2023 Conference on Systems Engineering Research

Literature Review and Research Design for Systems Integration: Case study in Defense Systems

Gaute Tetlie*, Gerrit Muller, Satyanarayana Kokkula

Systems Engineering, Univeristy of South-Eastern Norway, Kongsberg 3616, Norway

Abstract

This paper analyses existing literature to identify an integration strategy suitable for a Norwegian defence contractor. Various types of unknowns cause uncertainties in the system design. These uncertainties manifest as problems discovered during later project phases. To mitigate such uncertainties a criterion-driven integration strategy is suggested. Adding to this strategy, we recommend also identifying test-to-design areas. By doing so, uncertainties not directly captured by the chosen criterion may also be captured. Lastly, a research design with three iterations is recommended to validate the proposed integration strategy. This research shall be executed in Spring 2023, and the findings shall be published later.

Keywords: Systems integration; Defence systems; Integration strategy; Research design; System quality

1. Introduction

The defence industry is known for extensive projects ranging from a few years to decades on time scales. The intricacy and size of defence projects make them hard to plan, budget and execute. As a result, the margins in such projects are highly volatile for the companies fulfilling them. To mitigate the number of uncertainties in such projects, a Design-Build model is often adopted [1]. As opposed to the Design-Bid-Build method, the Design-Build methodology gives the organization ownership of both price and design. Although a Design-Build method gives the contractor more freedom to handle uncertainties, they still suffer from considerable cost overruns [2]. Which begs the question, of whether our current approaches to managing uncertainties are adequate for the defence industry.

Uncertainties rooted in unknowns is often a significant contributor to budget overruns [3]. These unknowns are gradually unraveled during the project lifecycle. Depending on the impact and lifecycle stage, handling these

* Corresponding author. Tel.: +47-949-855-53; fax: N/A.
E-mail address: Gaute.Tetlie@outlook.com

unknowns have an associated project cost. Along with technical unknowns, the defense industry is particularly haunted by unknowns related to organizational or legislative challenges [4]. For a defense project to be successful, the contractor must continuously manage these unknowns through the project lifecycle.

This research is based on a case study within a defense contractor, from here on referred to as the company. The company is a large defense contractor delivering, among others, ground-based air defense (GBAD) solutions. The company has recently experienced an increase in commercial opportunities. To facilitate new and concurrent projects, the project execution phases must be sharpened in terms of costs and hours. The company must thus establish better methods for dealing with uncertainties and unknowns in the product creation process.

GBAD projects typically range from 2-10 years in length, followed by decades of in-field operation. The long lifetime requires the product to have high consistency in performing the intended operation. The ability to consistently perform the intended operation is often referred to as ‘system quality’. During development, the primary source of failures is aspects that were not planned or known. As a result, the quality achieved during development is proportional to the number of unknowns present in the system. Product quality can be defined by the amount unknowns still present upon delivery. The need for product quality is determined by the cost of failures by the product. The amount of effort to disclose unknowns must thus be justifiable according to the intended operation and failure thereof. The amount of unknown reducing measures, often called Systems Engineering, are thus domain specific as shown in Fig. 1.

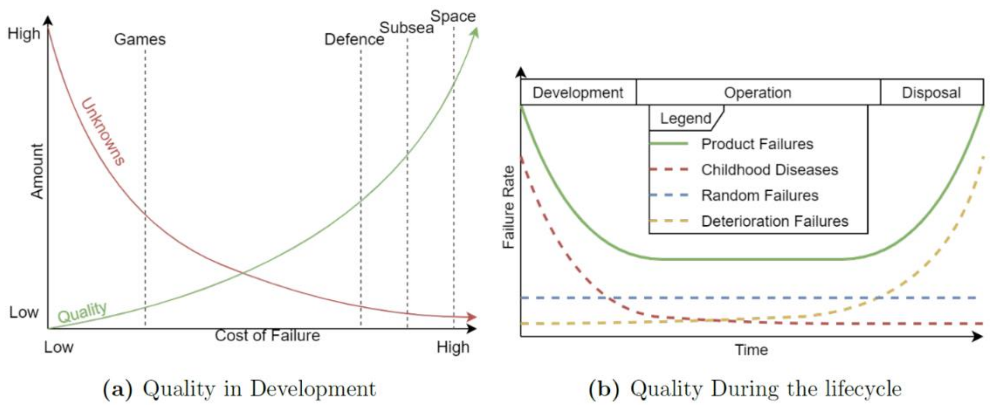


Fig. 1. System quality and unknowns

A recent initiative has been performed to identify possible areas of improvement in the company. One of the results of this initiative is that a significant portion of unplanned costs and hours occurs during the production and verification phases. Correcting faults during the late project lifecycles is often referred to as “late-stage” changes. Combating these late-stage changes is thus critical to reducing the effort related to the product creation process. Systems Engineering will help us to control the unknowns surfacing as late-stage changes.

1.1. Problem Statement

The company is experiencing high costs related to unknowns surfacing during the production and verification phases. These unknowns materialize as unwanted behavior threatening operational capabilities and product quality. The scope and impact of late-stage faults vary from minor software bugs to reworks of entire components. Historically, such issues have been resolved by allocating large amounts of resources temporarily. With an ever-expanding project portfolio, such prioritization efforts may be a luxury not affordable in the future. The importance of preventing late-stage faults has increased during the last two years. Due to a worldwide electronics shortage, the lead time of commonly used components has increased from days to months in many cases. This means that single hardware faults may delay entire projects for months, resulting in increasing project costs. It is thus crucial for the company to find ways of detecting and handling unknowns earlier in the project lifecycle.

To prepare the company for new projects and future markets, there is a need to reduce the number of late-stage changes. Due to a limited pool of resources, it is essential that the measures taken do not require an exhaustive approach. As unknowns are notoriously hard to predict, a certain level of project maturity if often needed for them to be uncovered. Paradoxically, the cost of the change imposed by the unknown increases with project maturity. The method must thus be applied in the sweet spot after the start of the implementation, but before the system production and verification. This phase in the project is often called systems integration. As allocating more resources must be kept to a minimum, the integration strategy itself must be adapted. This remodeled integration strategy must align with the required quality of the GBAD section of the defense domain. To successfully integrate a GBAD system as a whole, both organizational and project-specific elements must be united.

1.2. Research questions

The purpose of this paper is hence to survey the Systems Engineering body of knowledge for a suitable systems integration strategy. The paper will focus on the practical application of integration strategies in the GBAD domain. Through this paper we aim to explore the following research questions (RQ)

- RQ1: Why do significant amounts of unknowns' surface during the production and verification phases?
- RQ2: What are potential measures for earlier detection of these unknowns?
- RQ3: What practical approaches be implemented in future GBAD projects?

We organize the remainder of the paper as follows: First, the paper starts with an introduction of the subject, domain, and problem. This sets the stage for further analysis in the literature study, describing the current state of practice. Next, a research design is suggested for small-scale adaption and feedback collection. Finally, expected findings and potential risks are evaluated in the discussion and conclusions.

2. Literature study

2.1. Unknowns

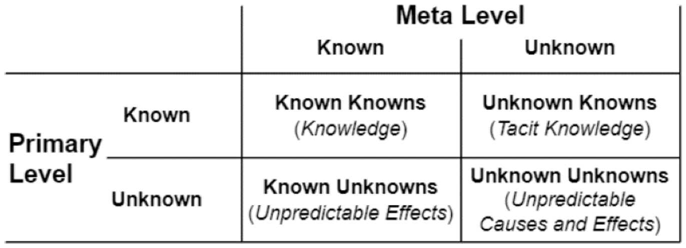
A key concept in Systems Engineering is the notion of unknowns. Unknowns and Knowns can be generally described by four archetypes. These archetypes are vigorously studied in psychology, statistics, scientific theory, and systems engineering. The description of unknowns often starts with Known Knowns. As described by Bammer et al. [5], known knowns are the knowledge in one or more people. Knowledge and unknowns are fluent and participates in the Cynefin Knowledge Flow [6]. We continuously strive to change unknowns into knowns as is rooted in human curiosity. Known Unknowns are the things we know that we do not know. For example, the climate crisis is a known unknown. We know that we must solve it, but how is unknown. Most efforts in the product creation process deal with known unknowns. Next, the Unknown Knowns is our tacit knowledge. Tacit knowledge is more elusive, as it in most cases materializes as intuition. In technical development, unknown knowns often come from the seasoned engineer at the back of the room. Their intuition, or ability to infer from previous experiences, without rationale, is due to unknown knowns. Lastly, there are Unknown Unknowns. These are the unknowns we do not know that we do not know. They are therefore impossible to predict [7]. The phenomenon of unknown unknowns has previously been observed in systems integration by Kjeldaas et al. [4] as "emergent behavior only detectable in hindsight". The three unknowns are often represented by a two-by-two matrix as shown in Fig. 2.

2.2. Knowledge, Problems, and Systems

Based on the concept of unknowns and knowledge, problems can be categorized into four types[6, 8]. First, there are simple problems that have a known solution as shown in Fig. 3. If enough simple problems are aggregated, they eventually become complicated problems. Complicated problems are problems that have joined together to the point where the relationship between cause and effect is no longer straightforward. Much like strings joining into a tangle. Complex problems are more of a challenge to solve. With these problems, we lack the knowledge of the internal mechanisms to reliably predict the outcome. Composite problems where the "whole" changes the problem dynamic (emergence) are typically complex problems. Lastly, there are chaotic problems. These are the problems that we do not understand and cannot consistently reproduce.

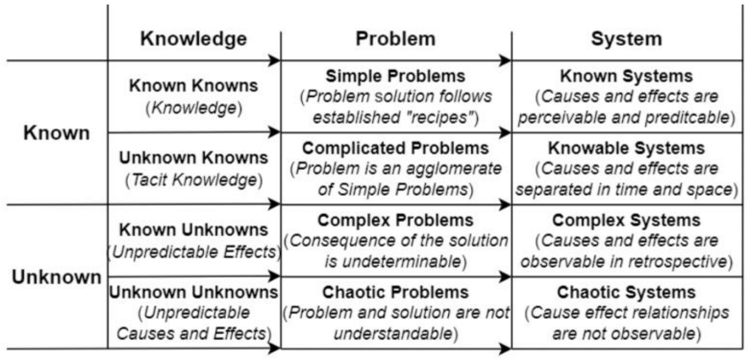
From a project perspective, systems are simply solved stakeholder problems. These problems, although often decomposed, may then be considered as knowledge yet to be gained. Participating in the Cynefin Knowledge Flow [6], knowledge, problems, and systems are therefore heavily interconnected. Kurtz and Snowden thus suggested applying the knowledge archetypes to systems and problems as well [6, 9]. From their discoveries, we get the four-by-three matrix describing the nature of knowledge, problems and systems as shown in Fig. 3. In general terms, our knowledge determines the nature of the problems we are facing. The problems then again define what type of system we can make with our current knowledge. In system development, it is unacceptable to deliver unpredictable systems. This means that we require a method to increase our knowledge during the project creation process to create predictable systems. This transdisciplinary method is commonly called Systems Engineering.

.



		Meta Level	
		Known	Unknown
Primary Level	Known	Known Knowns (*Knowledge*)	Unknown Knowns (*Tacit Knowledge*)
	Unknown	Known Unknowns (*Unpredictable Effects*)	Unknown Unknowns (*Unpredictable Causes and Effects*)

Fig. 2. Nature of unknowns [5]



	Knowledge	Problem	System
Known	Known Knowns (*Knowledge*)	Simple Problems (*Problem solution follows established "recipes"*)	Known Systems (*Causes and effects are perceivable and preditcable*)
	Unknown Knowns (*Tacit Knowledge*)	Complicated Problems (*Problem is an agglomerate of Simple Problems*)	Knowable Systems (*Causes and effects are separated in time and space*)
Unknown	Known Unknowns (*Unpredictable Effects*)	Complex Problems (*Consequence of the solution is undeterminable*)	Complex Systems (*Causes and effects are observable in retrospective*)
	Unknown Unknowns (*Unpredictable Causes and Effects*)	Chaotic Problems (*Problem and solution are not understandable*)	Chaotic Systems (*Cause effect relationships are not observable*)

Fig. 3 Knowledge, problems, and systems

2.3. Systems Engineering

The purpose of Systems Engineering is thus to convert as many unknowns as possible into knowns. At the start of all Systems Engineering projects, there is a period used to understand the need triggering the project. Determining to solve this need will create a top-level unknown in the form of a problem. An engineering system is then defined to take on the endeavor of solving the problem. Systems Architecture and Design is then applied to define a theoretical solution to the problem. Much like baking a cake, a recipe is created. The recipe is taken, and each ingredient is bought and prepared in the implementation. However, in systems development this recipe is often new and never tested before. The assumptions made in the recipe must therefore be validated for the cake to exhibit the desired emergent flavors and characteristics. This validation of design assumptions assimilates the parts into a collective whole and is called systems integration.

2.4. Systems Integration

Systems integration is a process that makes wholes out of parts. As described by Sols, system integration assembles all system parts to yield the desired functionality and characteristics [10]. While system design and architecture identify the parts, functionality, and characteristics, whereas system integration actualizes them. In some ways, systems integration can be considered the reality check of the design decisions made earlier in the project. Design decisions are the response to problems where there are multiple possible solutions. As earlier mentioned, there are different types of problems depending on our knowledge about them. The uncertainty in the decision is therefore directly proportional to the probability of problems occurring during integration. When systems become more complex, more problems occur during integration.

This is arguably one of the key drivers for the “do-a-little-test-a-little” approach adopted by the Agile community [11]. By testing every decision immediately after making, “wrong” decisions can be solved early. To ensure a successful integration it is therefore critical that the design is verified early enough to change it. This is the exact conclusion by Muller [12], where he describes: “the goal of integration is to find unforeseen problems as early as possible, to solve these problems in time”. Unknowns thus surface during production and verification due to design decisions not being reality checked or integrated early enough. This is again primarily due to problems in the unknown category of Fig. 3, being underestimated, miss communicated, or simply neglected.

2.4.1. Context of Systems Integration

A commonly proposed solution to late-stage changes is to “test everything” prior to production and verification. However, this is neither cost-effective nor practical. Testing everything means going through all possible permutations of the system functions. Furthermore, the system permutation must be evaluated against several contextual factors applicable to the system. As everything is not an option, we must look to the context in which systems integration resides. Systems integration is seldom a standalone process. In most cases, it is part of a project which again resides in a particular industry or domain. Through the project and domain, the resources allocated to systems integration are determined.

For example, in the space industry, failures in operation quickly result in the entire project being a failure. Due to the immeasurable loss upon failure, the space domain does, for all practical purposes, test every design decision. This practical maximum method of integration testing is known as integration through Technological Readiness Levels. This integration strategy is used by space organizations such as NASA [13]. In the defense domain, failures in operation are costly but not immeasurable so. The consequence is that significantly fewer resources are allocated to system integration compared to space systems. With fewer resources, the “test everything” strategy is not feasible from a project perspective.

Note that safety-critical elements have a dedicated process in the defense domain [14]. This process is not considered part of the systems integration. Such elements are tested regardless of integration strategy and will therefore not be further discussed in this article.

2.5. Organization

Technical challenges caused by unknowns is not the only threat to a successful systems integration. The organization which performs the integration also has a significant impact on both the process and the product itself. Organizations are constrained to create products with structures mimicking their organizational structure. This phenomenon is called Conway’s Law [15] and is widely observed in software systems. For systems integration, this means that systems with a larger deviation from the organizational structure are harder to integrate. This alternative version to Conway’s law is called Yawnoc’s Law [16].

Gary Langford describes the source of these patterns in his doctorate on systems integration [17]. As earlier mentioned, systems engineers utilize partitioning to simplify problems making them manageable. These partitions are mental borders we make for ourselves to keep the working set of parameters manageable. Partitions are only useful if they aid in performing the product creation process. These partitions are thus more organizational than technical structures. Going back to Conway’s and Yawnoc’s laws, they are simply stating that architectures that are not made for the users are not beneficial.

If the people constructing the system partitions are not working together, neither will the partitions. If the partitions are not working together, systems integration is impossible. Organizational challenges are therefore of great importance to system integration. Organizational challenges must therefore be considered in the integration strategy. Organizational challenges often include but are not limited to sub-suppliers, organizational/product structure mismatch, and scope uncertainty.

2.6. Integration Strategies

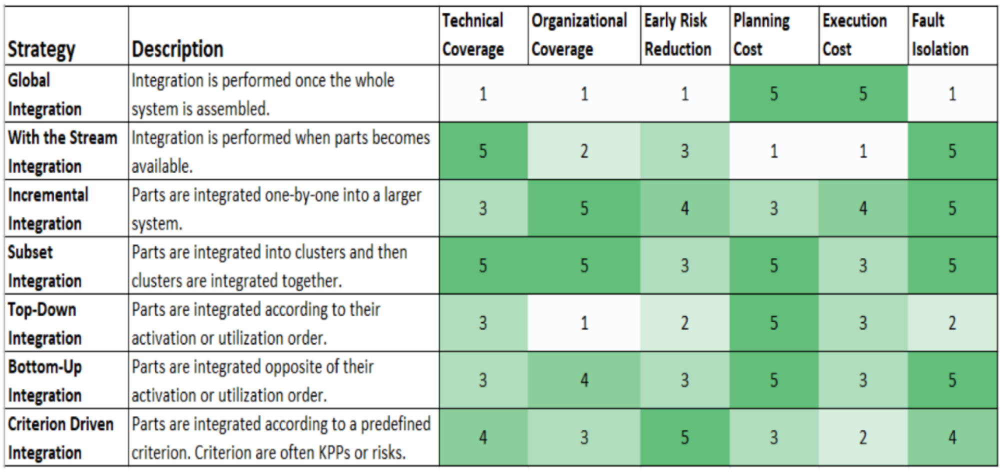
System integration has been around since humans started exploring complex systems. During this time, different strategies have been created to ease the task of integrating systems. INCOSE has collected commonly used strategies as listed in Fig. 4. Regarding the previous discussion, each strategy is accompanied by a qualitative analysis of advantages and disadvantages. However, strategies alone are not enough to ensure a successful integration. In many cases, these strategies give the impression that systems integration is an isolated process. For most systems, systems integration is heavily intertwined with everything from design to validation.

Employing an integration strategy without considering the system, project, and organization is bound to fail. Failures in the integration process lead to unknowns surfacing late and therefore creates late-stage changes. Continuing our discussion, the purpose of systems integration is to uncover unknowns early enough to deal with them. Project uncertainties must thus be identified to give the systems integration process a fighting chance at success. As stated by Kjeldaas et al. [4], the solution to complex problems is only observable in hindsight. The systems integration process must therefore strive to achieve this hindsight observation.

For systems to gain from systems integration, these observations must be made early enough to influence it. In this paper, we coin this concept as an area in need of test-to-design. Regarding problems of complex and chaotic nature, we are not capable of reliably predicting a solution. Solving these problems thus requires early observations before a final solution can be identified (e.g. test-to-design).

2.6.1. When to Integrate

Timing of systems integration is in many cases as important as the content. Although critical, the timing of systems integration is often a challenge. Project timing is the responsibility of the project manager who may or may not see the immediate benefit of systems integration. This becomes especially apparent where early integration is necessary. For early integration to be possible, multiple people from different domains must be aligned. In the project context, this means that the systems integration plan must influence both project and implementation plans. For complex systems, significant integration efforts must be made up-front to uncover complex and chaotic problems. Without this early effort, the design decisions are not validated early enough to be corrected. For complex systems, Slack et al. [19] recommend an effort graph as shown in Fig. 5. Less complex systems require less upfront effort and may follow a more right-shifted effort curve.



Strategy	Description	Technical Coverage	Organizational Coverage	Early Risk Reduction	Planning Cost	Execution Cost	Fault Isolation
Global Integration	Integration is performed once the whole system is assembled.	1	1	1	5	5	1
With the Stream Integration	Integration is performed when parts becomes available.	5	2	3	1	1	5
Incremental Integration	Parts are integrated one-by-one into a larger system.	3	5	4	3	4	5
Subset Integration	Parts are integrated into clusters and then clusters are integrated together.	5	5	3	5	3	5
Top-Down Integration	Parts are integrated according to their activation or utilization order.	3	1	2	5	3	2
Bottom-Up Integration	Parts are integrated opposite of their activation or utilization order.	3	4	3	5	3	5
Criterion Driven Integration	Parts are integrated according to a predefined criterion. Criterion are often KPPs or risks.	4	3	5	3	2	4

Fig. 4 INCOSE integration strategies [18]

2.6.2. Systems Integration for GBAD Systems

Defense systems are continually experiencing a technological race. The measurement of effectiveness (MOEs) of such systems will always be dependent on the countering systems. GBAD systems are no exception, as missiles, fighter jets, and UAVs are constantly trying to elude them. Pushing into uncharted territory means endorsing complex problems only observable in hindsight. The level of unknowns present in such systems calls for a robust integration strategy. However, a “test-everything” approach cannot be financially justified compared to the cost of failure.

.

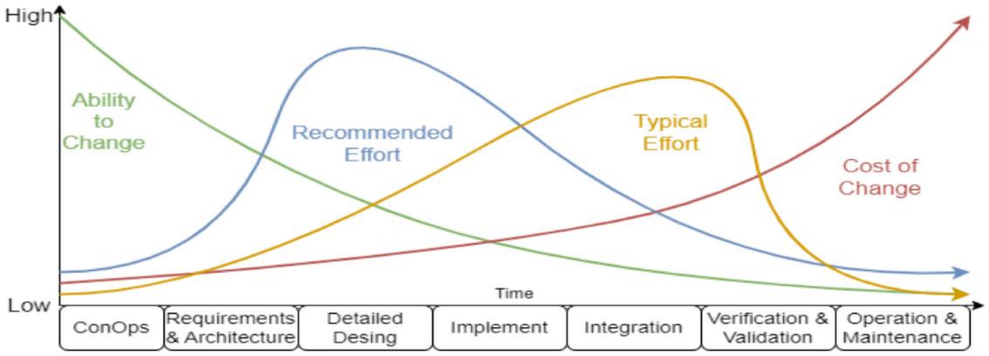


Fig. 5 Systems integration in the project lifecycle [19]

However, in the defense domain, a pure Key Performance Parameter (KPP)-driven integration strategy falls short in some areas. KPPs are emergent properties achieved through multiple parts or systems. This means that some of the first integrations require significant portions of the system to be present. These systems could be simulated, although doing so for most of the system is expensive. With limited resources allocated, convincing management of such expenditures might be difficult. We thus suggest an approach to first establish KPPs and then shift left appropriate low-confidence high-risk integrations. An illustration is shown in Fig. 6.

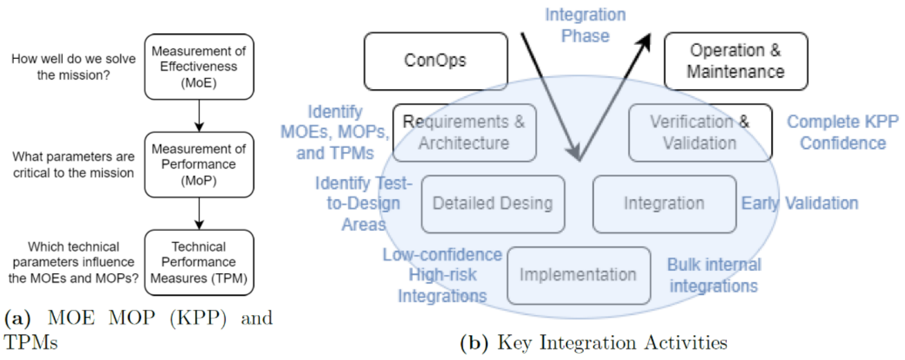

How well do we solve the mission?
Measurement of Effectiveness (MoE)
What parameters are critical to the mission
Measurement of Performance (MoP)
Which technical parameters influence the MOEs and MOPs?
Technical Performance Measures (TPM)
(a) MOE MOP (KPP) and TPMs
Integration Phase
ConOps
Operation & Maintenance
Identify MOEs, MOPs, and TPMs
Requirements & Architecture
Verification & Validation
Complete KPP Confidence
Identify Test-to-Design Areas
Detailed Desing
Integration
Early Validation
Low-confidence High-risk Integrations
Implementation
Bulk internal integrations
(b) Key Integration Activities

Fig. 6 Systems integration in GBAD systems

A simplistic flow diagram of the suggested strategy is shown in Fig. 7. Although the model seems sequential, it is assumed that it will follow the iterative nature of architecture and design. For the company, each concept is documented through design decisions. Implementing the suggested approach can therefore easily piggyback on this already established practice.

3. Research Design

The ideal test design for evaluating the approach would be to implement it into a real project. After project completion, the results could be evaluated against historical data of similar projects. However, the time horizon on most defense projects is ten-fold the period allocated to this research. A concept was thus devised to evaluate the approach within the given time frame as shown in Fig. 8. A prerequisite to the concepts is that the researcher is familiar with the company and has experienced or observed the symptoms. The concept starts with problem identification based on the observed symptoms. As a quality control, a series of interviews should be conducted. This ensures that the identified problem is a “real” problem. Non-real problems or problems solved for the purpose of solving them should be avoided. With the problem in mind, a measurable and realistic goal shall be defined.

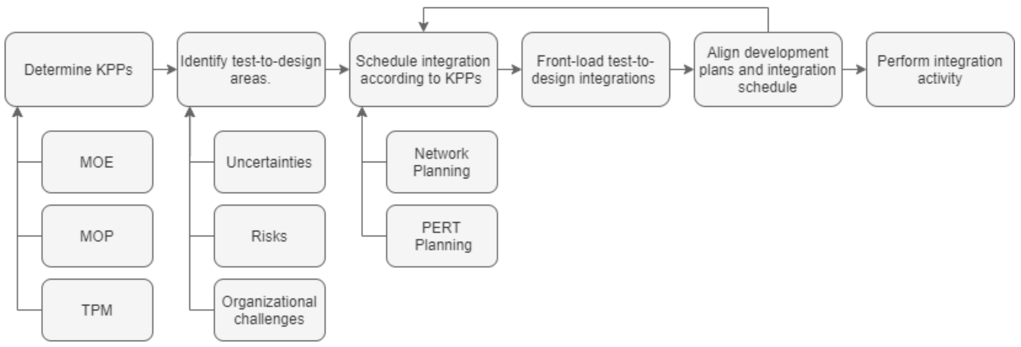

Determine KPPs
Identify test-to-design areas.
Schedule integration according to KPPs
Front-load test-to-design integrations
Align development plans and integration schedule
Perform integration activity
MOE
MOP
TPM
Uncertainties
Risks
Organizational challenges
Network Planning
PERT Planning

Fig. 7 Suggested integration strategy

Next, existing literature on the subject must be investigated. Based on the literature and the leeway within the company, an initial solution is to be chosen. The solution is then evaluated by the company expert group (Test Forum) and the academic supervisor. For early validation, a post-mortem of a previous project is conducted. By applying the new strategy to a completed project, an evaluation of possible benefits can be made.

After the post-mortem iteration, a working group implementation should be held. The working group iteration is dependent on the accessibility of a project in the right phase of the project lifecycle. If held, the project may choose to use the created integration strategy or disregard it. With the feedback from the working group, a draft solution can be made. This draft version will be the foundation for scientific research. Supported by the academic supervisor, a paper concluding the findings shall be made adding to the SE body of knowledge. Although outside of the research period, the next step for the company is a pilot project to implement the solution. Based on the outcome of this pilot, the solution may be added to the Product Creation Process through the process description

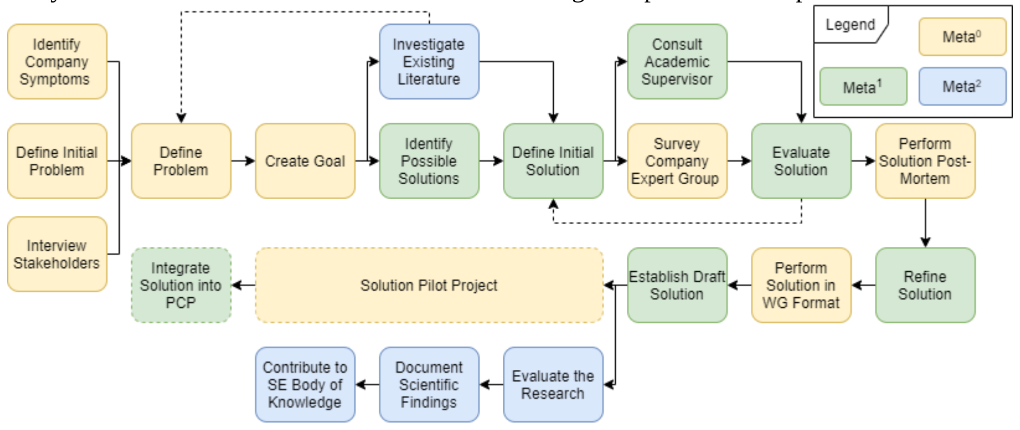

Legend
Meta⁰
Meta¹
Meta²
Identify Company Symptoms
Define Initial Problem
Interview Stakeholders
Define Problem
Create Goal
Investigate Existing Literature
Identify Possible Solutions
Define Initial Solution
Consult Academic Supervisor
Survey Company Expert Group
Evaluate Solution
Perform Solution Post-Mortem
Refine Solution
Perform Solution in WG Format
Establish Draft Solution
Solution Pilot Project
Integrate Solution into PCP
Evaluate the Research
Document Scientific Findings
Contribute to SE Body of Knowledge

Fig. 8 Research design

4. Discussion

4.1. Limitation of Integration

Defense contractors have been given larger freedom through the Design-Build methodology. This approach gives the contractor full ownership of the architecture and design of the system. As a result, the responsibility of identifying and classifying unknowns also resides with the contractor. Due to the unpredictable nature of certain unknowns, additional risk is placed upon the contractor. The system architect, with support from various specialists, is tasked with mitigating this risk. He/she does so by creating concepts that are likely to solve one or more problems. These concepts will inherit the uncertainty of the problem. Evaluating uncertainty requires substantial insight into the organization, technology, and previous systems. The suggested approach depends heavily on the identification of uncertainties. As a result, the integration strategy is heavily reliant on the skill of the system architect. Although this is arguably also true for all system integration. Systems integration is limited to the success of the system design but may improve it through removing uncertainties [17].

4.2. Evaluating Integration

The need for integration is dependent on the system domain and the nature of the project. A new development project may require significantly more resources allocated to integration compared to the further development of an existing system. An organization often builds its product creation process around its primary domain and most frequent project type. Once new projects are created, resource needs are determined according to guidelines, processes, and previous experience. This means that the context of the project often outweighs the need for integration when allocating resources. The resulting integration phase is then either lacking or exaggerated, both limiting the success of the project. Obtaining quantitative data on integration strategies is therefore close to impossible. Evaluating integration is a chaotic problem as we are unable to determine generic casual relationships across organizations.

4.3. Isolation of Integration

A great misconception has been created by the sequential project methodologies commonly applied in the defence domain. In these models, systems integration is presented as something that occurs sometime after implementation. However, an isolated integration phase is bound to be fruitless for anything but simple problems. The purpose of systems integration is to handle uncertainties early enough to resolve potential problems. Starting the systems integration after the system has been implemented means that there is no longer room left for change. Each problem uncovered in the integration will then require rework and increase project costs considerably. When a late integration phase is combined with a lack of resources and delays, the result is late-stage changes.

In the case of internal delays, project management will often attempt to prevent delivery delays. Since systems integration is often detached from the delivery itself, it is often the first process subjected to cuts. Once system integration has been sufficiently cut, problems start to occur during production and verification. Then one of two things happen. Either the project is delayed more than what was originally mitigated, or the workforce must put down extraordinary 12 effort to complete the project. This extraordinary effort often comes at the expense of the workers and other projects. The result is likely a cascading effect with dangerous consequences.

4.4. Expected Results

By performing this research, we except to gain further insight into how uncertainties in the design phase influences the system integration. More specifically, how to facilitate the necessary symbiotic relation between Architecture & design and system integration. For the company, the desired outcome is a strategy that promotes critical early validations without adapting an exhaustive approach. From this strategy, the systems engineering body of knowledge receive a list of important factors to considerate during systems integration in the defence domain. In addition, an indication of the efficiency of upfront investment in systems integration may be obtained.

5. Conclusion

The Design-Build methodology has given defense contractors more freedom in solving customer needs. However, freedom does not come for free. Freedom of design is intrinsically riddled with unknowns. These unknowns may cause problems and late-stage changes if not properly handled. Unknowns are detectable by uncertainties in design decisions. These uncertainties are sourced from both inside and outside the project. Our primary method for managing uncertainties is called systems integration. A multitude of these uncertainties is only observable in hindsight. This means that the systems integration must be performed early enough to handle possible problems caused by uncertainties.

Different strategies have been made by the systems engineering community to facilitate the systems integration phase. Out of the ones described by INCOSE, the Criteria Driven integration is most suitable for the company in question. Adding to this strategy, Muller [12] suggests using KPPs as the integration criterion. As such, each integration will be prioritized according to its yield toward customer value. However, it is recognized that this alone is not enough in defense projects. A pure criterion-driven integration has problems when identifying nontechnological factors. An adapted criterion-driven approach is thus suggested. By tapping into the test-to-design concept, the integration strategy may capture both technical and non-technical uncertainties. However, this modification comes at the cost of a more complicated strategy.

In the last part of the paper, a research design is proposed to validate the approach. The designed approach includes three iterations to improve the suggested approach. After the approach has undergone these iterations, a paper will be written documenting the findings. A secondary goal is to launch a pilot project within the company for further research.

References

1. Douglas D. Gransberg, James E. Koch, and Keith R. Molennar. Preparing for Design-Build Projects: A Primer for Owners, Engineers, and Contractors. ASCE Press, 2006, pp. 1–10. isbn: 978-0-7844-7139-5.
2. Qing Chen et al. “Time and Cost Performance of Design-Build Projects”. In: Journal of Construction Engineering and Management 142.2 (2016), p. 04015074. doi: 10.1061/ (ASCE)CO.1943-7862.0001056.
3. Tzvi Raz, Aaron J. Shenhar, and Dov Dvir: Risk management, project success, and technological uncertainty. R&D Management, 2002, pp. 101-109. doi: https://doi.org/10.1111/1467-9310.00243.
4. Kent Aleksander Kjeldaas, Rune Andr´e Haugen, and Elisabet Syverud. “Challenges in Detecting Emergent Behavior in System Testing”. In: INCOSE International Symposium (2021), pp. 1211–1228. doi: https://doi.org/10.1002/j.2334-5837.2021.00896.x.
5. Gabriele Bammer and Michael Smithson. Uncertainty and Risk: Multidisciplinary Perspectives. Earthscan, 2012, pp. 293–295. isbn: 978-1-8497-7360-7.
6. David Snowden. “Complex acts of knowing: paradox and descriptive self-awareness”. In: Journal of Knowledge Management (May 2002), pp. 100–111. doi: https://doi.org/10.1108/13673270210424639.
7. Nassim Taleb. “Silent Risk: Lectures on Fat Tails, (Anti)Fragility, and Asymmetric Exposures”. In: SSRN Electronic Journal (Jan. 2014), pp. 31–35. doi: 10.2139/ssrn.2392310.
8. Carmel M Martin and Joachim P Sturmberg. “General practice — chaos, complexity and innovation”. In: Medical Journal of Australia 183.2 (2005), pp. 106–109. doi: https: //doi.org/10.5694/j.1326-5377.2005.tb06943.x.
9. C.F. Kurtz and David Snowden. “The new dynamics of strategy: Sense-making in a complex and complicated world”. In: IBM Systems Journal 42 (Feb. 2003), pp. 462–483. doi: 10.1147/sj.423.0462.
10. Alberto Sols. Systems Engineering Theory and practice. ASCE Press, 2014, pp. 219–230. isbn: 978-8-4846-8539-5.
11. Sheetal Sharma, Darothi Sarkar, and Divya Gupta. “Agile Processes and Methodologies: A Conceptual Study”. In: International Journal on Computer Science and Engineering (May 2012).
12. Gerrit Muller. System Integration How-To. url: https://gaudisite.nl/SystemIntegrationHowToPaper. pdf.
13. John C. Mankins. “TECHNOLOGY READINESS LEVELS”. In: (1995). url: https: //aiaa.kavi.com/apps/group_public/download.php/2212/TRLs_MankinsPaper_1995.pdf.
14. Department of Defense. “DEPARTMENT OF DEFENSE STANDARD PRACTICE: SYSTEM SAFETY”. In: SAFT - SYSTEM SAFETY E (May 2012).
15. Melvin E. Conway. “How Do Committees Invent?” In: (1968), pp. 28–31. url: https: //hashingit.com/elements/research-resources/1968-04-committees.pdf.
16. Nigel Slack, Alistar Brandon-Jones, and Robert Johnston. Operations Management. Pearsonbooks, 2013, pp. 495–521. isbn: 978-0-273-77620-8.
17. Gary Langford. Engineering Systems Integration: Theory, Metrics, and Methods. Taylor & Francis Group, 2012, pp. 20–21. isbn: 978-1-4398-5289-7.
18. SEBoK. System Integration. url: https://sebokwiki.org/wiki/System_Integration# Boundary of Integration Activity.
19. R. van Ommering. “Software reuse in product populations”. In: IEEE Transactions on Software Engineering 31.7 (2005), pp. 537–550. doi: 10.1109/TSE.2005.84.