewing an old problem from a different vantage point, new insights may provide unique and holistic solutions.

THE SOLUTION SPACE

One of the chief causes of unsuccessful projects that experts cite is attributed to the increased complexity in systems, and the inability of associated requirements to capture this complexity (Bar-Yam, 2003, Katina, et al 2014). However, there is an essential determinate relating a successful system to its requirement, regardless of a system's complexity, and it is related to the mathematical concept of 'solution spaces'.

In mathematics, the solution space is typically defined as the possible set of feasible solutions for a problem (illustrated in figure 1). A "feasible solution space is the only area that contains values for the variables that are feasible, or does not violate the constraints" (Russell & Taylor, 2006) or also defined as "an area that satisfies all constraints simultaneously" (Russell & Taylor, 2006). Within the context of mathematics, given enough information, an optimized solution can be provided for a problem**.**

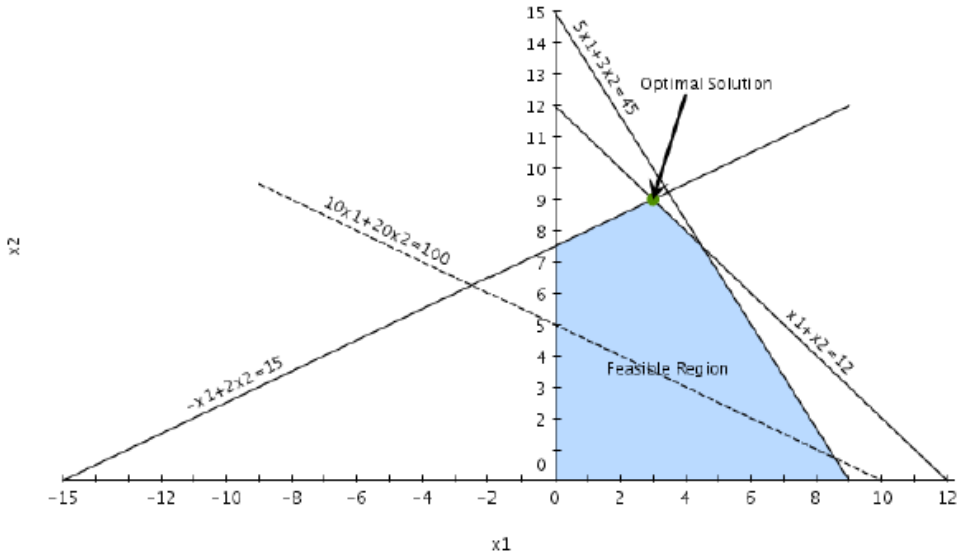

Figure 1. Illustration of a Solution Space with constraints in a Linear Model

Within the context of system engineering, the requirements can be viewed as constraints (Byun et. al, 2014), which define the feasible space for its respective system. This means the requirements must align to the system's purpose, in order to increase the likelihood of delivering a 'successful' system or project. In other words, the solution space associated with the requirement must reside within the intended solution space of the system.

COMPLEXITY

"In the face of these characteristics, the notion that requirements can be characterized by traditional attributes such as being unambiguous, consistent, complete, ranked, understandable, verifiable, traceable, and modifiable becomes tenuous for complex situations" (Katrina et al, 2014).

As depicted in figure 1, this practice can be relatively straightforward with regard to linear systems. However, non-linear or complex systems bring forth unique challenges in this regard (illustrated in figure 2). "For complex situations, the elicitation of requirements as a process needs to be re-examined and evaluated in the face of increasing complexity. More specifically, the nature of requirements, the role of the observer in the requirements process, and the environment within which requirements are generated offer important points of contrast" (Katrina, et al, 2014).

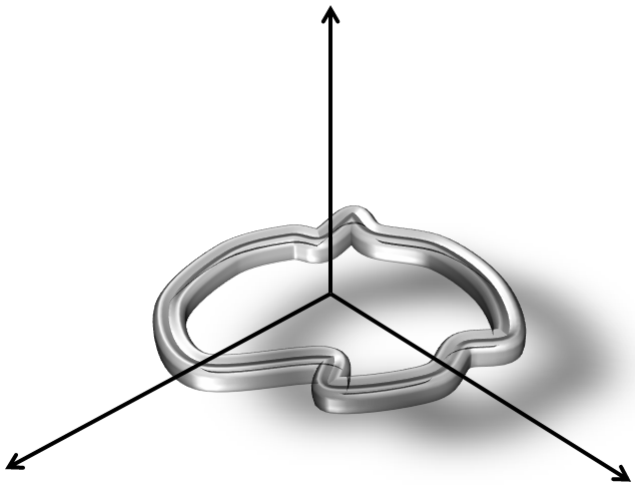

Figure 2. Illustration of a Possible Solution Space in Non-linear Models

Additional problems occur within the framework of systems engineering, in particular complex systems, where optimized solutions can be difficult, if not impossible to obtain (Keating, et al, 2008, Keating & Katina, 2011). This is typically due to the multitude of stakeholders with an interest in the system in question, and the disparate perspectives each possesses. Within this context, the most appropriate approach would be to 'satisfice', a term used by Simon (1956) as he compares the idea of optimizing vice satisficing in his discussion of organisms. "Evidently, organisms adapt well enough to "satisfice"; they do not, in general, "optimize" (Simon, 1956).

Systems & Boundaries

When systems are defined, many attribute as one of the fundamental characteristics is the idea of a system boundary (Weinberg, 1975, Flood & Carson, 1993, Rechtin, 2000). A system boundary is important because it defines the entities residing as a part of the system, and those elements that exist outside of the system. Boundaries assist observers by defining the system of interest, its capabilities and limitations. Although boundaries can be considered artefacts of the observer, they are useful in because boundaries allow the system in question to be scoped such that the system can be studied.

System requirements define the system. The summation of these requirements describes what the system is and what the system does. In effect, the requirements are the boundaries of the system.

"A functional requirement should define what, how well, and under what conditions one or more inputs must be converted into one or more outputs at the boundary being considered in order to satisfy the stakeholder needs" (Bayhill and Henderson, 2005).

To extend this principle, a system's solution space must be capable of containing the summation of all the system requirements. An additional prerequisite is that each solution space of the respective requirement must not oppose (or contradict) another requirement residing within the system solution space.

REQUIREMENTS

In order to understand the concept of the solution space concept, one must first re-examine what a requirement is, and its role with respect systems. Bijan, et al, (2013) defines a requirement with the following: "The IEEE Std 610.12-1990 Standard Glossary of Software Engineering Terminology [IEEE, 1990] states:

A requirement is a condition or capability needed by a user to solve a problem or achieve an objective.

A requirement is a condition or capability that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formality"

"A requirement is a statement that identifies a capability or function that is need by a system in order to satisfy its customer's need. A functional requirement defines what, how well, and under what conditions one or more inputs must be converted into one or more outputs at the boundary in question in order to satisfy the customer's needs" (Bayhill & Dean, 2009). Successful systems are in part attributed to well-formed requirement statements. These statements are normally considered 'unambiguous' and 'bounded'.

Single Dimensional Requirements

In order to illustrate the solution space concept using a simple practical example, consider the requirements of a desert operation military vehicle. There are a number concerns deploying a vehicle in the desert as opposed to deploying in a jungle or city environment (i.e. sand filtration mechanisms, cooling, etc). If one of the vehicle's operations require camouflage (the system's solution space), then a requirement should be stated requiring a paint scheme which aligns to the vehicle's overall purpose. In this example, the requirement (sand paint scheme) has a solution space as well; and its solution space was one dimensional.

System requirements should not contradict or oppose one another. Continuing use of the military vehicle example to illustrate the point; if camouflage is an essential aspect of this vehicle's use, the all of the items on the surface of the vehicle should have similar or identical paint scheme requirements. A bright red antenna on the top of the vehicle opposes or contradicts the neighbouring requirements.

Multi-Dimensional Requirements

However, to understand the concept of a two dimensional requirement, consider a requirement where an item has to operate within a range. For example, the desert vehicle must be able to maintain a velocity between 20-25 miles per hour (mph). Since there are an infinite set of

solutions exiting between 20 and 25 mph, (i.e. 20.1, 22.002, 22.0003, etc), this solution is continuous and linear.

If another dimension is added to the previous speed requirement, such as, the vehicle must reach a velocity of 20-25 mph within 20 seconds; the resultant requirement can either be two or three dimensional plane (depending on how the plane is drawn), thus creating a requirements 'feasibility space'. The requirement statement will become three dimensional if an additional constraint is added such that the statement is written as follows:

The vehicle shall reach an operating velocity of 20 -25 mph within 20 seconds, and must be capable of maintaining this velocity for at least 15 minutes.

If the fictional speed requirement were illustrated, figure 3 could a possible set of solutions which satisfies the military's needs:

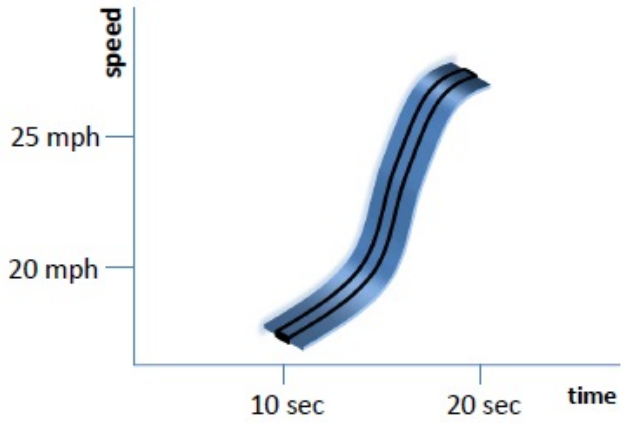

Figure 3. Illustration of speed vs time

However, one key to developing and fielding a successful system is aligning all of the system's requirements, such that the solution of one subsystem requirement does not interfere with the solution to another. As in the one dimensional example, meeting the requirement for one requirement should not violate the subsystem requirement of another. Consider the requirements in the previous example:

The vehicle shall reach an operating velocity of 20 -25 mph within 20 seconds, and must be capable of maintaining this velocity for at least 15 minutes.

The vehicle paint scheme shall simulate the color and shadow patterns that are commonly found in the desert.

If the speed requirement for the previously mentioned desert vehicle accelerates the peeling of camouflage and the military vehicle may be easy spot. Therefore requirements need to be aligned such that both feasible regions align as illustrated in figure 4.

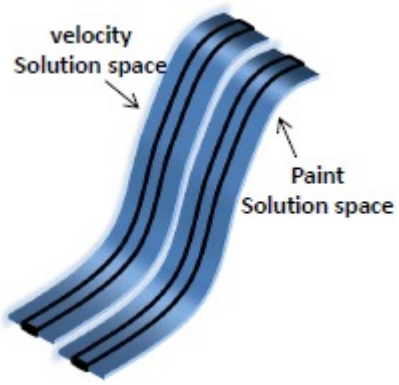

Figure 4. Illustration of multiple requirement feasible regions.

In this fictional example, completion of the overall mission and returning troops safely depends on stealth as well as escaping (when necessary), meeting both of these requirements are paramount. Alignment of the requirements 'feasibility space' with the system's intended solution space tends to result in successful design, or solving the right problem (Mitroff, 1998). This means systems requirements should be constructed that the subsystem's feasibility spaces align.

Characteristics of a Good Requirement

"The IEEE says that 'requirements must be unambiguous, complete, correct, traceable, modifiable, understandable, verifiable, and ranked for importance and stability'" (Bahill & Henderson, 2005). Young (2001, 2006) list the following qualities as characteristics of a good requirement:

- Necessary
- Verifiable
- Attainable
- Unambiguous
- Complete
- Consistent
- Traceable
- Allocated
- Implementation free
- Standard constructs
- Unique identifier

Although this is not an all-inclusive listing, the above still captures the majority of the criteria attributed to well written requirement statements. Good requirements play a pivotal role in successfully deploying system. And as part of the requirement engineering process, requirement generation is considered essential to proper design, development and deployment of the system of interest. Frequently issues arise in systems when requirements

Of the requirements engineering process, Mobasher and Cleland-Huang (2011) state, "These activities engage various stakeholders in the task of identifying and producing an agreed-upon set of requirements that clearly specify the functionality, behaviour, and constraints of the proposed system"

Katrina et al (2014) cites several issues limiting the requirement elicitation process with respect to complex systems, are due to complex systems are characterized by:

"• Diverse and potentially conflicting perspectives,
• High levels of ambiguity and uncertainty,
• Information that may be incomplete, incorrect, or nonexistent,
• Boundary conditions that are ambiguous and subject to change,
• Resources that are insufficient or subject to rapid shifts,
• Limited understanding of the problem or need
• High impact of nontechnical aspects of the problem (e.g., politics),
• Indeterminate entry point to address the situation,
• High levels of contextual influence (e.g., infrastructure, managerial worldviews)."

Unambiguous Requirements

One of the attributes of a good requirement is that it is unambiguous. "Can the requirement be interpreted in more than one way? If so, then the requirement should be clarified or removed. Avoid the use of synonyms and homonyms" (Bahill & Henderson, 2005). This means the statement or statements should only be interpreted one way. When statements are ambiguous, stakeholders can either misconstrue or interpret the objectives of the requirement. Ambiguous requirements can potentially lead to mistakes at the system level, where it can be costly to fix. Worse yet, if problems at the requirement engineering phase are not resolved in a timely fashion, the impact may be enormous in the latter stages of the system engineering process (Blanchard & Fabrycky, 2006).

If the ways a requirement can be interpreted are equated to a degree of freedom (DoF), the optimum requirement statement minimizes, to the greatest extent possible, the requirement's DoF. By extension, the sum of the all requirements' DoFs, should also be minimized to optimize system design. Mathematically this can be expressed as:

$$1)\ Min\left[\sum_{i=1}^{n} DoF_{Req}\right]$$

This term provides an initial foundation for designing an algorithm or 'tool' to check and minimize ambiguity in requirements. This tool would apply to both linear and non-linear systems to evaluate requirements and ensure unambiguous statements.

CONCLUSION

Often, increased costs results from rework and other issues related with ambiguous or incorrectly written system requirements. One of these issues results when a single requirement can be satisfied by a number of feasible solutions. This becomes problematic when a chosen solution doesn't fit within the overall paradigm of the intended solution space, or when the given

singular solution presents unintended consequences when combined with neighbouring solutions from other requirements with the system of interest, system of system, or the environment of interest. If requirements can be mapped to constraints and/or degrees of freedom, it may be possible to develop a tool to identify if a given set of requirement possesses: 1) a high degree of freedom -thus resulting in a high number multiple solutions; 2) a solutions that interfere with adjoining system requirements, and; 3) potentially cause unintended consequences with regard to the system of interest, system of system, or the environment of interest.

Although the focus of this paper centers on one specific characteristic of a requirement, the authors do not imply other attributes should be ignored. On the contrary, the intent is that similar analysis should be conducted with respect to each attribute, and with additional analysis to evaluate if these criteria should be refined and ranked. This paper is just an initial effort to relate the importance of developing a methodology of improving and checking the quality of a particular attribute, to understand the correlation to a good requirement extend that relationship to a successful project or system.

REFERENCES

Bahill, A. T. and Henderson, S. J. (2005). Requirements Development, Verification, and Validation Exhibited in Famous Failures. Systems Engineering. 8(1):1-14

Bar-Yam, Yaneer (2003). When Systems Engineering Fails - Toward Complex Systems Engineering, IEEE International Conference on Systems man and Cybernetics, 2: 2021-2028.

Bayhill, A. and Dean, F (2009). Discovering System Requirements, Handbook of Systems Engineering & Management., (Editors: Sage, A and Rouse, W). John Wiley & Sons, Inc. Hoboken, New Jersey

Bijan, Yu, Stracener & Woods (2013). "Systems Requirements Engineering—State of the Methodology" Systems Engineering. Vol. 16, No. 3, 267- 276.

Blanchard BS, Fabrycky WJ (2006) Systems engineering and analysis, 4th edn. Pearson Prentice Hall, Upper Saddle River

Byun, J., Rhew, J., Hwang, M., Sugumara, V., et al. (2014). "Metrics for Measuring the Consistencies of Requirements with Objectives & Constraints." Requirements Engineering Vol.19:89–104.

Flood, Robert L. & Carson, Ewart R. (1993). Dealing with Complexity, 2nd ed. Plenum Press, New York

Katina PF, Keating CB, Jaradat RM (2014). System requirements engineering in complex situations. Requirements Engineering 19:45–62

Keating CB, Katina PF (2011) System of systems engineering: prospects and challenges for the emerging field. Int J System of Systems Engineering 2:234–256

Keating CB, Padilla JJ, Adams K (2008) System of systems engineering requirements: challenges and guidelines. Engineering Management Journal 20:24–31

Mitroff I (1998) Smart thinking for crazy times: the art of solving the right problems. Berrett-Koehler, San Francisco

Mobasher, Bamshad and Cleland-Huang, Jane (2011). Recommender Systems in Requirements Engineering Artificial Intelligence (AI) Magazine, ISSN 0738-4602, 2011, Volume 32, Issue 3, p. 81

Rechtin, Eberhardt (2000). Systems Architecting of Organizations: Why Eagles Can't Swim, CRC Press , Boca Raton

Russell, Roberta S. & Taylor III, Bernard W. (2006) Operations Management: Quality and Competitiveness in a Global Environment, 5th ed. John Wiley & Sons, Hoboken, New Jersey

Simon, H. A. (1956). Rational Choice and the Structure of the Environment. Psychological Review 63 (2): 129–138. doi:10.1037/h0042769.

Weinberg, Gerald M. (1975). An Introduction to General Systems Thinking, John Wiley & Sons, New York

Young, R.R. (2001). Effective Requirements Practice, Addison-Wesley, Reading, Massachusetts

Young, R.R. (2006). Criteria of a Good Requirement. http://www.ralphyoung.net/artifacts/CriteriaGoodRequirement.pdf