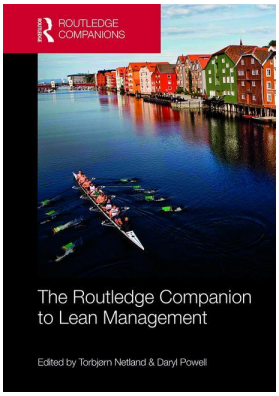

The Routledge Companion to Lean Management

Torbjørn H. Netland, Daryl J. Powell

Lean Systems Engineering

Publication details
https://test.routledgehandbooks.com/doi/10.4324/9781315686899.ch7
Cecilia Haskins, Bohdan W. Oppenheim
Published online on: 28 Dec 2016

7
LEAN SYSTEMS ENGINEERING

Cecilia Haskins and Bohdan W. Oppenheim

Introduction

Systems engineering (SE) has proven to be an established sound practice; but it is not always delivered effectively. A record of accomplishment of over 100 successful space launches between the years 2000 and 2015 demonstrates the result of systems engineering practiced well. However, recent US Government Accountability Office (GAO, 2008) and NASA (2007) studies of space systems criticized significant budget and schedule overruns, some exceeding 100 percent. These programs struggle with waste, poor coordination, unstable requirements, quality problems, and management frustrations. Ironically, the early years of American aerospace programs were quite lean and typified by small, collocated groups of enthusiastic engineers, little bureaucracy, and projects that were completed in record time.

Recent studies by the researchers at the MIT Lean Advancement Initiative (LAI) have identified a staggering amount of waste in government programs, concluding that value is created during only 12 percent of the program time—and the rest is waste (Oppenheim, 2004; McManus, 2005). This waste represents a vast productivity reserve in programs and an opportunity to improve program efficiency as illustrated in Figure 7.1.

Practices in classical systems engineering and program management (PM) have grown heavier and more wasteful over the past 60 years. At the same time, the number of top-level requirements in projects has grown, routinely reaching many hundreds and thousands. This drives the perceived

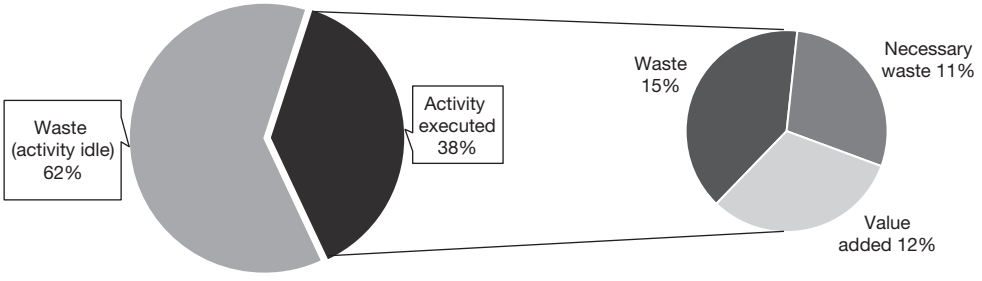

Figure 7.1 88 percent productivity reserve

Source: Oppenheim (2004); McManus (2005).

need for program bureaucracy and makes programs costlier, longer, and usually more frustrating to all stakeholders. This can be attributed to two influential factors: the uncompromising need for reliable system-level performance, and inefficient acquisition practices and incentives. Oppenheim (2015) outlines the history of this evolution that has created the paradox that plagues modern aerospace systems. On one hand, these projects are building some of the most technologically advanced and inspirational systems ever engineered by humans. On the other hand, current design and management processes manifest some of the least efficient organizational efforts ever practiced.

For comparison, it is fascinating to recall that the early Apollo program, without a doubt the most dramatically successful space program in the entire history of human civilization, started with only four requirements pronounced by President J. F. Kennedy (paraphrasing): "(1) take man to the Moon, (2) and back, (3) safely, (4) by the end of the decade!" Similarly, the highly successful U2 aircraft program started only with a few requirements defining the flight altitude, speed, endurance, and payload.

What is Lean Systems Engineering?

Systems engineering, which evolved from practices in the space industry to help deliver flawless complex systems, focuses on technical performance and risk management. Lean focuses on creating value through waste minimization, short schedules, low cost, flexibility, and attention to quality. The International Council on Systems Engineering (INCOSE) has developed the subsequent definition of *lean systems engineering* (LSE): lean systems engineering is the application of lean principles, practices, and tools to systems engineering in order to enhance the delivery of value to the system's stakeholders.

Lean thinking is the holistic management paradigm credited for the extraordinary rise of Toyota to the most profitable and near-largest auto company in the world (Womack and Jones, 1996). Toyota refers to its practice of combining excellent product development and SE as simultaneous engineering. For example, the Prius car design was completed nine months after the end of styling, a performance level previously unmatched by any competitor (Morgan and Liker, 2006).

LSE is the area of synergy of lean and SE with the goal to deliver the best life cycle value for technically complex systems with minimal waste. Emphatically, *LSE does not mean less SE*, or cutting corners. It means more and better SE, "with better preparations of the enterprise processes, people and tools; better program planning and frontloading; better workflow management; and leadership with higher levels of responsibility, authority, and accountability (RAA)" (Oppenheim, 2011: xv).

Challenges and Opportunities for LSE

In large projects, value formulation is a difficult process not only because of complex technology, but also because of unstable program funding, dissolved management, fragmented execution, and policy and politics. In many technologically complex programs customers lack expertise to describe their needs clearly and more often than not must be assisted in the task by value creators, or a proxy SE organization through extensive efforts of interaction, cooperation, and clarification.

> In lean systems engineering, both customers and contractors have a responsibility to formulate requirements as well as the state of the art permits, without blaming one another for inadequate effort while working together as a seamless team of honest, open, trustworthy partners who share the same goal.
>
> *(Oppenheim, 2011: 52)*

Requirements

In traditional SE, value to the customer is formulated using requirements, first the top level or customer requirements, then detailed derived requirements allocated for all subsystems at all levels of decomposition. The value proposition to be captured must involve not only explicit requirements and related documents, but also unspoken requirements defining needs, context, operations, interpretations, interoperability, and compatibility characteristics, as well as a good understanding of customer culture.

Experience indicates that many programs eager to get underway tend to rush through the process of capturing top-level requirements, resulting in incomplete, incorrect, or conflicted requirements that burden subsequent program phases with waste (Oppenheim et al., 2011). Poorly formulated requirements significantly increase program cost and lead time, and in extreme cases even cause cancellation of entire programs. Programs that require long durations often suffer requirement instability introduced by the need for changes which were unforeseen at the beginning of the program.

Requirement stakeholders often ignore the fact that a requirement is an imperfect and inherently ambiguous means to describe need. Typically, a requirement is a sentence containing several words. Written in a natural language, especially one as rich as English, where each word in a dictionary has several meanings, it is inherently ambiguous in the linguistic sense. Additional ambiguity arises because of handoffs: the person writing the requirement has in his/her mind the rich context of the need, while the person reading the requirement sees only the requirement text. Because of these structural communicative disjunctions, it is critically important not only to make every effort to make all requirements crystal clear and complete, but also to create means to clarify requirements without causing requirement creep, properly planning effective and efficient channels for clarifications.

Applying LSE we continue defining value using requirements, but in view of the above difficulties we place a significantly higher emphasis on the quality of the effort of formulating and clarifying the requirements, and make certain that all of the above potential pitfalls are minimized. We also promote development of a robust process for capturing and formulating requirements.

Interfaces

With "n" elements in the system, there are $n(n-1)/2$ possible simple connections or interfaces. A typical space vehicle or craft has tens of thousands of elements. This alone makes the interface definition effort formidable, as each interface is needed to write good specifications. Systems engineering practitioners anticipating interfaces understand the trepidation question: "Have we included all of them?," knowing that even one omitted interface may cause fatal failure. Particularly difficult to anticipate are the "wicked" human or higher-order interfaces. The Space Shuttle *Challenger* disaster serves as a dramatic example: engineers understood that the rubber O-rings in the Solid Rocket Boosters must not be used in cold weather, but they ignored the human interface between the O-ring and the shuttle flight management. The managers did not appreciate the risk of cold weather and ordered the flight, which led to the catastrophe (Challenger Commission, 1986).

Model-based systems engineering (MBSE), with its advanced functional and physical views of the system, greatly facilitates the management of interfaces, but cannot assure that all interfaces have been included. Model-based systems engineering can help in identifying possible interfaces by making the n-squared matrix easier to manage, but cannot fill in the details in each matrix cell. That task is still left to the experience and intuition of engineers. The problem is that the experience and intuition work well only for well-understood systems. This brings us to the

following important conclusion: it is not realistic that all interfaces in a complex system can be anticipated and defined early in the program. Since all interfaces need to be defined in order to write a complete set of requirements, it follows that it is not realistic to develop good, detailed requirements at the beginning of a program.

The topic of interfaces is further complicated by two widespread industrial practices:

1 Distribution of production among as many contractor sites and supplier locations as possible, driven by political and financial pressures to "spread the wealth" and secure broad political support for the project.
2 Overwhelmingly popular corporate policy to "stick to core competencies and subcontract the rest," with the vast majority of system parts built by a complex network of suppliers. This complex multitier supplier network is largely driven by the fact that higher-tier costs are higher than those at a lower tier, and the time between building parts at higher tiers is so long that many organizations decide to outsource lower-level design activities to specialty contractors who survive because they focus on just one type of activity for multiple buyers.

These practices make the program coordination and system integration challenging and increase the need for excellent classical systems engineering and program management. More specifically, prime contractors perform system design, major structural design, and systems integration, and subcontract subsystems and components to the established vendor base for major subsystems (e.g. engines, avionics). These second-tier subcontractors then subcontract for components (such as valves, pumps, etc.). These third-tier contractors subcontract for smaller parts such as sensors, actuators, seals, and so forth. It is normal for a supplier network that builds a complex system to include four tiers of suppliers. Therefore, lean SE strongly recommends that system design is fully completed and stable before allocating lower-level requirements to subcontractors. Allocating only fully mature and stable requirements prevents costly, lengthy, and frustrating requirements instabilities (Oppenheim, 2015).

INCOSE LSE Working Group

Waste in classical SE emanates from a variety of causes: the conviction of the engineers that everything is unique, requires a unique solution, and a unique set of processes. The result of this mindset is projects that are poorly planned, executed in an ad hoc manner, exhibit poor communications culture, poor coordination, and poor organizational preparation (Oppenheim, 2004). INCOSE is the world authority on systems engineering, with the stated goal to advance the state of the practice. The LSE Working Group was formulated in 2006. In 2010 the group issued the *Lean Enablers for Systems Engineering* (LEfSE), which subsequently was recognized as the INCOSE product of the year, and later received the Shingo Research and Professional Publication Award. LEfSE is a checklist of nearly 200 actionable practices for SE and closely related aspects of product development and enterprise management, including supply chain management (INCOSE, 2011). A series of validating surveys confirmed that the LEfSE recommendations aligned closely with NASA and GAO benchmarking results (see Table 7.1).

Lean Enablers for Systems Engineering

The LEfSE practices are organized according to the six lean principles. The checklists cover a large spectrum of SE practices, with a general focus on improving program value and stakeholder

Table 7.1 NASA best practices compared with LEfSE recommendations (NASA, 2007)

NASA "Best practices"	*LEfSE sample*
Leading with vision: sharing the vision; providing goals, direction and visible commitment	3.2.6 Identify a small number of goals and objectives that articulate what the program is set up to do, how it will do it, and what success criteria will align the stakeholders
Focusing on requirements: mission success driven requirements and validation process	3.2 Clarify, derive, prioritize requirements early and often during execution
Achieving robust systems: use rigorous analysis; robust design; HALT/HASS testing	3.2.5 *Fail early—fail often* through rapid learning techniques (e.g. prototyping)
Models and simulation: model-based systems engineering with "seamless" models, validated by experts	3.2.4 Use architectural methods and modeling for system representations that allow interactions with customers as the best means of drawing out requirements
Visible metrics: effective measures, visible supporting data for better decisions at each organizational level	2.6 Plan leading indicators and metrics to manage the program 3.7 Make program progress visible to all
Systems management: managing for value and excellence throughout the life cycle	4.2 *Pull* tasks and outputs based on need and reject others as waste—prevents execution of unnecessary tasks
Building culture: based on foundation "systems" principles; continuous improvement	6.2 Build an organization based on *respect for people*

satisfaction, and reducing waste, delays, cost overruns, and frustration. The full text of the LEfSE (Oppenheim et al., 2011) is too long for the present chapter, but a brief summary is given here. The full text is available online.

Under the *value principle*, the enablers promote a robust process of establishing the value of the end product or system to the customer with crystal clarity. The process should be customer-focused, involving the customer frequently and aligning the enterprise employees accordingly.

The enablers under the *value stream principle* emphasize waste-preventing measures; solid preparation of the personnel and processes for subsequent efficient workflow and healthy relationships between stakeholders (customer, contractor, suppliers, and employees); detailed program planning; frontloading; and use of leading indicators and quality metrics.

The *flow principle* lists the enablers that promote the uninterrupted flow of robust quality work and first-time right; steady competence instead of hero behavior in crises; excellent communication and coordination; concurrency; frequent clarification of the requirements; and making program progress visible to all.

The enablers listed under the *pull principle* are a powerful guard against the waste of rework and overproduction. They promote pulling tasks and outputs based on need (and rejecting others as waste) and better coordination between the pairs of employees handling any transaction before their work begins (so that the result can be first-time right).

The *perfection principle* promotes excellence in the SE and enterprise processes; the use of the wealth of lessons learned from previous programs in the current program; the development of perfect collaboration policy across people and processes; and driving out waste through standardization and continuous improvement. A category of these enablers calls for a more important role of systems engineers, with RAA for the overall technical success of the program.

Finally, the *respect-for-people principle* contains enablers that promote the enterprise culture of trust, openness, respect, empowerment, cooperation, teamwork, synergy, good communication, and coordination—enabling people for excellence.

The LEfSE were developed to be used as a checklist of good holistic practices. Some are intended for top enterprise managers, some for programs, and others for line employees. Some are more actionable than others, and some are easier to implement than others. Some enablers may require changes in company policies and culture. However, employee awareness of enablers that are the least actionable and most difficult to implement is useful.

A major follow-up project jointly managed by INCOSE, Project Management Institute (PMI), and the MIT Lean Advancement Initiative (LAI), with the participation of nearly 180 industrial companies, developed additional *lean enablers*, this time for *integrated systems engineering and program management* (Oehmen, 2012; Conforto et al., 2013). The guide has been tailored for programs that operate under complex political and financial constraints and applies to a broad range—commercial and governmental, including hardware, aerospace, energy, and infrastructure. The LEfSE were incorporated into the set of 326 enablers that aid programs to improve cost, schedule, and quality performance.

The Future for LSE

When speaking of lean practices, one must not fail to mention the exemplary lean company SpaceX, an almost entirely co-located and vertically integrated small firm building rockets and spacecraft which is well ahead of large governmental space programs, breaking from traditional SE methods with its independently developed lean principles and practices. Companies such as SpaceX provide the inspiration for the development of future lean enablers. At the time of writing a major study of SpaceX operations is in preparation. The illustrative case study summarizes the SpaceX way.

Case Study: Lean Systems Engineering in SpaceX

According to its website, "SpaceX designs, manufactures and launches advanced rockets and spacecraft. The company was founded in 2002 to revolutionize space technology, with the ultimate goal of enabling people to live on other planets" (spacex.com). Its historical milestones include December 2010 when it became the only private company ever to return a spacecraft from low-Earth orbit; in May 2012, its Dragon spacecraft attached to the International Space Station, exchanged cargo payloads, and returned safely to Earth—a technically challenging feat previously accomplished only by governments. Since then Dragon has delivered cargo to and from the space station multiple times, providing regular cargo resupply missions for NASA. The company's comparatively small, talented staff includes former NASA engineer John Muratore, author of "X38 Program System Engineering Lessons" (Bilardo et al., 2008). Muratore (2014) recently told a class at Loyola Marymount University that "classical methods only work well when you are building something that is completely understood – otherwise you need a crystal ball. . . ."

SpaceX only formally manages the payload interface requirements and verification with the customer. It replaces lower-level requirements management, allocations, verifications, and artifact bureaucracy with intense engineering optimization, super-efficient prototype development, design iterations focused on best overall system performance, and extensive multi-dimensional mission

assurance based on an extraordinary amount of testing of prototypes, subsystems, and entire integrated systems.

SpaceX uses a next generation testing infrastructure that permits rapid design-build-test prototyping cycles, rapid development and testing of software and software-hardware systems, and the use of 3D printers. It tests a fully integrated wet rocket in the "test what you fly" condition (in contrast to the traditional "test as you fly" which demands identical but not the same hardware and software to that used in flight). "Test what you fly" performs tests on the actual flight hardware and software, including testing on the launch pad. In order to enable such testing, SpaceX developed restartable engines and other multi-use hardware.

Rather than employ dedicated SE or PM organizations, SpaceX performs mission assurance with integrated SE and PM activities distributed throughout the company, using a unique system of integration and mission assurance responsibilities. For example, for each system element, SpaceX uses a life cycle integrator called "*responsible engineer*" who has full RAA for the element from early concept through prototype development and testing, design, production, integration, launch, and performance in flight, including disposal. This person knows the status of the element at any moment in time, and has the right to inquire, coordinate, and help stakeholders along the life cycle, and react immediately to any problems or delays. This practice enables excellent and efficient mission assurance for the element, traceability, configuration and risk management, and rapid decision making to optimize system design and qualify system parts. Intensive mission assurance and integration are also performed along several vertical dimensions: integration within a discipline (e.g. propulsion system), product integration, and integration for a flight. SpaceX engineers spend almost all their time on "great engineering," avoiding the frequent "bureaucracy of artifacts" of traditional SE and PM.

We estimate that SpaceX practices over 100 of the lean SE enablers. It is fascinating that these practices were developed at SpaceX totally independent of the INCOSE or PMI projects, as independent intellectual development. This is an independent validation of both the lean enablers and SpaceX practices. However, SpaceX has gone farther than the scope discussed in this chapter, and a future release of the lean enablers by INCOSE and PMI should address and incorporate these new revolutionary practices.

References

Bilardo Jr, V. J., Korte, J. J., Branscome, D. R., Langan, K., Dankhoff, W., Fragola, J. R., . . . and Sweet, R. E. (2008). Seven key principles of program and project success—A best practices survey. Available at: http://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/20080021182.pdf (accessed July 2016).

Challenger Commission (1986). *Report to the President by the Presidential Commission on the Space Shuttle Challenger Accident*, June 6, 1986, Washington, DC.

Conforto, E., Rossi, M., Rebentisch, E., Oehmen, J. and Pacenza, M. (2013). Survey report: Improving the integration of program management and systems engineering—Results of a joint survey by PMI and INCOSE. Presented at the *23rd INCOSE Annual International Symposium*, Philadelphia, June 2013.

GAO (2008). *Space Acquisitions—Major Space Programs Still at Risk for Cost and Schedule Increases*, GAO-08-552T. Washington DC. March 4, 2008. Available at: www.gao.gov/new.items/d08552t.pdf (accessed October 17, 2015).

INCOSE LSE WG (2011). Lean Systems Engineering Working Group website. Available at: www.lean-systems-engineering.org (accessed July 2016).

McManus, H. L. (2005). *Product Development Value Stream Mapping (PDVSM) Manual*. Release 1.0. Available at: https://dspace.mit.edu/bitstream/handle/1721.1/81908/PDVSM_V.1_2005.pdf?sequence=1 (accessed July 2016).

Morgan, J. M. and Liker, J. K. (2006). *The Toyota Product Development System: Integrating People, Process and Technology*, New York, Productivity Press.

Muratore, J. (2014, January 23). A traditional discipline of systems engineering in a non-traditional organization system. Lecture, Loyola Marymount University.

NASA (2007). *NASA Pilot Benchmarking Initiative: Exploring Design Excellence Leading to Improved Safety and Reliability*. Final Report, October 2007. Washington, DC, NASA.

Oehmen, J. (ed.) (2012). *The Guide to Lean Enablers for Managing Engineering Programs*. Version 1.0. Cambridge, MA, Joint PMI-MIT-INCOSE Community of Practice on Lean in Program Management. Available at: http://dspace.mit.edu/handle/1721.1/70495 (accessed July 2016).

Oppenheim, B. W. (2004). Lean product development flow. *Systems Engineering*, 7(4), 352–376.

Oppenheim, B. W. (2011). *Lean for Systems Engineering with Lean Enablers for Systems Engineering*, Hoboken, NJ, J. Wiley & Sons.

Oppenheim, B. W. (2015). *Program Requirements: Complexity, Myths, Radical Change, and Lean Enablers*. PMI White Paper. Available at: www.incose.org/docs/default-source/los-angeles/oppenheim_program-requirements.pdf?sfvrsn=2 (accessed October 17, 2015).

Oppenheim, B. W., Murman, E. M. and Secor, D. A. (2011). Lean enablers for systems engineering. *Systems Engineering*, *14*(1), 29–55.

Womack, J. P., and Jones, D. T. (1996). *Lean Thinking: Banish Waste and Create Wealth in Your Corporation*, New York, Simon & Schuster.