DISSERTATION

A HYBRID SYSTEMS ENGINEERING APPROACH FOR MANAGING COMPLIANCE AND ADAPTABILITY IN AEROSPACE AND DEFENSE

Submitted by

Maryam Gracias

Department of Systems Engineering

In partial fulfillment of the requirements

For the Degree of Doctor of Philosophy

Colorado State University

Fort Collins, Colorado

Spring 2026

Doctoral Committee:

Advisor: Erika Gallegos

Marie Vans
Jeremy Daily
David Prawel

ABSTRACT

A HYBRID SYSTEMS ENGINEERING APPROACH FOR MANAGING COMPLIANCE AND ADAPTABILITY IN AEROSPACE AND DEFENSE

The aerospace and defense industry operates in complex, safety critical, and highly regulated environments. Programs must maintain strict compliance while responding to evolving sustainment and operational needs. Traditional Waterfall methodologies provide structure, documentation, and traceability. However, they often struggle when requirements change late in the lifecycle, leading to increased cost and schedule delays. Agile approaches offer flexibility and frequent stakeholder engagement, but they can fall short of the verification, validation, and documentation rigor required in regulated aerospace programs.

This dissertation examines how a Hybrid approach outperforms pure Waterfall and Agile approaches in aerospace sustainment programs. The research is conducted in three stages. First, surveys and interviews capture practitioner perspectives on requirement management and development practices, showing that most aerospace programs already operate using Hybrid execution models, whether formally defined or not. Second, a comparative analysis of projects executed under Waterfall, Agile, and Hybrid identifies performance patterns and clarifies when Hybrid execution should lean more toward Agile or more toward Waterfall based on safety criticality, certification demands, and time pressure. Third, a detailed case study applies Model-Based Systems Engineering (MBSE) as an analytical tool to evaluate how these methodologies perform when applied to the same sustainment system.

The results show that the Hybrid approach reduces requirement rework and shortens verification and validation cycles. It also improves cost and schedule performance while maintaining compliance with safety and certification requirements. These findings are reinforced through surveys and interviews with industry practitioners, which capture current perspectives on requirement management across development methodologies. The data confirm that most aerospace and defense programs already operate in Hybrid environments.

This study concludes that Hybrid frameworks, supported by MBSE, provide a balanced and sustainable solution for aerospace sustainment programs. The results demonstrate not only why Hybrid outperforms pure Agile and Waterfall approaches, but also how MBSE enables Hybrid to deliver requirements faster, at lower cost, and with stronger alignment to stakeholder and regulatory needs.

ACKNOWLEDGEMENTS

I would like to sincerely thank my advisor, Dr. Erika Gallegos, for her invaluable guidance, support, and encouragement throughout this research. I am also grateful to my committee members, Dr. Marie Vans, Dr. Jeremy Daily, and Dr. David Prawel for their thoughtful feedback and direction, which greatly strengthened this dissertation.

I wish to acknowledge the faculty and staff at Colorado State University for providing the academic foundation and resources that made this work possible. I am also thankful to my peers and colleagues who offered encouragement and stimulating discussions during this journey.

Most importantly, I extend my deepest gratitude to my family and friends for their unwavering love, patience, and support. This accomplishment would not have been possible without them.

DEDICATION

This dissertation is dedicated to my husband, my parents, and my brother, whose love, patience, and encouragement have guided me through every step of this journey.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1. Introduction

The aerospace and defense (A&D) industry operates in a highly demanding environment defined by complexity, safety critical requirements, and strict regulatory oversight. Systems must deliver reliable performance across long lifecycles while adapting to evolving operational and technological needs. Meeting these demands requires development approaches that balance stability, traceability, and compliance with the flexibility to manage changing requirements and stakeholder expectations.

For decades, Waterfall methodologies have served as the standard approach for A&D programs. The linear progression from requirements to design, implementation, verification, and delivery, often represented by the systems engineering V-model, provides strong documentation discipline and traceability aligned with certification needs (Balaji and Murugaiyan, 2012). However, this structure assumes stable requirements. When changes occur late in the lifecycle, Waterfall projects often experience increased rework, cost overruns, and schedule delays (Palmquist et al., 2013).

To address these limitations, many organizations have adopted Agile methodologies. Agile emphasizes iterative development, early stakeholder involvement, and rapid response to change (Beck et al., 2001). Its application has expanded beyond software into large and complex engineering programs. Reported benefits include faster delivery, improved stakeholder alignment, and reduced defect rates (Meyer, 2014). Despite these advantages, Agile practices can be difficult to reconcile with the documentation rigor, auditability, and certification requirements common in A&D environments (Hoda et al., 2011).

As a result, many aerospace programs no longer operate strictly within a single methodological approach. Instead, they rely on Hybrid approaches that blend elements of Waterfall and Agile to address both compliance and adaptability needs. Hybrid frameworks combine the structure of Waterfall with the flexibility of Agile, allowing development practices to be tailored at the system or subsystem level. Safety-critical and highly regulated components may follow Waterfall-based sequencing, while software-intensive or analytical elements may benefit from Agile iteration. This shift toward Hybrid execution reflects how work is performed in practice.

Meanwhile, Model-Based Systems Engineering (MBSE) has emerged as a key enabler for implementing Hybrid frameworks. By shifting from document centric to model centric development, MBSE strengthens requirement traceability, supports early verification, and improves communication across multidisciplinary teams (Walden et al., 2015). Modeling tools such as MagicDraw support interconnected artifacts that remain consistent across Waterfall, Agile, and Hybrid workflows.

This dissertation investigates how Hybrid development approaches outperform pure Waterfall and Agile methods in aerospace sustainment contexts. The research is structured as a three-stage program. The first stage examines industry perspectives through survey and interview data, identifying how practitioners manage requirements and revealing that most organizations already operate in Hybrid environments, whether formally acknowledged or not. The second stage analyzes completed projects executed under Waterfall, Agile, and Hybrid approaches to identify performance patterns and determine how Hybrid execution should lean toward structure or flexibility based on safety criticality, certification demands, and time pressure. The third stage uses a detailed case study to evaluate these approaches within a real sustainment program.

For the case study, this chapter examines the application of Waterfall, Agile, and Hybrid methodologies within the SILA (Sustainment Integration for Legacy Aircraft) system, a pseudonym for a real legacy aircraft sustainment initiative. SILA involves safety-critical requirements, diverse stakeholder needs, and a mix of regulated and flexible subsystems, making it well suited for comparative analysis. In this stage, MBSE is used as an analytical tool to consistently represent requirements, stakeholder interactions, verification activities, and performance outcomes across methodologies. The case study is supported by survey and interview data collected from industry professionals.

By integrating the case study results with survey and interview data, this research investigates how Hybrid approach is better than Waterfall and Agile methods in sustainment programs. Across all three stages, the study focuses on requirement management, stakeholder collaboration, cost and schedule performance, and verification effectiveness. The results aim to provide practical guidance for implementing Hybrid frameworks that reflect real execution conditions while maintaining the rigor required in aerospace and defense programs.

1.1. Research Aims and Research Questions

This dissertation aims to investigate how Waterfall, Agile, and Hybrid frameworks perform in the sustainment of aerospace systems, with particular emphasis on Hybrid as a balanced pathway. The study integrates three sources of data: (i) survey data from industry professionals, (ii) interviews with subject matter experts, and (iii) a case study of the SILA system. Together, these methods provide both breadth and depth, enabling triangulation of findings across different perspectives and contexts.

1.1.1. Research Aim 1

The purpose of this research aim was to understand industry practitioners' perspectives and experiences with the different development methodologies. This constituted the first stage of the research, which initially focused on comparing Waterfall and Agile approaches. However, the findings from this stage highlighted the relevance of Hybrid methodologies as a distinct and important area of consideration. The following research questions are addressed in this Research Aim 1:

- **RQ1.1:** How does transitioning from Waterfall to Agile, in conjunction with MBSE, affect project management, time-to-market, and overall efficiency in the aerospace and defense industry?
- **RQ1.2:** What challenges arise when implementing MBSE during a transition from Waterfall to Agile in the aerospace and defense industry, and what strategies support successful adoption?
- **RQ1.3:** What future opportunities exist for integrating MBSE within Waterfall and Agile approaches in the aerospace and defense industry, and how can organizations maintain a competitive advantage?

1.1.2. Research Aim 2

The purpose of this research aim was to evaluate the impact of Waterfall, Agile, and Hybrid approaches on sustainment performance in real-world aerospace projects. This included assessing requirement change frequency, cost efficiency, and schedule performance, identifying change management bottlenecks, and examining the potential of Hybrid approaches to balance adaptability with regulatory compliance while minimizing rework. The following research questions are addressed in Research Aim 2:

- **RQ2.1:** How do Waterfall, Agile, and Hybrid approaches affect requirement changes, cost efficiency, and schedule performance in aerospace sustainment projects?
- **RQ2.2:** What change management and workflow bottlenecks result from rigid or poorly integrated requirement structures across the development approaches?
- **RQ2.3:** How effectively can a Hybrid approach optimize sustainment performance by balancing adaptability, regulatory compliance, and rework reduction?

1.1.3. Research Aim 3

The purpose of this research aim was to examine how MBSE supports the comparison of Waterfall, Agile, and Hybrid development methodologies. Using the same real-world case study as Research Aim 2, this research aim integrates MBSE artifacts with program performance data to evaluate methodological trade-offs in aerospace sustainment. The findings provide practical guidance on how Hybrid MBSE practices balance adaptability and regulatory compliance. The following research questions are addressed in Research Aim 3:

- **RQ3.1:** How does the use of MBSE impact requirement stability, verification and validation (V&V) timing, and stakeholder collaboration when applied within Waterfall, Agile, and Hybrid development methodologies?
- **RQ3.2:** How does the use of MBSE artifacts support key Measures of Effectiveness (MOEs), including cost and rework reduction, compliance integration, and defect detection accuracy across Waterfall, Agile, and Hybrid approaches?

- **RQ3.3:** How does the integration of MBSE modeling with real sustainment data from the SILA case study reveal trade-offs among Waterfall, Agile, and Hybrid methodologies in aerospace sustainment programs?

Chapter 2. Background

The A&D industry is marked by long development cycles, safety-critical systems, and stringent certification requirements. Programs must satisfy competing objectives: delivering reliable, compliant systems while adapting to evolving operational needs, emergent technologies, and extended sustainment lifecycles (Hiekata et al., 2016). These constraints have historically shaped the dominance of sequential, document-driven methodologies such as Waterfall, which provide traceability and predictability but often fall short in dynamic sustainment environments.

As aerospace systems transition from development into long-term sustainment, the tension between stability and adaptability becomes more pronounced. Sustainment programs must respond to in-service findings, integrate operational data, and introduce incremental upgrades, often years after initial certification. These realities challenge traditional lifecycle assumptions and expose the limitations of development approaches that assume requirements remain largely fixed over time.

A wide range of software and systems development methodologies have been introduced to improve efficiency, adaptability, and alignment with evolving user and business needs (Ke, 2019; Ahmed & Miller, 2022). Among these, the Waterfall model remains one of the earliest and most widely applied approaches. Its linear, sequential structure emphasizes upfront planning and extensive documentation, making it well suited for projects with stable requirements and clear certification expectations (Gupta, 2021). However, this same rigidity limits its ability to respond effectively when requirements change during development, often leading to increased rework and schedule delays. Studies in other engineering domains reinforce this challenge. Field-based needs analysis has shown that design decisions grounded in early assumptions often fail to reflect evolving operational realities, leading to suboptimal performance and economic outcomes over the system lifecycle (Ahmed & Miller, 2023a; Ahmed & Miller 2023b).

One key implication is that development methodologies in aerospace sustainment are evaluated not only on initial delivery success, but on their ability to support controlled change over extended operational lifetimes. Unlike development programs that end at certification, sustainment programs must repeatedly accommodate new data, emerging failure modes, and evolving mission requirements. This reality places sustained pressure on requirement management practices and exposes weaknesses in approaches that assume requirements stability across the lifecycle.

2.1. Waterfall Methodology

The Waterfall model has been the cornerstone of A&D project management for decades. As described by Balaji & Murugaiyan (2012), Waterfall progresses linearly through requirements, design, implementation, verification, and delivery as shown in Figure 1. The V-model, a derivative of Waterfall, strengthens traceability by pairing development phases with verification activities (Forsberg, Mooz, & Cotterman, 2005). Some of the Waterfall's advantages include: (1) Clear documentation and traceability, (2) Predictable planning and milestone tracking, and (3) Alignment with certification and compliance practices (Maharao, 2024).

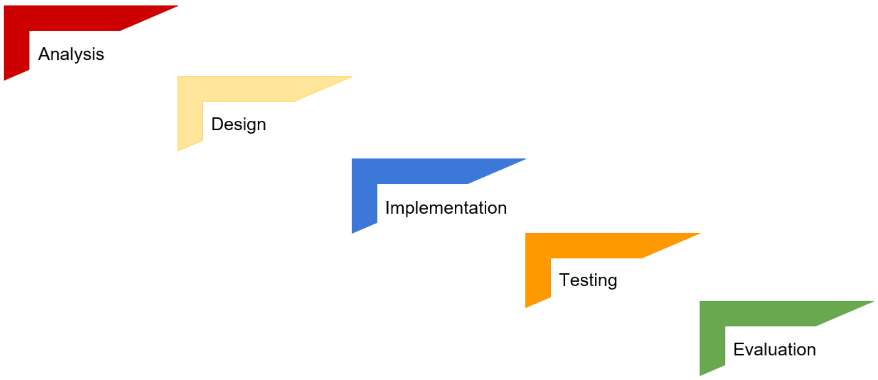


Figure 1. Waterfall Methodology, a Linear Approach

The Waterfall model offers a clear, well-defined structure that divides the project into distinct phases, such as requirements gathering, design, development, testing, deployment, and maintenance. Each phase has specific deliverables and objectives, making it easier to track progress and set expectations. This structured approach is especially appealing, where thorough documentation and adherence to standards are crucial. Clients often favor the Waterfall methodology because it provides a predictable timeline and budget, with clear milestones that establish a sense of control over the project's lifecycle. A major strength of Waterfall lies in its emphasis on documentation. Every phase requires detailed records, ensuring that all aspects of the project are well-documented for future reference, audits, or maintenance. This characteristic is particularly valuable in regulated industries where traceability and compliance are mandatory.

However, the rigidity of this model can lead to significant drawbacks, particularly regarding feedback and adaptability. Since the model operates sequentially, client feedback is typically gathered only at predefined milestones, such as the end of the requirements phase or during testing. This delayed feedback loop can result in late identification of issues or misalignments with client needs. Moreover, if changes are needed after a phase is completed, they require a formal change request process. This process can be time-consuming and expensive, often leading to delays and increased project costs. The lack of flexibility is another key limitation of Waterfall. Once a phase is finalized and the team moves to the next stage, revisiting earlier stages becomes challenging, if not impractical. This rigidity can be problematic in dynamic environments where requirements may evolve, or new insights emerge during development. The inability to quickly implement changes can hinder responsiveness.

Additionally, because testing and validation occur late in the process, defects or issues may not be discovered until significant resources have already been expended. This can lead to costly rework, impacting both the timeline and the budget. The sequential approach often falls short. But despite these challenges, Waterfall remains a viable methodology for projects with stable, well-defined requirements and minimal anticipated changes. Its predictable structure, robust documentation, and clear milestones continue to make it a trusted choice for certain clients and industries. However, its limitations highlight the need for alternative methodologies, such as Agile, in environments that demand greater flexibility and faster feedback loops.

However, its limitations are well-documented. Including Waterfall's inability to accommodate evolving requirements without significant rework (Palmquist et al., 2013). Kemp et al. (2015) also noted that Waterfall can lead to delayed defect discovery due to late verification phases and inefficiency in sustainment contexts where subsystems must be adapted mid-lifecycle. These weaknesses make Waterfall less suited to the dynamic, iterative needs of modern sustainment projects.

Beyond cost and schedule impacts, Waterfall can also limit timely stakeholder engagement. Feedback from operators and maintainers is often incorporated only at formal review points, which may occur long after issues are first observed in service. In sustainment environments, where early insight can prevent operational disruption, this delayed feedback loop increases both risk and response time (Birch et al., 2023).

In sustainment-heavy environments, these limitations become more pronounced. As systems remain in service for decades, late-stage discoveries are not exceptions but expected outcomes. Waterfall's sequential structure, while effective for initial certification, provides limited mechanisms for efficiently incorporating operational feedback once systems are deployed. This mismatch between lifecycle assumptions and sustainment realities has motivated practitioners to seek alternative execution models that preserve compliance while improving responsiveness.

2.2. Agile Methodologies

Agile methods emerged in the early 2000s, emphasizing flexibility, stakeholder collaboration, and iterative development cycles (Beck et al., 2001). Agile prioritizes incremental value delivery, rapid responsiveness, and continuous feedback, making it well suited to dynamic and uncertain environments as shown in Figure 2. Although initially developed for small software teams (Kahkonen, 2004), Agile practices have since been scaled to support larger and more complex engineering programs (Dikert et al., 2016; Salo & Abrahamsson, 2008). Studies have shown that Agile can improve responsiveness to change (Meyer, 2014), reduce defect rates while accelerating delivery in defense programs (Hoda, Noble, & Marshall, 2011), and strengthen stakeholder alignment through frequent engagement (Cocco, Mannaro, & Concas, 2011).

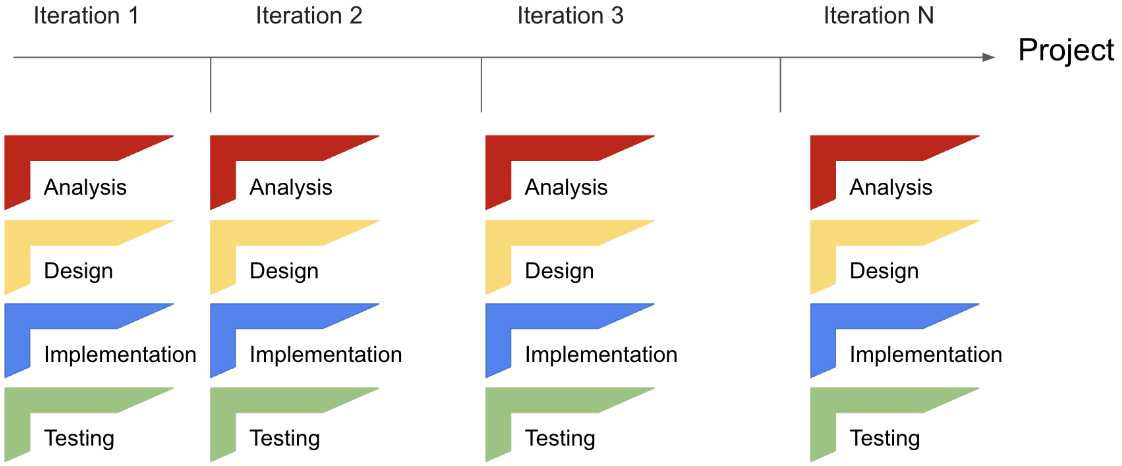


Figure 2. Agile Methodology, an Iterative and Adaptive Approach

Agile comes with several advantages that make it an appealing methodology, where project requirements often evolve. One of its greatest strengths is its flexibility and adaptability to change. In contrast to Waterfall, Agile allows teams to adjust to shifting priorities or new stakeholder requirements without derailing the entire project. This adaptability ensures that the final product remains relevant and aligned with customer needs, even as those needs evolve over time. Another key advantage of Agile is its focus on delivering functional software or system components quickly. By working in short development cycles or sprints, teams can produce usable increments of the system, enabling customers or stakeholders to provide feedback in near real-time. This rapid feedback loop fosters a dynamic development environment where issues can be identified and resolved early, reducing the risk of costly errors later in the project. Agile also emphasizes close collaboration between team members and stakeholders. This collaborative approach enhances communication and transparency, ensuring that everyone involved has a clear understanding of the project's goals and progress. It fosters a sense of shared ownership, which can lead to better team cohesion and higher-quality outcomes (Fagarasan, 2021).

However, Agile is not without its challenges. Its flexible nature can sometimes make the project less predictable. Without a fixed plan or timeline, teams may struggle to estimate costs or completion dates accurately, which can be a concern for stakeholders accustomed to more structured approaches. Another potential drawback is the risk of overlooking essential documentation. Since Agile prioritizes working software over comprehensive documentation, teams must be diligent in ensuring that critical information is not lost or omitted during the development process of the project (Gregory et al., 2015).

Despite these advantages, Agile adoption in aerospace and defense (A&D) contexts presents notable challenges. Agile’s emphasis on lightweight documentation often conflicts with stringent regulatory and compliance requirements, while continuous iteration can complicate configuration management and requirements traceability (Gupta et al., 2016). Furthermore, scaling Agile within large organizations introduces cultural, organizational, and process-related barriers that can limit its effectiveness (Boehm & Turner, 2005; Akif et al., 2012). Case studies, such as

Primavera Systems' successful adoption of Agile practices (Schatz & Abdelshafi, 2005), demonstrate the potential benefits of Agile at scale; however, broader surveys continue to report recurring obstacles, including backlog management, module integration, and insufficient training (Akif et al., 2012).

In sustainment settings, these challenges are often amplified. Agile teams may deliver frequent updates, but without structured governance, maintaining traceability across evolving baselines becomes difficult. Compliance evidence is frequently reconstructed after the fact, which reduces efficiency and undermines confidence during audits. As a result, many A&D programs adopt Agile selectively, applying iteration where it adds value while retaining traditional controls elsewhere (Hamid et al., 2015).

Within the Department of Defense (DoD), Agile adoption has been uneven. While organizations such as the U.S. Air Force have pursued Agile software acquisition strategies to accelerate delivery, progress has often been constrained by entrenched cultural norms and regulatory frameworks (Kenner, 2019; Bieler, 2018). These mixed outcomes suggest that Agile alone is rarely sufficient for safety-critical aerospace programs, particularly when applied across systems that combine software, hardware, and operational interfaces.

These challenges help explain why Agile is rarely adopted wholesale across aerospace programs. Instead, Agile practices are often constrained to software-intensive components or early prototyping efforts, while traditional controls remain in place for safety-critical elements. This selective adoption reflects a broader recognition that while Agile improves speed and feedback, it does not inherently address the governance, traceability, and certification demands of regulated aerospace systems.

Hybrid development approaches have therefore emerged less as a formal methodology and more as an adaptive response to real program constraints. Rather than representing a deliberate replacement of Waterfall or Agile, Hybrid execution often reflects how organizations pragmatically combine elements of both to manage competing demands for speed, rigor, and compliance. This organic evolution suggests that Hybrid practices are shaped by context, subsystem criticality, and organizational maturity rather than by prescriptive frameworks.

Recent studies increasingly point to Hybrid development models as a practical response to the limitations of purely plan-driven or purely iterative approaches (Krupa et al., 2023). Researchers have shown that integrating Agile practices, such as Scrum, within a broader Waterfall structure can help organizations retain documentation discipline while gaining the benefits of iterative delivery and stakeholder feedback (Sitorus et al., 2025). These hybrid configurations are often adopted to address challenges related to requirement volatility, stakeholder coordination, and late-stage defect discovery that are common in complex projects.

However, despite growing interest, standardized guidance for implementing Hybrid approaches remains limited. Much of the existing literature describes hybrid execution patterns informally, with variations driven by organizational context rather than shared frameworks or best practices. In addition, empirical evidence comparing Agile, Waterfall, and Hybrid approaches across diverse project types is still sparse, making it difficult to assess their relative effectiveness in a systematic way (Sitorus et al., 2025). This gap highlights the need for comprehensive comparative studies that examine not only performance outcomes, but also the practical trade-offs and challenges associated with each methodology in real-world settings.

Prior research has documented several challenges organizations face when transitioning from Waterfall to Agile development. One recurring issue is the difficulty of managing multiple projects simultaneously during a methodology shift, as organizations rarely transition a single project in isolation (Johnston, 2014). Coordinating different execution models across parallel programs can increase scheduling complexity and create uncertainty in resource allocation.

Task assignment and workload management also become more complex during transition. Johnston (2014) notes that shortening sprint durations and estimating work in hours rather than abstract story points can help teams adjust more naturally to iterative planning while improving transparency into individual effort and time distribution. These adjustments allow organizations to ease teams into Agile execution while maintaining visibility into progress.

Communication practices present another significant challenge. Waterfall development relies heavily on formal documentation and written communication, whereas Agile emphasizes frequent verbal interaction and collaboration. This shift often requires substantial cultural change, which can be difficult for teams accustomed to document-centric workflows (Francis, 2016). Compounding this issue, some studies report that organizations struggle simply because teams lack a clear understanding of what Agile execution looks like in practice, making it difficult to align behavior with intended process goals (Lester, 2013).

Across multiple studies, resistance to change consistently emerges as one of the most significant barriers to successful transition (Ortolano & Gallegos, 2025). Reluctance to adopt new practices, learn unfamiliar tools, or abandon established routines can slow or derail Agile adoption efforts. Collectively, these findings demonstrate that transitions between Waterfall and Agile involve both technical process changes and human, organizational factors. These challenges help explain why many organizations do not move directly from Waterfall to Agile but instead evolve toward Hybrid approaches that retain familiar structures while gradually introducing iterative practices. The transitions between methodologies are not purely technical shifts, but organizational transformations involving culture, communication, and workforce readiness. The persistence across industries helps explain why many organizations stabilize into Hybrid execution patterns, where familiar structures are retained while iterative practices are gradually introduced.

2.3. Model-Based Systems Engineering (MBSE) as a Tool

MBSE is defined as the formalized application of modeling to support system requirements, design, analysis, verification, and validation across the system lifecycle (Walden et al., 2015). By replacing document-driven engineering with model-driven practices, MBSE improves transparency, traceability, and collaboration. Tools such as MagicDraw enable the development of interconnected artifacts including requirement diagrams, block definition diagrams, activity models, and allocation matrices that maintain consistency across lifecycle phases (Weilkiens, Lamm, & Roth, 2015; Corl & Gallegos, 2024). Through these capabilities, MBSE enhances requirement clarity and traceability (Estefan, 2008), improves collaboration across multidisciplinary stakeholders (Bone & Cloutier, 2010), and reduces rework through simulation-driven verification (Madni & Purohit, 2019; Bumgarner et al., 2024), while remaining flexible enough to support both sequential and iterative development processes.

Importantly, MBSE does not prescribe how development should be executed. Instead, it provides a shared structure that can support Waterfall, Agile, or Hybrid execution depending on how models are governed and used. This flexibility makes MBSE especially valuable in Hybrid environments, where different subsystems may operate on different cadences while remaining connected through a common model baseline.

When combined with Agile, MBSE accelerates requirements validation and integrates continuous feedback (Call & Herber, 2023). Within Hybrid frameworks, MBSE strengthens compliance by preserving documentation while enabling iterative updates (Huss et al., 2023). As depicted in Figure 3, which was adapted from Call & Herber (2023) and Madni & Purohit (2019), the growing complexity of interactions typical in a traditional, document-based approach can be streamlined by allowing stakeholders to directly input and retrieve SE data within the model itself.

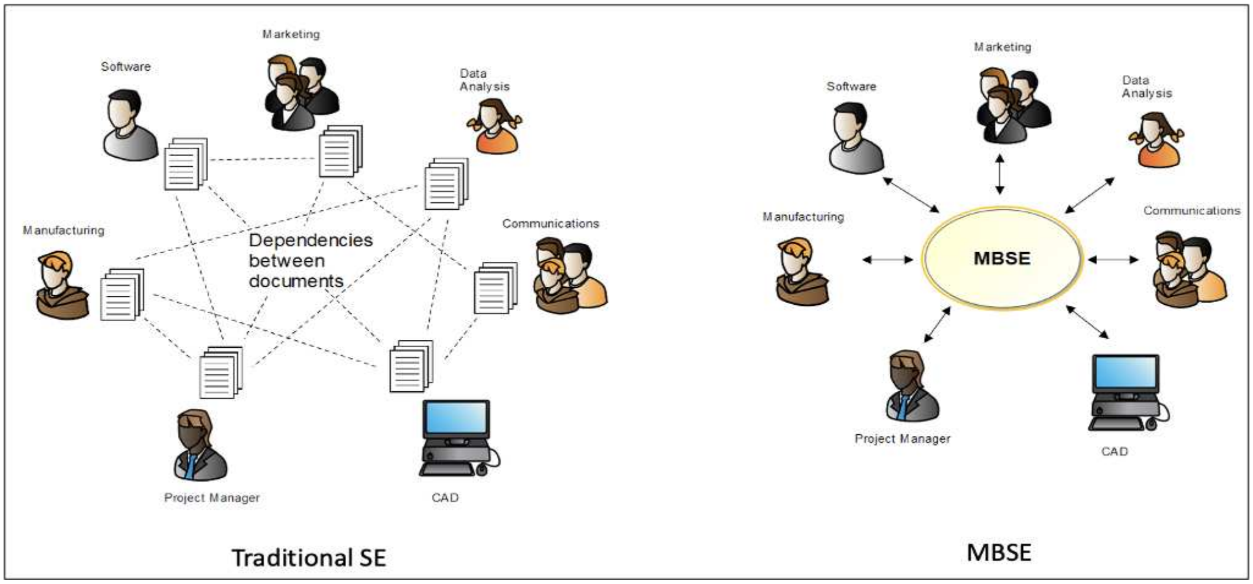


Figure 3. Traditional SE and MBSE Adapted (Call & Herber, 2023; Madni & Purohit, 2019)

2.4. Research Gaps

Although prior research has explored Agile, Waterfall, and MBSE independently, there remains a significant gap in studies that examine Agile, Waterfall, and Hybrid approaches together within the same setting. Much of the existing literature emphasizes the conceptual advantages of Hybrid methods without providing concrete guidance on how Agile and Waterfall practices should be integrated in practice. As a result, organizations are often left to define Hybrid execution strategies informally, with limited methodological consistency across programs.

In addition, relatively few studies define measurable and repeatable metrics for evaluating Hybrid effectiveness across different project contexts. Existing comparisons frequently rely on qualitative assessments or high-level surveys, offering limited quantitative evidence on how

Hybrid approaches perform relative to Agile and Waterfall in areas such as requirement stability, verification efficiency, cost, and stakeholder collaboration. This lack of standardized evaluation criteria makes it difficult to determine when Hybrid approaches provide meaningful advantages and under what conditions they should lean more toward Agile or Waterfall execution.

Another limitation of the current literature is the lack of aerospace specific, in-depth case studies. Many studies examine software development in general terms, without addressing the unique constraints of safety-critical and highly regulated domains such as aerospace and defense sustainment. In particular, very few studies examine how Hybrid execution emerges in response to real program pressures, including certification requirements, system complexity, and operational feedback during long lifecycle sustainment efforts.

While existing studies acknowledge the promise of Hybrid approaches, few examine how Hybrid execution emerges over time within a single program as constraints evolve. In particular, there is limited evidence that traces how programs transition from Waterfall to Agile and ultimately stabilize into Hybrid practices in response to certification pressure, system complexity, and operational feedback.

This dissertation addresses these gaps by integrating survey and interview data from industry practitioners with a detailed case study of the SILA propulsion diagnostic subsystem. By comparing Waterfall, Agile, and Hybrid execution within a single, real sustainment program, the research provides empirical evidence of how Hybrid frameworks balance adaptability and compliance in practice. The findings offer insight into the conditions under which Hybrid approaches improve requirement change management, cost and schedule performance, verification effectiveness, and stakeholder alignment in aerospace sustainment environments.

Chapter 3. Aim 1: Practitioner Perspectives - Survey

3.1. Introduction

This chapter investigates how organizations in the A&D sector perceive the transition from traditional Waterfall practices to Agile methodologies when integrating MBSE. The discussion is grounded in survey data that captures perspectives from industry professionals on project management efficiency, stakeholder collaboration, and system adaptability during this methodological shift. The survey results provide an empirical foundation for assessing both the opportunities and the barriers faced in embedding MBSE within Agile and Waterfall contexts. This chapter sets the stage for understanding how methodological perceptions shape adoption, and it contributes insights into the broader narrative of improving engineering practices in highly regulated, complex environments; and ultimately forms the basis for evaluating Hybrid methodologies. The following research questions are addressed in this chapter:

- **RQ1.1:** How does transitioning from Waterfall to Agile, in conjunction with MBSE, affect project management, time-to-market, and overall efficiency in the aerospace and defense industry?
- **RQ1.2:** What challenges arise when implementing MBSE during a transition from Waterfall to Agile in the aerospace and defense industry, and what strategies support successful adoption?
- **RQ1.3:** What future opportunities exist for integrating MBSE within Waterfall and Agile approaches in the aerospace and defense industry, and how can organizations maintain a competitive advantage?

The content of this chapter has been published in the *ITEA Journal of Test and Evaluation*, titled "Transitioning Perspectives: Agile and Waterfall Perceptions in the Integration of Model-Based Systems Engineering (MBSE) within Aerospace and Defense Industries" (Gracias & Gallegos, 2024).

3.2. Methods

An online survey was administered for this study. The survey was approved by the Institutional Review Board (IRB) at Colorado State University.

3.2.1. Participants

There were a total of 40 participants that completed the survey. The study surveyed employees from a prominent defense and aerospace company. The company is known for its significant contributions to defense, security, and aerospace technology, and is an ideal setting for this study due to its ongoing transition from traditional Waterfall methodologies to more flexible Agile practices. This provides a rich context for exploring the practical implications, challenges, and benefits of Agile practices in a highly regulated and technologically complex industry. The insights gained from the employees can offer valuable lessons and best practices that can be applied across the A&D sector, aiding other organizations in navigating similar transitions. Participants were recruited from various departments across the United States, seeking diverse representation of experiences and perspectives. This recruitment process aimed to capture a comprehensive understanding of how Agile methodologies are being implemented and perceived across different project types and stages.

All participants were selected based on their current or past roles as software or systems engineers and/or Agile-related job responsibilities. This focus ensures that the insights gathered are directly relevant to the technical and project management challenges inherent in the A&D industry.

3.2.2. Survey Design

The survey was conducted online through SurveyMonkey. The survey questions are listed in Appendix A. Data collection occurred March through August 2024. It took participants, on average, 10 minutes to complete the survey. Participation was both anonymous and voluntary, encouraging candid responses and ensuring that the data collected accurately reflects the participants' genuine experiences about the current and past projects. The anonymity of the survey was crucial in obtaining honest feedback, free from the potential biases or fears of repercussions.

The survey was designed to capture insights from professionals actively engaged in A&D projects, focusing on their roles, experience levels, and the specific methodologies employed. Participants provided detailed responses regarding the perceived effectiveness and challenges of Agile practices when integrated with MBSE. The survey consisted of several sections. The first section captured participants' roles, experience with Agile and Waterfall methodologies, and overall tenure in systems or software development. Then captured 5-point Likert scale data on attitudes towards Agile practices and experience with various practical applications of Agile methods (e.g., Scrum, Kanban, and Extreme Programming, XP). The survey then captured integration of MBSE into Agile projects, including the challenges, successes, and perceived effectiveness of MBSE in managing requirements. Lastly, there was an open-ended question for deeper qualitative insights.

3.3. Results

3.3.1. Participant Job Positions

The participants in the survey represented a diverse array of job roles, reflecting the multifaceted nature of project teams. Note, participants could select more than one job role, which several did. As shown in Table 1, the most prevalent job role was engineer, comprising 31.7% (N=19) of the total participants. These include software or system engineers, who play a crucial role in A&D projects, contributing their technical expertise to various aspects of system and software development. Next were Team Leads, who are responsible for overseeing project teams, coordinating activities, and ensuring project objectives are met within specified timelines and budgets. Then, Product Owners, Developers, and Designers each represented significant proportions of the participant pool. Product Owners are tasked with defining and prioritizing project requirements from a stakeholder perspective, while Developers and Designers are involved in the implementation and design aspects, contributing to the development of software and system components. Scrum Masters, Testers, and Business Stakeholders were also represented among the survey participants. Scrum Masters facilitate the adoption of Agile practices within project teams, ensuring adherence to Agile principles and facilitating collaboration and communication. Testers play a critical role in quality assurance, conducting thorough testing to validate system functionality and performance. Business Stakeholders provide valuable insights into project requirements and priorities from a business perspective, guiding decision-making and project direction.

Table 1. Survey Participants' Job Roles on Project Teams

Position	**Count**	**Percentage**
Engineer	19	31.7
Team Lead	9	15
Product Owner	5	8.3
Developer	7	11.7
Scrum Master	8	13.3
Designer	6	10
Business Stakeholder	2	3.3
Tester	3	5
Agile Coach	1	1.7

3.3.2. Experience with Agile and Waterfall Practices

The survey revealed a diverse range of experience with Agile practices. Participants reported experience ranging from 1 to 20 years, with a mean of 5.07 (SD = 3.78). These upper values indicate that many participants have substantial experience and insights into Agile methodologies. Notably, one participant reported 20 years of experience. Additionally, experience durations of 3 and 5 years were also common, showing a mix of moderate and more recent adopters. This diversity in experience levels contributes to a comprehensive understanding of Agile utilization in A&D projects.

Similarly, participants provided a wide range of experience with Waterfall practices, from 1 to over 40 years and a mean of 7.07 (SD = 7.46). Those with shorter tenures, around 1 year, likely reflect recent transitions to Agile or hybrid methodologies. Participants with 9 to 12 years of experience have practical insights into the successes and challenges of Waterfall projects. Extensive experience, surpassing 15 years and even up to 40 years, indicates deep-rooted familiarity with Waterfall methodologies, reflecting its historical dominance in the industry. This variation underscores differing perspectives on project management approaches and the potential benefits of integrating newer methodologies like Agile.

3.3.3. Adoption of Agile, Hybrid and Waterfall

Participants were asked to rate their team's use of Agile practices on a scale from 0 to 100, where 0 represents all Agile, 50 represents a hybrid approach, and 100 represents all Waterfall, see Table 2. The data collected indicates a diverse range of practices, with mean of 48.83 and a standard deviation of 16.28. The scores ranged from a low of 0 to a high of 70. Most teams positioned themselves closer to the hybrid model, specifically within the 50 to 70 range, suggesting a blend of Agile and Waterfall methodologies. This trend indicates that while Agile practices are being integrated to enhance flexibility and efficiency, the structured nature of Waterfall remains necessary for meeting stringent requirements and regulatory standards. Some teams are fully embracing Agile principles, while others still rely significantly on traditional Waterfall practices, reflecting the transitional nature of project management methodologies at BAE Systems.

Table 2. Use of Agile Practices, from 0 (all Agile) to 100 (all Waterfall)

Descriptive Statistic	**Value**
Mean	48.83
Standard Deviation	16.28
Range	0 to 70

The mean score of 48.83 indicates that, on average, teams at BAE Systems lean towards Agile methodologies but have not fully embraced them. This score suggests that most teams are adopting a hybrid approach that combines Agile practices with elements of Waterfall methodology. The standard deviation of 16.28 further highlights the variability in how different teams implement Agile practices, pointing to a wide range of adoption levels and practices within the organization. The analysis of Agile practice ratings reveals several key insights about the project management practices at BAE Systems. Firstly, the high mean score and the prevalence of hybrid approaches underscore the organization's pragmatic strategy of integrating Agile methodologies with traditional Waterfall practices. This integration allows teams to leverage the benefits of Agile's flexibility and iterative processes while addressing the need for structured planning and documentation. Secondly, the variability in scores and the presence of both fully Agile and predominantly Waterfall teams highlight the transitional nature of project management practices within the organization. Some teams have fully embraced Agile principles, benefiting from its adaptive approach, while others continue to rely heavily on Waterfall methods due to their suitability for certain types of projects. This transitional phase reflects the ongoing evolution in project management strategies at BAE Systems, as the organization seeks to balance flexibility with regulatory compliance. Thirdly, the trend towards hybrid approaches illustrates the organization's ability to tailor its project management practices to the specific needs and challenges of different projects. This tailored approach enables teams to optimize their methodologies, combining Agile's advantages with the necessary structure provided by Waterfall. It also highlights the importance of flexibility in project management, as teams adapt their practices to achieve the best outcomes for their projects. The survey results provide valuable insights into the application of Agile and Waterfall methodologies at BAE Systems. The trend towards hybrid practices reflects a strategic balance between flexibility and structure, allowing teams to effectively manage complex projects. Understanding this balance is crucial for optimizing project management practices and ensuring successful project outcomes in a dynamic and regulated environment.

3.3.4. Perceived Benefits of Agile Methodologies

Agile methodologies have gained significant traction due to their emphasis on flexibility, collaboration, and iterative progress. Understanding the acceptance and perceived effectiveness of Agile practices among employees is crucial for companies aiming to implement or refine their Agile processes. Participants were asked about their attitudes towards Agile acceptance, including responsibility completion, alignment with working style, effectiveness, quality of work, flexibility, compatibility, and control over work, see Figure 4. The survey data reveals a strong positive attitude towards Agile practices among employees surveyed. Specifically, 73% of participants agree/ strongly agree that Agile methodologies are beneficial overall. A significant 85% agree/ strongly agree that Agile enables flexibility in tasks, though only 34% find it compatible with all work facets. Additionally, 73% believe that Agile practices give them more control over their work, showcasing the overall positive reception of Agile methodologies within the company.

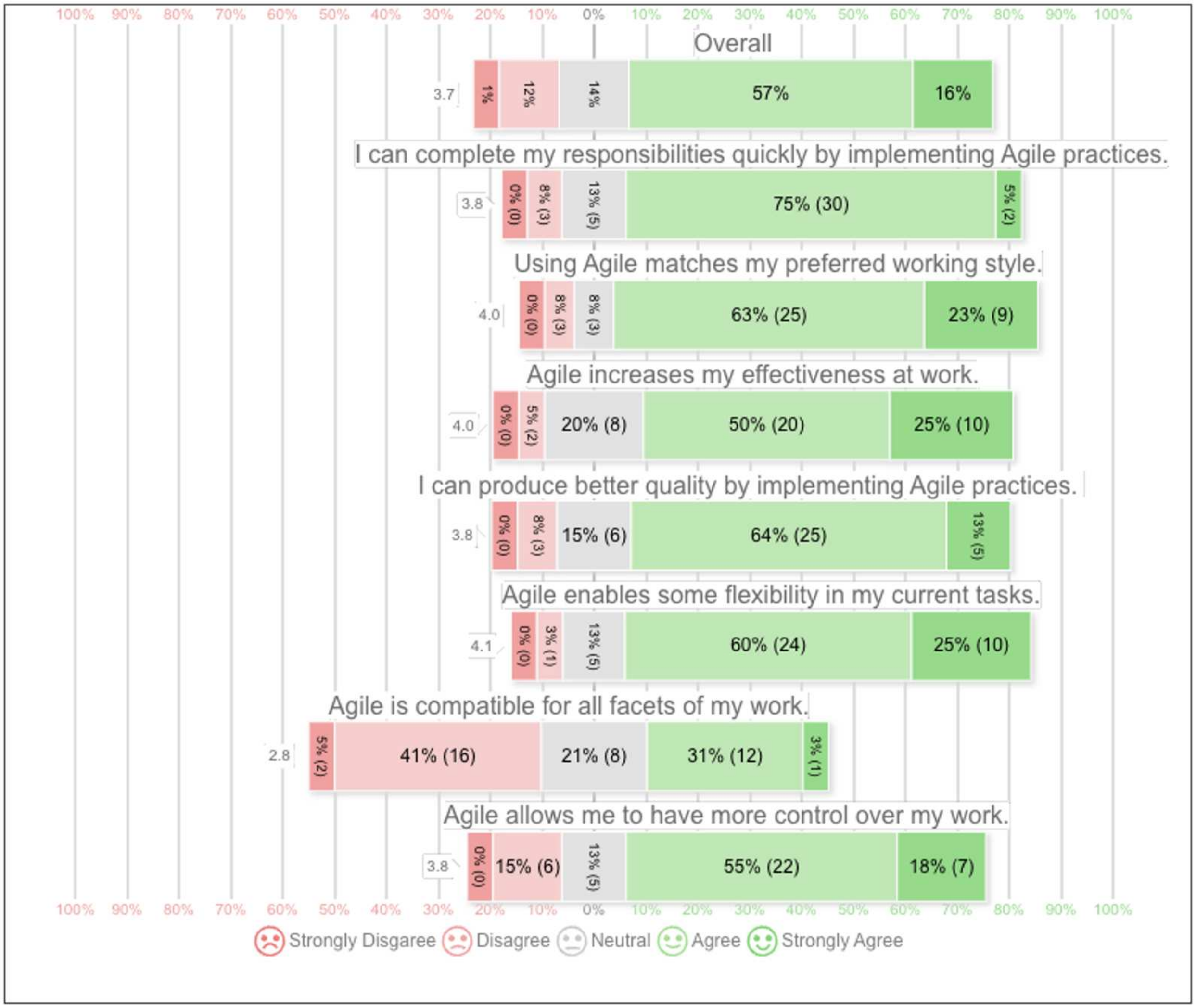


Figure 4. Perceptions of Implementing Agile Methodologies

The analysis in Figure 5 below presents survey results on the effectiveness and implementation of Agile methodologies observed among the participants. A majority 78% of participants believe that Agile positively impacts their work. Notably, 62% of participants agree/ strongly agree that using proper tools facilitates effective Agile usage. However, there are concerns regarding training, as 38% of participants feel that their management did not provide adequate training for Agile practices. These findings highlight the general effectiveness of Agile practices, while also pointing out areas for improvement, particularly in management training and tool utilization, to further optimize Agile implementation.

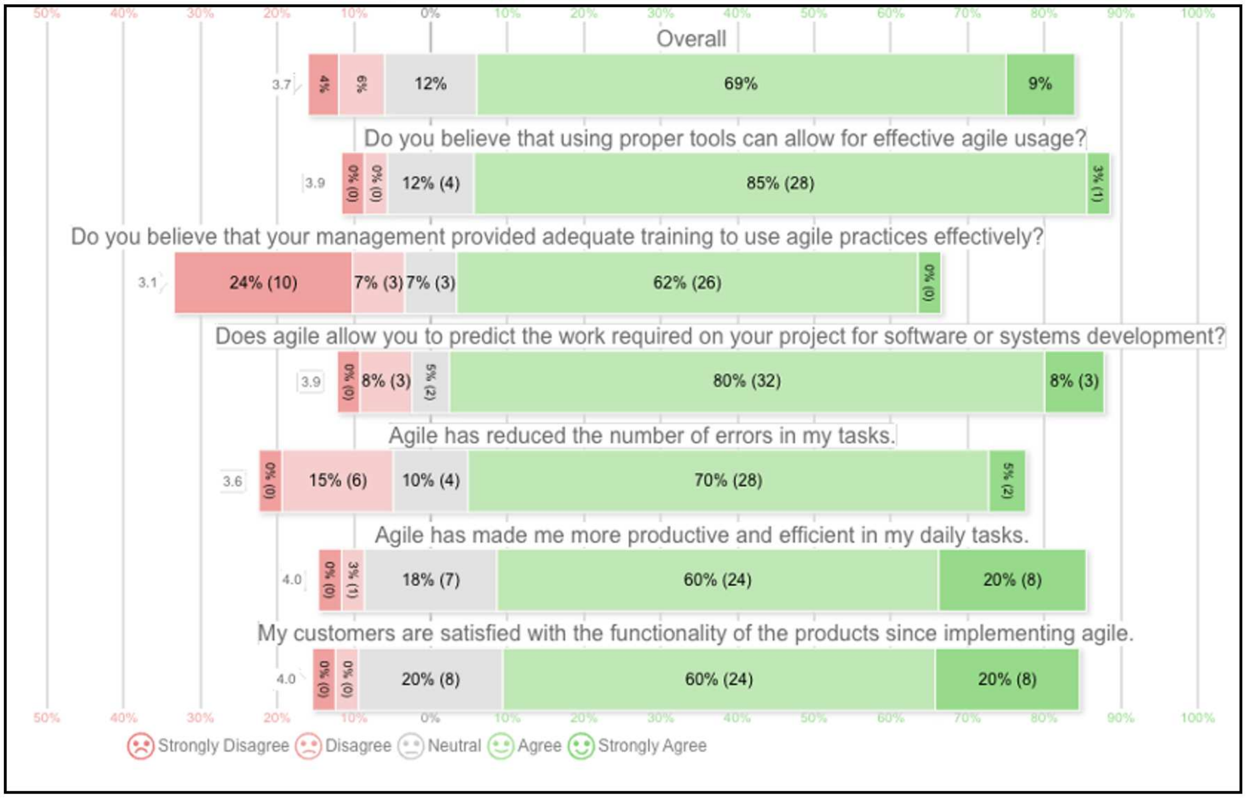


Figure 5. Perceived Impact of Agile Methodologies on Project Outcomes

The data in Figure 4 and Figure 5 were further analyzed to compare perceptions between Agile and Waterfall adoption within teams (as described in the section above “Adoption of Agile, Hybrid, And Waterfall”), where participants who were mostly practicing Agile (< 50) and mostly practicing Waterfall (> 50) were compared. Note, these questions were on a 5-point Likert scale of agreement, where responses above 2.5 are favorable towards agreement and below 2.5 represent disagreement towards the statement as shown in Table 3. The survey results highlight a generally positive attitude toward Agile practices among employees across different teams, regardless of whether they primarily adopt Agile or Waterfall methodologies. Employees who mostly practice Agile tend to rate Agile's effectiveness slightly higher on certain statements, such as "Using Agile matches my preferred working style" (mean = 4.12, SD = 0.64) and "Agile allows me to have more control over my work" (mean = 4.12, SD = 0.64), compared to those mostly using Waterfall who rated these statements at 3.91 (SD = 0.79) and 3.52 (SD = 1.03), respectively. This suggests that Agile adopters may feel a stronger alignment between Agile practices and their work style, as well as a greater sense of control over their tasks.

Across both groups, statements regarding the flexibility and productivity enabled by Agile received favorable ratings, with the overall mean for “Agile enables some flexibility in my current tasks” at 4.00 (SD = 0.67) and “Agile has made me more productive and efficient in my daily tasks” at 3.94 (SD = 0.71). These high ratings indicate broad agreement on Agile's role in enhancing task flexibility and efficiency, even among those who primarily practice Waterfall.

Interestingly, while both groups reported positive experiences with Agile, differences emerge in perceptions of training and compatibility. Participants primarily practicing Agile rated "Management has provided adequate training to use Agile practices effectively" slightly lower (mean = 3.00, SD = 0.92) than those mostly practicing Waterfall (mean = 3.39, SD = 0.99), which may reflect a perceived need for more formalized Agile training among Agile-focused teams. Additionally, the statement “Agile is compatible for all facets of my work” received mixed ratings, with a lower mean among mostly Waterfall teams (mean = 2.73, SD = 1.00), suggesting that these teams may face challenges integrating Agile into certain aspects of their work. Overall, the survey results underscore the positive reception of Agile practices across different experience levels, though there are areas particularly in training and full compatibility where further support may enhance Agile adoption and effectiveness in the workplace.

Table 3. Perceived Impacts (5-point Likert scale) of Agile based on Experience Level

Agile Impact Statement	**Overall** *Mean (SD)*	**Mostly Agile** *Mean (SD)*	**Mostly Waterfall** *Mean (SD)*
I can complete my responsibilities quickly by implementing Agile practices	3.72 (0.65)	3.62 (0.74)	3.78 (0.67)
Using Agile matches my preferred working style	3.92 (0.77)	4.12 (0.64)	3.91 (0.79)
Agile increases my effectiveness at work	3.94 (0.82)	4.00 (1.07)	3.95 (0.82)
I can produce better quality by implementing Agile practices	3.78 (0.79)	3.75 (1.03)	3.83 (0.72)
Agile enables some flexibility in my current tasks	4.00 (0.67)	4.12 (0.35)	4.04 (0.63)
Agile is compatible for all facets of my work	2.86 (1.01)	3.00 (0.92)	2.73 (1.00)
Agile allows me to have more control over my work	3.77 (0.93)	4.12 (0.64)	3.52 (1.03)
Using proper tools allow for effective agile usage	4.11 (0.57)	4.50 (0.76)	3.95 (0.47)
Management has provided adequate training to use agile practices effectively	3.28 (0.97)	3.00 (0.92)	3.39 (0.99)
Agile predicts the work required in the project	3.86 (0.68)	3.87 (0.99)	3.91 (0.51)
Agile has reduced the number of errors in my tasks	3.61 (0.84)	3.37 (1.06)	3.78 (0.67)
Agile has made me more productive and efficient in my daily tasks	3.94 (0.71)	3.87 (0.64)	4.00 (0.60)
My customers are satisfied with the functionality of the products since implementing agile	3.97 (0.65)	4.00 (0.53)	4.04 (0.70)

The analysis presented in Table 3 provides a nuanced view of survey results concerning the effectiveness and implementation of Agile methodologies within the organization. The data reveal several key insights into the participants' perceptions and experiences with Agile practices, shedding light on both the strengths and potential areas for improvement in the Agile adoption process. These findings collectively highlight the overall effectiveness of Agile methodologies, as perceived by the participants. However, the mixed responses concerning training and tool utilization signal areas that require further attention. To optimize Agile practices, it is crucial to address the identified training gaps by enhancing training programs to ensure comprehensive preparation for all employees. Additionally, the emphasis on proper tool utilization suggests a need for ongoing investment or refinement in technological support to better align with Agile workflows. By addressing these areas, the organization can enhance the effectiveness of Agile implementation and fully leverage its benefits.

These results provide valuable insights into the current state of Agile methodologies within the organization. The positive overall perception of Agile, combined with concerns about training and tool utilization, offers a foundation for targeted improvements. By focusing on enhancing training programs and optimizing tool use, the organization can better support Agile practices and achieve more effective project management outcomes.

3.3.5. Agile Implementation on Teams

Participants were asked what Agile methods they actively use, allowing them to select all that apply. The survey results reveal a clear preference for Scrum and Kanban among participants, as shown in Figure 6, highlighting their perceived effectiveness in managing complex, high-stakes projects. This question was structured as multiple-choice, allowing respondents to select all that they actively use. Scrum emerged as the most widely used methodology, with 77.5% (N = 31) of participants selecting it, reflecting its structured yet flexible framework, which is particularly valuable in project environments requiring iterative sprints and continuous improvement. Scrum's approach to breaking down large tasks into manageable increments aligns well to focus on rigorous project management and quality control. Additionally, Scrum's emphasis on team roles and regular feedback loops helps maintain alignment and ensures swift adjustments to meet project demands. Agile Modeling followed closely, with 55% (N = 22) of respondents indicating its use. Agile

Modeling emphasizes visual management and a continuous workflow, facilitating real-time adjustments and promoting transparency, a critical need in regulated, high-visibility sectors such as aerospace and defense. This methodology supports the need for precise tracking and adaptability, making it an effective choice for project managers who must balance innovation with regulatory compliance.

Notably, participants reported using hybrid approaches that combine elements from different Agile methodologies, such as Scrum and Extreme Programming (XP). This adaptability speaks to Agile's inherent flexibility and participants' ability to tailor practices to the specific demands of their projects. For instance, incorporating XP's emphasis on technical practices, like test-driven development, into Scrum's iterative framework can enhance code quality and stability, addressing the stringent standards often required in defense contracts.

The survey revealed that participants employed hybrid approaches, integrating elements from various Agile methodologies such as Scrum and Extreme Programming (XP). This hybridization demonstrates the flexibility and adaptability inherent in Agile practices, allowing teams to customize their methodologies to align with the specific needs and challenges of their projects. For instance, combining Scrum's structured sprints with XP's focus on technical excellence can create a robust framework that addresses both project management and technical requirements effectively. This tailored approach enables teams to optimize their methodologies to achieve better alignment with project goals and enhance overall project performance. The survey results reflect a strategic alignment between Agile methodologies and the needs of high-stakes projects. The combination of Scrum, Kanban, Agile Modeling, Lean practices, and hybrid approaches provides a comprehensive toolkit for managing complex projects effectively. These methodologies collectively contribute to enhanced project performance, flexibility, and responsiveness, demonstrating their value in addressing the organization's project management challenges. The preference for Scrum and Kanban, coupled with the use of Agile Modeling and Lean practices, underscores a sophisticated approach to Agile implementation. The incorporation of hybrid methodologies further illustrates the organization's commitment to optimizing project management practices and adapting Agile principles to achieve superior project outcomes.

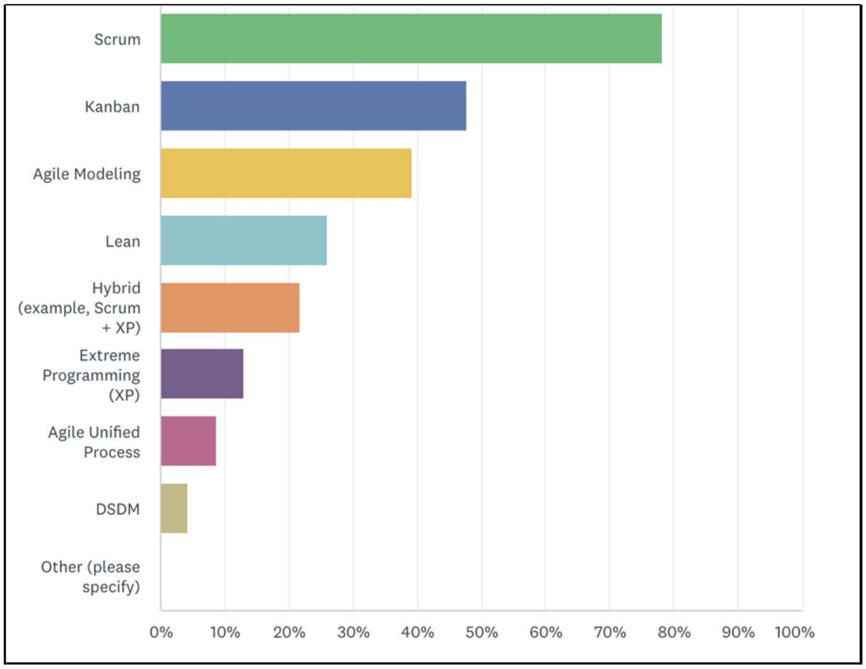


Figure 6. Most Popular Agile Methods Used Among Survey Participants

The survey also highlights a range of management tools used by professionals to ensure efficient workflow and project management. Participants were able to select multiple answers. The data reveals preferences for specific tools, indicating their perceived effectiveness and usability. Figure 7 highlights the prevalent use of various management tools within the company. Atlassian tools (82.5%) and Kanban boards (80%) are the most widely used, reflecting their effectiveness in supporting Agile project management through visualization, collaboration, and comprehensive tracking. GitLab is also used by 30% of participants for source code management. Other tools include custom-developed internal tools (12.5%), unit test tools (15%), and spreadsheets (17.5%), indicating a mix of standardized and tailored approaches to project management. The multiple tools usage shows the company's commitment to leveraging technology to optimize workflows, improve communication, and ensure project success.

The variety of tools used demonstrates the company's commitment to leveraging technology to optimize project management workflows and enhance communication. The high adoption rates of Atlassian tools and Kanban boards highlight their effectiveness in supporting Agile project management, suggesting that these tools play a central role in facilitating iterative development and real-time progress tracking. The diverse tool usage underscores the need for tailored solutions while emphasizing the importance of ongoing optimization to achieve maximum effectiveness and efficiency in project management practices.

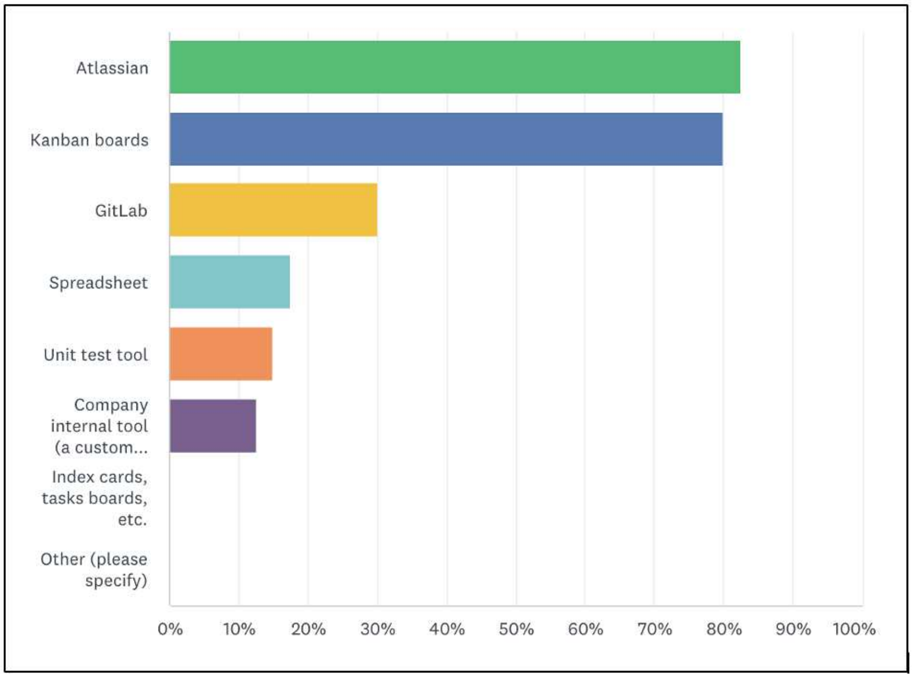


Figure 7. Most Widely Used Agile Tools Among Survey Participants

3.3.6. MBSE Integration

This section explores the extent to which MBSE has been integrated into Agile projects based on survey responses. Participants were asked whether they had integrated MBSE into any of their Agile projects. The results indicate a strong inclination towards the integration of MBSE in Agile projects, with a significant majority (78.3%) affirming its use. However, the remaining participants who have not integrated MBSE into their Agile projects suggest that there are still challenges and barriers to adoption. The results show the growing trend of integrating MBSE into Agile projects, driven by the complementary strengths of both methodologies. The results also indicate that a significant majority of participants find MBSE to be effective (74.36%) in Agile environments. The survey highlights that MBSE is viewed as an effective method for capturing and managing requirements in Agile projects. Participants rated MBSE implementation on their teams on a scale from 0 to 100, with a mean score of 55.82 and standard deviation of 16.38. This variability suggests that while some teams have successfully integrated MBSE into their Agile practices, others are still navigating the complexities of adoption. Tailoring implementation strategies and providing targeted support can help address these disparities and enhance overall effectiveness.

The strong inclination towards integrating MBSE into Agile projects highlights a growing recognition of the value that MBSE brings to Agile environments. The ability of MBSE to provide a structured approach to capturing and managing complex requirements complements Agile’s iterative and adaptive processes. The fact that 78.3% of participants have integrated MBSE suggests that many teams are leveraging its capabilities to enhance their Agile practices, particularly in areas such as requirements management, system modeling, and verification. Despite this positive trend, the remaining participants who have not integrated MBSE into their Agile projects indicate that challenges and barriers to adoption still exist. These obstacles may include issues related to tool compatibility, the complexity of MBSE methodologies, or resistance to change within teams accustomed to traditional practices. Addressing these barriers will be crucial for further adoption and effective utilization of MBSE in Agile contexts.

3.3.7. Participants' Comments on MBSE and Agile Implementation

The survey gathered insightful comments from participants on the integration of MBSE and Agile methodologies, highlighting their impact on workflow, efficiency, and team dynamics. A summary of these comments is provided below, corresponding to themes.

Enhanced Workflow and Efficiency: Many participants noted that incorporating MBSE and Agile significantly enhanced their workflow and efficiency. They reported that MBSE enabled the creation of precise, visual models that improved system understanding and facilitated better communication with stakeholders. This reduction in ambiguities and rework, coupled with Agile's iterative development and continuous feedback, helped teams stay adaptable and aligned with evolving requirements.

Challenges with Agile Implementation: Some participants expressed concerns about the proper implementation of Agile. They pointed out that resistance from management, who prefer traditional Waterfall processes, can hinder Agile's effectiveness.

Impact of Scrum Practices: Opinions were mixed regarding Scrum practices. While some participants acknowledged that Scrum processes improved productivity and communication, others found daily scrums overbearing and counterproductive. This suggests a need for balance in implementing Scrum to avoid potential negative impacts on team morale and efficiency.

Use of Additional Tools: Participants mentioned the use of various tools such as Tableau, Power BI, and Oracle in their Agile processes. These tools complemented MBSE and Agile methodologies by facilitating data analysis and project management tasks.

Role of Leadership and Training: The importance of experienced and pragmatic leadership in using MBSE tools effectively was emphasized. Knowing when to use specific tools and when to avoid them is crucial for successful Agile implementation. Additionally, several participants highlighted the need for more initial training in Agile methodologies. They believed that better training would have given their teams a clearer understanding of Agile's benefits and how to implement it effectively.

Quality and Integration Risks: Some participants pointed out that while Agile can speed up certain engineering processes, the quality of the output heavily depends on the skills of the individuals involved. They noted that Agile teams might introduce risks into complex integration efforts. These risks often manifest later in the project lifecycle, potentially leading to costly fixes. The complexity of projects can make Agile less efficient compared to simpler, faster-paced environments like software development.

Overall, the comments reflect a refined perspective on the integration of MBSE and Agile methodologies. While participants recognize the benefits of these approaches in improving workflow and efficiency, they also highlight significant challenges and areas for improvement, particularly in management support, training, and balancing Agile practices to suit complex project environments.

3.4. Discussion

The survey results highlight the acceptance and implementation of Agile methodologies. In companies where projects often involve intricate systems and interdependencies, Agile modeling helps teams visualize and manage complexity effectively. This is supported by the literature, where Agile methodologies such as Scrum and XP, initially designed for smaller projects (Kahkonen, 2004; Beck, 2000; Paasivaara et al., 2016), are increasingly being implemented in large-scale system development organizations (Salo & Abrahamsson, 2008). Participants from diverse roles, including product owners and engineers, generally reported improved efficiency, flexibility, and customer satisfaction with Agile practices. However, our survey respondents, as well as others in the literature such as Palmquist et al. (2013), describe challenges that remain in balancing Agile's iterative processes with MBSE's upfront modeling requirements. The adoption and success of MBSE may be hindered by a lack of widespread understanding within the systems engineering community regarding its precise definition. This ambiguity often leads to ineffective implementations. Early modeling, particularly in MBSE, plays a critical role in iterative development by enabling simulations that refine and validate requirements throughout the project life cycle. These simulations not only facilitate early detection of issues but also enhance the adaptability of the requirements definition process, ensuring alignment with Agile's iterative approach. By leveraging MBSE’s capabilities for continuous validation, organizations can streamline requirement development and reduce rework, thereby supporting project efficiency rather than hindering it.

The survey results regarding participants’ experience with both Agile and Waterfall methodologies highlight critical insights into the challenges of transitioning from Waterfall to Agile in the A&D sector. This range of experience levels from a few years to several decades indicates that organizations often find themselves in a hybrid state, caught between the established, document-heavy Waterfall approach and the iterative, collaborative Agile methodology. Participants with extensive Waterfall experience (over 15 years, up to 40 years) may naturally gravitate towards familiar processes, making them hesitant to fully embrace Agile principles. This entrenchment creates significant roadblocks when trying to foster Agile mindsets within teams. Waterfall’s rigid structure can be comforting for those accustomed to it, but its inflexibility hampers the adaptability required for modern systems engineering, especially when managing rapidly evolving requirements typical of A&D projects. Conversely, participants with Agile experience of 2 to 20 years provide insight into Agile’s benefits but may still face institutional resistance from those more comfortable with Waterfall practices. The survey reveals that, despite the push towards Agile, many organizations still operate within a hybrid framework. In such settings, teams are often required to adhere to both Waterfall’s hierarchical documentation requirements and Agile’s iterative, flexible cycles, which can lead to conflicts in project goals, resource allocation, and decision-making.

The hybrid approach is particularly detrimental because it can diminish the full potential of Agile methodologies. Agile’s strength lies in its ability to enhance collaboration, responsiveness, and iterative problem-solving. However, when teams are forced to toggle between Waterfall’s linear stages and Agile’s iterative sprints, project timelines can be delayed, communication between teams may become fragmented, and innovation stifled. This is supported in Davis (2012), who reported that the presence of different mindsets within the same organization can further exacerbate these issues, leading to misaligned priorities, increased inefficiencies, and a lack of cohesion across teams.

Ultimately, the data suggests that transitioning fully to Agile methodologies in A&D requires not just training in Agile principles, but a cultural shift that embraces Agile’s inherent flexibility. Failure to do so risks the organization becoming "stuck" in a hybrid state, which can lead to a suboptimal combination of both methodologies, causing confusion and limiting the benefits that Agile could provide to improving systems engineering and project management outcomes. Organizations must recognize these challenges and invest in a deliberate and systematic shift toward Agile practices, breaking away from legacy systems that may no longer serve the needs of complex modern projects.

In A&D organizations transitioning from Waterfall to Agile, the hybrid project management environment can pose significant challenges. While Agile practices offer increased flexibility, autonomy, and iterative progress, their coexistence with traditional methodologies such as Waterfall can cause friction. Agile thrives in environments where quick iterations, adaptability, and collaborative processes are valued. However, Waterfall's linear, document-driven, and often rigid nature can inhibit the full potential of Agile's iterative benefits. The result is that employees can struggle to seamlessly switch between Agile's fast-paced, adaptive approach and Waterfall's highly structured phases.

In organizations where Agile is layered onto existing Waterfall processes without full integration, the hybrid setup can lead to operational inefficiencies. For example, employees might be expected to generate comprehensive documentation (typical of Waterfall) while simultaneously performing iterative updates (characteristic of Agile), causing task duplication and a slowdown in project velocity. Additionally, team members may experience confusion regarding their roles and responsibilities, as Agile encourages shared ownership and rapid decision-making, which contradicts the hierarchical, approval-heavy processes of Waterfall.

Further complicating this hybrid dynamic is the notion that Agile is not universally applicable to all work facets. This limitation can contribute to operational bottlenecks, especially in areas like systems engineering, where strict compliance to protocols and safety standards is non-negotiable. Aerospace and defense projects often involve long-term commitments, and the emphasis on predictability, thorough documentation, and risk management may make certain tasks less suitable for Agile's fast iterations. This mismatch can hinder Agile's full benefits, leaving some aspects of the project stuck in inefficient workflows.

Without proper management and a clear strategy for balancing Agile and Waterfall practices, the hybrid approach can detract from overall project effectiveness. Employees might not fully buy into Agile's value if they find it burdensome to integrate into a predominantly Waterfall culture, leading to resistance, diminished productivity, and failed Agile implementation. For A&D companies, moving beyond the hybrid state and fostering a culture that supports Agile methodologies will be essential to achieve seamless project management and drive innovation in complex systems development.

Tools such as Atlassian, Kanban boards, and GitLab play a critical role in facilitating Agile processes. However, the survey revealed a need for improved training programs, as many participants feel inadequately trained. The findings align with existing research, indicating the importance of tool integration and user training for effective Agile implementation (Azizyan et al., 2011).

The survey results also indicate that a segment of participants expressed resistance to Agile methodologies, reflecting a preference for traditional Waterfall practices. This resistance often stems from a comfort with the structured, sequential nature of Waterfall, which provides clear roadmaps and predictability. Individuals who prefer Waterfall methodologies typically value its thorough upfront planning and well-defined stages, which they perceive as essential for managing complex projects with stringent requirements and regulatory standards. The hybrid approach is prevalent, reflecting the need to balance Agile's flexibility with Waterfall's control and predictability, essential for regulatory compliance (VanderLeest & Buter, 2009).

Despite the benefits of Agile, the survey underscores the necessity for tailored approaches to fit unique project demands and organizational cultures. MBSE integration within Agile frameworks is viewed positively, enhancing requirement traceability, early validation, and stakeholder communication, despite some challenges in balancing iterative development with upfront modeling (Noguchi et al., 2020; Biggs et al., 2024). While the integration is substantial, continued efforts are needed to address the challenges and barriers to broader adoption. Costly tool licenses are purchased, and engineers are trained on a modeling language and tool, with these engineers being designated as "modelers" on the development team. Development continues as usual, with the modelers documenting architecture and design decisions after they are made. Occasionally, the model helps identify potential issues, but it's mostly used for recording decisions rather than as a design tool. Over time, the model's use declines as it hinders more than it helps, leading to the conclusion that MBSE is a passing trend and not a solution for integration and quality challenges. The question arises about what defines a successful MBSE approach that can truly benefit an organization. Delligatti builds upon the INCOSE definition and highlights three core components of MBSE: a modeling language, a modeling tool, and a modeling methodology (Delligatti, 2013). A modeling methodology is crucial as it outlines the objectives of the modeling effort and specifies the tasks and standards to be followed in creating the model. This emphasizes that simply having a system model does not automatically translate into value for an organization, nor does it represent a genuine MBSE approach. The real value is derived from the systems engineering processes employed and the actual development of the model (Haskins, 2011). By overcoming these obstacles, the company can fully leverage the benefits of MBSE and Agile, optimizing their project management processes and improving outcomes in the complex and regulated environment (Sundaram & Brownlow, 2018).

In terms of project performance, 55.3% of participants in our study noticed improvements with MBSE in Agile projects, though the benefits varied depending on project circumstances. Literature supports these findings, showing quantitative benefits of integrating MBSE within Agile, including better estimation reliability, productivity, and lower defect rates (Huss et al., 2023). Qualitative feedback further underscores the need to address cultural and structural barriers to Agile adoption (George, 2023). Overall, our survey data align with existing literature, emphasizing the advantages of Agile methodologies and the need for continued support and training to optimize project management practices. Participants discussed how Agile methodologies and MBSE have been integrated into their projects to enhance development efficiency and collaboration. For example, in "Project X," bi-weekly sprint planning, daily stand-ups, and continuous integration/continuous deployment facilitated iterative progress and rapid feedback, while MBSE ensured comprehensive requirement traceability and early validation through SysML models.

Despite initial implementation delays, significant successes included improved team collaboration and product quality. Another example highlighted the challenges of a hybrid Agile-Waterfall approach, where contracts followed Waterfall, but actual work used Atlassian tools. This hybrid approach led to delays, indicating a pure Agile framework might have expedited product acceptance. Another study by Kusters et al. (2017) examines the challenges and risks that arise when integrating Agile and traditional development methods within hybrid organizations, particularly regarding their impact on coordination and collaboration. Combining these approaches is notably difficult for many organizations due to the differing cultures and conditions inherent to each method, despite their respective strengths and benefits. Understanding these challenges is crucial for effectively managing the integration process. This study aims to identify and validate a comprehensive overview of such issues (Kusters et al., 2017).

Various projects demonstrated Agile's effectiveness in managing requirements and adapting to evolving needs, with practices like weekly scrums and bi-weekly sprints keeping teams organized and stakeholders informed. Transitioning to Agile often required adjustments in team mindset and approach. Initial challenges were overcome by collaborative modeling and real-time feedback, fostering continuous improvement and better adaptability.

The integration of Agile and MBSE consistently led to enhanced project outcomes, despite the difficulties faced with hybrid methodologies. The survey responses underscore the benefits of fully embracing Agile practices for improved collaboration, product quality, and adaptability in project management. The hybrid methodology combining MBSE and Agile approaches in managing complex systems can be detrimental due to integration challenges, communication barriers, and resource allocation issues. The differing principles of MBSE and Agile can lead to conflicts and misalignment, while continuous communication between analysts and model developers might cause misunderstandings and delays. Balancing resources and ensuring consistent quality can be difficult, and managing stakeholder expectations can become complex. These potential drawbacks need careful management to avoid negatively impacting the project's success (Power et al., 2021).

3.4.1. Methodological Implications of Survey Sample

3.4.1.1. Sample Size Adequacy

The survey component of this research included 40 participants from an aerospace and defense organization. The sample size warrants discussion regarding statistical power and inferential strength. To evaluate the adequacy of the survey sample size for inferential comparisons, an a priori power analysis was conducted using conventional assumptions in organizational research. For a two-tailed independent-samples t-test comparison with $\alpha = 0.05$ (Fisher, 1925) and desired power of 0.80 (Cohen, 1988), the required per-group sample size can be approximated as:

$$n_{per\ group} = \frac{2(Z_{\frac{\alpha}{2}} + Z_{\beta})^2}{d^2} = \frac{2(1.96 + 0.84)^2}{d^2} = \frac{15.68}{d^2}$$

Where $Z_{\alpha/2} = 1.96$, corresponding to the 97th percentile of the standard normal distribution for a two-tailed $\alpha = 0.05$ test, and $Z_{\beta} = 0.84$, corresponding to the 80th percentile of the standard normal distribution and reflecting 80% statistical power. This experimental design compares two groups: Agile users and Waterfall users.

For a medium effect size ($d = 0.50$), $d^2 = 0.25$, resulting in $n_{\text{per group}} = 15.68 / 0.25 = 62.72$, which rounds up to approximately 63 respondents per group (approximately 126 total for two balanced groups). For a small effect size (d = 0.30), $d^2 = 0.09$, resulting in $n_{\text{per group}} = 15.68 / 0.09 = 174.22$, which rounds up to approximately 175 respondents per group (approximately 350 total). The values 0.30 and 0.50 come from Cohen's (1988) conventional definitions of small and medium standardized effect sizes commonly used in power analysis.

Given the observed sample size in this study, statistically non-significant results should be interpreted cautiously because the study is underpowered to reliably detect small effects and may also be moderately underpowered for medium effects. Consequently, the survey findings are best interpreted as identifying directional patterns within the studied context.

3.4.1.2. Implications of Small Sample Size

With a total of 40 participants, the survey provides meaningful insight into perceptions of Hybrid MBSE within the organization, but it also carries inherent statistical limitations. A smaller sample size reduces the ability to detect subtle differences in perception and increases the likelihood that non-significant findings may reflect limited statistics. Individual responses also carry more proportional weight in a smaller dataset, which can amplify variability. Additionally, when the data are segmented by role, experience level, or functional domain, subgroup sizes become quite small, limiting the reliability of comparative interpretation. For these reasons, the survey findings should be interpreted as identifying directional trends and perception patterns. The purpose of the survey was not broad statistical generalization across the aerospace and defense industry, but structured insight within a defined sustainment environment.

3.4.1.3. Implications of Unbalanced Sample Distribution

The composition of the participants pool also warrants consideration. Mid-career engineers were more heavily represented than early-career personnel or senior technical leaders. This imbalance may influence how the results are interpreted. Individuals closer to day-to-day execution often experience governance processes differently than senior leaders responsible for compliance oversight and enterprise risk management. As a result, the survey may reflect operational-level friction points more strongly. This does not invalidate the findings, but it does shape their lens. Future research would benefit from stratified sampling approaches that intentionally balance tenure, role, and authority level, thereby strengthening confidence that observed patterns reflect organization-wide experience.

3.5. Conclusion

The objective of this chapter was to identify challenges, limitations, and opportunities in the transition from Waterfall to Agile methodologies in the A&D sector from actual experiences from industry, with a focus on the role of MBSE in this transition. The survey results provide a comprehensive overview of the current state and perceptions of integrating MBSE in Agile at a prominent A&D company. The findings indicate a strong preference and positive reception towards Agile practices among industry professionals with a significant number of participants expressing favorable views. Agile methodologies are praised for their flexibility, improved efficiency, and enhanced collaboration, which align well with the dynamic needs of projects. Participants reported varying levels of experience with Agile practices, with the majority having substantial experience that contributes to an understanding of its benefits and challenges. Those with extensive experience, such as 20 years, provide valuable insights into the long-term application and evolution of Agile methodologies. Conversely, the experiences with Waterfall practices varied widely, with some participants indicating over 40 years of experience. This diversity underscores the historical dominance of Waterfall in the company and its continued relevance, particularly for projects with well-defined requirements and stable environments. However, the rigidity and longer development cycles of Waterfall pose significant challenges in the fast-paced environment at the company, as reflected in participant feedback. The findings highlight a critical tension between traditional and emerging methodologies, emphasizing the need for adaptability and innovation in project management.

The integration of MBSE into Agile projects has been notably effective in capturing and managing requirements, with 74.36% of participants finding it very effective or effective. This highlights MBSE's role in enhancing the precision and traceability of project requirements, thereby improving overall project outcomes. Early-stage modeling and simulation, as supported by MBSE, are particularly beneficial in facilitating iterative design and reducing rework, contributing to time-to-market improvements.

The diversity in participant experience with Agile methodologies ranging from those with extensive experience of up to 20 years to those with varying levels of engagement provides valuable insights into both the benefits and challenges associated with Agile adoption. The substantial experience of many participants contributes to a nuanced understanding of Agile's impact and evolution over time. Conversely, the broad range of experience with Waterfall methodologies, some extending over 40 years, highlights the historical significance and ongoing relevance of Waterfall practices, particularly for projects characterized by well-defined requirements and stable conditions. However, the rigidity and extended development cycles of Waterfall methodologies present notable challenges in the fast-paced and adaptive environment. This tension between the traditional Waterfall approach and the more dynamic Agile practices reflects a broader shift within the company towards embracing Agile principles while grappling with the limitations of established methodologies. The integration of MBSE into Agile projects has emerged as a particularly effective strategy for managing and capturing requirements.

In summary, the survey reveals a clear trend towards adopting Agile methodologies, supported by the effective integration of MBSE practices. This shift reflects a broader industry movement towards more adaptive and responsive project management approaches. The findings address the importance of continued training and organizational support to maximize the benefits of Agile and MBSE, addressing the challenges of transitioning from traditional Waterfall practices. Specifically, leadership buy-in and cultural transformation are essential factors for fostering a successful transition.

However, a limitation of this is often budget and further research into the financial impact and scalability of these methodologies in complex systems would be valuable. This study focused on perceptions of employees on their teams' transitions from Waterfall to Agile, and future research could survey additional companies to get a more diverse representation of experiences and approaches. Alternatively, while employee buy-in and perspectives are important elements to successful transitions towards Agile, an evaluation of case studies could provide valuable insight into quantifying the benefits of completely Agile, completely Waterfall, or a hybrid approach in project outcomes.

In conclusion, the purpose of this chapter was to explore how MBSE supports the transition from Waterfall to Agile methodologies and its impact on project management processes, efficiency, and time-to-market in the A&D sector. As companies evolve, embracing these methodologies will be crucial for maintaining competitiveness and achieving strategic objectives.

Chapter 4. Aim 1: Practitioner Perspectives - Interviews

4.1. Introduction

The results of the survey study from Research Aim 1 revealed an unexpected but important pattern. Rather than serving only as a temporary step toward full Agile adoption, Hybrid development methodologies emerged as a prominent and sustained approach in practice. As such, an interview study was performed to better explore this Hybrid approach and serve as information gathering for Research Aims 2 and 3. The qualitative insights from these interviews position Hybrid methodologies as an enabling mechanism for methodological change, highlight areas where Agile practices require adaptation, and help identify gaps in current guidance that inform future research directions related to hybrid adoption, MBSE integration, and organizational readiness.

4.2. Methods

Between March and September 2024, interviews with 25 participants were conducted with industry practitioners from the same company as in the surveys of Research Aim 1. The interview questions are listed in Appendix B. All interviews were conducted virtually to accommodate schedules. A consistent set of open-ended questions was used across all interviews (listed in Appendix B), with the flexibility to ask follow-up questions when clarification or deeper insight was needed.

Each interview lasted approximately 60 minutes and focused on participants' real-world experiences with Waterfall, Agile, and Hybrid development approaches, as well as the use of

MBSE in complex programs. Participants were intentionally selected to represent a range of technical and leadership perspectives, including systems engineers, software engineers, project engineers, and senior technical and program leads. Many of the interviewees held leadership or decision-making roles and provided insight not only from an execution standpoint but also from a management and organizational perspective.

The interviews were designed to complement the quantitative survey data by capturing practical challenges, lessons learned, and organizational factors influencing methodology adoption particularly during transitions from Waterfall to Agile. This input added important context to the survey findings and enabled a richer understanding of how MBSE and development methodologies are applied in practice.

4.3. Results

4.3.1. Challenges in Transitioning from Waterfall to Agile

A recurring theme in the interviews was the difficulty of transitioning from Waterfall to Agile, particularly for organizations with deeply entrenched practices rooted in Waterfall methodologies. Participants frequently noted resistance from management, who were accustomed to Waterfall's detailed planning, rigid structures, and comprehensive visibility into project timelines. As one participant observed, "Transitioning from Waterfall to Agile can be challenging, especially when senior management is accustomed to Waterfall's visibility and detailed planning." This sentiment reflects the need for organizations to address entrenched mindsets and adapt their processes to embrace the iterative and flexible nature of Agile.

Leadership buy-in emerged as a critical factor for overcoming these challenges. Several participants highlighted the importance of leadership in driving cultural change, with one emphasizing, "We need a cultural shift, driven by senior leaders." Others noted that insufficient leadership engagement created obstacles to the effective implementation of Agile practices, stating that leaders often failed to provide the necessary support and advocacy for change. One participant underscored this need, stating, "Need more leadership buy-in. We need to embrace change, and leaders need to drive it."

Another significant challenge was the lack of adequate preparation for teams transitioning to Agile. Many participants expressed concerns about the insufficient training and resources provided during the transition. One remarked, "We need more initial training for the entire team in Agile before implementation. Better understanding of Agile could have been achieved rather than jumping straight into it." This feedback highlights the importance of equipping teams with the knowledge and skills necessary to navigate Agile methodologies successfully. Without proper training, teams may struggle to fully adopt Agile practices, leading to inefficiencies and reduced confidence in the approach.

4.3.2. Benefits of Agile for Managing Complex Projects

Despite the challenges, participants overwhelmingly recognized the benefits of Agile for managing complex projects. Agile's iterative and adaptable nature was praised for enabling teams to respond to changes quickly and effectively. One participant highlighted this advantage, stating, "For managing complex projects, Agile is preferred due to its adaptability and iterative nature." Compared to Waterfall's rigid, linear structure, Agile's flexibility allows teams to break down tasks into manageable increments, refine their processes based on feedback, and adjust their priorities as project requirements evolve.

Agile's focus on iterative development, sprints, and backlogs was also seen as a significant advantage. One participant explained, "Agile allows for more flexibility with its backlog and sprints, enabling teams to switch tasks and adapt to changes, unlike Waterfall, which is constrained by a single, detailed plan." This flexibility was particularly valuable for addressing the uncertainty and complexity often encountered in large-scale projects. Participants also noted that Agile fostered better communication and collaboration among team members, with some crediting Scrum practices for improving productivity. However, a few participants voiced concerns about daily scrums, describing them as potentially overbearing and counterproductive if not managed effectively.

The ability of Agile to enhance the quality of project outcomes was another key theme. One participant remarked, "Agile combined with MBSE is a game changer," emphasizing how Agile's adaptability, when paired with MBSE's focus on requirement traceability, can significantly improve project success rates. However, some participants cautioned that Agile practices could falter if not implemented correctly, with one noting, "Agile can work if properly implemented. However, resistance from some managers who insist on using Waterfall can render Agile ineffective." This underscores the importance of consistent and well-supported Agile implementation.

4.3.3. Integration of MBSE into Agile Frameworks

The interviews also explored the integration of MBSE into Agile practices, with many participants viewing this combination as a promising strategy for managing complex projects. MBSE was described as a tool that ensures comprehensive requirement traceability and supports iterative design development, aligning well with Agile's principles. One participant emphasized this alignment, stating, "We need to integrate MBSE into Agile projects to create a model that supports iterative development and rapid prototyping, making it possible to test and refine designs efficiently."

While participants expressed enthusiasm for the potential of MBSE to enhance Agile workflows, they also highlighted the challenges associated with its adoption. Many stressed the need for experienced and pragmatic leadership to guide the integration process, with one participant noting, "MBSE tools need to be combined with experienced and pragmatic leadership." Leadership support was viewed as essential for ensuring that MBSE tools are used effectively and that their integration into Agile frameworks yields the intended benefits.

4.4. Conclusion

The interviews revealed several key recommendations for improving the transition from Waterfall to Agile and enhancing the integration of MBSE into Agile frameworks. First, organizations need to invest in comprehensive training programs to ensure that teams are well-prepared for the transition. These programs should cover both the technical aspects of Agile practices and the cultural changes required to support their adoption. One participant suggested that better initial training could have mitigated many of the challenges faced during the transition, stating, "Better understanding of Agile could have been achieved rather than jumping straight into it."

Second, leadership buy-in and engagement were identified as critical factors for success. Senior leaders must take an active role in driving cultural change, advocating for Agile practices, and providing the necessary resources and support. As one participant put it, "We need more leadership buy-in. Leaders need to embrace change and drive it." This level of commitment is essential for overcoming resistance and ensuring that Agile practices are implemented effectively.

Finally, the integration of MBSE into Agile frameworks should be approached strategically, with a focus on aligning MBSE tools with Agile principles. Participants emphasized the importance of combining MBSE with experienced leadership to maximize its potential, suggesting that this integration could serve as a "game changer" for managing complex projects. By leveraging the strengths of both MBSE and Agile, organizations can enhance their ability to manage complexity, ensure requirement traceability, and deliver high-quality outcomes.

The interviews provided valuable insights into the experiences of participants transitioning from Waterfall to Agile and integrating MBSE into their workflows. While challenges such as resistance to change, lack of training, and leadership disengagement were frequently cited, participants overwhelmingly recognized the benefits of Agile for managing complex projects. The integration of MBSE into Agile frameworks was seen as a promising approach, but its success hinges on strong leadership, comprehensive training, and strategic implementation. These findings underscore the importance of addressing cultural and organizational barriers to fully realize the potential of Agile and MBSE in complex project environments.

Chapter 5. Aim 2: Project Success Metrics

5.1. Introduction

This chapter examines how Waterfall, Agile, and Hybrid development methodologies influence sustainment strategies in aerospace projects, with particular focus on requirement change, cost and schedule performance, and workflow efficiency. The analysis draws on findings from the SILA system case study, which serves as a representative example of a sustainment-focused aerospace program. These perspectives provide both practical and organizational insights into how different methodologies support or hinder requirement adaptability, stakeholder alignment, and compliance within the A&D industry. The following research questions are addressed in this chapter:

- **RQ2.1:** How do Waterfall, Agile, and Hybrid approaches affect requirement changes, cost efficiency, and schedule performance in aerospace sustainment projects?
- **RQ2.2:** What change management and workflow bottlenecks result from rigid or poorly integrated requirement structures across the development approaches?
- **RQ2.3:** How effectively can a Hybrid approach optimize sustainment performance by balancing adaptability, regulatory compliance, and rework reduction?

The content of this chapter has been published in the *ITEA Journal of Test and Evaluation*, titled “Balancing structure and flexibility: evaluating agile, waterfall, and hybrid methodologies in aerospace and defense projects” (Gracias & Gallegos, 2025).

5.2. Methods

This study adopts a qualitative case study methodology to evaluate the use of three development approaches (Waterfall, Agile, and Hybrid) within the SILA system. SILA is a de-identified, real-world aircraft sustainment initiative that supports aging aircraft systems. By examining comparable subsystems developed using each approach, this study investigates how different development methodologies Waterfall, Agile, and Hybrid affect change management processes, cost, and schedule efficiency within complex A&D environments.

The SILA system presents unique challenges in requirements management, as it must ensure operational readiness while addressing part obsolescence, diminishing manufacturing sources, and evolving maintenance needs. These constraints require a development approach that balances stability with adaptability. As such, SILA exemplifies the organization's long-term commitment to sustaining in-service aircraft well beyond the closure of production lines. Its core objectives include proactively supporting maintenance through subsystem modeling, failure prediction, and preventive maintenance strategies to minimize unplanned aircraft downtime, as well as streamlining sustainment efforts by managing spare parts, maintenance workflows, and system updates to ensure continued operational capability and compliance with evolving mission requirements.

This research examines three distinct development approaches as applied to subsystems within SILA. Waterfall-based subsystems followed a sequential, documentation-intensive process in which all requirements were defined, reviewed, and approved upfront prior to design and implementation. In contrast, Agile-based subsystems were developed through iterative sprints, allowing requirements to evolve over time while emphasizing continuous integration and frequent stakeholder feedback throughout development. Hybrid subsystems combined elements of both methodologies, applying structured Waterfall practices to compliance-driven activities such as requirements baseline establishment, formal validation, and final system qualification, while leveraging Agile cycles for design elaboration, prototyping, and subsystem integration to enable iterative refinement.

The three classifications provide a direct comparison across development styles, supporting the identification of trade-offs and potential best practices for sustainment-focused projects operating in highly regulated environments.

5.2.1. Research Design

A case study design was chosen for its effectiveness in providing rich, context-specific insights into complex engineering and organizational dynamics. The SILA system designed to sustain legacy aircraft subsystems after production offers a practical platform for analyzing how Hybrid methodologies operate under varying developmental constraints. By examining how both Waterfall and Agile methodologies were applied, and more importantly, how Hybrid strategies emerged to bridge the two, the research captures critical lessons for future implementation.

This design enables an in-depth analysis of project execution, from managing evolving requirements to maintaining regulatory compliance. The research design also evaluates how Hybrid methodologies influence cost, time, and sustainment outcomes, providing actionable recommendations for integrating flexible, scalable development models suitable for the dynamic demands of the A&D sector.

5.2.2. Hybrid Methodology Definition

This study categorizes SILA subsystems into three approaches: Waterfall SILA, Agile SILA, and Hybrid SILA. Each category represents a distinct methodological approach applied to comparable subsystems ensuring that the analysis reflects differences attributable to development methodology. The Hybrid SILA approach is defined as a deliberate and structured integration of select Waterfall and Agile practices, applied to balance the traceability and compliance requirements of the aerospace domain with the adaptability needed for evolving sustainment needs. The Hybrid model was an intentional architectural response to the limitations observed in using either methodology exclusively. Specifically, Hybrid SILA subsystems adopted the following Waterfall components:

- Formal requirements baselining and approval at project initiation, ensuring traceability to original stakeholder intents.
- Structured design reviews and verification gates to support compliance with safety and airworthiness standards.
- Rigorous configuration management and change control processes, aligning with aerospace regulatory expectations for documentation and auditability.

In parallel, the Agile components embedded in Hybrid SILA subsystems included:

- Use of sprint cycles and incremental development to accelerate prototyping and subsystem
- Continuous stakeholder engagement, including maintainers and field engineers, to rapidly incorporate operational feedback.
- Frequent internal demonstrations and backlog grooming, allowing early identification and resolution of emergent issues.

This hybridization allowed teams to partition development activities: high-assurance tasks (e.g., interface definition, regulatory compliance) were conducted using Waterfall practices, while lower-risk or iterative tasks (e.g., interface mockups, non-safety-critical logic updates) followed

Agile workflows. In doing so, Hybrid SILA sought to maximize responsiveness to change without compromising system integrity or certification requirements.

5.2.3. Subsystems and Corresponding Methodologies

The classification of subsystems into Waterfall, Agile, and Hybrid categories was derived through a combination of document analysis of lifecycle artifacts (e.g., requirement specifications, sprint logs, system review packages), interviews with engineering leads and project managers to understand the rationale behind methodological choices, and configuration reviews, identifying formal development practices applied to the subsystem. Data was collected over a 13-month period, allowing the capture of impacts such as change propagation, rework costs, and time-to-field metrics.

Each subsystem included in the study (see Table 4) was selected based on functional similarity and scope, ensuring valid comparisons across the three methodological approaches. This structured classification enabled a systematic evaluation of how development methodology influences outcomes such as requirement volatility, cost efficiency, and sustainment adaptability. In summary, the Hybrid SILA methodology represents a tailored engineering strategy that draws upon the predictability of Waterfall and the agility of iterative methods, designed specifically for the nuanced challenges of long-term aircraft subsystem sustainment.

Table 4. Summary of Subsystems and Methodologies Compared

Lifecycle Element	Methodology	SILA Subsystem
Requirement Changes	Waterfall	Propulsion system software
	Agile	
	Hybrid	
System Integration and Test	Waterfall	Integrated avionics management system
	Agile	Flight management software suite
	Hybrid	Aircraft health monitoring system
Performance Testing and Optimization	Waterfall	Autonomous navigation and guidance module
	Agile	Engine performance monitoring system
	Hybrid	Cybersecurity interface
Earned Value Management System	Waterfall	Flight Control Data
	Agile	Navigation Planning Processor
	Hybrid	Avionics Health and Diagnostics

5.3. Results

The cost estimates have been rounded to the nearest half million to protect sensitive data while maintaining the integrity of the comparative analysis. This rounding preserves relative trends and outcomes across Waterfall, Agile, and Hybrid methodologies without disclosing precise financial figures.

5.3.1. SILA System Requirement Changes Comparison

Table 5 presents a comparative analysis of how propulsion system software requirements for the SILA system were managed using Waterfall, Agile, and Hybrid methodologies. This table compares three comparable propulsion subsystem projects developed independently under

Waterfall, Agile, and Hybrid methodologies. Each row outlines how key engineering and project management factors were handled under each method. Requirement scopes were functionally equivalent to allow fair benchmarking of time, cost, rework, and stakeholder outcomes.

The Waterfall-based subsystem experienced late-stage requirement gaps and minimal stakeholder input, resulting in a 9-month timeline 2 months beyond the planned schedule and a total cost of $4 million, including $1 million in rework due to inflexible architecture and delayed validation. In contrast, the Agile approach supported iterative refinement of requirements particularly for diagnostics and fault detection enabled by early stakeholder involvement and simulation-in-the-loop validation. This led to on-time delivery in just 15 weeks. While Agile was both the fastest and most cost-effective, the Hybrid approach yielded a strategic middle ground. It combined upfront planning with controlled iteration, completing delivery in 18 weeks at the same total cost of $2 million.

The Hybrid approach limited rework by identifying requirement changes early (e.g., via model reuse and staged updates) and improving stakeholder alignment through monthly feedback sessions. This also enabled a 35% reduction in V&V duration, and only one defect escaped testing. Though not as rapid as Agile, Hybrid demonstrated superior control and consistency traits critical for sustaining complex legacy systems by balancing structure and flexibility to mitigate risks and align with regulatory and operational constraints. All cost savings and timeline deviations are compared to contractual estimates provided to the client. The data supports that while Agile maximizes speed and cost savings, Hybrid offers a robust alternative for environments that demand adaptability without sacrificing system integrity. Cost and schedule comparisons are based on contractual estimates, and all savings/overruns are measured against those baselines.

Table 5. System Requirement Changes of the SILA System for Propulsion System Software

Factor	**Waterfall**	**Agile**	**Hybrid**
Requirement Stability	Initial requirement: “The subsystem must support fixed-point diagnostics for engine anomalies using predefined fault codes.” Issue: Inflexible design led to a 3-month delay when unexpected sensor types were introduced.	Evolved requirement: “The diagnostic engine must process both fixed and variable fault data.” Result: Refined over 6 iterations in 8 weeks, with no delay.	Combined initial planning with staged updates. Result: Adaptable requirement set integrated after 2 iterations, completing changes in 10 weeks.
Change Management	Example change: Add fault detection for variable-speed compressor anomalies. Process: Took 6 months due to documentation and formal review cycles.	Same change as in Waterfall implemented after Sprint 2. Result: Completed in 3 weeks, saving $250K under budget.	Same change identified in iteration 1 and implemented in 5 weeks. Result: saved through partial reuse of previously verified models.
Stakeholder Collaboration	Frequency: Quarterly milestone reviews. Issue: UI mismatches with pilot expectations noticed too late. Impact: $500K redesign.	Frequency: Bi-weekly stakeholder demos. Result: UI refinements made mid-development; no redesign required.	Frequency: Monthly reviews with integrated feedback sessions. Result: UI issues caught early and resolved in design sprint; reduced stakeholder review time by 40%.
Verification & Validation	Initial Requirement (Waterfall): “The system shall validate fault flags post-cycle via simulation only.” Issue: Late-stage defects detected during final test phase.	Continuous V&V using simulation-in-the-loop throughout development. Result: Reduced test cycle time by 50% (from 4 weeks to 2 weeks).	V&V conducted through phased simulation and lab testing. Result: Only 1 defect escaped final test; test duration reduced from 4 weeks to 2.6 weeks (35%).

Time & Cost Efficiency	Timeline: 9 months (2 months overrun) Total Cost: $4M Rework: $1M	Timeline: 15 weeks (delivered on time) Total Cost: $2M by avoiding late-stage rework and UI redesigns (compared to Waterfall).	Timeline: 18 weeks Total Cost: $2M Result: Balanced performance 60% less rework than Waterfall, with more controlled changes than Agile. Completed in 2 iterations vs. Agile's 6.
Impact	High cost and schedule overrun driven by late discovery of requirement issues and rigid processes.	High adaptability enabled rapid delivery and cost control but required strict sprint discipline and heavy stakeholder involvement.	Demonstrated optimal balance: up-front clarity combined with flexible delivery. Enabled sustained cost control, reduced testing burden, and improved stakeholder alignment.

5.3.1.1. Requirement Stability

The Waterfall approach suffered from rigidity in managing evolving requirements. The initial requirement, “The subsystem must support fixed-point diagnostics for engine anomalies using predefined fault codes” resulted in a 3-month delay and $1M in rework when unexpected sensor types were introduced late in development. In contrast, Agile’s evolved requirement “The diagnostic engine must process both fixed and variable fault data” was refined iteratively over 6 sprints in 8 weeks without delays. Hybrid combined structured upfront planning with staged adaptability, completing changes in 10 weeks after just 2 iterations, with only $250K in adjustments. This highlights that while Agile excels at rapid iteration, Hybrid offers flexibility with fewer change cycles and tighter control.

5.3.1.2. Change Management

In Waterfall, implementing changes such as “adding fault detection for variable-speed compressor anomalies” required 6 months due to lengthy documentation and review cycles,. Agile integrated the same change in just 3 weeks after Sprint 2 using telemetry logs. Hybrid flagged the change early during its first iteration and implemented it within 5 weeks, achieving $300K savings through partial reuse of previously verified models. Hybrid’s proactive identification of changes reduced both cost and time compared to Waterfall, though Agile remained the fastest.

5.3.1.3. Stakeholder Collaboration

Waterfall limited stakeholder collaboration to quarterly milestone reviews, which led to misaligned user interfaces (UI) with pilot expectations and a $500K redesign. Agile employed bi-weekly stakeholder demos, allowing UI refinements mid-development with no redesigns needed. Hybrid implemented monthly reviews and integrated feedback sessions, enabling early UI adjustments and reducing review time by 40%. This demonstrates that while Agile provides the fastest feedback loops, Hybrid still captures early insights effectively, reducing late-stage changes.

5.3.1.4. Verification & Validation (V&V)

In Waterfall, verification was performed late in the cycle, relying solely on simulation. The requirement, “The system shall validate fault flags post-cycle via simulation only” resulted in rework due to late defect detection. Agile incorporated continuous V&V through simulation-in-the-loop, reducing test cycle time by 50% (from 4 weeks to 2 weeks). Hybrid phased V&V with both incremental simulation and lab testing reduces the test phase to 2.6 weeks a 35% improvement over Waterfall.

5.3.1.5. Time & Cost Efficiency

The Waterfall approach took 9 months (2 months overrun) with a total cost of $4M, including $1M in rework. Agile, by contrast, delivered in 15 weeks on budget at $2M, avoiding rework through early defect detection and iterative stakeholder collaboration. Hybrid delivered in 18 weeks at the same total cost of $2M but with reduced rework- 60% lower than Waterfall. While Agile achieved the fastest delivery, Hybrid maintained tighter control with fewer requirement changes.

5.3.1.6. Overall Impact

Waterfall was characterized by high costs and schedule overruns caused by rigid processes and late identification of requirement gaps. Agile provided rapid delivery and cost savings but required rigorous sprint discipline and continuous stakeholder involvement. Hybrid demonstrated a balanced performance, combining upfront clarity with flexibility to accommodate evolving requirements, while sustaining cost control, reducing testing effort, and improving stakeholder alignment.

5.3.2. SILA System Integration and Test (SIT) Phase Comparison

The comparative analysis of the SILA System Integration and Test (SIT) phase in Table 6 highlights distinct trade-offs among Waterfall, Agile, and Hybrid methodologies regarding cost, schedule, adaptability, and validation efficiency. The Waterfall approach used for integrating the Integrated Avionics Management System required the most time 12 months and the highest expenditure at $4 million. While the system’s inherent complexity contributed to this, the rigid, sequential Waterfall process further extended the schedule by dedicating three months to extensive documentation, six months to integration testing, and three months to validation. The lack of early feedback loops led to significant rework during validation, especially due to hard-coded interfaces that limited flexibility when requirements changed.

Conversely, the Agile methodology applied to the Flight Management Software Suite enabled completion in just four months at \$3 million. Through four iterative sprints, Agile facilitated early subsystem testing and incorporated maintainer feedback quickly, which enhanced responsiveness. However, this accelerated pace resulted in reduced system-wide traceability and some inconsistencies in subsystem outputs, necessitating additional verification and validation steps after delivery. The Hybrid approach, implemented for the Aircraft Health Monitoring System, balanced these trade-offs by integrating Agile sprints within a structured Waterfall milestone framework. Over six months and \$3 million, it achieved moderate cost and time efficiency while maintaining compliance and traceability.

Table 6. System Integration and Test (SIT) Phase of Aircraft Systems Sustainment

Variable	**Waterfall**	**Agile**	**Hybrid**
Subsystem	Integrated Avionics Management System	Flight Management Software Suite	Aircraft Health Monitoring System
Duration	12 months	4 months	6 months
Cost	\$4M	\$3M	\$3M
Approach	Sequential (Plan, Build, Test)	4 Iterative Sprints	Agile Sprints Aligned to Waterfall Milestones
Highlights	3 months for documentation; 6 months integration testing; 3 months validation	Feedback from maintainers; Subsystem testing in Sprint 1; Interface tweaks integrated by Sprint 2	Milestone 1: Compliance planning (1 month); Sprints 1–3: Development & testing (3 months); Milestone 2: Formal reviews (2 months)
Challenges	Delayed feedback Rework in validation phase; Hard-coded interfaces led to issues when updates were needed	Lacked traceability; Inconsistencies in subsystem outputs; Required additional V&V steps post-delivery	Maintained traceability and compliance; Enabled quick adaptation to maintenance feedback; Reduced rework compared to Waterfall

5.3.3. SILA Performance Testing and Optimization

The SILA Performance Testing and Optimization data in Table 7 highlights distinct differences in efficiency and effectiveness among the three development methodologies. The Waterfall method, applied to the Autonomous Navigation and Guidance Module, was the most time- and cost-intensive, requiring six months and $2.5 million to complete. The process was heavily sequential, with three months dedicated to initial simulations and stress testing, followed by bottleneck analysis and a final month of optimization and certification. Issues were often discovered late, leading to high rework efforts and delays primarily due to the inflexible documentation and lack of iterative testing cycles. In contrast, the Agile approach used for the Engine Performance Monitoring System completed performance testing in just three months at a cost of $2 million. This was achieved through three focused sprints addressing stress testing, bottleneck identification and fixes, and final optimization with iterative testing. However, despite faster delivery, Agile faced challenges in maintaining certification traceability and sometimes missed broader system-wide impacts due to sprint-focused efforts. The Hybrid model, implemented for the Cybersecurity Interface, struck a strong balance between speed, cost, and rigor. Completed in 4.5 months and costing $2 million, it began with Waterfall-style simulations and baseline setup, followed by two months of Agile sprints for performance tuning, and concluded with 1.5 months of certification alignment and automated traceability. This combined approach reduced bottleneck resolution time by 40% compared to Waterfall while preserving necessary documentation and compliance rigor. Overall, the Hybrid model demonstrated superior adaptability and control, making it well-suited for complex, compliance-driven environments demanding both responsiveness and audit readiness.

Table 7. Performance Testing and Optimization of Autonomous Systems Resilience

Variable	**Waterfall**	**Agile**	**Hybrid**
Subsystem	Autonomous Navigation and Guidance Module	Engine Performance Monitoring System	Cybersecurity Interface
Duration	6 months	3 months	4.5 months
Cost	$2.5 million	$2 million	$2 million
Activities	- 3 months: Initial simulations and stress testing - 2 months: Bottleneck analysis - 1 month: Final optimization and certification	- Sprint 1: Initial stress testing - Sprint 2: Bottleneck identification and fix - Sprint 3: Final optimization and iterative testing	- Month 1: Waterfall-style simulation and baseline setup - Months 2–3: Agile sprints for performance tuning - Final 1.5 months: Certification alignment and automated traceability
Challenges	- Late discovery of issues - High rework effort - Inflexible documentation and testing cycles	- Certification traceability issues - Sprint focus sometimes missed system-wide impacts	- Balanced iterative feedback with documentation rigor

5.3.4. SILA Earned Value Management System (EVMS) Analysis

The following Table 8, Table 9, and Table 10 illustrates evolving cost and value trends across three comparable subsystems developed under distinct methodologies within the Aircraft Mission Support: the Flight Control Data (Waterfall), the Navigation Planning Processor (Agile), and the Avionics Health and Diagnostics (Hybrid). Over the eight-month period, the Waterfall-based Flight Control Data shows a structured increase in Planned Value (PV), rising from $0.5M in Month 1 to $4.5M by Month 8. However, Earned Value (EV) begins to fall behind planned value from Month 3 and ends at $4.0M, while Actual Cost (AC) accelerates, also ending at $4.5M. This indicates early and sustained cost overruns, with a final $0.5M gap between expected value and actual cost highlighting inefficiencies and slower progress than planned. The Navigation Planning Processor, developed using Agile, shows a more proportional trend.

Planned and expected values remain closely aligned from start to finish, both ending at $4.0M. Actual costs stays consistently controlled, mirroring expected value across most months and finishing at $4.0M. The tight alignment among the three metrics suggests efficient execution, with minimal variance between what was planned, accomplished, and spent. The Hybrid approach, used for the Avionics Health and Diagnostics, demonstrates the most consistent and balanced performance. Planned value rises steadily to $4.5M, closely matched by expected value, which also reaches $4.5M by Month 8. Actual cost climbs in parallel, finishing at $4.0M. Variances between all three indicators remain small throughout the period, reflecting controlled costs and predictable progress. The Hybrid model effectively combines structured planning with adaptive delivery, leading to strong schedule and budget alignment across the development timeline.

Table 8. Waterfall EVMS

Month	**Waterfall Planned Value**	**Waterfall Expected Value**	**Waterfall Actual Cost**
1	$0.5M	$0.5M	$0.5M
2	$1.0M	$1.0M	$1.5M
3	$2.0M	$1.5M	$2.0M
4	$2.5M	$2.0M	$2.5M
5	$3.0M	$3.0M	$3.5M
6	$3.5M	$3.5M	$4.0M
7	$4.0M	$4.0M	$4.0M
8	$4.5M	$4.0M	$4.5M

Table 9. Agile EVMS

Month	**Agile Planned Value**	**Agile Expected Value**	**Agile Actual Cost**
1	$0.5M	$0.5M	$0.5M
2	$1.0M	$1.0M	$1.0M
3	$2.0M	$1.5M	$1.5M
4	$2.5M	$2.5M	$2.0M
5	$3.0M	$2.5M	$2.5M
6	$3.5M	$3.0M	$3.0M
7	$4.0M	$3.5M	$3.5M
8	$4.0M	$4.0M	$4.0M

Table 10. Hybrid EVMS

Month	**Hybrid Planned Value**	**Hybrid Expected Value**	**Hybrid Actual Cost**
1	$0.5M	$0.5M	$0.5M
2	$1.0M	$1.0M	$1.0M
3	$1.5M	$1.5M	$2.0M
4	$2.5M	$2.0M	$2.5M
5	$3.0M	$3.0M	$2.5M
6	$3.5M	$3.5M	$3.0M
7	$4.0M	$4.0M	$3.5M
8	$4.5M	$4.5M	$4.0M

5.3.4.1. Planned Value (PV) Comparison

Looking at the Planned Value (PV) graph in Figure 8, all three subsystems exhibit structured cumulative budget allocation across eight months. The Flight Control Data (Waterfall) subsystem maintains a steady planned value increase from \$0.5M in Month 1 to \$4.5M by Month 8. This front-loaded approach aligns with Waterfall's planning-intensive philosophy, where funding is committed early to support documentation, design, and review cycles (Boehm et al., 2003). In contrast, both the Navigation Planning Processor (Agile) and Avionics Health and Diagnostics (Hybrid) subsystems demonstrate similarly gradual planned value growth patterns reaching \$4.0M and \$4.5M respectively reflecting more adaptive budgeting models. Agile's incremental allocation supports sprint-based delivery and reprioritization, while Hybrid balances early funding for critical components with iterative reallocation, integrating both flexibility and oversight (Highsmith, 2001; Serrador & Pinto, 2015).

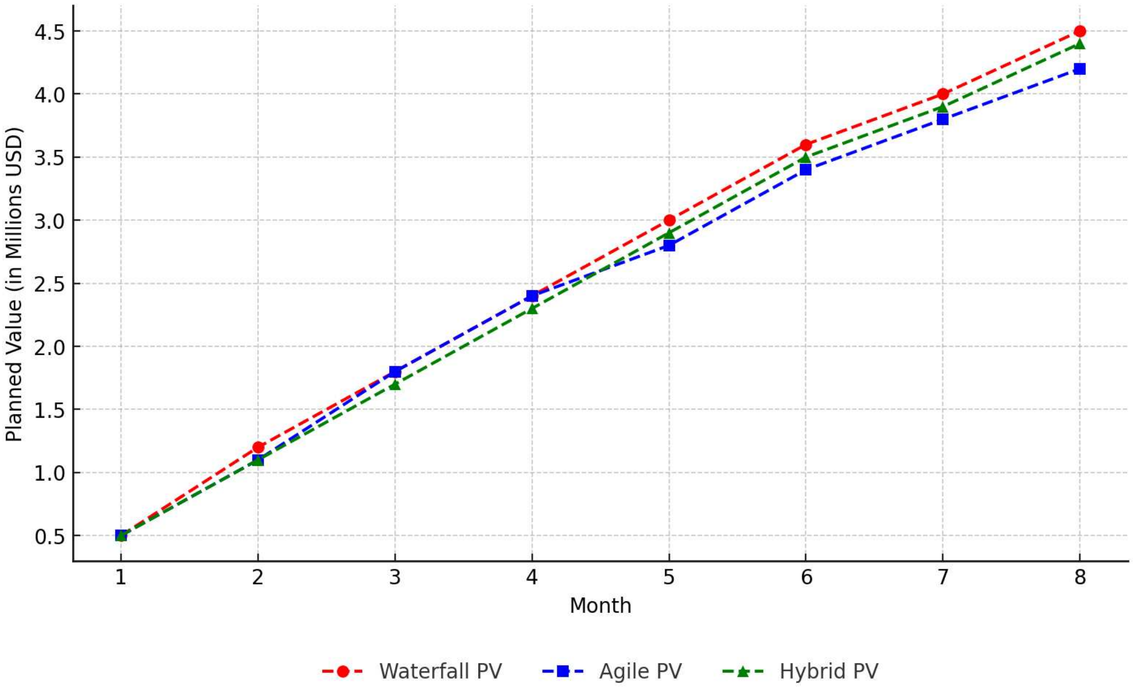


Figure 8. Planned Value (PV) in SILA

5.3.4.2. Earned Value (EV) Comparison

The Earned Value (EV) graph in Figure 9 reveals notable differences in delivery efficiency. By Month 8, Avionics Health and Diagnostics (Hybrid) leads with \$4.5M in expected value, followed by Navigation Planning Processor (Agile) and Flight Control Data (Waterfall), both at \$4.0M. Hybrid's ability to align earned value with planned value demonstrates an effective combination of pre-defined structure and agile responsiveness. The Hybrid team baselined critical system elements early mimicking Waterfall strengths while using Agile iterations to incorporate feedback and refine deliverables. This allowed for fast prototyping and high user engagement without compromising traceability or documentation (Boehm et al., 2003; Vinekar et al., 2006). By contrast, Agile's strong start slowed around Months 5–7, likely due to limited upfront architecture planning and decentralized coordination (Stettina et al., 2014), although the final expected value (\$4.0M) still matched its planned value. The Waterfall-driven Flight Control Data subsystem reflected the most delayed progress, with expected value plateauing at \$4.0M half a million below its planned value. The linear sequencing inherent in Waterfall, which delays value realization until phase completion, limited its adaptability to emerging needs (Maharao, 2024), further reinforcing the relative strength of Hybrid models in sustainment environments (Conforto et al., 2014).

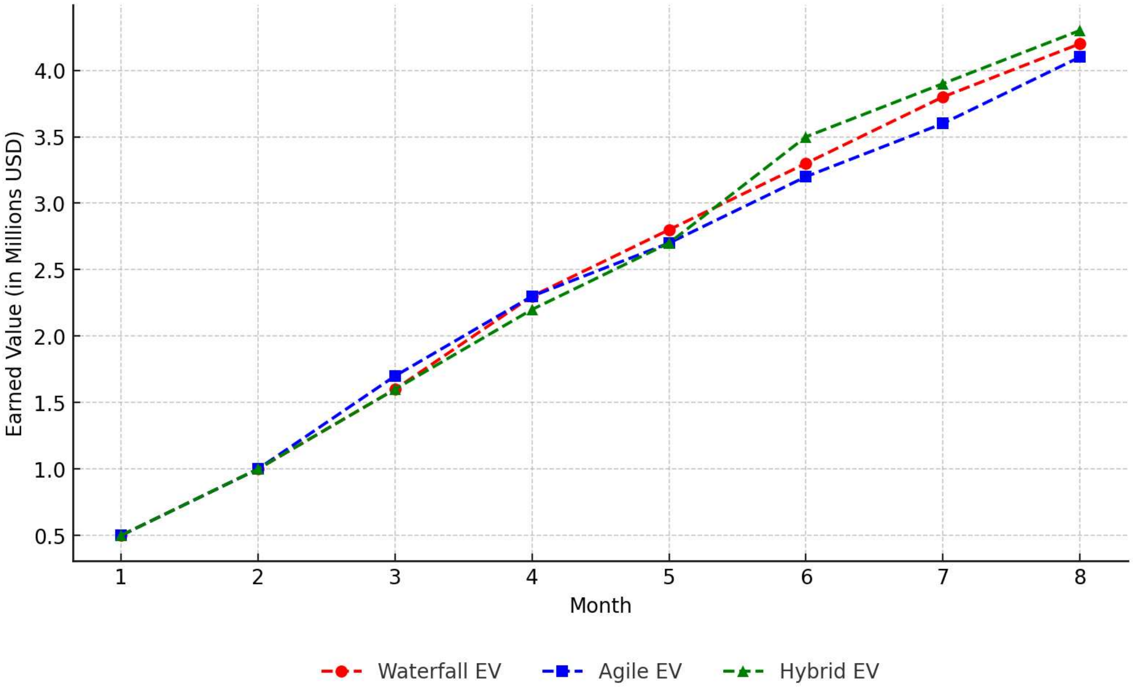


Figure 9. Earned Value (EV) in SILA

5.3.4.3. Actual Value (AV) Comparison

The Actual Cost (AC) graph in Figure 10 reveals clear distinctions in expenditure behavior. By Month 8, the Flight Control Data subsystem incurs the highest cost at $4.5M, matching its planned value but exceeding its EV. This consistent overrun began in Month 2, when AC jumped to $1.5M while expected value remained at $1.0M indicating inefficiencies, likely caused by delayed requirement validation or rework during integration (Boehm et al., 2003). In contrast, the Navigation Planning Processor (Agile) maintained tight cost control, concluding with an actual cost of $4.0M, in exact alignment with its planned and expected values. This reflects Agile's efficient cycle management and reduced overhead stemming from early stakeholder involvement and scope refinement (Highsmith, 2009). Meanwhile, the Avionics Health and Diagnostics (Hybrid) subsystem presents a middle ground reaching $4.0M in actual cost, which is lower than its final EV of $4.5M. The Hybrid cost curve remains steady, reflecting strategic planning for compliance-heavy tasks early in the cycle and efficient iterative refinement for less critical updates. This stratified resource allocation contributed to predictable spending and reduced volatility across the timeline (Conforto et al., 2014; Serrador & Pinto, 2015).

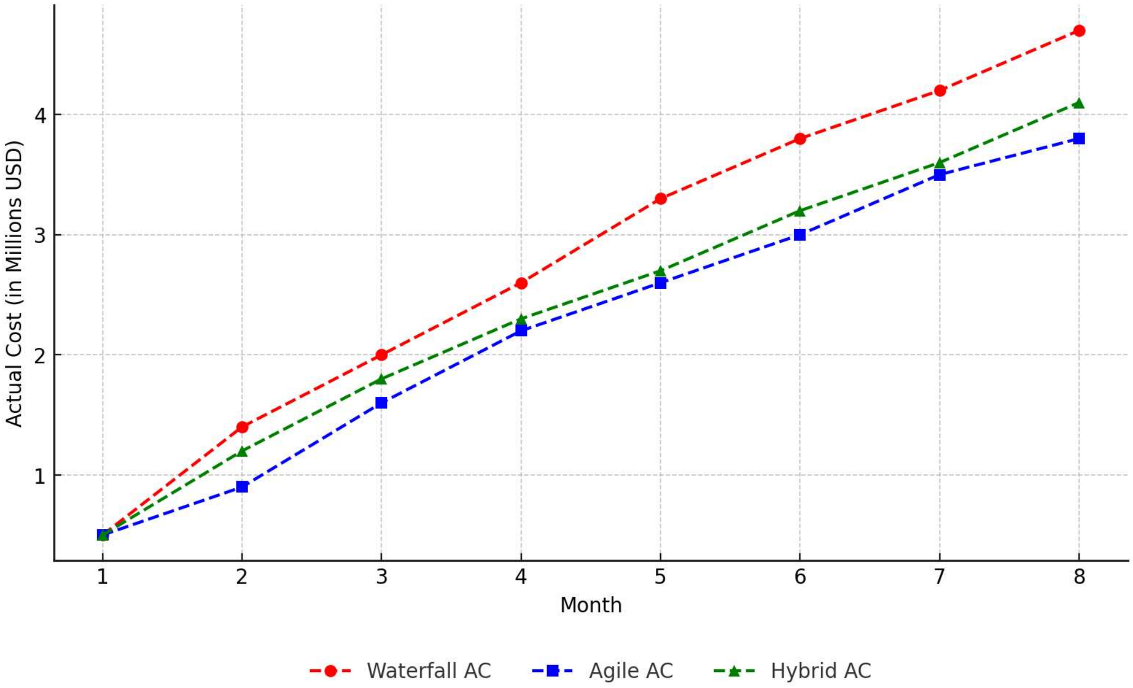


Figure 10. Actual Cost (AC) in SILA

5.4. Discussion

The findings from the SILA case study illustrate that neither Waterfall nor Agile, when applied in isolation, fully meets the operational and compliance demands of aerospace and defense sustainment projects. Instead, the results reinforce the value of a strategically tailored Hybrid methodology that draws on the strengths of both approaches leveraging Agile's adaptability for rapid, low-risk updates while retaining Waterfall's governance for high-assurance phases. This balance proved particularly effective in SILA, where subsystem-level iterations could proceed at speed without compromising the formal verification, documentation, and certification processes essential to regulated environments.

5.4.1. Requirement Changes Overall

One of the most critical areas of divergence across the three methodologies was requirement stability. In the SILA case, each propulsion subsystem shared a comparable diagnostic requirement, yet the ability to manage requirement change varied significantly by method. Under the Waterfall approach, requirements were rigidly front-loaded and documented with minimal room for adaptation. When new sensor types were introduced mid-development, the design lacked the flexibility to accommodate them. This led to a 3-month schedule delay, as late-stage changes required revalidation of integrated components. This outcome reflects the common limitations of linear models in dynamic sustainment environments, where requirements may shift based on operational feedback or evolving system interfaces (Silva et al., 2023).

In contrast, Agile's iterative model allowed requirements to evolve through six development sprints across eight weeks. The team was able to adapt the fault diagnostics algorithm to handle both fixed and variable fault data types without triggering any schedule delay. This was enabled through continuous collaboration and simulation-in-the-loop testing, which allowed incremental refinement aligned with real-time feedback. While Agile clearly excelled in responsiveness, it also required a higher degree of coordination and sprint discipline, and its lightweight documentation practices posed traceability concerns particularly in highly regulated sectors (Arthur et al., 2017).

The Hybrid approach demonstrated a more structured yet adaptive path. Requirements were defined up front and refined through two planned iterations, completing the updates in 10 weeks. This structure preserved traceability while avoiding the rework costs of Waterfall and the overhead of repeated Agile iteration cycles. The outcome shows that strategic flexibility, when intentionally embedded within staged development gates, can minimize cost and schedule impacts without compromising traceability (Dugbartey et al., 2025).

Change management further illustrated the burden of rigid process adherence. A representative engineering update adding fault detection for variable-speed compressor anomalies required six months to integrate under Waterfall due to formal documentation and approval processes. Agile addressed the same change after Sprint 2 using real-time telemetry logs and completed it in just 3 weeks. However, Agile's rapid integration also posed configuration management risks when formal change logs were delayed or incomplete. The Hybrid model implemented the change within 5 weeks during its first iteration, leveraging partial reuse of previously verified model elements. The result supports the notion that Hybrid methods can offer speed without sacrificing auditability.

Stakeholder collaboration revealed a similar pattern. In Waterfall, feedback was limited to quarterly milestone reviews, which proved too infrequent to identify UI misalignments with pilot expectations. Agile's bi-weekly demos enabled earlier input from stakeholders and resolved UI issues during development eliminating the need for redesign altogether. The Hybrid approach scheduled monthly feedback sessions with interim design reviews, enabling UI refinements within the first design sprint and reducing review effort by 40%. This cadence effectively captured feedback without overwhelming stakeholders, offering a middle ground between Waterfall's inflexibility and Agile's intensity (Shereni, 2015).

Verification and Validation (V&V) strategies also diverged in terms of timing, traceability, and cost impact. Waterfall deferred all validation to a post-development test phase, where several critical defects emerged. This resulted in a 4-week test window. Agile integrated simulation-in-the-loop testing during development, allowing earlier defect identification and reducing the final test phase to 2 weeks, avoiding the rework entirely. Hybrid followed a phased testing approach, combining simulation and physical lab validation at key milestones. This approach limited escaped defects to one and shortened the final V&V phase to 2.6 weeks, a 35% reduction compared to Waterfall. Importantly, Hybrid preserved formal test documentation needed for certification, striking a balance between flexibility and compliance (Shekhar, 2019).

Finally, the cost and schedule performance summarized the trade-offs across methodologies. Waterfall took 9 months and cost $4 million, with $1 million attributed to rework. Agile completed the project in 15 weeks at a total cost of $2 million, by resolving issues earlier and avoiding rework. Hybrid required 18 weeks, slightly longer than Agile, but matched the $2 million budget and reduced rework to a 60% improvement over Waterfall. While Agile was the fastest and most cost-efficient, Hybrid offered more predictability in change management, traceable artifacts, and compliance support attributes essential for sustainment programs within aerospace and defense domains.

These results reinforce Objective 1, confirming that Agile approaches deliver speed and cost efficiency in dynamic, change-prone environments. However, the Hybrid methodology emerged as the most balanced, offering a viable alternative in sustainment contexts where traceability, certification readiness, and controlled evolution of requirements are equally important. Waterfall retained value in traceability but showed significant performance degradation in the face of evolving system needs.

5.4.2. System Integration and Test (SIT) Phase Overall

The SILA System Integration and Test (SIT) phase results reveal important distinctions in how development methodologies impact integration duration, cost effectiveness, and defect resolution. Using the Waterfall approach, the SIT phase spanned 12 months and incurred costs of approximately $4 million. This extended timeline was partly due to the system's complexity but was further prolonged by Waterfall's rigid, sequential structure, which dedicated the first three months exclusively to documentation and interface specifications before any integration began. This front-loaded process delayed critical feedback from integration testing, leading to defects being discovered late during validation and necessitating significant rework, which caused schedule overruns and increased expenses. These outcomes align with prior studies highlighting inefficiencies caused by delayed defect detection and rigid phase transitions typical of sequential models (Boehm et al., 2003).

Conversely, the Agile implementation completed the SIT phase in only 4 months at a reduced cost of $3 million, representing a 30% cost saving. Agile's iterative sprints embedded integration and testing activities early and continuously, enabling earlier identification of interface mismatches and faster incorporation of stakeholder feedback. This iterative cadence allowed for quicker issue resolution and minimized the propagation of integration defects, a pattern observed in other large defense projects employing Agile methodologies effectively (Tanner et al., 2014). However, Agile's rapid pace sometimes compromised formal traceability and led to inconsistencies among subsystem outputs, particularly when deliverables from different sprints were not fully synchronized (Womack et al., 2008).

These challenges address the benefits of Hybrid methodologies. The Hybrid approach adopted in the SILA SIT phase combined Agile sprints with structured milestone-based reviews, effectively balancing agility with regulatory compliance. By front-loading compliance planning and integrating formal validation gates, the Hybrid model reduced rework by 50% compared to Waterfall, while maintaining traceability and adapting quickly to feedback. Such Hybrid frameworks provide a strategic middle ground, embedding flexibility within a rigorously controlled process, a necessity for high-assurance aerospace projects (Shekhar, 2019).

5.4.3. Performance Testing and Optimization Overall

The results of the SILA performance testing and optimization phase further highlight systemic inefficiencies associated with traditional Waterfall approaches, especially within sustainment workflows. Waterfall's fixed-phase progression delayed the detection and resolution of performance bottlenecks, with the testing cycle taking six months and $2.5 million to complete. Bottlenecks were only identified during the later stages, leading to extensive rework and delayed certification. This lag in feedback echoes longstanding critiques of Waterfall's rigidity, particularly its limited capacity to adapt to emergent system behaviors during integration and testing (Larman et al., 2003). By contrast, the Agile model achieved a 50% reduction in duration and a 20% reduction in cost, completing testing in three months at $2 million. Agile's iterative, feedback-driven development and early validation practices embedded stress testing within sprints, enabling earlier bottleneck detection and resolution before issues could propagate downstream. These results align with prior studies demonstrating that Agile's continuous integration and testing enhances responsiveness and reduces late-stage failures (Dingsøyr et al., 2010).

The Hybrid approach, combining upfront structured planning with iterative tuning and automated traceability, provided the most balanced solution. Completed in 4.5 months and costing $2 million, it reduced bottleneck resolution time by 40% compared to Waterfall, while maintaining documentation rigor essential for certification. This supports Boehm and Turner's (2003) argument that Hybrid models enable organizations to leverage Agile's adaptability and speed while preserving plan-driven controls. Furthermore, Hybrid strategies reduce integration debt by aligning iterative development with formal milestones (Stettina et al., 2014). These findings directly address RQ 2.2 of this chapter, quantifying how Waterfall's rigid phase-gated structure contributes to rework and inefficiency in sustainment workflows. Agile's shortened feedback loops are especially valuable in environments requiring ongoing optimization based on evolving operational data. However, the traceability and audit readiness limitations suggest that Agile alone may not fully satisfy regulated industry standards. Therefore, a Hybrid approach emerges as the optimal strategy for performance testing in Aerospace & Defense contexts, balancing faster issue resolution with compliance, traceability, and certification integrity.

5.4.4. Earned Value Management System (EVMS) Overall

The three Figure 8, Figure 9, and Figure 10 collectively provided a comparative Earned Value Management (EVM) analysis for three SILA subsystems, each developed under a distinct methodology: the Flight Control Data subsystem (Waterfall), the Navigation Planning Processor subsystem (Agile), and the Avionics Health and Diagnostics subsystem (Hybrid). EVM is a critical project control technique that integrates scope, cost, and schedule parameters to provide an objective measurement of project performance. It does so through three core metrics: Planned Value (PV) the authorized budget assigned to scheduled work; Actual Cost (AC) the realized cost incurred for the performed work; and Earned Value (EV) the value of work completed to date. These indicators, when analyzed collectively, offer a multidimensional view of cost-efficiency, schedule adherence, and overall value delivery. This is particularly vital in A&D programs, where complex interdependencies, long development cycles, and evolving requirements often lead to significant scope creep and increased sustainment costs if not proactively managed (Kerzner, 2017).

The Flight Control Data (Waterfall) subsystem shows strong alignment with planned effort but suffers from higher costs and slower value realization due to integration bottlenecks and late rework. The Navigation Planning Processor (Agile) subsystem demonstrates strong cost control and value alignment, although its performance decelerates slightly under complex integration scenarios. The Avionics Health and Diagnostics (Hybrid) subsystem outperforms both in value delivery and cost-effectiveness, maintaining consistent trajectories across planned value, expected value, and actual cost metrics. These findings reinforce broader project management literature that advocates for Hybrid frameworks in regulated, evolving environments like A&D, where both structure and flexibility are essential to program success (Kerzner, 2017).

5.4.5. Synthesis of Hybrid Methodology

Based on the findings, a conceptual diagram (Figure 11) was developed to illustrate how Hybrid methodologies can be adapted by drawing from both Agile and Waterfall approaches, depending on factors such as time pressure and safety-critical requirements. Specifically, when time constraints are tight and safety demands are high, certain elements of Waterfall may be prioritized to ensure rigor and documentation, whereas in low-risk, time-flexible scenarios, Agile practices may offer greater efficiency and adaptability. The diagram highlights this spectrum and provides guidance on how to balance these methodologies in various operational contexts.

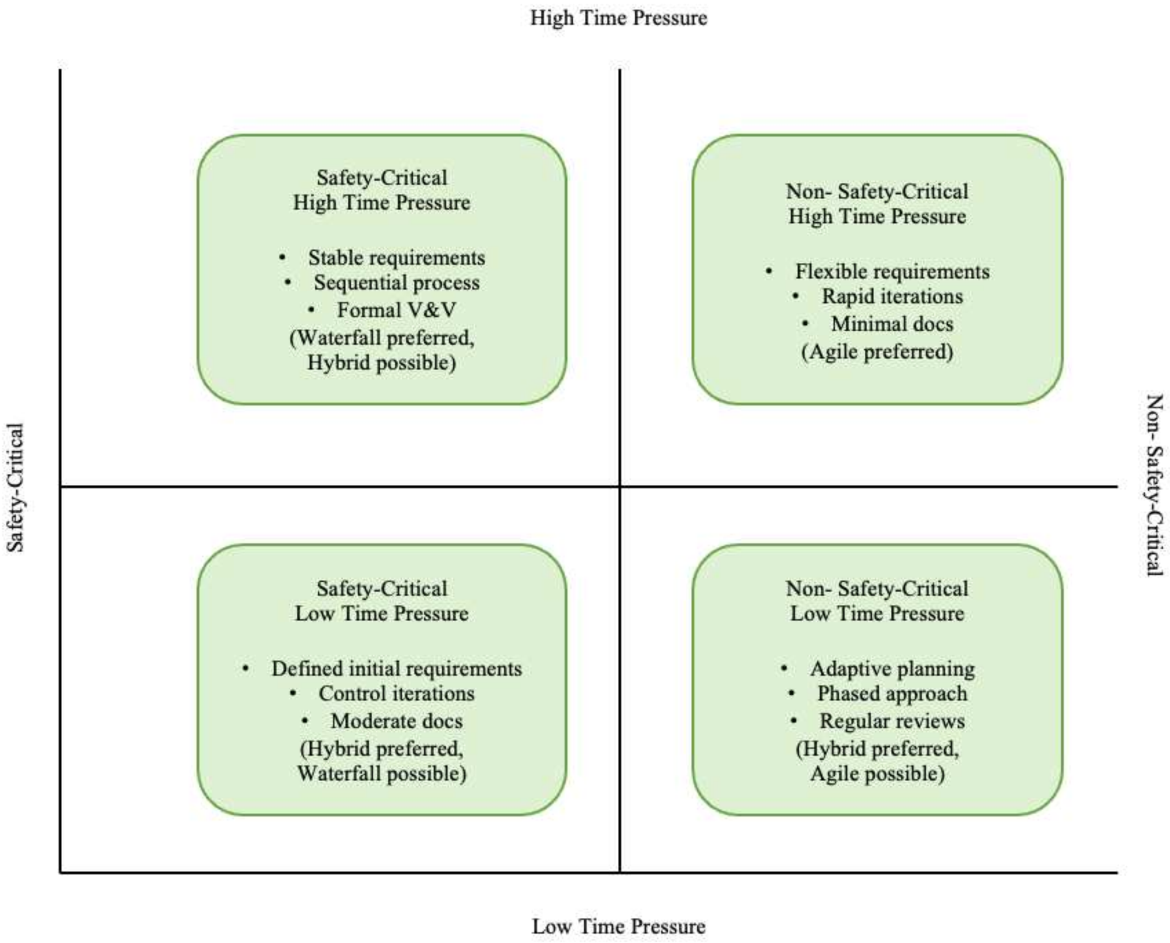


Figure 11. Hybrid Methodologies Recommendations Based on SILA Project

Overall, looking at Hybrid methodologies quadrant diagram (Figure 11), the Earned Value Management System (EVMS) discussion highlights the specific benefits of adopting Hybrid methodologies in scenarios characterized by lower safety requirements and minimal time pressures. Under these conditions, SILA’s implementation of a Hybrid approach enabled regular reassessments and flexible, phased implementation, aligning closely with evolving project goals. Through moderate documentation and targeted stakeholder involvement, the EVMS updates allowed for precise tracking of project costs, resource allocation, and schedule adherence, resulting in enhanced adaptability and responsiveness without the excessive administrative overhead typical of pure Waterfall methodologies. This approach fostered a balanced environment that supported informed decision-making and rapid adjustments, crucial for optimizing the value derived from project investments.

In the Performance Testing and Optimization discussion, the effectiveness of Hybrid methodologies emerged clearly in high safety but low urgency contexts. Initially defined yet flexible requirements provided a strong foundation, allowing structured iterations and moderate documentation. These controlled iterative cycles enabled the team to integrate real-time performance data and make incremental improvements while maintaining compliance with rigorous safety standards. Stakeholder engagement was consistently balanced, involving the right individuals at strategic milestones, thus enabling informed adaptations based on comprehensive performance insights. The result was a finely tuned optimization process that maintained system integrity and reliability, demonstrating the critical role Hybrid methodologies can play in environments where safety and performance must coexist with adaptive flexibility.

The System Integration and Test (SIT) Phase discussion emphasized Waterfall's particular strengths and limitations when managing projects with stringent regulatory and safety requirements under high time pressure. The detailed, sequential process and extensive documentation inherent in Waterfall methodologies ensured comprehensive verification and validation (V&V), crucial for meeting strict aerospace regulations. However, this rigid structure presented substantial drawbacks, especially when unforeseen requirements emerged or operational conditions rapidly changed. The prolonged feedback loops and limited adaptability led to delays and cost escalations. These limitations underscore the need for more flexible methodologies or hybrid approaches for certain components or project phases, where adaptability could mitigate these traditional drawbacks without sacrificing the meticulous compliance required in high-stakes environments.

The Requirement Changes discussion distinctly illustrated Agile methodologies' superior capability to manage projects facing dynamic requirements and high urgency with lower safety constraints. Agile’s iterative sprints, frequent stakeholder interactions, and continuous feedback mechanisms empowered the SILA team to respond rapidly to evolving propulsion system requirements. This iterative approach reduced turnaround times significantly and minimized rework by swiftly identifying and addressing changes as they arose. Although Agile’s minimal documentation posed some risks regarding traceability and regulatory compliance, its rapid responsiveness greatly enhanced SILA’s operational agility, ultimately resulting in improved system performance and reduced overall sustainment costs.

A comprehensive analysis of the quadrant diagram further reveals the inherent value of strategically deploying different methodologies based on clearly identified project contexts. The SILA case study demonstrates the limitations and strengths of each approach, emphasizing the critical advantage of hybrid methodologies in effectively navigating the nuanced trade-offs between documentation rigor, adaptability, regulatory compliance, and operational responsiveness. Hybrid frameworks, which integrate structured Waterfall phases with Agile flexibility, emerge as particularly advantageous, offering an optimal blend tailored specifically to the project's risk profile, stakeholder expectations, regulatory demands, and time constraints. Thus, the quadrant analysis not only provides a valuable visual and strategic tool but also articulates a clear pathway towards methodological agility and excellence in aerospace sustainment projects.

5.4.6. Limitations

While the SILA case study provides valuable insights into the practical application of Hybrid methodologies in aerospace and defense sustainment, several limitations should be acknowledged. First, the analysis is based on a single program within a specific organizational and regulatory context, which may limit the generalizability of findings to other A&D projects with different scales, governance structures, or operational priorities. Second, the study relied on retrospective data and stakeholder accounts, introducing the potential for recall bias and subjective interpretation of process effectiveness. Additionally, quantitative measures of cost and time efficiency were constrained by the availability and granularity of project documentation, which may have obscured smaller-scale variations in performance. Finally, the research did not include a formal experimental comparison between Hybrid, Agile, and Waterfall approaches, meaning causality between the methodology and observed outcomes cannot be definitively established. These limitations suggest the need for broader, multi-program studies with prospective data collection to more rigorously evaluate hybrid frameworks across varied sustainment environments.

5.5. Conclusion

The SILA case study underscores the growing necessity of hybrid methodologies in the sustainment of complex aerospace and defense systems. Traditional Waterfall approaches delivered vital structure, formal documentation, and regulatory compliance elements that remain indispensable in high-assurance, safety-critical domains. However, their rigidity and lengthy change cycles often hindered timely adaptation to evolving operational needs, especially as new failure modes and sustainment demands emerged over time. In contrast, Agile methodologies significantly improved SILA's responsiveness by enabling iterative development, rapid stakeholder feedback, and continuous integration of diagnostic enhancements. These qualities proved critical in reducing turnaround times, lowering cost overruns, and enhancing the system's predictive maintenance capabilities. Yet, Agile's inherent informality introduced risks to consistency, traceability, and certification core requirements in defense sustainment programs. This chapter illustrates that a Hybrid methodology, combining Waterfall's structured compliance with Agile's iterative adaptability, provides the most effective framework for sustainment success. Within SILA, Hybrid implementation allowed the organization to retain the control and traceability demanded by regulatory bodies, while embedding short, Agile-driven sprint cycles to manage subsystem updates, respond to maintenance insights, and refine diagnostic algorithms in near real-time. Notably, the Hybrid model enabled earlier defect detection, reduced rework through continuous testing, and promoted cross-functional collaboration across engineering, program management, and maintenance personnel.

For example, Agile sprints allowed updates to predictive algorithms to be delivered within weeks, particularly beneficial in scenarios characterized by high time pressure and low safety concerns, where rapid iterations and minimal documentation enhanced efficiency. Concurrently, Waterfall milestones ensured that subsystem integration and testing critical in high safety, high time-pressure contexts aligned with overarching certification and contractual requirements, ensuring robust formal documentation and rigorous verification processes. In situations marked by high safety yet lower urgency, such as SILA's Performance Testing and Optimization, the Hybrid methodology balanced initial requirement definitions with controlled iterative developments and moderate documentation, effectively harmonizing structure with flexibility. For the Earned Value Management System (EVMS) Analysis, which presented lower safety and time pressures, the Hybrid approach facilitated adaptive planning, phased implementations, and regular stakeholder reviews, optimizing cost control and adaptability.

While Agile clearly delivers superior performance in fast-turnaround sustainment environments, it is not universally applicable across all phases of aerospace engineering. Hybrid methodologies offer a pragmatic solution, enabling teams to adopt Agile’s sprint-driven approach for lower-risk iterative activities, while preserving Waterfall’s rigor for phases involving certification, documentation, and V&V. In the SILA case, a hybrid structure was utilized by integrating Agile practices into subsystem-level development while applying Waterfall gates for formal reviews and compliance checks.

In conclusion, the SILA case study demonstrates that a Hybrid methodology offers the optimal balance between stability and adaptability in aerospace sustainment projects. As illustrated in Figure 11 (Hybrid quadrant diagram), by strategically blending the formal rigor of Waterfall with the responsiveness of Agile tailored specifically to the safety and time-pressure contexts organizations can better navigate the complexities of system evolution, regulatory compliance, and operational readiness, ultimately ensuring more resilient, cost-effective, and mission-aligned sustainment outcomes. To fully realize the benefits of Hybrid development in aerospace and defense, greater emphasis must be placed on workforce training and formal education particularly through institutions like the Defense Acquisition University (DAU). Most current engineering programs still treat Agile and Waterfall as mutually exclusive, leaving a gap in preparing future engineers for Hybrid application in regulated environments. Embedding Hybrid methodology into DAU coursework and related university curricula would better equip acquisition professionals and engineers to strategically tailor development approaches based on safety, compliance, and time-pressure demands.

Chapter 6. Research Aim 3 - MBSE Artifacts

6.1. Introduction

This chapter focuses on how MBSE can be used as a tool to demonstrate when and why Hybrid development approaches outperform pure Waterfall and Agile execution in aerospace sustainment programs. Using data from the SILA propulsion diagnostic subsystem, the chapter examines how Hybrid practices affect requirement stability, verification timing, cost and rework, and stakeholder alignment when compared against alternative methodologies. MBSE is applied as a consistent modeling and data-integration mechanism that enables side-by-side comparison across methodologies. The results illustrate how Hybrid execution emerges as a practical response to system complexity and sustainment pressures, and how it achieves faster, more controlled delivery while maintaining compliance in a safety-critical environment. This chapter addresses the following research questions:

- **RQ3.1:** How does the use of MBSE impact requirement stability, verification and validation (V&V) timing, and stakeholder collaboration when applied within Waterfall, Agile, and Hybrid development methodologies?
- **RQ3.2:** How does the use of MBSE artifacts support key Measures of Effectiveness (MOEs), including cost and rework reduction, compliance integration, and defect detection accuracy across Waterfall, Agile, and Hybrid approaches?
- **RQ3.3:** How does the integration of MBSE modeling with real sustainment data from the SILA case study reveal trade-offs among Waterfall, Agile, and Hybrid methodologies in aerospace sustainment programs?

The contents of this chapter have been submitted and is under review at a peer reviewed journal, with the manuscript title "Demonstrating Model-Based Systems Engineering (MBSE) with Agile, Waterfall, and Hybrid Methodologies: A Comparative Case Study in Aerospace."

6.2. Methods

This study employs a case study methodology to examine how Waterfall, Agile, and Hybrid development approaches, when using MBSE, affect requirements management in a safety-critical aerospace context. The SILA propulsion diagnostic subsystem serves as the single case for this paper. While anonymized for publication, SILA reflects a real sustainment initiative aimed at extending the lifecycle of legacy aircraft systems.

The research integrates both quantitative and qualitative analyses. MBSE serves as the central analytical framework to model requirements, stakeholder interactions, and V&V processes. Model-based diagrams were created in MSOSA to represent each methodological scenario. The resulting models were then evaluated against defined Measures of Effectiveness (MOE), such as requirement stability, rework costs, verification cycle times, and stakeholder engagement frequency. The project initially followed a Waterfall execution model. As system complexity increased, Agile practices were formally introduced. However, full Agile execution proved difficult to sustain due to certification, verification, and governance constraints. As a result, teams transitioned toward Hybrid execution practices that combined iterative development with structured compliance and verification controls. These transitions occurred sequentially as execution practices evolved in response to program needs.

Data was collected from project artifacts, engineering logs, and program records. Metrics were attributed to Waterfall, Agile, or Hybrid execution based on the governing methodology in place at the time the work was performed. This approach captures how Waterfall execution initially struggled under increasing complexity, how Agile improved responsiveness but introduced compliance gaps, and how Hybrid practices emerged organically to balance adaptability with regulatory rigor.

6.2.1. Research Design

The SILA propulsion diagnostic subsystem was selected because it exemplifies the challenges of sustainment in aerospace: balancing rigorous certification requirements with the need for adaptability in predictive diagnostics and fault detection. The propulsion subsystem also represents a well-bounded case with clear interfaces (i.e., cockpit displays, avionics bus, ground maintenance systems, and OEM suppliers), making it suitable for systematic modeling and comparison. The research design unfolded in three phases: Case Framing, MBSE Model Development, and Evaluation Against Measures of Effectiveness. These are further described as follows.

Step 1: Case Framing. Table 11 summarizes the modeling method requirements defined for evaluating the SILA propulsion diagnostic subsystem under Waterfall, Agile, and Hybrid methodologies. These requirements focus on how MBSE supports requirement stability, verification efficiency, and stakeholder collaboration across development methods.

Table 11. Process-Oriented MBSE Requirements Evaluated

ID	**Name**	**Definition**
R1	Requirement Stability	The subsystem shall maintain a controlled baseline of subsystem requirements, enable iterative updates while preserving traceability across versions.
R2	Cost and Rework Reduction	The subsystem shall enable reuse of validated model elements to minimize rework and reduce total verification cost.
R3	Verification Cycle Duration	The subsystem shall demonstrate accelerated verification through phased V&V integration and reuse of validated models across increments.
R4	Stakeholder Collaboration Frequency	The subsystem shall capture stakeholder feedback and decision points at each iteration, maintaining a record of model changes and rationale.

R5	Compliance Integration	The subsystem shall generate traceable compliance evidence at gated reviews, ensuring certification alignment without duplicate documentation.
R6	Defect Detection Accuracy	The subsystem shall identify and document model inconsistencies early through incremental reviews to reduce defect escape rates.

Step 2: MBSE Model Development. Systems modeling language (SysML) models were created to show how Waterfall, Agile, and Hybrid approaches behave when applied to the same propulsion diagnostic subsystem. Two types of diagrams were developed in three separate versions to represent process behavior that changes across development styles:

- Activity Diagrams (Waterfall, Agile, Hybrid): illustrate the step-by-step development and verification flow.
- Use Case Diagrams (Waterfall, Agile, Hybrid): show how operational roles interact with the subsystem under each methodology.

The remaining diagrams were created as single shared models, since the underlying system structure and requirements remain constant regardless of methodology:

- Requirements Diagram: captures the six SILA requirements used across all three processes.
- Block Definition Diagram (BDD): defines subsystem components and their interactions with external actors.
- Allocation Map: connects requirements to verification activities, MOEs, and model artifacts.

These models provided a stable system foundation while allowing the process-specific diagrams to highlight how each methodology uniquely affects workflow, iteration patterns, and stakeholder involvement.

Step 3: Evaluation Against Measures of Effectiveness (MOEs). Each methodological scenario was assessed using six common MOEs, defined within MBSE models to ensure traceability and objective measurement. These are described in Table 12.

Table 12. Measures of Effectiveness for Evaluating Across Methodologies

ID	**Name**	**Definition**
MOE 1	Requirement Stability	Measured by baseline freezes versus iteration rate, providing traceability in MBSE models showing change frequency and approval gates.
MOE 2	Cost and Rework Reduction	Measured by the reduction in duplicated modeling or late-stage fixes, as captured through MBSE allocation matrices linking requirement updates to rework costs
MOE 3	Verification Cycle Duration	Measured by the average time required to complete a verification loop in MBSE-linked test artifacts (e.g., 4 weeks in Waterfall vs. 2 weeks in Agile).
MOE 4	Stakeholder Collaboration Frequency	Measured by the frequency of modeled stakeholder review events in MBSE activity diagrams (e.g., quarterly vs. bi-weekly vs. monthly).
MOE 5	Compliance Integration	Measured by the number of certification artifacts automatically generated from MBSE traceability exports, demonstrating alignment between model verification and compliance deliverables.
MOE 6	Defect Detection Accuracy	Measured by the number of defects detected after verification stages as tracked in MBSE validation reports (e.g., ≤ 1 per release).

By embedding these MOEs directly into SysML artifacts within MSOSA, MBSE provided the structured framework to quantify performance across Waterfall, Agile, and Hybrid approaches, ensuring that evaluations were traceable, consistent, and comparable. This case design ensures analytic depth by embedding methodological comparisons within one bounded system. The approach provides insights into how MBSE enabled methodologies can be tailored for aerospace sustainment contexts.

To quantify the MOEs for Waterfall, Agile, and Hybrid, raw SILA project data was converted into proportional percentages using the below formulas. The goal of these calculations was to normalize metrics of different types so they could be compared on the same scale.

Requirement Stability (MOE 1) was calculated as the proportion of requirements that remained unchanged during a development cycle, see Equation 1.

$$Requirement\ Stability = \frac{Number\ of\ unchanged\ requirements}{Total\ requirements\ in\ cycle} \quad 1$$

Cost and Rework Reduction (MOE 2) was measured by comparing the number of rework tasks recorded in each methodology against the baseline rework observed in Waterfall, see Equation 2.

$$Cost\ and\ Rework\ Reduction = \frac{Rework\ tasks\ in\ method}{Baseline\ methodology\ rework\ tasks} \quad 2$$

Verification Cycle Duration (MOE 3) was calculated by converting the average number of weeks per verification loop into a proportional speed metric, see Equation 3.

$$Verification\ Cycle\ Duration = \frac{1}{Average\ verification\ cycle\ time\ (weeks)} \quad 3$$

Stakeholder Collaboration Frequency (MOE 4) was calculated by counting the number of review events recorded in the MBSE activity diagrams for each method, see equation 4.

$$Stakeholder\ Collaboration\ Frequency = \frac{Number\ of\ stakeholder\ reviews}{Maximum\ reviews} \quad 4$$

Compliance Integration (MOE 5) was measured by calculating the proportion of required certification artifacts that were generated automatically from MBSE traceability exports, see Equation 5.

$$Compliance\ Integration = \frac{Generated\ compliance\ artifacts}{Total\ required\ artifacts} \qquad 5$$

Defect Detection Accuracy (MOE 6) was computed as the proportion of defects identified before system integration, based on validation reports and SIL test logs, see Equation 6.

$$Defect\ Detection\ Accuracy = \frac{Defects\ caught\ preintegration}{Total\ defects\ recorded} \qquad 6$$

6.2.2. Modeling SILA in MSOSA

This paper models a single subsystem (i.e., the SILA propulsion diagnostic subsystem) under three methodological approaches: Waterfall, Agile, and Hybrid, each applied within comparable sustainment deliverables from the SILA program. This study draws on actual project data and verifies artifacts developed for SILA, ensuring that findings reflect variations in process performance. For the purposes of this research, the same subsystem was analyzed under all three methodologies to enable direct comparison. The SILA system was represented in MSOSA using artifacts that defined system requirements, captured use cases, and mapped component-level interactions. Each methodology was modeled to capture how requirements, verification activities, and stakeholder interactions progress across its lifecycle. By modeling the methods, the study enabled a direct, model-based comparison of how each approach governs requirement evolution, verification timing, and stakeholder collaboration. The methodology serves as an enabler of MBSE by defining both the nature of artifacts to be produced and the structured sequence in which system engineering activities are executed.

All model views presented in this paper were created in MSOSA (version 2026x R1), which serves as the underlying modeling framework through its containment structure and integrated model repository. Within this framework, standardized modeling practices from the MagicGrid and Object-Oriented Systems Engineering Method (OOSEM) methodologies were applied to decompose stakeholder needs into system requirements, behaviors, and verification elements. OOSEM was originally developed through collaboration between Lockheed Martin Corporation and the Systems and Software Consortium (SSCI) and was later refined and standardized under INCOSE guidance. The OOSEM methodology follows a top-down, scenario-driven approach that leverages SysML to analyze, specify, design, and verify complex systems. This approach supports consistent integration of software, hardware, and verification activities across the system lifecycle (Friedenthal et al., 2015).

The MagicGrid methodology was developed in response to industry demand for a more structured yet practical way to apply MBSE using SysML. It incorporates best practices from real-world projects while remaining flexible enough to adapt to different organizational and program needs. MagicGrid is organized as a matrix composed of Domains and Pillars, which guide system development activities rather than define a standalone framework. The methodology defines three primary domains: Problem, Solution, and Implementation. Each domain represents a distinct stage of the system lifecycle. The process begins in the Problem Domain, where stakeholder needs and system requirements are analyzed and captured in SysML models, which then inform the development of solutions in subsequent phases (Aleksandravičienė et al., 2021).

In alignment with the three foundational pillars of MBSE (i.e., tool, method, and language), this research demonstrates a formally structured implementation of MBSE within the SILA case study. MBSE is defined in this work as the application of modeling to support requirements development, architectural design, analysis, verification, and validation activities across the system lifecycle. From a tool perspective, Cameo served as the modeling environment used to construct, manage, and maintain the system model repository. The model functioned as the technical baseline, enabling requirements traceability, subsystem decomposition, interface definition, and workflow analysis. Model elements were interrelated through defined dependencies, ensuring structural consistency across views.

From a methodological standpoint, model development was guided by the MSOSA framework. MSOSA aligns conceptually with established MBSE methods such as MagicGrid by emphasizing early System-of-Context definition, stakeholder identification, external interface characterization, capability decomposition, and requirements allocation across subsystems. This structured sequencing ensured that model development followed a repeatable engineering process. The language pillar was implemented using SysML v1.7. Requirements diagrams traced high-level sustainment capabilities to derived and allocated subsystem requirements, enabling bidirectional traceability. Block definition diagrams (BDDs) captured structural hierarchies and interface connectivity, while activity diagrams represented sustainment workflows and operational processes. Through the integrated application of Cameo (tool), aligned with MagicGrid principles (method), and SysML v1.7 (language), this research demonstrates a process-based execution of SILA.

The model view in Figure 12 shows a Block Definition Diagram (BDD) that captures the structural relationships among aircraft, flights, sensors, telemetry frames, and diagnostic elements within the SILA propulsion diagnostic subsystem, independent of Waterfall, Agile, and Hybrid methodologies. The subsystem was modeled in MSOSA to define system structure and preserve traceability across aircraft metadata, sensor inputs, diagnostic logic, and operational artifacts, providing a coherent framework for linking stakeholder needs to system elements. At a structural level, the Aircraft block is associated with Flights and Sensors, while Telemetry Frames represent recorded mission data generated by sensors during flight operations. Diagnostic Logic, Fault Events, Anomaly Logs, and Maintenance Actions are modeled as distinct system elements that interact through associations rather than composition. Composition relationships are limited to true ownership boundaries, while associations are used to indicate triggering, recording, and evaluation relationships without implying lifecycle dependency. This modeling approach reflects the fault events, anomaly logs, and maintenance actions may exist independently of any single log instance, even though they remain traceable within the overall diagnostic architecture.

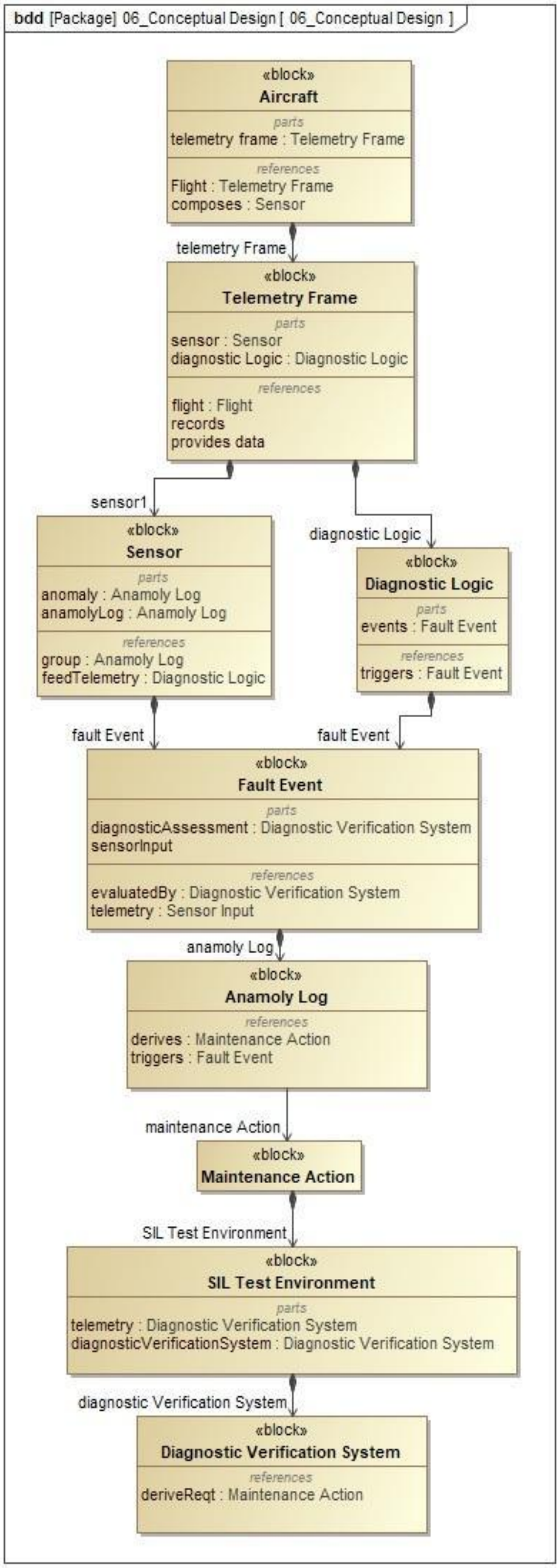


Figure 12. SILA Propulsion Subsystem Block Definition Diagram (BDD)

This structural view highlights how the subsystem integrates aircraft metadata, telemetry, and diagnostic outputs into a unified framework, supporting both operational responsiveness and compliance needs. The use of SysML modeling in MSOSA aligns stakeholder requirements for low-latency cockpit alerts, predictive diagnostics, compliance traceability, and lifecycle cost reduction with explicit data structures (Bone, 2010).

6.3. Results

6.3.1. SILA Case Study Data

The resulting allocation matrix (Table 13) demonstrates how each requirement (R1- R6) maps to specific Measures of Effectiveness (MOE1- MOE6), each supported by evidence from project data and MBSE analysis. The "Outcome" column separates findings by method clarifying how Waterfall, Agile, and Hybrid each performed relative to the same subsystem metrics. The percentages were derived from 8 months of SILA subsystem activity, including test logs, defect reports, verification schedules, and MBSE model exports. Each MOE value reflects metrics defined in Section 2 and was computed directly from recorded project artifacts and logs, then expressed as proportional values to support comparison across methodologies. Converting all of these raw counts and durations into percentages made it possible to evaluate Hybrid, Agile, and Waterfall on a consistent scale while preserving the relative performance differences observed in the SILA dataset. Quantitative measures such as rework cost, verification cycle duration, and stakeholder review frequency were obtained from actual program metrics as described in Table 3, while qualitative insights were drawn from project documentation and team debriefs.

Table 13. Allocation Matrix Between SILA Requirements and MOEs

Requirement	MOE	Agile Outcome	Waterfall Outcome	Hybrid Outcome
R1 Requirement Stability	MOE1 Requirement Stability	**75%** Stability achieved through continuous iteration and evolving baselines.	**85%** Stable baseline maintained, but inflexible to requirement changes.	**90%** High baseline stability with 2–3 controlled updates per cycle.
R2 Cost and Rework Reduction	MOE2 Cost and Rework Reduction	**50%** Rework reduced through early defect discovery and iterative development.	**20%** Higher rework costs driven by late defect discovery and sequential validation.	**60%** Rework reduced through reuse of validated models and controlled iteration.
R3 Verification Cycle Duration	MOE3 Verification Cycle Duration	**2 weeks** Short verification cycles enabled by sprint-based integration testing.	**4 weeks** Longer verification cycles due to sequential reviews.	**2.5 weeks** Accelerated verification using phased model reuse within gates.
R4 Stakeholder Collaboration Frequency	MOE4 Stakeholder Collaboration Frequency	**98%** Bi-weekly reviews ensured continuous stakeholder feedback.	**17%** Quarterly reviews limited timely feedback and delayed issue discovery.	**50%** Monthly checkpoints balanced responsiveness with governance.
R5 Compliance Integration	MOE5 Compliance Integration	**55%** Compliance artifacts generated incrementally, requiring supplemental steps.	**30%** Compliance evidence generated after development completion.	**99%** Traceable compliance artifacts generated at gated reviews.
R6 Defect Detection Accuracy	MOE6 Defect Detection Accuracy	**90%** Most defects detected pre-integration through continuous testing.	**50%** Fewer defects detected before integration, resulting in higher rework costs.	**85%** High pre-integration detection with minimal late-stage defects.

6.3.2. Measure of Effectiveness

The design process MOE Block Definition Diagram (Figure 13) defines a common measurement structure used to evaluate development effectiveness across different execution approaches. At the top of the model is a MOE Holder block, which captures the primary Measures of Effectiveness applicable to safety-critical aerospace programs, including requirement stability, cost and rework reduction, compliance integration, stakeholder collaboration, verification cycle duration, and defect detection accuracy. Each MOE is defined as a quantitative or time-based metric, enabling consistent comparison across methodologies. The MOE structure is intentionally process-agnostic. The diagram establishes a shared set of effectiveness measures that can be populated with values derived from project data.

SILA is used as the empirical basis for populating the MOE values shown in the results. Project data from the SILA propulsion diagnostic subsystem, including requirement changes, verification timing, rework effort, stakeholder review frequency, compliance artifacts, and defect detection results, were mapped to the MOE definitions defined in the design process diagram. This approach allows the same MOE framework to be applied consistently while ensuring that comparisons between Waterfall, Agile, and Hybrid execution reflect observed performance in a real sustainment context.

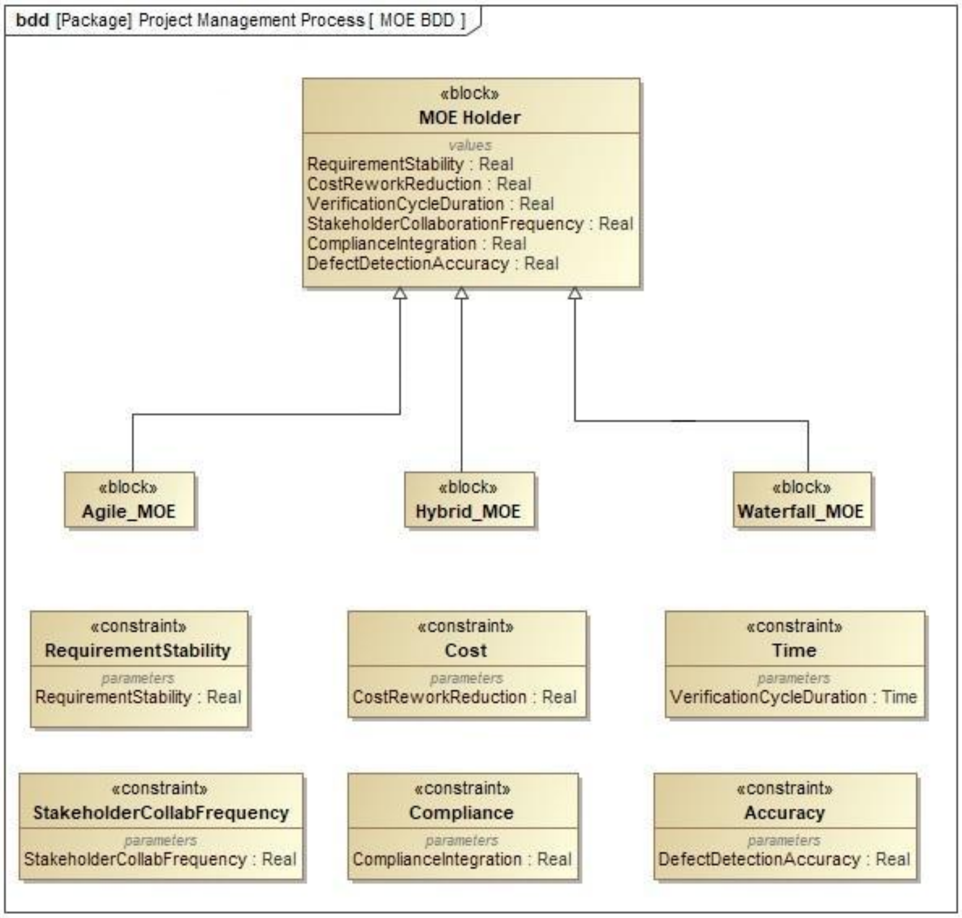


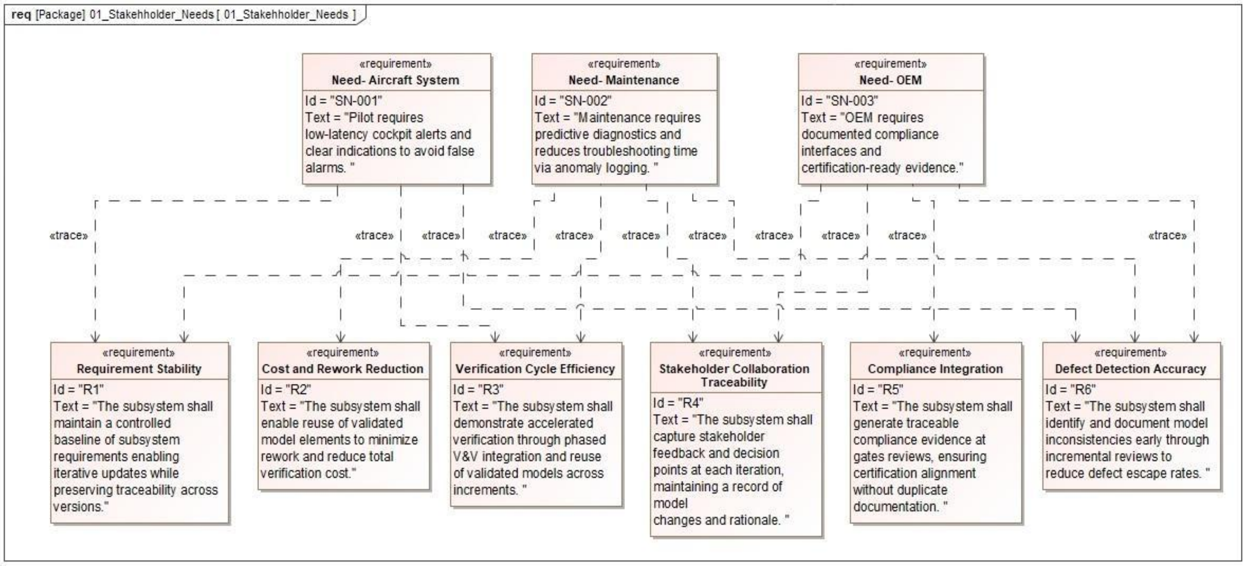
Figure 13. Design Process Measures of Effectiveness (MOE)

6.3.3. Measure Modeling the SILA Propulsion System in MSOSA

The MBSE framework was used to integrate Waterfall, Agile, and Hybrid development processes within the SILA propulsion diagnostic subsystem by providing a structure for analyzing stakeholder needs and tracing them to system behavior. Stakeholders' needs represent the operational expectations of pilots, ground maintenance personnel, OEM suppliers, and sustainment customers. These needs include low-latency cockpit alerts, compliance-ready maintenance outputs, lifecycle cost control, and reliable, traceable iteration requirements that directly influence how the propulsion diagnostic system must perform. Each development approach satisfies these needs in different ways. MBSE, implemented using MSOSA, serves as the unifying mechanism that connects all Agile, Waterfall and Hybrid development approaches to stakeholder needs. This framework includes requirement representations, activity flows, allocation matrices, and traceability mapped to the Measures of Effectiveness. Through this shared modeling environment, the same stakeholder needs can be evaluated consistently across Waterfall, Agile, and Hybrid execution, while revealing how different process choices affect requirement stability, verification timing, compliance integration, and overall system effectiveness.

The requirements diagram in Figure 14 shows how the 6 SILA subsystem requirements (R1–R6) were formally derived from the three primary stakeholder needs groups: Aircraft System (SN-001), Maintenance (SN-002), and OEM Supplier (SN-003). The diagram demonstrates a clear hierarchical flow, where each stakeholder need is connected through satisfy relationships to specific, measurable subsystem requirements through trace relationships that indicate evaluative support as shown in Table 14. For example, the pilot's need for low-latency cockpit alerts (SN-001) is supported by requirements related to verification cycle duration (R3) and defect detection accuracy (R6). Faster verification cycles enable earlier validation of alert behavior, while higher defect detection accuracy reduces false or missed alerts, both of which directly influence alert reliability and timeliness. Maintenance needs for predictive diagnostics and reduced troubleshooting time (SN-002) are supported by requirements associated with cost and rework reduction (R2), verification efficiency (R3), stakeholder collaboration frequency (R4), and compliance integration (R5), reflecting the operational importance of efficient diagnostics, coordinated feedback, and compliant maintenance outputs. Similarly, OEM needs for compliance-ready documentation and interface stability (SN-003) are supported by requirements addressing requirement stability (R1), cost and rework reduction (R2), stakeholder collaboration frequency

(R4), and compliance integration (R5), all of which contribute to controlled change management and certification readiness.



«requirement»
Need- Aircraft System
Id = "SN-001"
Text = "Pilot requires low-latency cockpit alerts and clear indications to avoid false alarms. "

«requirement»
Need- Maintenance
Id = "SN-002"
Text = "Maintenance requires predictive diagnostics and reduces troubleshooting time via anomaly logging. "

«requirement»
Need- OEM
Id = "SN-003"
Text = "OEM requires documented compliance interfaces and certification-ready evidence."

«requirement»
Requirement Stability
Id = "R1"
Text = "The subsystem shall maintain a controlled baseline of subsystem requirements enabling iterative updates while preserving traceability across versions."

«requirement»
Cost and Rework Reduction
Id = "R2"
Text = "The subsystem shall enable reuse of validated model elements to minimize rework and reduce total verification cost."

«requirement»
Verification Cycle Efficiency
Id = "R3"
Text = "The subsystem shall demonstrate accelerated verification through phased V&V integration and reuse of validated models across increments. "

«requirement»
Stakeholder Collaboration Traceability
Id = "R4"
Text = "The subsystem shall capture stakeholder feedback and decision points at each iteration, maintaining a record of model changes and rationale. "

«requirement»
Compliance Integration
Id = "R5"
Text = "The subsystem shall generate traceable compliance evidence at gates reviews, ensuring certification alignment without duplicate documentation. "

«requirement»
Defect Detection Accuracy
Id = "R6"
Text = "The subsystem shall identify and document model inconsistencies early through incremental reviews to reduce defect escape rates. "

Figure 14. SILA Stakeholder Needs and MOE Requirement Traceability

Table 14. Mapping of Stakeholder Needs to SILA Requirements

Stakeholder Need	**R1 Requirement Stability**	**R2 Cost & Rework Reduction**	**R3 Verification Cycle Duration**	**R4 Stakeholder Collab Traceability**	**R5 Compliance Integration**	**R6 Defect Detection Accuracy**
SN-001: Aircraft System (Pilot)	Applicable	Not Applicable	Applicable	Not Applicable	Not Applicable	Applicable
SN-002: Maintenance	Not Applicable	Applicable	Applicable	Applicable	Not Applicable	Applicable
SN-003: OEM Supplier	Applicable	Not Applicable	Not Applicable	Applicable	Applicable	Applicable

6.3.4. System Context

The System Context diagram as shown in Figure 15 defines the environment in which the SILA propulsion diagnostic subsystem operates. SILA is shown as part of a larger corporate structure, indicating that all development activities operate under shared constraints regardless of whether Agile, Hybrid, or Waterfall methods are used. The diagram identifies the core roles involved in SILA development, which remain constant across all methodologies and provide a stable organizational baseline for comparing process behavior. The aircraft is included as an external operational system to show where SILA's diagnostic inputs and outputs originate and are applied. Telemetry from the aircraft, particularly from the propulsion system, drives SILA analysis, while SILA outputs support maintenance and sustainment decisions. By holding the system context constant, the diagram supports a fair comparison of how different development methods affect requirements management and verification within the same organizational and operational environment.

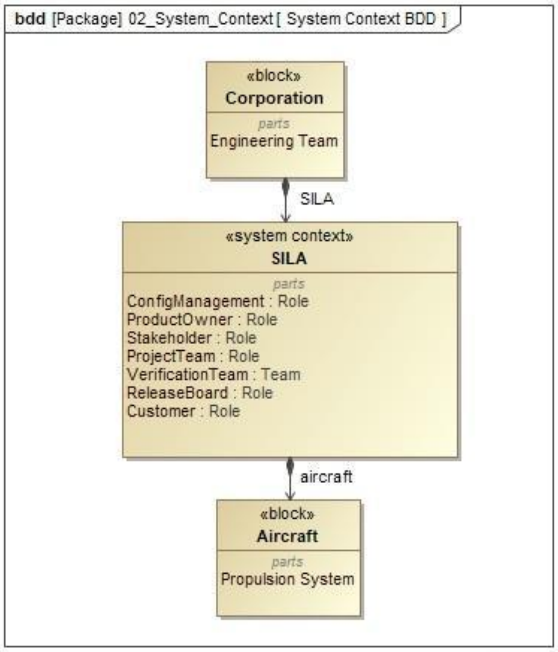


Figure 15. System Context

6.3.5. Use Case Diagrams

The design process use case diagram shown in Figure 16 is high level and provides a reference point for how system design work can be carried out using Agile, Hybrid, or Waterfall processes. The Engineering Team represents the individuals responsible for carrying out design and analysis tasks, while the Stakeholder represents those who provide input, review progress, and accept outcomes. Each use case reflects a different way of organizing the same underlying design responsibility. Agile, Hybrid, and Waterfall are shown as alternative approaches to executing design work. Sequencing, iteration, and formal governance steps are explored later through the activity diagrams, where differences in process behavior become explicit. When applied to SILA, the same structure holds, with the Engineering Team representing SILA diagnostic developers and the Stakeholder representing product owners, sustainment leadership, and customers. The diagram establishes a common conceptual baseline that supports comparison across development methodologies before introducing workflow-level detail.

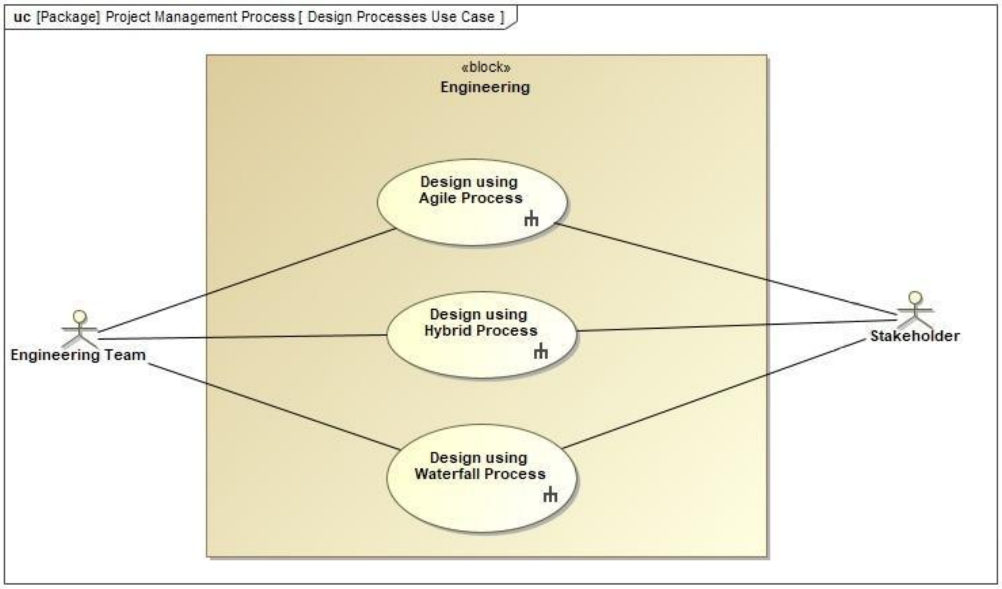


Figure 16. Design Process Use Case Diagram

6.3.6. Activity Diagrams

The activity diagrams for Agile, Hybrid, and Waterfall show the operational differences within the design process by modeling how workflows span the Product Owner, Engineering Team, and Stakeholder (Customer) swim lanes. The diagrams represent a common set of core actions that occur across all three approaches. These shared actions include requirement intake, engineering analysis and design, implementation, verification and validation, and communication of results to stakeholders. Establishing this common baseline reflects fundamental systems engineering practices that are independent of development methodology.

Each diagram represents a distinct pattern of coordination, decision-making, verification, and communication. Building on the shared baseline, the diagrams show how the same actions are sequenced, iterated, or gated differently across the three approaches. Differences emerge in feedback frequency, decision authority, stakeholder involvement, and verification timing. Together, these models demonstrate how iterative, gated, or sequential behaviors fundamentally shape system development outcomes. The figures demonstrate how development methodology influences stakeholder interactions, engineering decision-making, verification placement, and the overall flow of requirements through the lifecycle.

6.3.6.1. Agile Activity Diagram

The Agile Design Process Activity Diagram (Figure 17) illustrates the cyclical and feedback-driven nature of Agile development by showing how work continuously moves between the Product Owner, Engineering Team, and Stakeholder. In the context of SILA, the process starts with the Product Owner prioritizing the backlog using telemetry data, defect history, and feedback from earlier iterations. Once the backlog is agreed on, the Engineering Team plans and executes a sprint. Each sprint delivers small, testable updates such as changes to fault-detection logic, simulation behavior, or telemetry-driven diagnostic algorithms, which are developed and tested in the SILA test environment. Unlike Waterfall, reporting is not a separate step. Test results, logs, and status information are captured continuously through sprint activities and tooling, which is why the diagram does not include explicit “Generate Report” or “Evaluate Report” actions. Stakeholders review each increment, examine diagnostic outputs and fault logs, and provide feedback that either feeds the next sprint or confirms the increment as acceptable. All Agile artifacts are tracked using the Atlassian toolset to maintain traceability across iterations. Overall, the diagram emphasizes short verification loops, frequent stakeholder feedback, and early use of operational data, which allows SILA to adapt quickly while reducing late-stage rework.

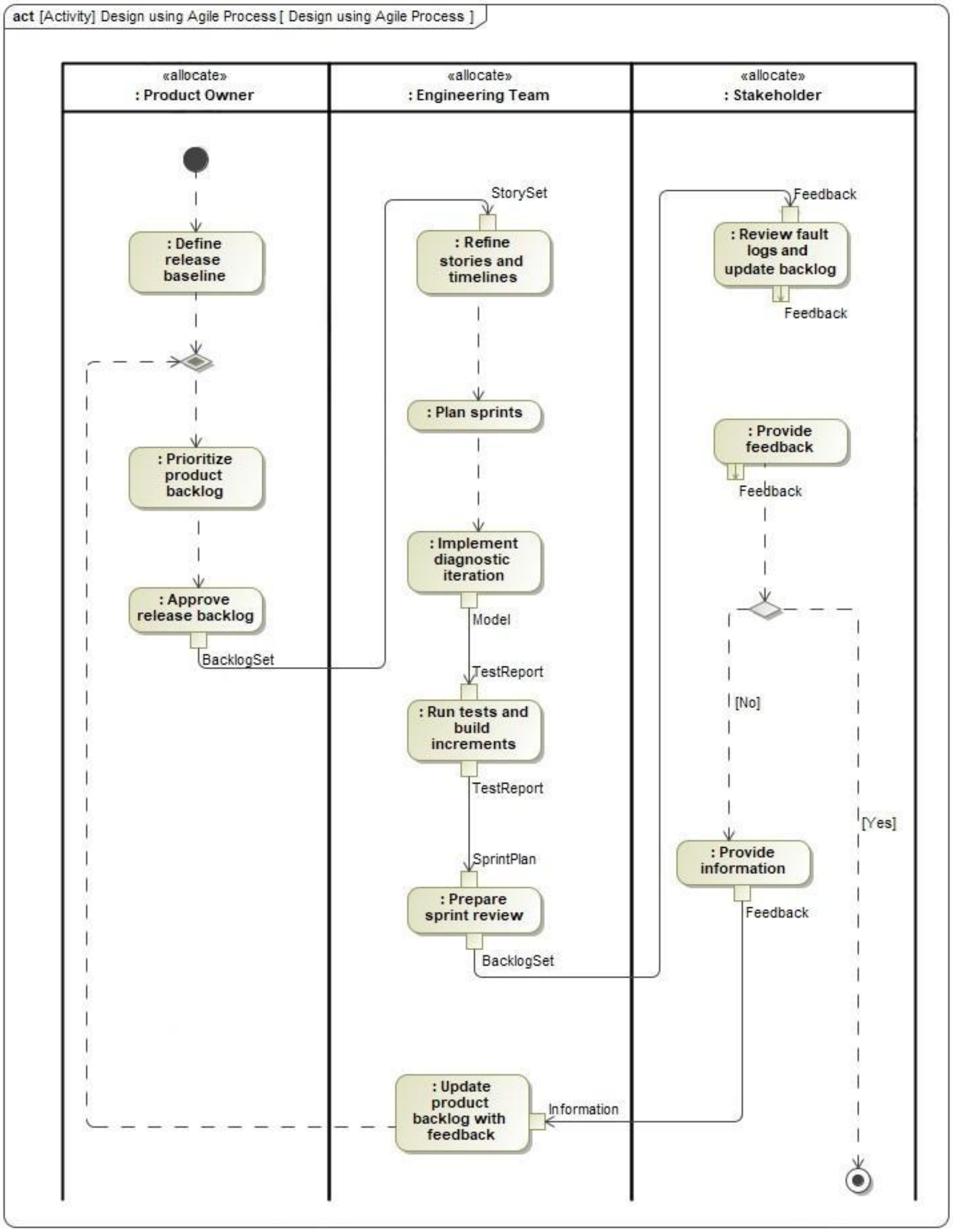


Figure 17. Agile Design Process Activity Diagram

6.3.6.2. *Waterfall Activity Diagram*

The Waterfall Design Process Activity Diagram as shown in Figure 18 represents a traditional, linear development approach where work progresses through fixed stages with little opportunity for iteration. The process begins with the Product Owner defining the release baseline, which sets the scope for the entire development cycle. For SILA, once this baseline is established, the Engineering Team executes a predefined sequence of activities, including loading diagnostic data into the test bench, integrating the system, running tests in a controlled environment, and generating formal test reports. Unlike Agile, reporting in Waterfall is modeled as a distinct and explicit activity. Steps such as “Generate Report” and “Evaluate Report” reflect the formal documentation and assessment required at phase completion to demonstrate readiness and compliance. For SILA, these reports include diagnostic results, verification checklists, and compliance evidence produced only after full test execution. Artifacts move strictly forward through the process, and earlier stages are not revisited once completed. Stakeholders become involved after engineering activities are complete. They perform structured verification using documented procedures and compliance criteria, followed by a formal release review. If issues are identified, additional testing is performed, but the overall process structure remains unchanged. Once accepted, SILA verification and compliance artifacts are compiled into a certification-ready package and archived in configuration-controlled repositories such as SharePoint and internal document libraries. Overall, the diagram shows the predictability and documentation focus of the Waterfall approach. In the SILA context, this structure provides strong control and traceability, but it also highlights a key limitation: defects discovered late in the cycle often require repeating large portions of the process, increasing schedule and rework costs.

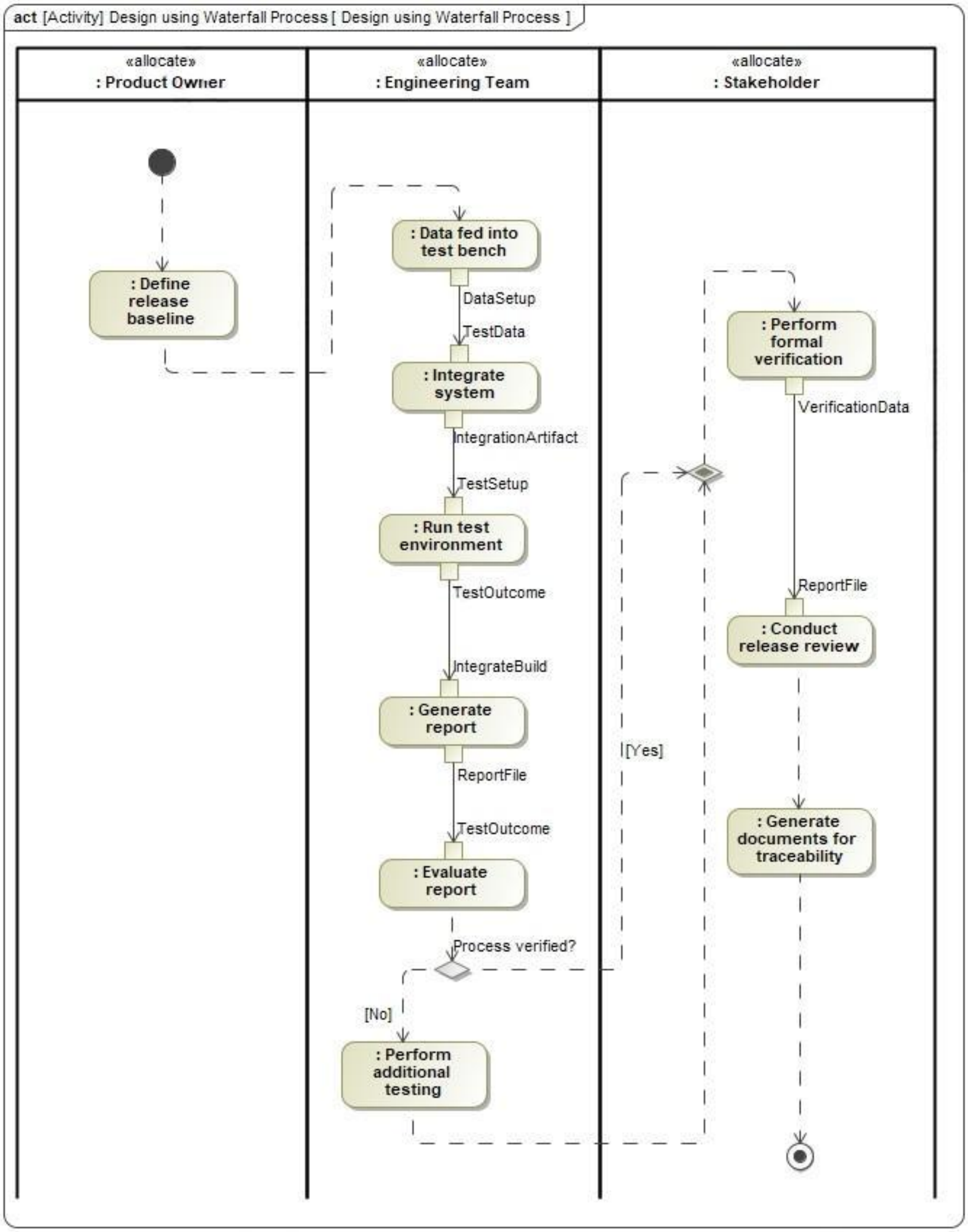


Figure 18. Waterfall Design Process Activity Diagram

6.3.6.3. Hybrid Activity Diagram

The Hybrid Design Process Activity Diagram (Figure 19) shows how the team blends structured, stage-based controls with flexible iteration loops. The process begins with the Product Owner defining the release baseline and prioritizing the backlog, establishing upfront scope similar to Waterfall. Unlike Waterfall, however, requirements are not locked for the entire project. Instead, the Engineering Team works through short, controlled iterations to refine requirements, develop diagnostic updates, and perform testing. The core engineering activities in Hybrid remain the same as those used in Agile and Waterfall, including requirement refinement, diagnostic development, testing, verification, stakeholder review, and release preparation. Each iteration includes an explicit compliance check that acts as a lightweight gate, confirming that models, test results, and artifacts meet expected standards before work progresses. These checks do not introduce new engineering tasks but rather make governance and verification activities explicit earlier in the lifecycle. After several iterations, the team conducts a formal gate review. This review blends Agile's incremental progress with Waterfall's structured oversight. Incremental test results and logs are generated continuously during iterations, while formal reports and compliance artifacts are produced at gate reviews. Stakeholders participate at these gates to review results, provide feedback, and determine release readiness. If issues are identified, feedback returns to the backlog for the next iteration. If approved, verification evidence and traceability artifacts are packaged and archived. When applied to SILA, the Hybrid approach follows the same diagnostic workflow used in Agile and Waterfall but reorganizes when verification, compliance, and stakeholder coordination occur. Diagnostic updates are developed and tested incrementally using the SILA test environment, while certification evidence is assembled at defined gates. This structure allows SILA to adapt to evolving operational data while maintaining the documentation rigor required for certification, demonstrating that Hybrid execution reorders existing activities rather than adding new ones.

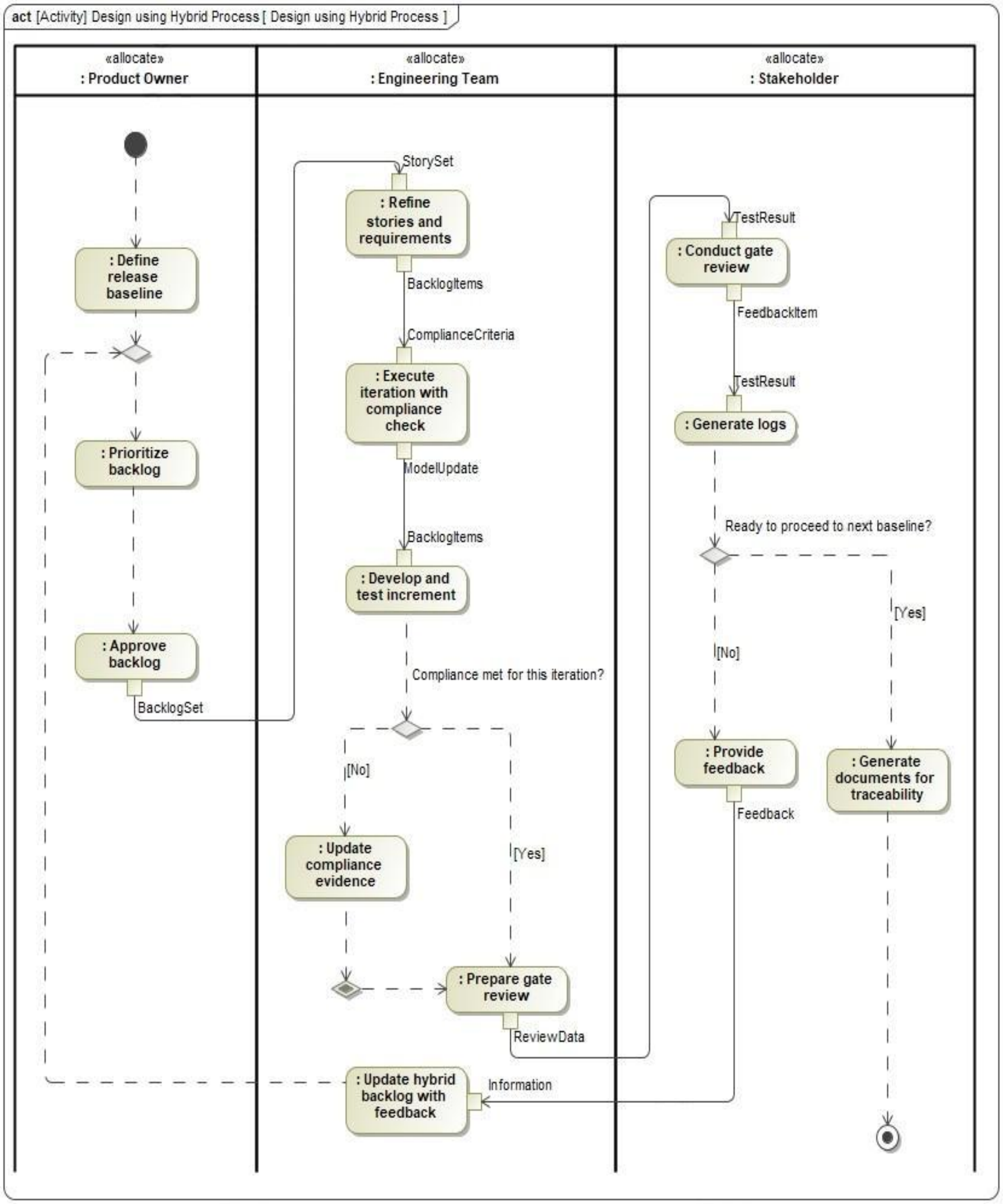


Figure 19. Hybrid Design Process Activity Diagram

6.4. Discussion

The SILA case study shows how MBSE can show meaningful differences between Waterfall, Agile, and Hybrid methods in an aerospace sustainment environment. MBSE gave the project a single modeling baseline, so performance changes came from process differences and not subsystem changes. This strengthened the comparison and supports the growing view of MBSE as a "single source of truth" for A&D programs (Call et al., 2023). The 6 MOEs used in this study highlight how each method affects stability, rework, verification timing, collaboration, compliance, and defect detection. The three methods also produced different types of deliverables. Agile generated many small outputs, such as model updates, logs, and sprint artifacts that are stored in Jira, Confluence, and Bitbucket. Waterfall produced fewer but much larger packages stored in SharePoint and controlled document repositories. Hybrid created both iterative outputs and formal gate documents. These differences influenced how quickly SILA could update diagnostic logic, how often teams interacted with stakeholders, and how early defects were found.

6.4.1. Waterfall's Compliance Strengths and Responsiveness Limits

The Waterfall version of SILA confirmed well-known strengths: stable baselines, predictable documents, and clear certification paths (Ijomah, 2024). About ≥ 90% of requirements were frozen by design review, giving the team a strong audit trail. This aligns with Elm and Goldenson (2012), who argue that document-driven stability supports compliance but limits evolution. The SILA data showed the same pattern. Fixed baselines kept the model consistent but made it harder to correct issues that appeared late in testing. Verification cycles stretched to about four weeks, and rework increased when diagnostic faults were discovered near the end. The sequential "deriveReqt" and "satisfy" relationships also restricted mid-cycle changes, echoing

Boehm and Basili's (2001) findings on the cost of late defects. In summary, Waterfall supported certification but struggled with fast-changing operational needs.

6.4.2. Agile's Responsiveness and Certification Gaps

The Agile configuration emphasized rapid feedback and adaptability. The SysML activity and use-case diagrams captured bi-weekly stakeholder interactions through "trace" links to requirements like ≤ 1s cockpit alerts and predictive maintenance needs. These short cycles improved responsiveness and defect detection, like findings by Hajjdiab et al. (2011). Verification loops averaged about two weeks, and continuous simulation caught ≥ 90% of defects before integration. However, Agile also showed weaknesses. Compliance evidence was produced inconsistently across sprints, and some certification requirements (R5) lacked full traceability. This support concerns raised by Maharao (2024) and Hüllmann (2024) about Agile's gaps in audit readiness. For SILA, this means Agile improves speed and quality but requires additional controls to satisfy regulatory expectations.

6.4.3. Hybrid as a Pragmatic Balance

Hybrid provided the most balanced outcome across all 6 MOEs. Its gated review points preserved baseline stability while allowing iteration between milestones. Requirement stability remained high (85–90%). Verification cycles averaged about 2.5 weeks. Monthly stakeholder reviews improved responsiveness without creating excessive coordination overhead. Compliance performance was especially strong: automated MBSE exports produced 99% of required traceability artifacts. Defect detection reached about 85%, close to Agile levels but with higher audit confidence. These results support earlier findings by Madni & Sievers (2018) and Dori (2016), who note that Hybrid MBSE can reduce technical debt by aligning iteration with verification. For SILA, Hybrid showed that a governed iterative process can support both adaptability and certification.

The emergence of Hybrid execution within the SILA case aligns with prior findings from C5ISR sustainment programs, where practitioners reported that neither fully sequential nor fully iterative approaches were sufficient for managing long-term system evolution (Chellin & Miller, 2023; Chellin & Miller, 2024). Instead, effective obsolescence management required structured baselining combined with iterative reassessment, mirroring the Hybrid patterns observed in this study. Within SILA, MBSE artifacts served as a critical enabler of this balance by supporting controlled iteration while preserving traceability and compliance across evolving requirements.

6.4.4. Broader Implications for A&D Programs

The SILA results provide insights for broader A&D program management. Waterfall's quarterly reviews created long feedback gaps. Agile's bi-weekly reviews were fast but sometimes overwhelming for teams and stakeholders. Hybrid's monthly rhythm provided timely feedback while keeping workloads manageable. This supports the view that methodology is not only a technical choice but also an organizational one that affects collaboration, governance, and culture (Wenyue et al., 2022).

SILA also shows that MBSE can unify different methodologies under a single modeling framework. The SysML structure was nearly identical across all three methods; what changed was iteration speed, review cadence, and change control. This suggests organizations can adopt MBSE as the backbone of their engineering process while tailoring the surrounding governance. This aligns with recommendations from Estefan (2008) and Wang (2020), who argue that MBSE should be treated as an enabler of methodological integration. Hybrid MBSE may be especially valuable for sustainment programs. By embedding compliance checks into iterative cycles, it reduces both certification delays and technical debt. This matches trends in digital engineering and digital thread research, which emphasize keeping compliance evidence synchronized with model updates (Hecht et al., 2021; Hashimoto et al., 2025). For long-lifecycle systems like aircraft propulsion, this approach supports continuous updates without losing certification integrity.

SILA fills a gap in literature by providing a direct comparison of Waterfall, Agile, and Hybrid within the same sustainment environment. Most prior studies examine these methods separately. By applying all three to the same subsystem, SILA quantifies how each method affects requirement stability, rework, verification timing, collaboration frequency, and compliance integration. These findings offer practical guidance for program managers, systems engineers, and regulators. Hybrid scheduling structures may offer the best balance, combining Agile sprints with Waterfall milestones. MBSE artifacts can help synchronize stakeholder collaboration with governance needs. Regulators may also need to adjust oversight frameworks to recognize Hybrid MBSE as a valid, verifiable compliance path.

6.5. Conclusion

The SILA case study shows that modern aerospace sustainment cannot rely on a single development methodology. Waterfall, Agile, and Hybrid each produced different outcomes when modeled through MBSE, and those differences are showcased in the performance data. Waterfall provided predictable documentation, stable baselines, and strong certification artifacts. However, it also locked requirements too early and created long verification cycles, which made late changes expensive and slowed responsiveness. Agile delivered the opposite pattern: fast verification loops, frequent stakeholder reviews, and high defect detection rates. But without added structure, it struggled to generate consistent compliance evidence.

Recognizing these limitations, A&D organizations are moving toward hybrid process structures (Gracias & Gallegos, 2025). The SILA results support this trend. The Hybrid model maintained high requirement stability (85-90%), reduced rework by ~60%, achieved 99% compliance integration through automated MBSE exports, and delivered verification cycles shorter than Waterfall but more controlled than Agile. These outcomes came from blending Agile-style iteration with Waterfall-style gated reviews. Within SILA, MBSE acted as the integrating framework that kept requirements, stakeholder needs, and verification activities aligned across all three methodologies. This allowed a direct comparison using the same subsystem and eliminated ambiguity about what caused performance differences. MBSE can function as the "digital backbone" for complex system development (Dori, 2016). SILA provides empirical support for that claim by showing that hybrid MBSE practices preserved audit rigor while cutting rework by more than half compared to Waterfall. These findings carry important implications for program leadership. Governance structures should place MBSE at the center and allow agility where it adds value, while maintaining the discipline needed to satisfy certification requirements. As others have argued, document-driven processes alone cannot support this pace of change; programs need adaptability, but not at the expense of mission assurance (Wenyue et al., 2022).

SILA shows how hybrid MBSE can help sustainment programs modernize. Engineers can use SysML artifacts not only to capture requirements but also to coordinate stakeholders and manage tradeoffs between speed and stability. Managers can design roadmaps that combine iterative learning with milestone reviews. Certification authorities can begin recognizing hybrid MBSE pathways as credible compliance mechanisms rather than relying solely on static documents. MBSE-enabled hybrid approaches offer this balance. They do not dilute the strengths of Waterfall or Agile; instead, they combine them into a process that fits the realities of long lifecycle, safety-critical work. Rather than a compromise, hybrid MBSE represents a practical strategy for achieving technical rigor, regulatory confidence, and operational agility within the same development framework. It provides a realistic blueprint for how future sustainment efforts can meet mission needs while supporting continuous innovation.

Chapter 7. Conclusion

7.1. Overview

This dissertation contributes to advancing knowledge on Hybrid development methodologies integrated with MBSE in aerospace and defense sustainment projects, using the SILA (Sustainment Integration for Legacy Aircraft) system as a representative case. Through surveys, interviews, and case study analysis, the research provides an empirical comparison of Waterfall, Agile, and Hybrid approaches, showing that Hybrid balances regulatory compliance with adaptability by combining upfront planning and gated reviews with iterative validation and stakeholder feedback. The research defines Hybrid as a distinct methodology modeled through MBSE artifacts, identifies bottlenecks and enablers in adoption, and develops a framework for tailoring Hybrid practices to subsystem criticality, project scope, and compliance requirements. This conclusion chapter consolidates these contributions, presents peer-reviewed publications, and addresses the research questions established at the outset of the document.

7.2. Research Contributions

This dissertation makes several contributions to advancing knowledge on the application of Hybrid development methodologies supported by MBSE in A&D sustainment projects. In environments characterized by stringent compliance requirements and evolving operational needs, Hybrid approaches blending elements of Waterfall and Agile offer a pragmatic balance between structure and adaptability. This research contributes by modeling and evaluating Hybrid methods within the context of the SILA system. A central contribution of this research is the explicit definition, modeling, and empirical evaluation of what constitutes a Hybrid methodology within an A&D sustainment context, grounded in real project data and practitioner experience.

7.2.1. Comparison between Waterfall, Agile, and Hybrid Methodologies

The first contribution is a comparative evaluation of Waterfall, Agile, and Hybrid methodologies with respect to requirements management, cost and schedule performance, and change control. Using survey data, interviews, and the SILA system case study, this dissertation quantitatively and qualitatively assesses how each methodology impacts requirement stability, verification timing, stakeholder collaboration, and rework. The findings demonstrate that while Waterfall provides strong upfront control and Agile enables responsiveness, Hybrid approaches deliver measurable improvements by reducing rework, improving verification efficiency, and enabling controlled adaptability across the project lifecycle.

7.2.2. Definition of Hybrid Methodology for A&D Sustainment

The second contribution is the characterization and definition of Hybrid as a distinct methodology for A&D sustainment projects. This research defines Hybrid as a structured development approach in which system-level planning, baseline requirements, and gated compliance reviews are governed by Waterfall principles, while iterative development, incremental verification and validation (V&V), and continuous stakeholder feedback are governed by Agile practices. Additionally, MBSE serves as the integrating mechanism that connects these modes of execution. By embedding Hybrid practices directly into MBSE artifacts such as requirement diagrams, activity models, allocation matrices, and traceability views, this research provides a rigorous, model-based definition of Hybrid tailored for sustainment environments.

7.2.3. Organization and Bottlenecks in Hybrid Adoption

The third contribution is the identification of organizational and technical bottlenecks, as well as key enablers, associated with Hybrid adoption. The findings highlight recurring challenges, including rigid requirement baselines, fragmented feedback loops, and cultural resistance to iterative change. At the same time, the research demonstrates how Hybrid approaches mitigate these challenges through incremental V&V, model reuse, simulation-driven testing, and continuous traceability enabled by MBSE. These insights translate directly into actionable recommendations for organizations seeking to refine or modernize sustainment workflows without compromising compliance or safety.

7.2.4. MBSE-Supported Methodological Tailoring Framework

Finally, this dissertation proposes a methodological tailoring framework for sustainment projects, grounded in MBSE. The framework provides guidance on when to emphasize structured compliance activities and when to introduce iterative updates, based on factors such as subsystem criticality, project scope, risk profile, and regulatory constraints. By using MBSE artifacts as decision-support tools, the framework enables teams to apply Hybrid practices in a disciplined and repeatable manner, rather than relying on intuition or one-size-fits-all processes.

Collectively, these contributions show the Hybrid approach reinforced by MBSE not as a compromise between Waterfall and Agile, but as a deliberate, model-driven methodology that enables operational agility while preserving the rigor required for A&D programs.

7.3. Discussion of Research Questions

7.3.1. Research Aim 1 Findings

RQ1.1: How does transitioning from Waterfall to Agile, in conjunction with MBSE, affect project management, time-to-market, and overall efficiency in the aerospace and defense industry?

Answer: Survey results showed that integrating MBSE with Agile significantly improved efficiency, traceability, and responsiveness, with 74.36% of participants rating the approach as effective or very effective. Compared to Waterfall, Agile supported faster time-to-market and reduced rework, while MBSE ensured rigor and requirement clarity in high-stakes projects.

RQ1.2: What challenges arise when implementing MBSE during a transition from Waterfall to Agile in the aerospace and defense industry, and what strategies support successful adoption?

Answer: Key challenges identified included cultural resistance, legacy system constraints, and gaps in training. Strategies for overcoming these barriers included early architectural planning, strong organizational support, and targeted training to embed MBSE practices into Agile workflows.

RQ1.3: What future opportunities exist for integrating MBSE within Waterfall and Agile approaches in the aerospace and defense industry, and how can organizations maintain a competitive advantage?

Answer: Future opportunities include leveraging MBSE-enabled digital twins, tighter SysML-driven traceability, and automation of compliance documentation. These advancements will enable organizations to move beyond rigid document-driven approaches, combining Agile adaptability with Waterfall discipline for competitive advantage.

7.3.2. Research Aim 2 Findings

RQ2.1: How do Waterfall, Agile, and Hybrid approaches affect requirement changes, cost efficiency, and schedule performance in aerospace sustainment projects?

Answer: Interviews and the SILA case revealed that Waterfall struggled with frequent requirement changes, leading to overruns and rework, while Agile enabled faster adaptation but required intensive stakeholder engagement. The Hybrid approach demonstrated cost efficiency and schedule predictability by integrating Agile's adaptability with Waterfall's compliance rigor.

RQ2.2: What change management and workflow bottlenecks result from rigid or poorly integrated requirement structures across the development approaches?

Answer: Rigid requirement baselines in Waterfall delayed updates and created costly approval bottlenecks. Agile minimized these issues but occasionally overlooked long-term integration risks. The Hybrid method addressed bottlenecks by using staged review gates and model reuse, enabling structured yet flexible updates.

RQ2.3: How effectively can a Hybrid approach optimize sustainment performance by balancing adaptability, regulatory compliance, and rework reduction?

Answer: Hybrid approaches in SILA reduced rework by 60% compared to Waterfall while maintaining certification compliance. This balance allowed the program to implement iterative improvements while still meeting sustainment, safety, and cost requirements.

7.3.3. Research Aim 3 Findings

RQ3.1: How does the use of MBSE impact requirement stability, verification and validation (V&V) timing, and stakeholder collaboration when applied within Waterfall, Agile, and Hybrid development methodologies?

Answer: In the SILA system, Waterfall maintained stable requirement baselines but delayed verification and validation activities until late in the lifecycle. Agile accelerated validation and stakeholder engagement through frequent iteration but required continual requirement updates and rebaselining. The Hybrid approach balanced baseline stability with incremental verification, enabling earlier stakeholder alignment while preserving control over requirement changes.

RQ3.2: How does the use of MBSE artifacts support key Measures of Effectiveness (MOEs), including cost and rework reduction, compliance integration, and defect detection accuracy across Waterfall, Agile, and Hybrid approaches?

Answer: SysML based artifacts developed in MagicDraw established clear traceability between stakeholder needs, system requirements, and verification events. The model-based structure reduced rework by improving requirement clarity and consistency across teams. It also enabled compliance evidence to be generated more efficiently than traditional document driven methods, while supporting earlier detection of requirement and design defects.

RQ3.3: How does the integration of MBSE modeling with real sustainment data from the SILA case study reveal trade-offs among Waterfall, Agile, and Hybrid methodologies in aerospace sustainment programs?

Answer: Waterfall provided strong rigor and oversight but limited adaptability to evolving sustainment needs. Agile maximized responsiveness and iteration speed but introduced risks related to compliance and verification continuity. The Hybrid approach achieved a balanced trade-off by maintaining structured oversight while enabling controlled iteration. These results show how Hybrid MBSE practices can be tailored based on subsystem criticality and regulatory constraints, providing a practical framework for future aerospace sustainment programs.

7.4. Limitations and Future Work

While this research offers meaningful insights, several limitations should be acknowledged. First, the survey sample size was limited to 40 participants, and interviews were conducted within a single organization. Although participants represented a wide range of roles, experience levels, and leadership positions, the findings may not be fully generalizable across all A&D organizations or international contexts. Second, the SILA case study, while detailed and data-rich, represents a specific sustainment environment and may not capture the full diversity of development programs across the industry.

Additionally, the MBSE models analyzed in this research were necessarily simplified abstractions of highly complex systems. While these models were sufficient to support comparative analysis, they do not capture every technical dependency or operational nuance. Computational constraints and tool interoperability limitations also influenced the scope of model validation that could be performed.

Future research can build on this work in several important ways. Expanding the dataset across multiple organizations and programs would improve statistical robustness and generalizability. Studies could examine how MBSE-supported Hybrid approaches perform over extended program lifecycles, particularly during major upgrades or sustainment transitions. Further research is also needed to quantify the cost, schedule, and risk impacts of Hybrid MBSE implementations using standardized metrics such as EVMS data across multiple programs.

Significant opportunities also exist in advancing MBSE integration with emerging digital technologies. One of the most promising directions is the integration of MBSE with digital twins, enabling real-time synchronization between physical systems and virtual models. For systems like SILA, this integration could support predictive maintenance, proactive sustainment strategies, and real-time performance optimization. By continuously feeding operational data back into system models, organizations can reduce downtime, extend system life, and improve mission readiness.

Another major opportunity lies in the adoption of SysML v2. Compared to SysML v1, SysML v2 offers improved expressiveness, a text-based and API-driven modeling approach, and stronger interoperability with modern software and DevOps pipelines. These capabilities align closely with Agile development and digital engineering initiatives, making SysML v2 particularly well-suited for hybrid environments. Its ability to support automation, model queries, and tighter integration with verification tools positions it as a tool for future MBSE-driven programs.

Finally, tools such as MagicDraw will continue to play a central role in enabling hybrid execution. Its ability to support version-controlled models, automated traceability, and integration with both Agile and Waterfall documentation structures demonstrate how tooling can reduce the traditional tradeoff between agility and compliance. As digital engineering matures, future work should explore deeper toolchain interoperability and automated assurance mechanisms to further reduce friction between development speed and regulatory rigor.

Although this research is grounded in a single sustainment case study, its contribution extends well beyond the SILA subsystem itself. The most important takeaway is to the broader lesson that Hybrid lifecycle governance, when intentionally integrated with MBSE, can help aerospace and defense programs balance compliance with adaptability. Across the industry, organizations are not truly choosing between Agile and Waterfall, they are navigating a structural tension between regulatory rigor and evolving operational demands. Hybrid approaches, when deliberately structured and supported by disciplined model-based traceability, offer a practical way to reconcile that tension. For other defense and aerospace companies, this means that Hybrid adoption should not be treated as an informal blending of practices. Programs need clarity around where stage gates remain essential, where iteration is appropriate, and how model-based artifacts can replace or streamline traditional documentation. Cultural alignment and leadership sponsorship matter as much as process design; friction during methodological transitions often arises from ambiguity and uncertainty. When viewed at this broader level, the contribution of this research is less about a single subsystem and more about demonstrating a governance framework that is transferable across regulated engineering environments. While the statistical scope of the study is bounded, the structural principles identified, risk-informed governance, disciplined hybridization, and model-based traceability, offer practical guidance for aerospace and defense organizations.

7.5. Publications

The results of this dissertation are disseminated through three journal publications (two published and one under review) and two conference presentations (one completed and one accepted). These publications are described below.

1. Gracias, M. H., & Gallegos, E. E. (2024). Transitioning perspectives: Agile and waterfall perceptions in the integration of model-based systems engineering (MBSE) within aerospace and defense industries. *ITEA Journal of Test and Evaluation, 45*(4). https://doi.org/10.61278/itea.45.4.1006
    a. *Focus:* Research Aim 1
2. Gracias, M. (2024). Embracing Agile Excellence: A Comparative Analysis of Model-Based Systems Engineering (MBSE) in Agile versus Waterfall Methodologies in Aerospace and Defense Industries. *Presented at 27th Annual Systems and Mission Engineering Conference of National Defense Industrial Association.* 1820785. October 29, 2024: Norfolk, VA.

   a. *Focus:* Research Aim 1

3. Gracias, M. H., & Gallegos, E. E. (2025). Balancing structure and flexibility: Evaluating agile, waterfall, and hybrid methodologies in aerospace and defense projects. *ITEA Journal of Test and Evaluation, 46*(3). https://doi.org/10.61278/itea.46.3.1003
   a. *Focus:* Research Aim 2
4. Gracias, M. H., Gallegos, E. E., & Daily, J. (under review). Demonstrating model-based systems engineering (MBSE) with agile, waterfall, and hybrid methodologies: A comparative aerospace case study. *Submitted to INCOSE Systems Engineering Journal.*
   a. *Focus:* Research Aim 3
5. Gracias, M. H. (2026). From Sustainment to Delivery: Governing Defense Programs with Hybrid MBSE. *To be Presented at the PGCS Project and Program Management Symposium.* August 11-13, 2026: Barton, Australia.
   a. *Focus:* Research Aims 2 and 3

REFERENCES

Ahmed, H., & Miller, E.E. (2022). Human-systems integration of agricultural machinery in developing economy countries: Perceptions of adoption. *Proceedings of the INCOSE Human Systems Integration Conference*, 18-28, San Diego, CA: Nov 2021. doi.org/10.1002/iis2.12868

Ahmed, H., & Miller, E.E. (2023). Needs analysis and payback models for tractor design based on field data from traditional farmers in Sudan. *World, 4*(4), 698-708. doi.org/10.3390/world4040044

Ahmed, H., & Miller, E.E. (2023). Quantifying the economic impact on farmers from agricultural machinery: A case study of farmers in Sudan. *World, 4*(2), 347-359. doi.org/10.3390/world4020022

Azizyan, G., Magarian, M. K., & Matsson M. K. (2011). Survey of Agile Tool Usage and Needs. Proceedings of the 2011 Agile Conference, 29-38. Salt Lake City, UT: Aug 2011.

Balaji, S. & Murugaiyan, M. S. (2012). Waterfall vs. V-Model vs. Agile: A comparative study on SDLC. International Journal of Information Technology and Business Management, 2(1), 26-30.

Beck, K. 2000. Extreme Programming Explained: Embrace Change. Addison Wesley Longman Inc.

Beck, K. et al. (2001). Manifesto for Agile Software Development. Accessed: 2022-12-14. Retrieved from http://www.Agilemanifesto.org

Bieler, E. A. (2018). Analyzing the United States Air Force Agile transformation using a systems thinking approach. Massachusetts Institute of Technology. Retrieved from https://dspace.mit.edu/handle/1721.1/120896

Biggs, T., Lanigan, T., Ruddell, D., Gallegos, E.E., & Daily, J. (2024). Modeling a heavy-duty vehicle data collection process for authenticating driver identify and analyzing driver behavior under duress. *Proceedings of the 19th Annual System of Systems Engineering (SoSE) Conference*, 256-263. Tacoma, WA: June 2024. doi.org/10.1109/SOSE62659.2024.10620958

Birch, D.S., Miller, E.E., & Bradley, T.H. (2023). Human reliability analysis using a human factors hazard model. *Journal of System Safety, 58*(2), 7-29. doi.org/10.56094/jss.v58i2.251

Boehm, B., & Basili, V. (2001). Software defect reduction top 10 list. IEEE Computer, 34(1), 135–137. doi:10.1007/3-540-27662-9_26

Boehm, B., & Turner, R. (2003). Balancing Agility and Discipline: A Guide for the Perplexed. Addison-Wesley Professional.

Boehm, B., & Turner, R. (2005). Management challenges to implementing Agile processes in traditional development organizations. IEEE Software, 22(5) 30-39.

Bone, M., & Cloutier, R. (2010). The current state of Model Based Systems Engineering: Results from the OMG SysML request for information. INCOSE International Symposium, 20(1), 1186–1199.

Bumgarner, S., Rudder, S., Gallegos, E.E., & Dailey, J. (2024). Human factors engineering (HFE) considerations for mounting internal interfaces in industry vehicles. *Proceedings of the Human Factors and Ergonomics Society (HFES) Annual Meeting*. Phoenix, AZ: Sept 2024. doi.org/10.1177/10711813241282266

Call, J., & Herber, D. (2023). A Case for Model-Based Systems Engineering in an Agile World and Principles for Growth. INCOSE International Symposium, 33(1):1612-1626. doi:10.1002/iis2.13102

Chellin, M.D., & Miller, E.E. (2023). Frustrated with obsolescence – Try changing your mental model. *Defense Acquisition Research Journal, 30*(3), 228-249. doi.org/10.22594/dau.23-903.30.03

Chellin, M.D., & Miller, E.E. (2023). Proactive obsolescence management methods for C5ISR systems: Insights from practitioners. *Defense Acquisition Research Journal, 30*(1), 24-44. doi.org/10.22594/dau.21-886.30.01

Chellin, M.D., & Gallegos, E.E. (2024). Visualization tool for obsolescence management: Reducing supply chain risk for C5ISR systems. *Defense Acquisition Research Journal, 31*(1), 36-61. doi.org/10.22594/dau.23-914.31.01

Cohen, J. (1988). Statistical power analysis for the behavioral sciences (2nd ed.). Lawrence Erlbaum Associates.

Conforto, E. C., Salum, F., Amaral, D. C., da Silva, S. L., & de Almeida, L. F. M. (2014). Can Agile project management be adopted by industries other than software development? Project Management Journal, 47(3), 21–34. doi:10.1002/pmj.21410

Corl, K., & Gallegos, E.E. (2024). A method for human systems integration requirements within model based systems engineering. *Proceedings of the International Council on Systems Engineering (INCOSE) International Symposium, 34*(1), 1787-1806. Dublin, Ireland: July 2024. doi.org/10.1002/iis2.13237

Davis, B. (2012). Agile practices for waterfall projects: Shifting processes for competitive advantage. J. Ross Publishing.

Delligatti, L. (2013). SysML Distilled: A Brief Guide to the Systems Modeling Language. Pearson Education.

Dikert K., Paasivaara M., & Lassenius C. (2016). Challenges and success factors for large-scale Agile transformations: A systematic literature review Journal of Systems and Software, 119, 87-108.

Dingsøyr, T., Dybå, T., & Moe, N. B. (2010). Agile Software Development. Springer. 10.1007/978-3-642-12575-1.

Dori, D. (2016). Model-Based Systems Engineering with OPM and SysML. Springer. doi:10.1007/978-1-4939-3295-5

Dugbartey, J., Amoako, E., & Mensah, I. (2025). Optimizing project delivery through agile methodologies: Balancing speed, collaboration and stakeholder engagement. World Journal of Advanced Research and Reviews 25(1):1237-1257. doi:10.30574/wjarr.2025.25.1.0193

Elm, J., & Goldenson, D. (2012). The Business Case for Systems Engineering Study: Results of the Systems Engineering Effectiveness Survey. Software Engineering Institute. doi: 10.1184/R1/6585080.v1

Estefan, J. 2008. Survey of Candidate Model-Based Systems Engineering (MBSE) Methodologies. INCOSE-TD-2007-003-02. Retrieved from http://www.omgsysml.org/MBSE_Methodology_Survey_RevB.pdf

Fagarasan, C., Popa, O., Pisla, A., & Cristea, C. (2021). Agile, waterfall and iterative approach in information technology projects. IOP Conference Series Materials Science and Engineering 1169(1):012025. doi: 10.1088/1757-899x/1169/1/012025

Fisher, R. A. (1925). Statistical methods for research workers. Oliver & Boyd.

Francis, K. (2016). From Waterfall to Agile: A Guide for Project Managers Retrieved from http://inviqa.com/blog/from-waterfall-to-agile-project-managers

Friedenthal, S., Moore, A., Steiner, R., & Rogers, M. (2015). A Practical Guide to SysML. doi: 10.1016/c2013-0-14457-1.

George, L. B. (2023). Zhivete Model - a Hybrid of V Model and Agile Scrum for Product Development. SSRN.

Gracias, M. H., & Gallegos, E. E. (2024). Transitioning perspectives: agile and waterfall perceptions in the integration of model-based systems engineering (MBSE) within aerospace and defense industries. ITEA Journal of Test and Evaluation, 45(4). https://doi.org/10.61278/itea.45.4.1006

Gracias, M. H., & Gallegos, E. E. (2025). Balancing Structure and Flexibility: Evaluating Agile, Waterfall, and Hybrid Methodologies in Aerospace and Defense Projects. ITEA Journal of Test and Evaluation, 46(3). https://doi.org/10.61278/itea.46.3.1003

Gracias, M. H., Gallegos, E. E., & Daily, J. (2026). Demonstrating model-based systems engineering (MBSE) with agile, waterfall, and hybrid methodologies: A comparative aerospace case study. *Manuscript Submitted for Publication.*

Gregory, P., Barroca, L., Taylor, K., Salah, D., & Sharp, H. (2015). Agile challenges in practice: a thematic analysis, In Proceedings of the 16th International Conference on Agile Software Development, Helsinki, Finland, 1-13.

Gupta, A. (2021). Comparative Study of Different SDLC Models. International Journal for Research in Applied Science and Engineering Technology 9(11):73-80. doi:10.22214/ijraset.2021.38736

Gupta, R. K., Manikreddy, P., & GV, A. (2016). Challenges in Adapting Agile Testing in a Legacy Product. IEEE 11th International Conference on Global Software Engineering (ICGSE), pp. 104–108. 10.1109/ICGSE.2016.21

Hajjdiab, H. & Taleb, A.S. (2011). Adopting Agile Software Development: Issues and Challenges. International Journal of Managing Value and Supply Chains 2(3):1-10. doi:10.5121/ijmvsc.2011.2301

Hamid, S., Nasir, M., Othman, M., & Ahmadi. R. (2015). Factors limiting the implementations of Agile practices in the software industry: a pilot systematic review, Indian Journal of Science and Technology, 8(30), 1-11.

Hashimoto, D., Kawamura, K., & Ong, R. (2025). Integration of MBSE and Agile Development by Seamlessly Creating Test Plans from Model Simulations in SDV Development. INCOSE 35th International Symposium.

Haskins, C. (2011). A historical perspective of MBSE with a view to the future. INCOSE International Symposium, 493–509.

Hecht, M., & Chen, J. (2021). Verification and Validation of SysML Models. INCOSE International Symposium 31(1):599-613. doi:10.1002/j.2334-5837.2021.00857.x

Hiekata, K., Mitsuyuki, T., Goto, T., & Moser, B. (2016). Design of software development architecture comparison of waterfall and agile using reliability growth model. Transdisciplinary Engineering: Crossing Boundaries (pp. 471-480). IOS Press.

Highsmith, J. (2009). Agile Project Management: Creating Innovative Products. Addison-Wesley.

Highsmith, J. & Cockburn, A. (2001), Agile software development: the business of innovation. IEEE Xplore, Vol. 34, no. 9, pp. 120-127. 10.1109/2.947100.

Hüllmann, J. A., Kimathi, K, & Weritz, P. (2024). Large-Scale Agile Project Management in Safety-Critical Industries: A Case Study on Challenges and Solutions, Information Systems Management 42(1):1-23. doi:10.1080/10580530.2024.2349886

Huss, M., Herber, D. R., & Borky, J. M. (2023). Comparing Measured Agile Software Development Metrics Using an Agile Model-Based Software Engineering Approach versus Scrum Only. Software, 2(3), 310-331.

Ijomah, T. (2024). Advancements in project management methodologies: Integrating agile and waterfall approaches for optimal outcomes. World Journal of Advanced Science and Technology 1:1-012. doi: 10.51594/estj.v5i7.1312.

INCOSE, INCOSE Systems Engineering Handbook, 4th ed. 2015.

Johnston, J. (2014). How to Start the Transition from a Waterfall to an Agile Process Retrieved from http://www.mightybytes.com/blog/transition-waterfall-to-agile/

Kahkonen, T. (2004). Agile methods for large organizations - building communities of practice Agile Development Conference, 2-10.

Ke, Y. (2019). Assessing various software development methodologies and matching software development methodologies with projects. Massachusetts Institute of Technology.

Kemp, D., K. Akroyd-Wallis, et al. (2016). Fifty Shades of Agile: AN ANALYSIS OF DIFFERENT PERSPECTIVES OF Agile SYSTEMS ENGINEERING. INCOSE International Symposium, 691–712.

Kenner, B. T. (2019). Too Agile? - DevOps Software Development Challenges in a Military Environment. University of South Carolina Scholar Commons. Retrieved from https://scholarcommons.sc.edu/etd/5396

Kerzner, H. (2017). Project Management: A Systems Approach to Planning, Scheduling, and Controlling. 13th Edition. Wiley.

Krupa, M. & Hajek, J. (2023). Hybrid Project Management Models: A Systematic Literature Review. International Journal of Project Organisation and Management, doi:10.1504/IJPOM.2024.10056237

Kusters, R. J., Leur, Y., Rutten, W., & Trienekens, J. (2017). When Agile Meets Waterfall - Investigating Risks and Problems on the Interface between Agile and Traditional Software Development in a Hybrid Development Organization. Proceedings of the 19th International Conference on Enterprise Information Systems, 2, 271-278.

Larman, C. & Basili, V. R. (2003). Iterative and Incremental Development: A Brief History. IEEE Xplore, 36(6), 47–56. 10.1109/MC.2003.1204375

Lester, R. (2013). Transitioning to Agile: Seven Strategies for Business Executives. Scrum Alliance, 2-8.

Madni, A.M., & Sievers, M. (2018). Model-Based Systems Engineering: Motivation, Current Status, and Needed Advances. Disciplinary Convergence in Systems Engineering Research. Springer, Cham. doi: 10.1007/978-3-319-62217-0_22

Madni, A.M., & Purohit, S. (2019). Economic Analysis of Model-Based Systems Engineering. Systems, 7(1):12. doi.org/10.3390/systems7010012

Maharao, C.S. (2024). A study on Agile Project Management in it: Challenges and Best Practices. ShodhKosh: Journal of Visual and Performing Arts 5(1). doi:10.29121/shodhkosh.v5.i1.2024.2284

Miller, E.E. (2013). *Effects of roadway on driver stress: An on-road study using physiological measures.* Master’s thesis, University of Washington. Available from: https://digital.lib.washington.edu/researchworks/handle/1773/23592

Miller, E.E., & Boyle, L.N. (2013). Variations in road conditions on driver stress: Insights from an on-road study. *Proceedings of the Human Factors and Ergonomics Society 57th Annual Meeting,* 1864-1868. San Diego, CA: Sept-Oct 2013. doi.org/10.1177/1541931213571416

Meyer, B. (2014). Agile! The Good, the Hype and the Ugly. Springer International Publishing, Switzerland.

Noguchi, R. A., Martin, J. N., & Wheaton, M. J. (2020). Using MBSE to Architect and Implement the MBSE System. INCOSE International Symposium, 30(1), 18-35.

Ortolano, L.F., & Gallegos, E.E. (2025). Advancing human-AI collaboration in small and medium-sized enterprises: A systems engineering approach. *Systems Engineering,* e70031. doi.org/10.1002/sys.70031

Paasivaara M., & Lassenius C. (2016). Challenges and success factors for large-scale Agile transformations: A research proposal and a pilot study Proceedings of the Scientific Workshop Proceedings of XP'16, Association for Computing Machinery, New York, NY.

Palmquist, S., Lapham, M., Garcia-Miller, S., Chick, T., & Ozkaya, I. (2013). Parallel Worlds: Agile and Waterfall Differences and Similarities. Software Engineering Institute. Retrieved from https://apps.dtic.mil/sti/pdfs/ADA610501.pdf

Power, W., Wylie, M., Mellen, P. & Bodegom, P. (2021). A Hybrid between Model-Based Systems Engineering and Agile Methodologies for Simulation of Complex Weapon Systems of Systems. Proceedings of 2021 IEEE Aerospace Conference, 1-15.

Salo, O., & Abrahamsson, P. (2008). Agile methods in European embedded software development organisations: a survey on the actual use and usefulness of extreme programming and scrum. IET Software, 2, 58-64.

Schatz, B. & Abdelshafi, I. (2005). Primavera gets Agile: A Successful Transition to Agile Development, IEEE Software, 36-41.

Serrador, P., & Pinto, J. K. (2015). Does Agile work? A quantitative analysis of agile project success. International Journal of Project Management, 33(5), 1040–1051. 10.1016/j.ijproman.2015.01.006

Shekhar, P. C. (2019). Agile vs. Waterfall: A Comprehensive Analysis of Software Testing Method. International Journal of Innovative Research and Creative Technology 5 (5):1-12.

Sitorus, G. B. M. T., Efendi, R. A. G., Napitupulu, J. F., Pranoto, H., & Hermawan, E. S. (2025). Bridging the Gap: A Comparative Evaluation of Agile, Waterfall, and Hybrid Methodologies in Modern Software Projects. Procedia Computer Science

Stettina, C. J., & Hörz, J. (2014). Agile portfolio management: An empirical perspective on the practice in use. International Journal of Project Management, 33(1), 140-152. 10.1016/j.ijproman.2014.03.008

Sundaram, V. & Brownlow, L. (2018). MBSE Based Digital Thread and Digital System Model for AF DCGS. Proceedings of AIAA Sci Tech. Kissimmee, FL: Jan 2018.

Tanner, M. & Willingh, U. (2014). Factors leading to the success and failure of Agile projects implemented in traditionally waterfall environments, International Conference on Human Capital without Borders: Knowledge and Learning for Quality of Life, Portoroz, Slovenia, 693-701.

VanderLeest, S. H., & Buter, A. (2009). Escape the waterfall: Agile for aerospace. Proceedings of the 2009 IEEE/AIAA 28th Digital Avionics Systems Conference, 6.D.3-1-6.D.3-16. Orlando, FL.

Vinekar, V., Slinkman, C. W., & Nerur, S. (2006). Can Agile and Traditional Systems Development Approaches Coexist? An Ambidextrous View. Information Systems Management, 23(3), 31–42. 10.1201/1078.10580530/46108.23.3.20060601/93705.4

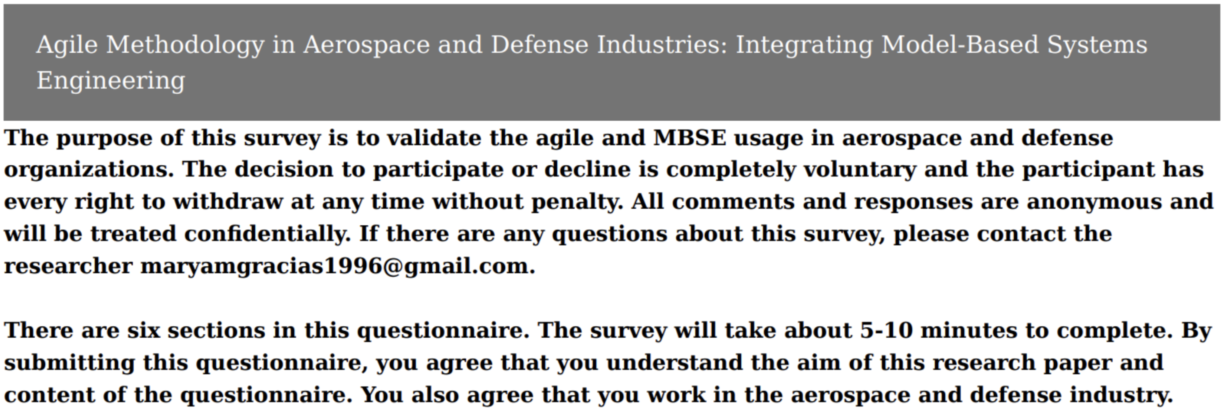
Wang, G. (2020). Implementing a model-based, digital engineering enterprise for a defense systems integrator - an ongoing journey. INCOSE International Symposium, vol. 30, no. 1, pp. 783–798. https://doi.org/10.1002/j.2334-5837.2020.00755.x

Wenyue, W., Junjie, H., & Yinxuan, M. (2022). Application and development of MBSE in aerospace. Journal of Physics Conference Series 2235(1):012021. doi:10.1088/1742-6596/2235/1/012021

Womack, J. P., & Jones, D. T. (2008). Lean Thinking: Banish Waste and Create Wealth in Your Corporation (2nd ed.).

APPENDIX A: SURVEY

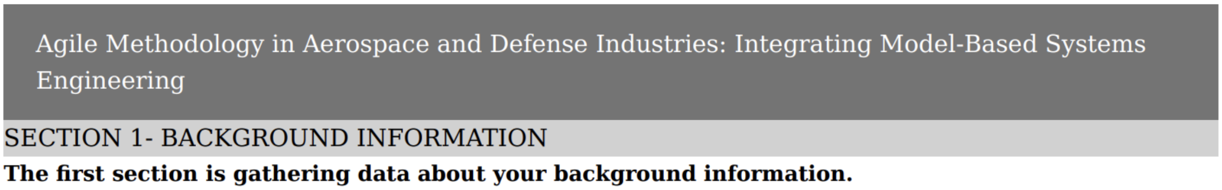
The survey had six sections for participants to complete.



Agile Methodology in Aerospace and Defense Industries: Integrating Model-Based Systems Engineering

The purpose of this survey is to validate the agile and MBSE usage in aerospace and defense organizations. The decision to participate or decline is completely voluntary and the participant has every right to withdraw at any time without penalty. All comments and responses are anonymous and will be treated confidentially. If there are any questions about this survey, please contact the researcher maryamgracias1996@gmail.com.

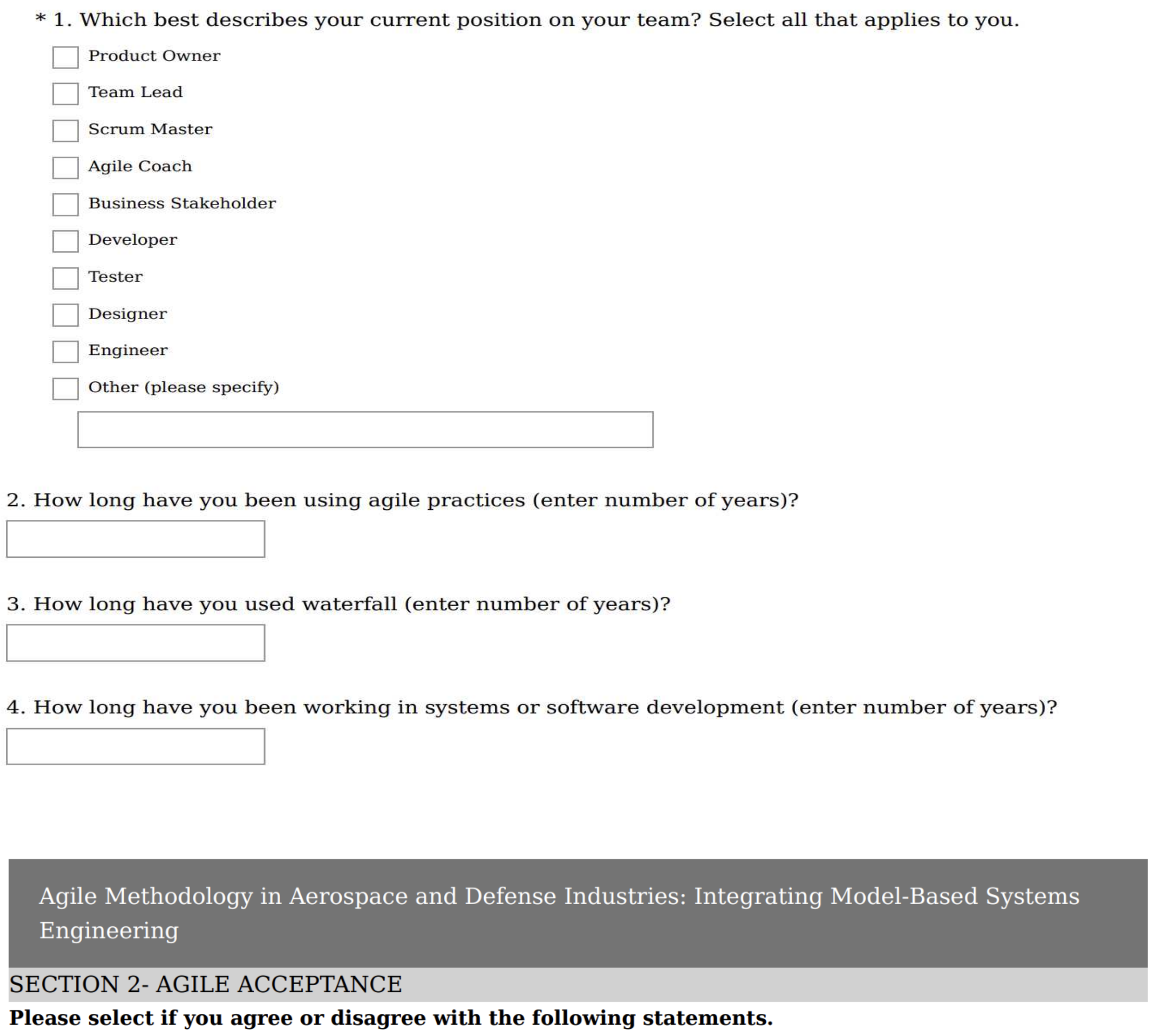
There are six sections in this questionnaire. The survey will take about 5-10 minutes to complete. By submitting this questionnaire, you agree that you understand the aim of this research paper and content of the questionnaire. You also agree that you work in the aerospace and defense industry.



Agile Methodology in Aerospace and Defense Industries: Integrating Model-Based Systems Engineering

SECTION 1- BACKGROUND INFORMATION

The first section is gathering data about your background information.



* 1. Which best describes your current position on your team? Select all that applies to you.

- ☐ Product Owner
- ☐ Team Lead
- ☐ Scrum Master
- ☐ Agile Coach
- ☐ Business Stakeholder
- ☐ Developer
- ☐ Tester
- ☐ Designer
- ☐ Engineer
- ☐ Other (please specify)

2. How long have you been using agile practices (enter number of years)?

3. How long have you used waterfall (enter number of years)?

4. How long have you been working in systems or software development (enter number of years)?

Agile Methodology in Aerospace and Defense Industries: Integrating Model-Based Systems Engineering

SECTION 2- AGILE ACCEPTANCE

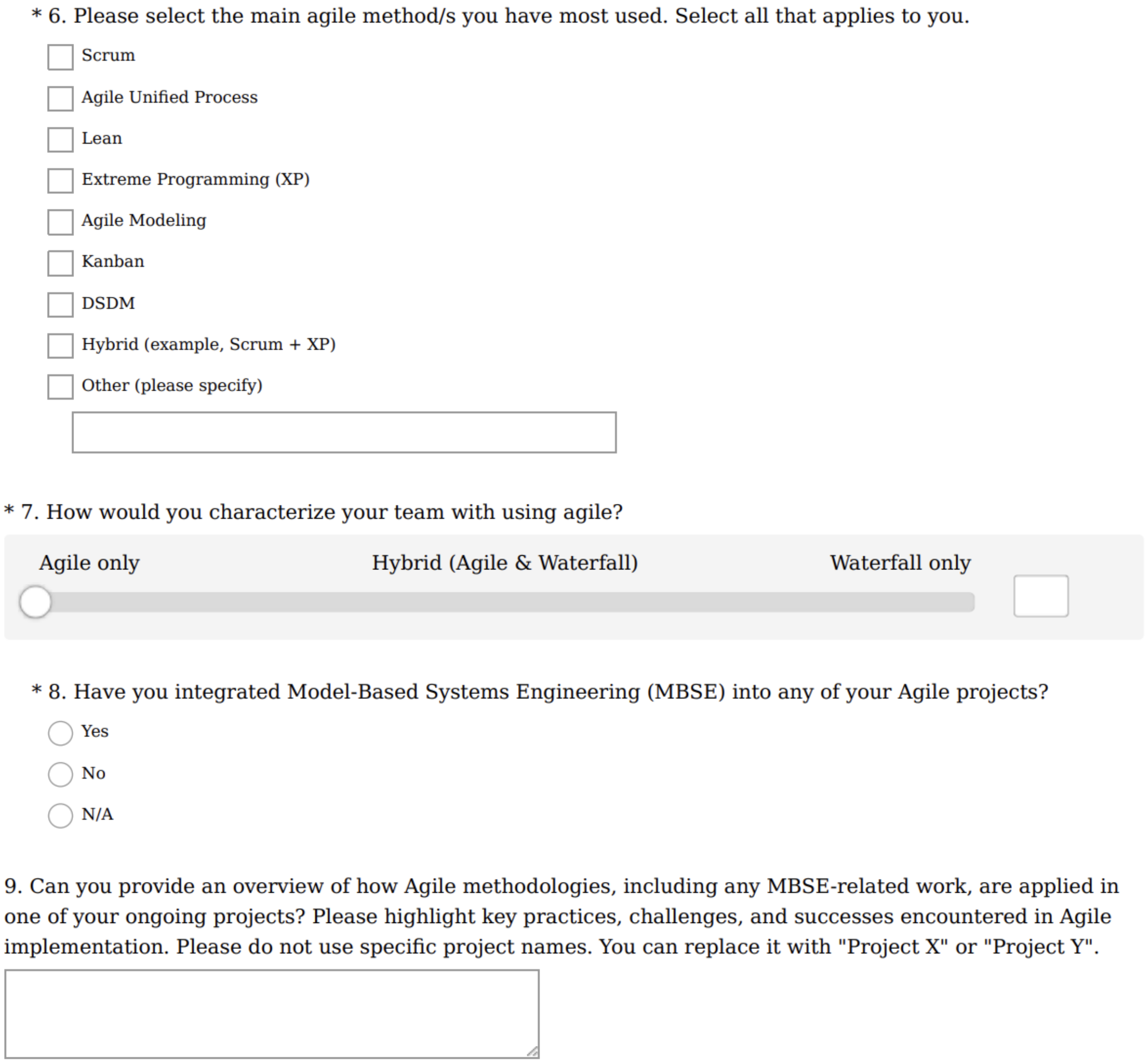
Please select if you agree or disagree with the following statements.

5. Please select your best choice.

	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
I can complete my responsibilities quickly by implementing Agile practices.	○	○	○	○	○
Using Agile matches my preferred working style.	○	○	○	○	○
Agile increases my effectiveness at work.	○	○	○	○	○
I can produce better quality by implementing Agile practices.	○	○	○	○	○
Agile enables some flexibility in my current tasks.	○	○	○	○	○
Agile is compatible for all facets of my work.	○	○	○	○	○
Agile allows me to have more control over my work.	○	○	○	○	○

Agile Methodology in Aerospace and Defense Industries: Integrating Model-Based Systems Engineering

SECTION 3- AGILE USAGE



* 6. Please select the main agile method/s you have most used. Select all that applies to you.

☐ Scrum

☐ Agile Unified Process

☐ Lean

☐ Extreme Programming (XP)

☐ Agile Modeling

☐ Kanban

☐ DSDM

☐ Hybrid (example, Scrum + XP)

☐ Other (please specify)

* 7. How would you characterize your team with using agile?

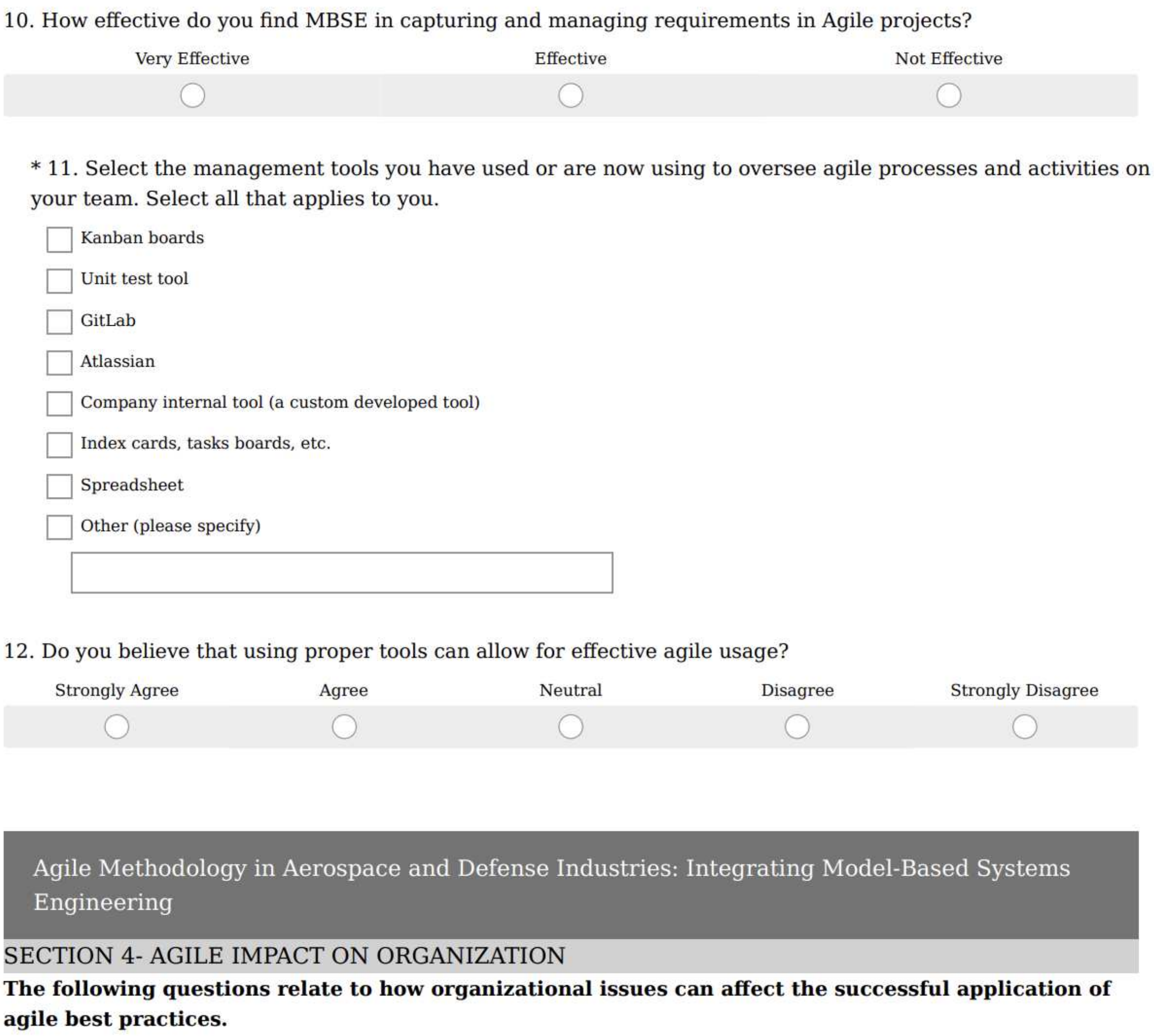
* 8. Have you integrated Model-Based Systems Engineering (MBSE) into any of your Agile projects?

○ Yes

○ No

○ N/A

9. Can you provide an overview of how Agile methodologies, including any MBSE-related work, are applied in one of your ongoing projects? Please highlight key practices, challenges, and successes encountered in Agile implementation. Please do not use specific project names. You can replace it with "Project X" or "Project Y".



10. How effective do you find MBSE in capturing and managing requirements in Agile projects?

Very Effective	Effective	Not Effective
○	○	○

* 11. Select the management tools you have used or are now using to oversee agile processes and activities on your team. Select all that applies to you.

☐ Kanban boards
☐ Unit test tool
☐ GitLab
☐ Atlassian
☐ Company internal tool (a custom developed tool)
☐ Index cards, tasks boards, etc.
☐ Spreadsheet
☐ Other (please specify)

12. Do you believe that using proper tools can allow for effective agile usage?

Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
○	○	○	○	○

Agile Methodology in Aerospace and Defense Industries: Integrating Model-Based Systems Engineering

SECTION 4- AGILE IMPACT ON ORGANIZATION

The following questions relate to how organizational issues can affect the successful application of agile best practices.

13. Please indicate the type of methodology that was used in your company.

☐

14. Do you believe that your management provided adequate training to use agile practices effectively?

Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
○	○	○	○	○

* 15. How would you rate the level of Model-Based Systems Engineering (MBSE) implementation in your organization?

Not implemented Partially implemented Fully implemented ☐

* 16. Have you noticed improvements in project performance metrics (e.g., time-to-market, cost, quality) with MBSE in Agile projects?

○ Yes
○ No
○ Unsure
○ N/A

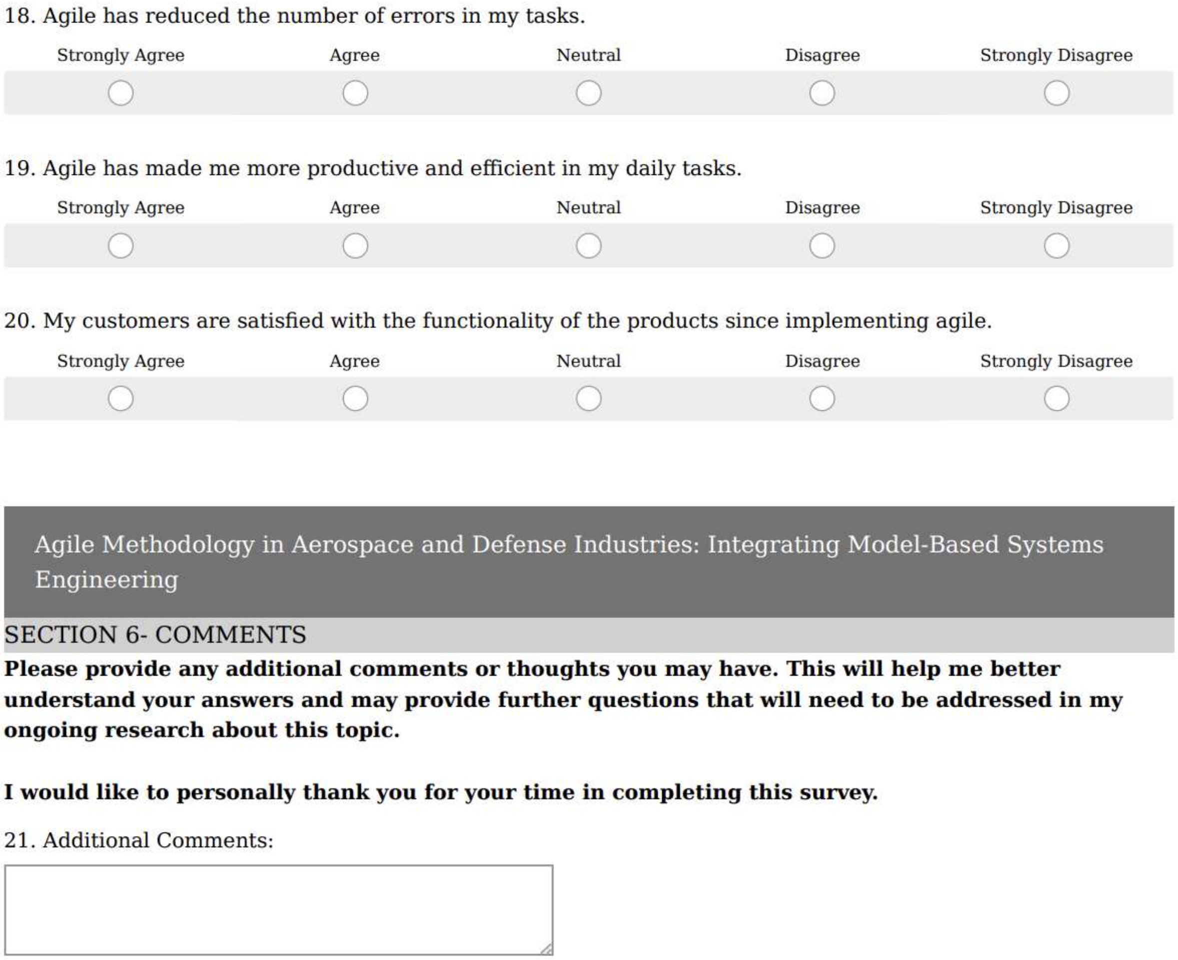
Agile Methodology in Aerospace and Defense Industries: Integrating Model-Based Systems Engineering

SECTION 5- INDIVIDUAL IMPACT

The following questions relate to how agile impacts your work. Please select the best option.

17. Does agile allow you to predict the work required on your project for software or systems development?

Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
○	○	○	○	○



18. Agile has reduced the number of errors in my tasks.

Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
○	○	○	○	○

19. Agile has made me more productive and efficient in my daily tasks.

Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
○	○	○	○	○

20. My customers are satisfied with the functionality of the products since implementing agile.

Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
○	○	○	○	○

Agile Methodology in Aerospace and Defense Industries: Integrating Model-Based Systems Engineering

SECTION 6- COMMENTS

Please provide any additional comments or thoughts you may have. This will help me better understand your answers and may provide further questions that will need to be addressed in my ongoing research about this topic.

I would like to personally thank you for your time in completing this survey.

21. Additional Comments:

APPENDIX B: INTERVIEW QUESTIONS

Below are some of the interview questions asked:

- How long have you been working with either (Agile/ Waterfall) projects?
- Can you describe the project, and how its design evolved from Waterfall to Agile?
- How has Agile changed the approach to software updates (e.g., code reload versus updating sections)?
- How did the Waterfall approach impact project management, and what changes were made to accommodate Agile?
- What are the benefits of Agile software compared to the Waterfall approach, particularly in terms of flexibility and adaptability?
- What are the differences between the Waterfall and Agile approaches in software architecture design and implementation?
- Can you describe how you would implement Agile in a software environment that originally used Waterfall?
- What are the challenges in transitioning from Waterfall to Agile, and how did you overcome them?
- How have you integrated Model-Based Systems Engineering (MBSE) into your projects, and what does that process look like?
- How does MBSE compare to traditional systems engineering methods, and what are its advantages?
- Can you describe a project where MBSE was used in an Agile/ Waterfall context?
- What specific tools and techniques did you use for MBSE in Agile projects?

- Which methodology (Waterfall or Agile) do you find more effective for managing complex projects, and why?
- Can you provide examples of projects where Waterfall and Agile were implemented, and how they differed in terms of change orders, budgets, and timelines?
- Are there other points of contact that could provide further insights?

LIST OF ABBREVIATIONS

Below is a list of acronyms used in this paper:

Acronym	Definition
A&D	Aerospace and Defense
AC	Actual Cost
BDD	Block Definition Diagram
CPI	Cost Performance Index
DAU	Defense Acquisition University
DevOps	Development and Operations
DoD	Department of Defense
EV	Earned Value
EVMS	Earned Value Management System
INCOSE	International Council on Systems Engineering
IRB	Institutional Review Board
MBSE	Model-Based Systems Engineering
MOE	Measure of Effectiveness
MSOSA	Magic Systems of Systems Architect
NDIA	National Defense Industrial Association
OEM	Original Equipment Manufacturer
OOSEM	Object-Oriented Systems Engineering Method
PV	Planned Value
SD	Standard Deviation
SE	Systems Engineering
SILA	Sustainment Integration for Legacy Aircraft
SIT	System Integration and Test
SysML	Systems Modeling Language
V&V	Verification and Validation
XP	Extreme Programming