Application of an Agile Framework in Assessing and Aligning Digital Twin Use Cases Across Product Classes in a Large Organization

by

Jeffrey Miller

B.S., Aerospace Engineering
University of Virginia, 2011

Submitted to the MIT Sloan School of Management and
Department of Aeronautics and Astronautics
in partial fulfillment of the requirements for the degrees of

Master of Business Administration

and

Master of Science in Aeronautics and Astronautics

in conjunction with the Leaders for Global Operations program
at the
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
May 2022

Author .....
MIT Sloan School of Management and
Department of Aeronautics and Astronautics
May 6, 2022

Certified by .....
Steven Spear, Thesis Supervisor
Senior Lecturer, MIT Sloan School of Management

Certified by .....
R. John Hansman, Thesis Supervisor
T. Wilson Professor of Aeronautics and Astronautics

Accepted by .....
Jonathan P. How, Chair, Graduate Program Committee
R. C. Maclaurin Professor of Aeronautics and Astronautics

Accepted by .....
Maura Herson
Assitant Dean, MBA Program, MIT Sloan School of Management

THIS PAGE INTENTIONALLY LEFT BLANK

Application of an Agile Framework in Assessing and Aligning Digital Twin Use Cases Across Product Classes in a Large Organization

by

Jeffrey Miller

Submitted to the MIT Sloan School of Management and
Department of Aeronautics and Astronautics
on May 6, 2022, in partial fulfillment of the
requirements for the degrees of
Master of Business Administration
and
Master of Science in Aeronautics and Astronautics

Abstract

In order to fully realize the benefits of the Fourth Industrial Revolution, companies must migrate legacy software and processes to state-of-the-art systems that fully incorporate technologies enabled by the Internet of Things, namely Digital Twin and Digital Thread. Digital Twin technologies hold the promise of significant advantages in design and manufacturing, particularly for complex products with a long lifecycle. In an early effort to adopt these technologies, large organizations are investing resources in defining and iterating upon their nascent Digital Twin capabilities. However, particularly in a large organization, use cases for Digital Twin technologies can differ according to product class. This thesis presents as a case study one such migration process in a large aerospace manufacturing organization. The research defined and applied an Agile framework to a team's current processes to assist them in better understanding the Digital Twin use cases present across the full enterprise, and to align their efforts accordingly. In applying the Agile framework to improve team alignment and gather additional feedback, a measurable change in Digital Twin representation specifically within the context of configuration definition and change management was effected, making Digital Twin configuration representations and change management tools more broadly applicable across product classes.

Thesis Supervisor: Steven Spear
Title: Senior Lecturer, MIT Sloan School of Management

Thesis Supervisor: R. John Hansman
Title: T. Wilson Professor of Aeronautics and Astronautics

THIS PAGE INTENTIONALLY LEFT BLANK

Acknowledgments

I'd like to thank Don Shingler, Greg Meboe, the PLCS team, the many Boeing professionals I interviewed, and my academic advisors John Hansman and Steve Spear.

I'd like to extend special thanks and dedicate this thesis to Julia Buckingham.

THIS PAGE INTENTIONALLY LEFT BLANK

Contents

List of Figures

THIS PAGE INTENTIONALLY LEFT BLANK

List of Tables

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 1

Background and Introduction

1.1 Project Motivation

This research focuses on the migration of legacy software and design tools at a large aerospace company. Legacy software suites for product design have only limited built-in functionality. Even add-on functionality that was programmed later can only extend the legacy suite's utility up to the limit of its inherent technological capabilities. For most legacy software systems, this technological limit is intrinsic to the world-view of the third industrial revolution, failing to account for the rise of the Internet of Things and predicting the utility of the technologies it enables, namely Digital Twin and Digital Thread. Companies wishing to become early adopters of these nascent technologies are thus faced with the challenge of upgrading their software and the design and production engineering processes that go hand-in-hand. This thesis puts forth as a case study one example of this software migration from a disparate set of legacy design tools to a single, state-of-the-art design software suite. This migration ensures the organization can fully incorporate Industry 4.0 technologies that leverage in-depth product representations, real-time, and historical data to simulate, model, and test the product throughout its lifecycle.

This research was undertaken at a large aerospace company ("The Company"). The researcher embedded in a team of software systems engineers ("The Team") whose initiative involved deriving requirements for configuration management functionality capable of working with the next generation of product management software. This team's effort was part of a large Systems Engineering initiative at The Company to move towards new software that is fully integrated for various products across the company from start to finish. This capability allows for the "Digital Thread" of an increased electronic footprint throughout the design and manufacturing process.

The overall challenge that The Company was trying to solve is a migration from "legacy" engineering design software programs to the state-of-the-art, part of a larger initiative to take advantage of Industry 4.0 initiatives Digital Twin and Digital Thread. The research presented here did not seek to solve that whole issue – rather, it intended to focus on a small subproblem under that umbrella. In moving to the new software, The Company risked losing Digital Twin capability currently in existence with legacy engineering design programs, namely the ability to regularly update product definitions and configurations and maintain a complete history of all engineering drawings that were ever associated with a particular, single instance of a product. The Team into which the researcher embedded intended to address that capability gap. The research focused identifying gaps between The Team's current processes and an ideal Agile method of performing the task. The research subsequently addressed those gaps to create a measurable difference in the resulting requirements for configuration definition within the Digital Twin.

The research focused around deriving and iteratively improving upon requirements for configuration management software. In furtherance of those efforts, this thesis performed a review of academic literature to create an Agile framework to use as instrumentation to compare and improve current processes. As a result of the application of the Agile framework, the researcher gathered feedback to assess enterprise-wide

configuration management needs and Digital Twin use cases. An additional review of Digital Twin academic literature was performed to create a framework against which this feedback data could be compared to improve alignment in Digital Twin use cases across The Company.

1.1.1 Systems Engineering and Agile Methods Literature Review

Process Requirements

At a high level, The Company was currently undergoing a detailed systems engineering process to recreate product definition capabilities for its next generation of software. The brand of systems engineering to which The Company professes to adhere is Model-Based Systems Engineering (MBSE). Moreover, The Company and The Team have stated that they want to take an Agile approach to product development as well as organization of functional teams, specifically adopting the Scaled Agile Framework (SAFe Agile) to organize teams and prioritize work. This information provided a reasonable background for reviewing academic literature at strategic (high), operational (medium), and tactical (low) levels to create an Agile framework for use within a model-based systems engineering environment.

The Team was at what it called "Milestone 2" of their process in developing Digital Twin configuration management software – defining the product through requirements. What follows is a review of the expected actions at such a milestone, as defined from a systems engineering, model-based engineering, and Agile perspective.

The two authoritative resources on systems engineering are INCOSE and NASA. The INCOSE Systems Engineering (SE) Handbook would rephrase The Company's "Milestone 2" as the Concept Stage. INCOSE defines this as the step in which a

team assesses new business opportunities and develops preliminary requirements and a feasible design solution. "The first step is to identify, clarify, and document stakeholders' requirements [19]." NASA's Systems Engineering Handbook similarly defines this stage, terming it the System Design process. Per the NASA SE Handbook, during System Design, the team is expected to "define and baseline stakeholder expectations, generate and baseline technical requirements, and convert the technical requirements into a design solution that will satisfy the baselined stakeholder expectations [10]."

For a tactical approach, the research turns to an Agile perspective. Agile software development began as a collaboration between several thought leaders in the industry attempting to unify and improve the leading strategies of their time [5]. It is defined by its ability to enable teams and enterprises to "create and respond to changes quickly in order to profit in a turbulent business environment [4]." Agile has gone through many iterations since its initial concept; the implementation that The Company has adopted is based off the Scaled Agile Framework (SAFe), which is an effective method to align their Agile software development ecosystem more closely with Model-Based Systems Engineering practices [3]. In the early stages of product design, teams are organized into Agile Release Trains (ARTs) to collaborate, develop, and demonstrate or implement the solutions [4].

In summary, the research suggests that a method for defining requirements using an Agile approach is one in which a team is created and aligned to identify and survey a comprehensive set of stakeholders to inform the requirements. Thus, any Agile framework utilized must provide academic justification for team formation and alignment as well as stakeholder identification and survey for requirements generation.

Team Formation and Alignment

The initial step in the Agile framework presented is team formation and alignment within the organization. The research reviewed here presents justification for how

teams are created and aligned using Agile and Model-Based Systems Engineering methodologies. The research is broken down into three parts, first addressing what team alignment strategies are prescribed by systems engineering in general, then moving to MBSE resources, and finally to what SAFe Agile describes as the specific requirements for team alignment. It is important to note that none of these systems (Systems Engineering, Model-Based Systems Engineering, and SAFe Agile) are in conflict, but rather describe activities at a strategic, operational, and tactical level, respectively.

Using INCOSE as a reference for Systems Engineering, describes team formation and alignment strategies under the Enterprise and Agreement Processes section, specifically calling out the need to align projects within portfolios based on strategic objectives. The process activities include defining team roles and responsibilities, as well as aligning the portfolio of projects [19]. The MBSE literature goes in greater depth. A model-based systems engineering approach to team formation and alignment is considered by researchers Borkey and Bradley in "Effective Model-Based Systems Engineering." Team roles include the Chief Architect, whose primary role is sourcing and providing feedback to improve the product. Additional roles include a broad array of Subject Matter Experts (SMEs), each responsible for the product's content under their area of expertise [2]. Borkey and Bradley describe this stage within the context of "constructing an architecture" and generating adherence to a "Reference Model [2]." The Reference Model enables the creation of an architecture or ecosystem of a consistent and interoperable portfolio of operational systems. This provides portfolio projects with consistent overall rules and policies, common vocabulary, a governance approach, and application guidance. Borkey and Bradley provide a relevant example Reference Model, the System Product Line Reference Architecture, in which projects are grouped within a portfolio because their "systems and systems components... have many features in common and are used for similar purposes [2]."

SAFe Agile's Agile Release Trains (ARTs) achieve precisely that aim. Knaster and Leffingwell, the founders of SAFe Agile, define ARTs as "a long-lived, self-organizing, team-of-Agile-teams and other stakeholders (approximately 50–125 people total) that defines new functionality and plans, commits, executes, and delivers solutions together [5]." Furthermore, Podeswa specifies that ARTs should be assembled not by competency, but rather to fulfill a set of mission principles. Of particular importance is the mission principle to "maximize self-sufficiency [12]." Podeswa restates this as organizing by value instead of competency, "so that each team can deliver value to an end user or customer with minimal dependencies on other teams [12]." Highsmith stresses that the team atmosphere must be "collaborative" and on this basis forms the idea of teams working in an ecosystem, akin to the idea of an ART [4].

Regarding individual team composition within each ART, the literature identifies several key roles for success. At its simplest, Rossburg states that any Agile team must have "one product owner, one Scrum master, and three to nine developers [14]." Knaster and Leffingwell echo this, adding that the developers should cover all the necessary roles to deliver value [5]. At this stage of the systems engineering process, delivering value equates to delivering comprehensive requirements, so developer teams should consist of representatives of all stakeholder perspectives. Shortcomings in this area can be mitigated by other roles within the ART, specifically the Release Train Engineer, whose role is to remove impediments to delivering value, as well as Product Managers, who interface with stakeholders and project owners to facilitate communication and validate solutions [5]. The process of continually interfacing with stakeholders and refining requirements is not unique to Agile and is emphasized in Pohl et al.'s research "Model-Based Engineering of Embedded Systems." "Continuous development by means of integration of various engineering activities... remains essential... [A]rtifacts must be continuously elicited, documented, modified and refined, starting during requirements engineering [13]."

From the high-level, strategic perspective provided by general Systems Engineering references to the operational guidance of MBSE, down to the tactical implementation of SAFe Agile, the research deduces what the teams and organizational alignment look like under an ideal Agile Framework. Teams should be diverse in perspective, incorporating a broad array of backgrounds so as to be self-sufficient and representative of the stakeholders. Additionally, these teams should be aligned within the organization to form a reference model or ecosystem along a shared system product line.

Definition and Generation of Product Requirements

Beyond forming and aligning teams, an Agile framework applied to improve the requirements generation phase must define how these requirements are defined and generated within the Systems Engineering and MBSE context. The following is an aggregation and summary of relevant systems engineering, MBSE, and Agile research on the process of defining and generating requirements.

Returning to INCOSE for a Systems Engineering perspective, a key component of the Concept Stage is to define stakeholder requirements. This includes identifying the major stakeholders of the project and eliciting their feedback on system services and capabilities. INCOSE states that "[t]he purpose of the Stakeholder Requirements Definition Process is to elicit, negotiate, document, and maintain stakeholders' requirements [19]." Importantly, INCOSE also defines stakeholders as any individual or organization "with a legitimate interest in the system [19]." Providing a prescriptive method for doing so, INCOSE includes the following activities:

- Identify stakeholders who will have an interest in the system throughout its entire life cycle.
- Elicit system services and capabilities – what the system must accomplish and how well.

...

- Establish critical and desired system performance – thresholds and objectives for system performance parameters that are critical for system success and those that are desired but may be subject to compromise in order to meet the critical parameters

  ...

- Analyze requirements for clarity, completeness and consistency.
- Negotiate modifications to resolve unrealizable or impractical requirements.

  ...

- Record and maintain stakeholder requirements throughout the system life cycle, and beyond for historical or archival purposes. [19]

Crucially, INCOSE calls out the need for open communications between systems engineers and stakeholders, as well as the importance of identifying a complete set of stakeholders [19]. Calling out Agile methods of requirements iteration specifically, INCOSE states that requirements capture is a continuous, iterative activity, and that techniques for elicitation include surveys and interviews [19].

Similarly, NASA's Systems Engineering Handbook goes into further detail. "The Stakeholder Expectations Definition Process is the initial process within the SE engine that establishes the foundation from which the system is designed and the product is realized. The main purpose of this process is to identify who the stakeholders are and how they intend to use the product [10]." NASA delineates expected inputs and outputs to the Stakeholder Expectations Definition process. These include upper level requirements and expectations, for instance, requirements that come down from a higher level for the particular product ecosystem. NASA also identifies a list of customers and stakeholders as an input to this stage. As an output, NASA states that top-level requirements and expectations of the product being developed are typical at this stage [10].

Model Based Systems Engineering resources echo a similar sentiment. Borkey and Bradley, Micouin, and others all dedicate significant portions of the early stages of

a project to identifying and consulting with stakeholders. Borkey and Bradley state that identifying stakeholders and their interests before starting to model is valuable in maximizing efficiency and minimizing errors and rework [2]. Regarding stakeholder analysis, they state that early stakeholder involvement (while requirements can be more easily changed) is a key element to success, and that addressing stakeholder concerns in a requirements baseline is "extremely important [2]." Adding techniques for dealing with stakeholders, they mention that ongoing dialogue is through formal meetings (such as program reviews) as well as informal, more frequent channels such as discussions between team members are key to ensuring interests are satisfied, the requirements are not departing from expectations, and giving stakeholders a sense of being heard and valued [2].

Micouin's research in Model Based Systems Engineering: Fundamentals and Methods dedicates two chapters on requirements. He states that design process consists of two subprocesses, each consisting of individual tasks. The first subprocess defines the requirements, which informs the second subprocess, defining a solution. In the defining requirements subprocess, Micouin specifies two tasks – gathering requirements cascaded from higher levels in the organization and gathering requirements from other stakeholders of the project [9]. Micouin states that the requirements must be "sufficiently complete," meaning that any overlooked requirements do not make the product unacceptable to any of the stakeholders, which is determined through an ongoing validation process [9].

In Model-Based Engineering of Embedded Systems, Pohl et al. describe the process by which requirements go from being solution-neutral to solution oriented. From the requirements standpoint, there is a separation between stakeholder intentions and solution-oriented requirements. In the solution-neutral stage, the requirements "describe the intentions of the stakeholders and the added benefit that can be gained for the stakeholders [13]." With this distinction, they make it clear that the inten-

tions of the stakeholders must be established as a prerequisite to solution-oriented requirements generation of any type.

Douglass bridges the gap between MBSE and Agile in "Agile Model Based Systems Engineering Cookbook." He elaborates further on the requirements generation process, making a point to note that in an Agile approach to MBSE, the needs of the stakeholders must be constantly revisited and revised as the product is being refined. He recommends focusing on incrementally creating the product requirements, as well as frequently offering those requirements to the stakeholders [3]. Douglass specifically indicates the presence of feedback loops in Figure 1-1.

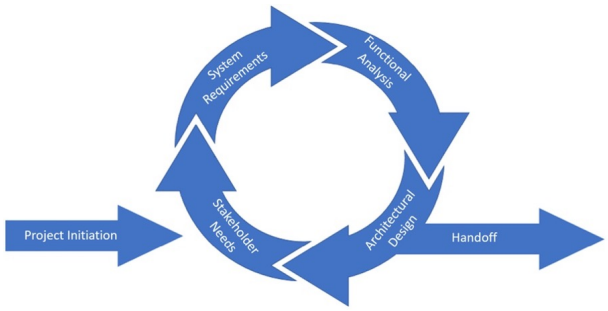

Figure 1-1: Basic Agile systems engineering workflow [3]

This leads to discussion on what Agile literature has to say about defining a product and enumerating product requirements. Beginning with the progenitors of SAFe Agile, Knaster and Leffingwell's process starts with developing from a vision to solution intent through collaboration with stakeholders. This entails an initial knowledge of requirements as well as communication with the project stakeholders. An initial solution intent records requirements and aligns stakeholders to a common understanding. Then, through an iterative and collaborative process between the ART and the stakeholders, the solution intent evolves to clearly spell out the product capabilities and features [5].

Similarly, Podeswa's "The Agile Guide to Business Analysis and Planning" calls this process "Visioning." At this early stage, stakeholders share ideas and communi-

cate rationale for building the product and its major features [12]. Podeswa states that identifying and key stakeholders should be done "as soon as possible" and that this identification process is ongoing [12]. Podeswa clearly identifies strategies for when important stakeholders cannot be fully integrated within the product development team or ART. Any strategy for engaging stakeholders should plan for both the method of collaboration and communication, as well as ensure engagement is frequent and ongoing. For those stakeholders whose collaboration must come externally, instead of via full-time incorporation within the team, collaboration expectations and communication priorities must be set via discussion with the Project Owner to ensure feedback arrives at appropriate levels of granularity, timing, and frequency [12]. In the spirit of ongoing identification, stakeholder engagement often leads to the next interviewee, who in turn can identify additional stakeholders for the project [12].

LeMay's report on Agile business processes stresses the importance of diversity in the stakeholder array, particularly at this early stage in the product lifecycle [7]. Highsmith states that any Agile practices must at the minimum include constant feedback and customer intimacy, highlighting the need for any ecosystem/ART maintain feedback loops with stakeholders. This in turn provides "quick wins" when frequently demonstrating progress [4]. Sabharwal agrees in his Hands-On Guide to AgileOps. He refers to these initial steps of defining requirements as "business planning" and emphasizes the importance of improving agility by including stakeholder feedback. He includes "Continuous Feedback" among his 12 Agile Principles [15].

Throughout all the references, from the highest-level overviews to the detailed implementations of Agile, all agree that stakeholder analysis is a critical component of the early product definition stages. Furthermore, particularly in an Agile environment such as SAFe, frequent communication with a diverse array of stakeholders is a universally agreed upon feature that improves the final product and increases stakeholder satisfaction. Finally, the greater the project visibility, the better. LeMay

references a requisite level of executive buy-in as being an effective enabler of aligning stakeholder expectations and effecting project communications across a large and diverse organization [7].

1.1.2 Digital Twin Literature Review

Among the core tenets of model-based engineering are the concepts of Digital Thread and Digital Twin. Particularly relevant to this thesis is the concept of Digital Twin (DT), how different product classes represent a product's configuration within the DT concept, and whether or not the DT concepts can be aligned across disparate product classes. This research conducts a review of available literature in order to develop a framework defining the concept of the DT and its utility within the configuration management space. The DT framework is then used as the instrument to compare DT use-cases across product classes and evaluate them for alignment.

The Digital Twin concept has its roots in the Fourth Industrial Revolution. The first industrial revolution involved replacing man with the first manufacturing machines, e.g., the cotton gin. The second industrial revolution was the advent of electrification and the moving assembly line, e.g., Ford's vehicle production in the early 20th century. The shift from mechanical tools towards digital electronics is the basis of the third industrial revolution. And finally, the advent of the Internet of Things, full integration of sensor data, and automation are the foundation of the fourth industrial revolution, often termed Industry 4.0 [11]. Digital Twin is borne of this revolution – while sources vary on its precise definition, all have in common the idea of a completely digital representation of a product and the value added by such a representation.

NASA defines Digital Twin in an aerospace context in its "Modeling, Simulation, Information Technology & Processing Roadmap." "A digital twin is an integrated multi-physics, multi-scale, probabilistic simulation of a vehicle or system that uses

the best available physical models, sensor updates, fleet history, etc., to mirror the life of its flying twin [17])." A DT uses high-fidelity models of physical components, integrates on-board sensor data, and includes detailed information on the product instance's history, including maintenance, manufacturing, and other fleet data [17].

The definition of DT has undergone revisions and updates that mirror the changes in enabling technology. Kritzinger et al. talk about the origins of DT as referring to a digital mirror of a product, referring to this simple use case as a Digital Mirror. This has evolved to digital representations not just of the product, but also the processes involved, including manufacture. This detailed digital representation combines with fully integrated data flows between the virtual and physical objects to become the Digital Twin. The physical and virtual objects are so closely integrated that state changes of one object (physical or virtual) directly lead to a corresponding change of state in the other [6]. Barricelli et al. describe these as the three primary elements for a DT. There is a physical object, a virtual representation of that object, and a link between the two [1]. It is the last element, the data link between the physical and virtual objects, that enables continuous convergence and synchronization between the two [1]. It can thus be inferred that the usefulness of the DT depends not just on the accuracy of its representation, but also on the data acquired from the sensors. Pal et al. make this point in the textbook Fundamental Concepts to Applications in Advanced Manufacturing [11].

One would not attempt to implement a vigorous undertaking such as DT without the promise of significant benefits. One such benefit is the fact that because the DT operates on real-time data, it can be made to have the power to predict product performance and errors. Taking this a step further, if errors are found, the DT could suggest corrective measures in real-time. This has the benefit of preventing product downtime and costly inspections and decreasing preventive maintenance [11]. DT systems on-board could also mitigate damage or degradation by changing (or recom-

mending changes to) mission profile characteristics or system performance, prolonging product lifespan and increasing the probability of mission success [17]. Furthermore, beyond mere reactionary insights, data gathered through the DT can be aggregated across products to improve simulations and predictive capabilities [11]. The applications of data and insight enabled by DT are deeply integrated in defining the idea and operations of Product Lifecycle Management (PLM). "[A]n effective PLM system should always have access to a faithful view of the product status and information, from when it is planned and manufactured, through its time of use, and until the time of disposal [1].

Of particular concern to this research are the advantages enabled by DT technologies within the context of configuration management. Each DT model contains the as-built configuration. West et al. point out that by including detailed manufacturing information about each instance of a product (not just one DT per product class, but every individual product having its own, individualized DT information), the DT has the potential for large savings over the operational life. They specifically call out the importance of incorporating "the full range of 'manufacturing anomalies'" as an enabler to this level of insight [20]. As previously mentioned, DT-enabled advantages are transformative within the context of Product Lifecycle Management. Indeed, Singh et al. refer to the useful insights accrued from DT data and information as the "conceptual ideal for PLM," improving the development process, consistency in manufacturing, and the possibilities for reuse in subsequent lifecycle stages [18].

Product lifecycle management as a discipline covers the full operational life of the product, from design to disposal, as well as the full integration of people, data, processes, and systems throughout [8]. This review does not intend to cover all possible advantages of DT across that full timeline, but rather focus on advantages within the specific context of configuration management. Put differently, any DT framework used in the context of this research must focus on the returns gained in

the design and/or manufacturing process through careful definition, updates, and historical maintenance of a product configuration. Lim et al. describe a few of these benefits. By more closely integrating a product's design with its production, the DT wields greater influence in manufacturing, with the potential of making the production process more efficient, reliable, and adaptable [8]. Digitalization of a product through a DT allows for optimization, individualization, and closer monitoring through the design process, as well as greater flexibility in modeling [8]. Continual review of design and production data enables more rapid iterations in the design process as well as refinements in manufacturing [8]. Schluse et al. propose experimentation with DT representations as a way to improve all aspects of the product lifecycle. Specific to design and manufacturing, DTs enable simulations to improve systems and production engineering processes by optimizing and validating requirements, as well as leading to "cost-efficient development processes, better designs, and more reliable systems [16]."

1.2 Problem Statement

1.2.1 Current Situation and Approach

As stated in the introduction, this research was undertaken within the context of recreating complex product configuration definition and update capabilities for the next generation of engineering design software that previously existed with legacy software. These capabilities included an object unique to each product instance that contains a full history of that instance's assigned parts, assemblies, manufacturing anomalies, etc., as well as service that runs to regularly update that object according to the latest engineering and manufacturing data.

The research began by breaking down the overall mandate of recreating all of legacy program functionality into several different capabilities. The initial focus was on generating requirements for how the product would ultimately be defined and the

regularly running service to update that product definition. Moreover, the research explored a broader mandate, expanded from creating a product-specific solution to generating requirements for a solution that would be product-agnostic, applying to various product classes across The Company.

This thesis focuses on the application of an Agile framework to The Team's processes as they stood at the onset of the internship with the imposition of the broader, product-agnostic mandate. The researcher contends that by application of the proposed Agile framework, The Team's processes can be improved, additional use cases for their product assessed, and additional feedback generated. The researcher further contends that this additional feedback can then be compared to the proposed Digital Twin framework to improve alignment in DT use cases across the various new product classes.

Chapter 2

Method

2.1 Definition of Agile Framework

Agile is a method for organizing and structuring team efforts to iteratively define and improve upon a product based on frequent feedback from the full array of stakeholders. Agile consists of network analysis to optimize team alignment, stakeholder mapping to understand the full breadth of the product's operating environment, and solicits and implements frequent feedback from the identified stakeholders. Agile works because the network analysis ensures efficient use of resources to implement changes and iterate most quickly. The stakeholder mapping ensures feedback is holistic for requirements to remain on-target across the full breadth of network needs and use-cases. Lastly, the frequent feedback, iterations, and demonstrations mean that no product, requirement, or MVP gets too far off-target from the stakeholders' expressed needs. Moreover, this framework works because it can be applied at any point or points in the product lifecycle to improve team processes and the resultant product.

The literature provides an in-depth review of the systems engineering and Agile practices in an academic context. A framework is needed to refine the Agile method

from a general, academic review into a usable course of action for assessing and aligning the needs of various business units. This section clearly states the Agile framework applied in assessing and aligning systems engineering processes in generating requirements for configuration management needs across product classes.

The following framework was developed by conducting a literature review and distilling the salient points, as well as consulting with systems engineering project leaders on the appropriate and feasible methods for implementing the Agile process within the context of an existing and functioning team.

The literature review provides adequate background to define what Agile looks like for generating requirements in the product definition phase of systems engineering. First, the team must be properly aligned within the organization to fit within an ecosystem of similar classes of solution. Second, the team or its ecosystem must consist of an adequate representation of stakeholders so as to maintain self-sufficiency. In this case, it was impractical to bring in product design expertise full-time from across The Company, so identification of stakeholders and solicitation of feedback through interviews and surveys were used as a substitute. The process of positioning team efforts within an ecosystem, identifying adequate additional stakeholders representative of alternate product classes, and soliciting, translating, and dispositioning their feedback on configuration management needs represents the Agile framework under which Digital Twin use cases can be assessed and aligned.

The following is a tabular representation of the Agile framework for assessing and aligning DT use cases.

Agile Framework Step	Method
1. Ecosystem Alignment	Network Analysis
2. Identify Additional Use Cases	Stakeholder Mapping
3. Solicit, Translate, and Disposition Feedback	Surveys and Interviews

Table 2.1: Tabular representation of Agile framework

Rigorous application of each step in the Agile framework detailed here will lead

to improvements in generating Digital Twin configuration definition requirements. By applying this Agile framework, a large engineering and manufacturing organization can assess and align Digital Twin use cases across disparate product classes for configuration management.

2.2 Definition of Digital Twin Framework

A Digital Twin (DT) is a complete digital representation of each instance of a physical product, down to the part level, that contains its full history including all engineering, manufacturing, and sensor data. Leveraging DT features allows for improvements in all aspects of a product's lifecycle, including design and manufacturing. Within the design phase, DT features enable faster iteration and simulation capabilities against product requirements (obviating the need for a physical model for testing). Within the manufacturing phase, a mature DT representation informs all aspects of production and allows for adjustments to the process based off of real-time data. DT works because full integration of engineering, manufacturing, design, and requirements data feeds models and simulations, the Digital Thread. Utilizing the increased level of detail for each product at every stage of its lifecycle throughout the Digital Thread provides both micro-level adjustments and insights for individual product performance enhancements, and macro-level improvements to design and test a product line more rapidly against requirements, as well as optimize the manufacturing process. Because the DT is the prime input to generating the insights gathered by Digital Thread, its capabilities and functionalities must be carefully analyzed (and in this case, preserved) to fulfill the Digital Thread objectives.

The following framework will be used as the basis for discussing and assessing DT use cases. The DT is a developing concept in multiple disciplines of engineering and manufacturing whereby a physical product is represented in its entirety in a digital

space. Initial concepts of the DT began solely with virtual models of a physical product and have evolved as time and technology have allowed. Kritzinger et al. refer to these early stages as Digital Models, with information flowing manually from a physical version of the product back to its digital model. These static models were of only limited utility throughout the product's lifecycle since the data and insights only flowed unidirectionally from the physical world into the model. This constraint arose in part due to the limitations of legacy software systems, as well as available computing power, and data availability/portability [11].

Current formulations and future visions of DT technology include fully integrating digital information from physical sensors on individual instances of a product to automatically feed a persistent, hyper-detailed virtual version throughout the entirety of the product's lifecycle (from design to manufacture to service life to disposal). Furthermore, data and simulations from that virtual model will automatically feed the Digital Thread – documentation of and improvements to real-world processes for the product, to include corrective and preventive maintenance, performance improvements, etc.

The ultimate goal of DT technologies is full integration of real-world sensors and complex virtual modeling to feed the Digital Thread, transforming and adding value to all stages of the product lifecycle. Looking specifically from a Product Lifecycle Management (PLM) standpoint, DT has use cases throughout the phases a product undergoes: design, manufacturing, distribution, usage, and disposal. Of particular interest to this research are applications of DT in the design and manufacturing stages. As the literature review shows, within the design stage, DT has been proposed as a method for faster iteration on design implementations and more quickly assessing product effectiveness by simulating against requirements. From a manufacturing standpoint, DT has the potential to heavily influence operations, from digitalizing, optimizing, and individualizing production, to performing one-off or systemic ad-

justments to the production process based off of continuous monitoring enabled by DT. A key feature enabling these advantages is the method by which the product's configuration is represented within the DT.

Appropriately defining a product's representation within the DT thus enables the advantages detailed in Table 2.2. Feedback received from stakeholders on methods to improve product representation can be evaluated against these metrics to assess utility and potential for inclusion and dispositioned accordingly.

Design	Manufacturing
• Faster design iteration • Simulations of product effectiveness against requirements	• Digitalize, optimize, and individualize production • Adjustments to production process based off real-time data

Table 2.2: Tabular representation of DT framework

This paper will use the feedback provided by stakeholders identified during the Agile framework implementation to look at the depth of DT information required across product classes. This feedback will be assessed under this framework for DT use cases in configuration management and discuss the option value in maintaining additional capabilities beyond the current contractually or regulatorily required needs. By comparing each product class's DT configuration definition requirements to the advantages enabled by DT under this framework, this paper will assess and improve the alignment of DT use cases across product classes.

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 3

Data

3.1 Current Operations

The Company sought to create a product-agnostic configuration change management solution. This research sought to improve that process by application of an Agile framework against The Team's current processes. Thus, the first step was to document the processes and organizational alignment as were and to apply the Agile framework against the operations to improve The Team's processes. Data about The Team's processes was gathered by observing interactions, attending meetings, conducting interviews, and recording progress.

The Team was aligned under the office of the Chief Engineer, within the Commercial Engineering vertical. All project reporting, particularly with respect to Agile Release Train alignment, went through Commercial Engineering leadership channels. Additionally, The Team's mandate changed from creating a commercial product configuration change management solution to creating a product-agnostic configuration change management solution. This change in mandate was reflected in The Team's name change, from Commercial Team to Product Team, illustrative of the change in business need for the final solution to be agnostic of product class.

The Team consisted of engineers with exclusively commercial product engineering experience. Each engineer had deep experience in the technical matter at hand, specifically configuration definition and change management. However, that technical depth was narrow in scope, stretching solely across the commercial product class and not extending to use cases in other business units.

The Team had been working in an iterative manner, creating minimum viable product (MVP) milestones and planning progress according to SAFe Agile principles. MVPs were for developing requirements to define product representation from engineering and manufacturing part representations within the design software. These MVPs and the requirements derived under them were informed by the expert knowledge of the commercial product engineers on the team and the Project Owner. While The Team's overall mandate spanned several types of product configuration definition aspects and functionalities, focus was limited to improving individual MVPs, mainly those dealing with requirements generation for specific features and capabilities under the product configuration representation object and the software service that updates the product configuration.

The Team's typical activities included requirements gathering meetings held between experts to document product definition and change management capabilities in legacy software, as well as desired capabilities and definition traits for the next generation and translating those into functional engineering requirements/descriptions. Team members were dedicated to recording terminology and functionality in The Company's formal documentation methods. Finally, there were additional meetings to translate requirements from functional engineering descriptions to generic, hierarchical requirements for submission to The Company's requirements management software and IT development groups.

There were also existing standing meetings for interfacing with groups external to The Team. There were requirements reviews with stakeholders who had been

identified previously in the systems engineering process. These included commercial production engineering teams who were identified as consumers of the products of The Team's proposed functionalities. Team members were also responsible for creation of an Interface Control Document (ICD) to manage expected external interactions with the proposed software. Along the same lines, there were team members documenting team interactions and information flows. These were interpersonal interfaces, as opposed to logical/software interfaces covered in the ICD, specifically covering the flow of requirements and documentation to The Company's IT and Architecture groups.

Finally, there were team members whose focus was in demonstrating functionality. There were two types of configuration definition and update functionality proposed: a prototype of The Team's proposed basic configuration object and update service, as well as native configuration management functionality within the next generation of design software.

Under the former, narrower mandate, The Team was conducting all the required steps to create requirements for a commercial, product-specific configuration management solution to work with the next generation of engineering and design software. The team was aligned properly to report through the Commercial Engineering vertical and consisted of deeply experienced commercial product engineers who were intimately familiar with the capabilities that needed to be recreated to work with the next generation of software. All the expected steps in the Define Product/Milestone 2 stage were underway. Product functionality was broken down into iterative blocks by Minimum Viable Product (MVP). Team experts were generating functional requirements for each MVP, translating them into more generally applicable hierarchical requirements, reviewing those requirements with previously identified stakeholders, and documenting all the interactions, interfaces, and terminology.

Having documented The Team's current processes and the broadened scope of the project's mission, it became clear that there were gaps between where the team was

currently operating and what they would need to do to accomplish the new, broader mission. By applying the aforementioned Agile framework, The Team's processes could be improved and additional feedback generated. This additional feedback would in turn be assessed against the Digital Twin framework to improve DT use case alignment across The Company in fulfillment of the newly broadened mandate.

3.2 Comparison against Agile framework

For the purposes of this research, the researcher established a baseline for The Team's operations. Here the research shifts to see how The Team's operations changed from the baseline when the proposed Agile framework was applied. The Agile framework addressed the delta between where The Team's former processes (creating requirements for capabilities solely informed by commercial product experience and expertise and siloed in a Commercial Engineering vertical) and the desired end state, which was proposing capabilities available and applicable to all products across the enterprise.

When The Team's mandate changed, the additional breadth of scope created a delta that was not addressed through current team activities. Through application of the Agile framework, the research identified gaps in the systems engineering process underway and began closing them according to best practices suggested in the available systems engineering, model-based systems engineering, and SAFe Agile literature.

Applying the Agile framework involved three distinct parts. The first was the network analysis which evaluated the realignment of The Team's statement of work into the Product Lifecycle Management (PLM) vertical. The second step of implementing the Agile framework was identifying additional stakeholders to fill in the expertise gaps created by the broader mandate. The Stakeholder Value map identified those teams whose input was most valuable in informing the end state of The Team's pro-

posed capabilities. Finally, the Agile framework required solicitation of stakeholder feedback that would subsequently be translated, compared against the Digital Twin framework, and dispositioned to improve the resultant functionality's applicability across product classes.

3.3 Network Analysis

The first step in applying the Agile framework discussed in Chapter 2.1 was evaluating The Team's ecosystem for proper alignment. The purpose of applying the network analysis methods here was to better understand the organizational structure, hierarchy, and architecture, and to get a better idea of where The Team was situated within The Company and thus how they relate to the different parts of the enterprise. Specifically, this translated to knowing how The Team was integrated with other teams under the office of the Chief Engineer, and where the providers and consumers of The Team's data were situated. By fully assessing and aligning The Team's work under a Reference Model, The Team would be more closely adhering to the Agile framework. This would improve the subsequent stakeholder analysis and streamline future feedback channels.

Data for the network analysis was gathered entirely via networking and interviews. After several initial interviews within The Team to gather information about outside contacts, the researcher was recommended to speak with a technical expert on a team working on a similar problem in parallel. Additionally, the research was recommended to speak with veterans within The Company who had more visibility on The Company's efforts to align engineering functions, the roles of the various teams and ecosystems under the Engineering vertical, and better insight into configuration definition use cases across the enterprise. It was of course impossible to conduct interviews with all members of The Company's Engineering organization; however,

through a review process with The Team's manager and Project Owner, the data gathered was deemed sufficiently representative to understand the different product functionalities and capabilities being satisfied within the Engineering vertical.

As previously mentioned, The Team was realigning their statement-of-work, as directed by senior management within The Company's Engineering organization, to better organize products by their function, according to the Reference Model framework. There existed two similar teams in two different parts of the enterprise working on the question of how configuration would be defined in next-generation software. The first was The Team into which the researcher was embedded, and the second ("Team 2") existed under the completely separate Product Lifecycle Management (PLM) Engineering vertical. Additional interviews would be necessary to understand how these solutions differed and if a vertical realignment was justified under this Agile framework.

The Company had several major units directly under the CEO responsible for the development of the vast array of products offered as well as functionality and operation of the business. To simplify slightly there were functional organizations and product business units. Functional organizations included the Office of the Chief Engineer, Chief Information Officer, as well as Finance, Human Resources, and Legal. These were distinct from the product business units like Commercial Products, Defense Products, and Customer Support. The Team was under the Office of the Chief Engineer. The Engineering Organization was broken down into several divisions including one for each of the product business units, with a division for Defense, Commercial, Customer Support, and Research and Technology. The Team's statement-of-work fell under the Commercial division, reporting specifically into the Commercial Product Engineering Integration team. This directly explained why The Team's work had been exclusively focused on commercial applications, since the team was situated within a Commercial branch of the organizational tree up to the executive level.

Under the Agile framework's assessment of ecosystem alignment, The Team remained under the office of the Chief Engineer but was entirely removed from the Commercial division. The Team integrated instead into the Engineering Strategy & Operations division (which is not affiliated with any specific product business unit). Within Engineering Strategy & Operations, The Team was moved into the Model Based Engineering (MBE) Agile Release Train (ART), alongside other teams that are working on challenges in Product Lifecycle Management (PLM). These PLM challenges included configuration definition questions that The Team was trying to solve and were universal to The Company's product lines (though they appear in varying degrees). The new MBE ART had no delineation between Commercial, Defense, or Customer Support at any branch or node of the organizational tree. From a Reference Model ecosystem perspective, this represented an immediate shift in the focus of The Team from a Commercial-focused team sourcing inputs from the other business units to a more universal team whose inputs, products, and processes should come from across the enterprise.

In addition to the vertical realignment within Engineering, The Team found itself more horizontally aligned with teams under the PLM ART. Existing teams under the PLM ART were performing essentially the same function of defining product configurations in the new software. This represented the same "solution kit" as The Team, with a couple distinct differences. Existing teams under the PLM ART were exploring solutions native to the new software, not an add-on piece of software like was being proposed by The Team. Furthermore, interviews with engineers within the PLM ART indicated that the native functionality they were exploring had only limited utility in defining a product's configuration and rudimentary configuration management functionality. Additionally, the PLM ART had existing and robust post-delivery feedback loops to a limited set of stakeholder customers. Each completed requirement was demonstrated for end-users in the Military product class and feedback solicited for

incorporation in future releases. This led naturally to the second step in implementation of the Agile framework.

3.4 Stakeholder Value Mapping

The second step in implementing the Agile framework was identifying additional use cases via stakeholder value mapping. By changing The Team's mandate to be product-agnostic, but with The Team's personnel consisting solely of Commercial engineering experience, stakeholder value mapping would improve the breadth of expertise that could be consulted and included in forming product requirements. This was done through interviews with project management and owners in the new PLM ART, as well as an analysis of shortfalls in expertise created by the team's composition.

The researcher conducted interviews with project managers and owners both within The Team and its former commercial-specific hierarchy as well as the new PLM ART and its hierarchy to get a better understanding of which product classes would be considered the most representative candidates and whose input would be most valuable. To confirm the value of the stakeholders, during the subsequent interviews and surveys, stakeholders were asked to self-assess the relevancy of the functionalities that The Team was proposing. These self-assessments were used to weight their feedback and rank the stakeholders accordingly.

Concerning The Team's expanded scope, initial interviews with The Team's manager and Project Owner indicated that the team desired to better understand additional stakeholders and their needs under the broader mandate. These stakeholders did not need to represent an exhaustive list of all product development teams outside of Commercial Products, but rather key stakeholders within other product business unit verticals (Customer Services and Defense). These key stakeholders would be

representative of product classes and would be at the appropriate stage of development to provide relevant feedback on how they currently implement configuration management solutions.

Working on information gathered through the network analysis, interviews were conducted with members of Team 2 within PLM to better understand the existing stakeholder network in the new ART. Team 2 regularly demonstrated change management software functionality to the Military product class. Further interviews with Team 2 pointed towards change management experts in the Military-Derivative of Commercial Product (MD) class. Subsequent interviews and surveys confirmed that the Military-Derivative class as a major stakeholder whose input would be important, given that the primary MD product is a military-derivative of a line of commercial products and likely would avail itself of the complex configuration management functionality proposed by The Team. These MD stakeholders formed a subnetwork of contacts within the Defense organization who would be knowledgeable about the configuration change management processes currently in-place in the MD product class and how they were developed alongside commercial products.

Additional interviews identified the Sub-Assembly Internal Supplier class as a candidate for inclusion. Products in this class were not sold by The Company as stand-alone units, but rather incorporated into The Company's major product class designs. Lastly, The Team's Project Owner also suggested a candidate stakeholder product class within Customer Services. Further interviews with the Repurchase and Conversion class team within Customer Services confirmed that this product class had specific, unique configuration management needs. In summary, the interviews and surveys identified five product classes: Commercial, Military, Military-Derivative, Sub-Assembly Internal Supplier, and Repurchase and Conversion (Figure 3-1). The combination of the depth of commercial experience contained within The Team with the additional feedback solicited across these product classes would add value and

more holistic breadth to the discussion of how disparate product classes across The Company utilize DT configuration management.

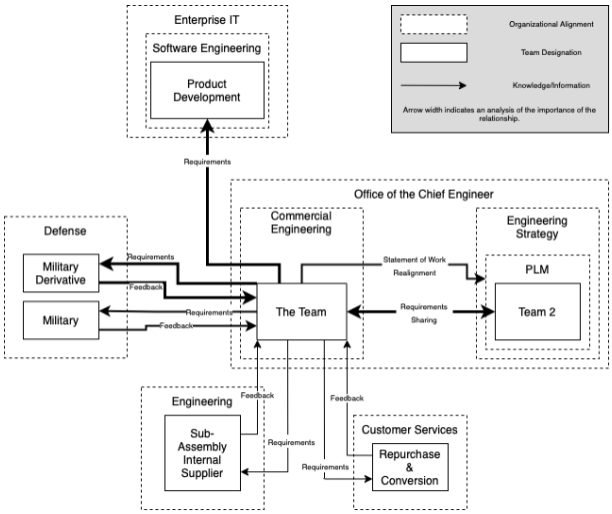

Figure 3-1: Stakeholder Value Map

3.5 Survey and Interview Questions and Distribution

Having identified the additional stakeholders whose input would improve breadth of applicability of The Team's end product, the next step in implementing the Agile framework was to solicit their feedback via interviews and surveys. By requirement, these questions were targeted enough to make best use of the limited time that interviewees had available, but general enough to provide degrees of freedom in the response for the interviewees to elaborate.

It was necessary to craft a set of questions that the interviews and surveys needed to ask to achieve the objective of identifying areas where The Team's product could be improved to be made more broadly applicable to the previously identified product classes across The Company. Within the Agile framework, the feedback solicitation sought to fill two objectives. The first was to socialize basic capabilities proposed by

The Team to other product classes across The Company that may not have been as familiar with them. The second objective was to receive feedback from the stakeholders both on how their teams conducted configuration change management, including the software and computing environments used, as well as their opinions on the applicability of the proposed functionalities to the product class on which they were working.

The first step was to introduce The Team and a general description of its focus, configuration change management. Additionally, the surveys had to introduce the proposed configuration change management functionality. To descope the problem and focus on The Team's immediate needs, the surveys and interviews focused on the product configuration definition object and service run to update that object. Draft descriptions of The Team, configuration change management, and these two features were created through study of available documentation and consultation with configuration change management experts. These were subsequently published to The Team for comment and refinement. Upon incorporation of feedback from The Team's configuration management experts, these descriptors were included in the surveys and interviews in advance of the questions that would pertain specifically to them.

The questions were created through an analysis of The Team's needs and formed to solicit information to meet those needs. Through consultation with The Team's Project Owner, this problem was phrased as a desire to better understand the enterprise needs and business opportunities that The Team could fulfill for additional, non-Commercial stakeholders. Digging deeper, this meant understanding what configuration change management processes had already been established in other product development teams, how well they worked, and how they could be improved. Furthermore, it was necessary to understand how those classes of product development teams would be able to use the functionality proposed by The Team (specifically

the configuration definition object and update service) and how they thought those solutions could be improved or more closely tailored to their product class's needs. Open-ended responses via survey or interviews allowed the subject matter experts to provide sufficient depth to and elaborate on their responses. Additionally, asking the interviewees and survey respondents to attach a numerical rating to their opinion of the proposed functionality's utility assisted in ranking stakeholder and feedback value.

The surveys and interviews began by asking for individual information about the respondent, as well as providing the MIT-required confidentiality statement. Any personally identifiable information in the responses was scrubbed before public presentation to maintain the subject's public anonymity. The questions then continued by asking about the change management processes and software currently in use on the subject's team, as well as how those processes could be improved, prior to introducing The Team and functionality definition, to avoid preemptively presenting information to the subject that might bias their response.

The surveys and interviews then provided descriptors to introduce The Team and its subject matter. An abstracted version of the surveys is provided in Appendix A. The questions and descriptors in Appendix A provided a structure for both surveys and interviews. These fulfilled the dual objectives of introducing The Team, its view on configuration change management, and the initial proposed functionalities, as well as soliciting both free-response and numerical feedback on the configuration change management processes in place elsewhere in the business and an opinion of the fit of these proposed functionalities. Since the additional stakeholders were identified through the network analysis and stakeholder mapping, the surveys were distributed and interviews conducted once the question sets were finalized.

At the conclusion of the interviews and surveys, there was a significant amount of raw data, especially in text form. While much of it was useful in informing The Team

of the individual's perspective, it was necessary to distill out the genuinely valuable pieces of feedback from the overall information provided.

3.6 Feedback Aggregation and Translation

The surveys and interviews provided extensive opinions, facts, references, and even diatribes. Not all of it was germane to the topic of configuration change management or the functionalities proposed by The Team. It thus became necessary to institute a process to translate the raw data gathered in interviews and surveys into specific feedback items that could be discussed by The Team.

Raw feedback from interviews and surveys was sanitized of any identifiable information per MIT's Committee On the Use of Humans as Experimental Subjects (COUHES) policy. This involved removing names where provided, as well as specific career information that might be used to identify an individual. Leaving only a generic descriptor of the respondent's role and team, the remainder of the feedback was discussed during daily meetings with a configuration change management expert whose experience helped inform the data extraction process. Individual line items of specific feedback were pulled from the raw data and entered into an Excel spreadsheet for further discussion and disposition by all members of The Team.

3.6.1 Commercial Data

In order to better understand the model-based systems engineering processes in-use at The Company, it was necessary to understand how each product group utilized DT representations to capture their product configurations. The Team wished to understand the other methods of product representation around the enterprise, so the first step was to establish a baseline of how the Commercial product class utilized models to represent their product configurations, as compared against the DT framework

discussed in Chapter 2.2.

Through comprehensive review of Commercial product configuration management training resources as well as daily mentoring sessions with a configuration change management expert, the researcher systematically reviewed the in-place practices, procedures, and literature on how a product goes from a set of options presented to a customer to a representation in design software and how that representation is maintained and changes over time.

Each line and sub-model of a Commercial product begin as a set of options. Many of these options are unique to the line and sub-model of the product itself and are not selectable by the customer, however many other options are selectable by the customer. The full slate of selectable options is contained in a digital object called the Customer Option Slate (COS). Once a customer selects the desired options for a specific product, this set of selected options is stored in a digital object called the Production Option Slate (POS), as well as the non-selectable options that are germane to the product definition. The POS thus contains the full set of selected options that define a product; however, it does not contain any part-specific engineering or manufacturing data.

For current major models of Commercial products, engineering and manufacturing data is created in various engineering design software suites. While the particular environment may change for a given major model, all Commercial products follow a similar process for modeling individual parts and assemblies. The engineering and manufacturing data for a part is stored in a drawing. Parts are assembled together digitally to form "installations" or "modules," with the verbiage depending on the particular software. Each module is assigned an option expression, defined by the engineer, that specifically identifies which options in the COS/POS this module fulfills. Each module is also assigned by its engineer an availability statement, which indicates which line numbers (unique identifiers for each instance of a product) to

which the module can be applied.

Module approval is controlled through a state assignment, the details of which and the approval process are outside of the scope of this research. But it is important to note that, with limited exceptions, modules must be in an "Approved" state in order to be included in a product's ultimate configuration. Once a module is approved, it contains the full engineering and manufacturing data for all of its parts, as well as the specific options to which it applies, and the product line number to which it can be made available.

The Production Design Object (PDO) is a persistent digital object unique to each line number (individual instance of a product) that contains all of the installations/modules (assemblies of parts) that have ever been associated with the product. This includes all installations that satisfy the customer's selected Options, as well as installations that have been force-included or –excluded by Design or Manufacturing Engineers. The attribute indicating whether or not a particular change/revision of a module is associated with a particular line number of a product is called "effectivity." Because this object is persistent, it is a complete record of all design and manufacturing changes that a particular instance of a product has undergone during its design and build lifecycle, from customer order until product delivery. Historic iterations of the PDO contain a relationship to installations/modules that are potentially not part of the final product but were associated with the product at some point in its history. From the PDO, a report can create the Engineering and Manufacturing Bill of Materials (EBOM and MBOM).

The Update service is a custom-built configuration management algorithm used for Commercial products. It is a nightly software algorithm that ingests input data from engineering design software to update the PDO for all Commercial products. At a high level, the Update algorithm first evaluates for each product instance where in its lifecycle it currently resides, and then updates the PDO according to three filters.

The first step of the Update algorithm is to evaluate the PDO for the production status of its product (Forecasted/Implemented/Certified/Delivered). Forecasted products are those that have been ordered, but for which construction has not begun. Implemented products are currently under construction. Upon completion of construction, a product must become Certified, and then following delivery to the customer, it transitions to Delivered. The Update service follows specific rulesets for each production status, the details of which are beyond the scope of this paper.

Following the PDO state evaluation, the Update algorithm runs filters to ensure that each Commercial product's PDO contains the most up-to-date information about which engineering and manufacturing drawings are satisfying the selected product Options. The product of the Update algorithm is an identification of which installations are associated to the PDO, which in turn contains a complete record of all engineering and manufacturing data associated with that particular product instance's production lifetime.

3.6.2 Military Data

With the understanding of how The Company's Commercial business unit models a product and its configuration, the goal of The Team was to better understand how teams across other business units within the enterprise conducted similar product modeling. The first team interviewed and surveyed was the Military product class. The representative product in this class was being developed under The Company's Defense business unit and competing for contract bids beyond the Conceptual Design Review process. Through interviews and surveys of experts in the Defense business unit, the research sought to improve the understanding of how the Military product class defined model configurations.

Military product configuration data was stored in an offshoot of the older engineering design software in-use in Commercial products, in a similar manner to how

configurations are stored in the Commercial business unit. Changes to parts and modules were initiated via a Change Request system and managed and approved by engineering and managerial experts prior to incorporation in any final configurations. Where the Military configuration definition departs significantly from the Commercial product definition is in the detail required for final configurations. Whereas Commercial products require a full history of all modules, parts, and assemblies ever associated with the product throughout its design and production history due to Government Agency certification rules, the Military product certification process is less rigorous. This is in part due to contractually-desired homogeneity between product models – the purchasing military entity preferred that all Military products be configured as similarly as possible to facilitate maintenance, interoperability, and interchangeability. While all Military-class products required a Bill of Materials (BOM) that contained a complete list of all the modules and parts, including the engineering and manufacturing data, this needed only be representative of the configuration at that particular point in time. As such, no service like The Team's Update service existed in the legacy software environment, and any product's configuration was fully represented in a BOM, as opposed to a complete history like the PDO object.

In experimenting with product modeling and DT representations, Team 2 under the PLM ART was able to sufficiently encapsulate the product detail using the next-generation software's limited, native effectivity functionality. While the native effectivity function was limited in its history, it was sufficient to represent a final product configuration. For any given instance of a product, a report could be run natively within the software environment to produce a BOM containing detailed configuration data that satisfied military requirements for tracking final product configuration.

3.6.3 Sub-Assembly Internal Supplier (SAIS) Data

The next class whose product definition and modeling methods were explored here is the Sub-Assembly Internal Supplier (SAIS) team. SAIS was an internal division under The Company's Engineering business unit that had its own vertical, independent of Commercial or Defense engineering. Their mission was to design and manufacture guidance, navigation, and control assemblies for incorporation in various products across The Company's portfolio. Interviews and surveys were conducted with managers and engineers within the SAIS team to better understand the configuration change management processes in place, as well as how the product serves the various needs of the business.

SAIS configurations were intended for incorporation in both Commercial and Military applications, which are maintained in disparate engineering design software. SAIS configuration data was itself represented in a legacy engineering design tool. However, because SAIS serves as an internal supplier to various product teams, the configuration data is ported over to the end product design team's native environment. As such, it is treated essentially as a module for effectivity inclusion within a final product representation in its destination program.

The fact that the SAIS configurations are designed to be incorporated into other products' final models makes them unique among the teams interviewed here. Much of the feedback received by these teams dealt with interoperability concerns, rather than how the modeling process can be improved, which will be discussed later.

3.6.4 Repurchase and Conversion Data

The Repurchase and Conversion (R&C) program under the Customer Services business unit purchased used Commercial products from carriers outside of The Company for the purpose of overhauling and retrofitting them to be resold under a new purpose. In order to better understand this product class's DT configuration definition needs,

the researcher conducted surveys and interviews of engineers in the R&C team.

The process of producing Commercial products is significantly different from the process of modifying or converting one. Commercial production starts from a blank slate and manages a model configuration from assembled parts. However, R&C is a modifications process, not a production process. These products have been in-service for 12-15 years outside of The Company's process control. Modifications and conversions on products of this type often begin on incomplete or out-of-date configuration data. Each product repurchased by R&C arrived customized per the previous owner's specifications, but R&C's job was to remove all customer options and individuality from the products. The R&C products are designed by contract to look as similar to each other as possible after R&C completes the conversion process. If a customer purchasing a converted product wishes additional customization, then it is sent to the Modifications customer support team, but this is outside of the purview of R&C. The engineers, the customers, or the certifying authorities do not care about changes made to the product on the production line up to the point of sale, only how the product looks when it leaves the conversion factory. These final configuration models are built and stored on the same engineering design software as the original models, but upon completion of the process, there is no need for an PDO object containing full instance history, but rather a simple BOM report generated off of current effectivity for a given product instance. Modules in this configuration representation are instead formatted to feed service bulletins that contain detailed information about the product.

3.6.5 Military Derivative Data

The Defense product class with by far the most in common with commercial product manufacturing was the Military Derivative (MD). The MD product class was borne of the commercial product design and was produced in-line alongside all the other

commercial products from which it is derived. As such, understanding the similarities and differences in how the MD team models product configurations was integral to building the future solution set of configuration change management software. Data was gathered here through several interviews and surveys with managers, contractors, and engineering experts both currently on the project and who were present at the MD product's inception and assisted in architecting the changes to legacy software systems that would facilitate in-line manufacturing of MD products alongside commercial ones.

The MD product was originally proposed to be a modification of an already produced and certified commercial product, however the contract was rewritten to specify that the MD product would be produced in-line, as opposed to modified or retrofitted to meet military specifications. As such, MD would use the same engineering software as Commercial to manage its configuration, including full use of the PDO and Update service. Since the MD is ultimately a Commercial design under the same production system, product representation at a high level is essentially identical to how Commercial products are represented, with several key differences that facilitated meeting government and military compliance regulations and standards.

Beginning with similarities in the design and production systems, both MD and Commercial use the PDO to represent their product configurations and the Update service to maintain that object. Modules and part assemblies are represented using the same software, revision, and approval processes. Deficiencies in legacy software were identified in common to both product classes. MD and Commercial products required greater flexibility to effect changes at lower levels (individual parts and submodules) without affecting the certification of the overall design. The Team had already taken this feedback into account with product requirements addressing the need for greater flexibility in part manufacturing, ordering, and inclusion in the Bill of Materials.

MD products, and Military products in general, require documentation pulled from the contract and placed into the engineering and manufacturing data down to the level of individual parts. In the legacy software systems, this information had to be added on via flags and design notes. While outside the scope of configuration definition, it was desired that the next generation of software have document management capabilities to automatically incorporate this functionality. This is an example of feedback that, while inapplicable to The Team's immediate mission, was easily forwarded to other subject matter experts within the PLM ART whose role was exploring and defining the new software's native capabilities, effected by the ecosystem realignment.

The choice to manufacture MD products alongside Commercial products represented a departure from the then-current use-case of the configuration definition software and required several modifications in order to meet government and military regulations. These changes were integral in the MD program's success at its inception and enabled a strategic competitive advantage.

MD products require departures from the standard set of rules that allow only parts with certain levels of governmental certification to be added to the final configuration. The Update service must be able to read and interpret this additional data from the product top-level down to the part level that enables in-line production and delivery of MD products. Additionally, many parts on an MD product undergo revision from the Commercial standards to meet DoD specification requirements. These revisions and departures from standard specifications need to be tracked, typically a manual process done for each product instance. Interviewees suggested that this type of tracking be done at the top level of a configuration. This could be accomplished through a separate configuration representation specifically calling out the altered modules or having the PDO automatically track the removal of Commercial-grade components and their replacement with comparable Military-grade components.

When it comes to configuration change management, this research has discussed the implications of the module state on its inclusion in the Update algorithm. As far as the mechanics of the algorithm, particularly its three-phase evaluation, this feedback does not change the process. The algorithm will either run for a given product or it will not run, based on its assessment of the phase the product is in. However, the differences in states for how part assemblies/modules are represented across military and commercial programs will trigger a different response by the algorithm in its module state filter. These changes are addressed in a basic feature of The Team's product architecture – ingesting generic engineering and manufacturing data to build a configuration. The Update service must be modified to accept and properly interpret the additional states used by MD program.

Additionally, the PDO and Update service must be able to determine the appropriate and current level of certification of modules or parts at the PDO level for a product. In addition to the module state, military-derivative programs have separate certification requirements for products, both at the individual part/module level, as well as up to complete assembly of the final product. Since the Update service runs to create a PDO for a given product instance, any changes in how military-derivative programs certify parts or define a final configuration must be considered, or else the ability to conduct in-line manufacturing of these products could be affected. One area where these differences are most frequently realized is in the certification of individual parts or modules. In Commercial products, parts that have not undergone the rigorous governmental and commercial certification process cannot be included in final configurations. However, on military-derivative products, many parts are not certified through this process, instead undergoing certification separately, in some cases even after production and delivery. Thus, there remains a need for certain uncertified parts to be included in the Update filter and in the final configuration object of certain military products. This feedback specifically calls attention to these uncertified

objects and the engineers will ensure that the configuration service will include or exclude parts based on flagged information.

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 4

Discussion

4.1 Value generated from network analysis

Through the network analysis methods of the Agile framework discussed in Chapter 2, it became clear that the reorganization of The Team under the PLM ART was an improvement. The Team was better aligned to take advantage of existing data pipelines, networking channels, and stakeholder expertise within the MBE group to source requirements, demonstrate capability to end-users, and receive feedback. The Team's move also aligned the two DT configuration management solutions under a Reference Model, given both the similarities and the differences between the two products. Because the two programs were working on the same problem of configuration change management but from two different approaches, they will necessarily come to two different solutions. Team 2's solution to configuration change management was simple, lightweight, native to the software application, and appropriate for simple products that did not require rigorous product definition complexity. Many Defense products fell into this characterization. However, Team 2's solution is not usable for extremely complex products like those offered by the Commercial business unit. It is for these more-complex products that The Team's solution to configuration change

management existed as an add-on software functionality.

In addition to team alignment under a Reference Model and grouping products by solution groups, the network analysis also provided value for the second step of the Agile framework. The realignment provided ready access to existing stakeholder networks, as well as expertise eager to point the direction for future exploration. The network analysis allowed the researcher to utilize and expand the existing network of contacts in the PLM ART, taking advantage of the Agile principle of fostering a whole-team culture and maximizing self-sufficiency. Taking this approach allowed the researcher to introduce The Team's mission to key stakeholders who were already intimately familiar with the stakeholder network already in existence within the PLM ART while simultaneously incorporating knowledge The Team's understanding of the network, adding value by filling knowledge gaps in both teams. These additional contacts became instrumental in the stakeholder mapping stage of applying the Agile framework.

4.2 Additional product classes identified through stakeholder mapping

The stakeholder mapping step of the Agile framework served to identify additional product classes whose methods for configuration definition and change management were deemed sufficiently broad to improve The Team's understanding of DT use cases across The Company. The product classes identified (Military, Sub-Assembly Internal Supplier, Repurchase and Conversion, and Military Derivative) all incorporated configuration change management in some form, and the inquiry step of stakeholder mapping allowed the researcher to identify key personnel within those teams whose input would be valuable. This value extended beyond informing The Team's immediate needs and proposed functionality. By establishing the broader network of expertise,

The Team gained a better understanding of the full enterprise's configuration change management needs as well as a deeper pool of resources from which to draw expertise for future configuration change management functionality.

There were drawbacks to the survey and interview approach. Because many of these experts were employed full-time to their projects, it was difficult to generate adequate buy-in and participation rates. This was especially true for the government-contracted projects, which accounts for all the Defense product teams, where cost and time accounting was especially stringent. Without a dedicated charge code, these surveys had to be completed incidentally to those projects and charged to overhead. This represented a significant drawback to this technique and reflected why the literature suggested that teams consist of a broad representation of the stakeholders and subject matter experts assigned full-time to the project. This was also one way in which the lack of executive buy-in referenced by LeMay hurt the project. Because knowledge of this internship project was localized to The Team and its immediate vertical, the research effort didn't have full visibility across the enterprise. This led some teams to question its legitimacy (potentially thinking the survey or interview communications were a phishing attempt), as well as running into the aforementioned cost and time accounting issues.

4.3 Assessment and alignment of Digital Twin use cases using DT framework

The greatest impact sought by applying the Agile framework was a change in understanding and implementation of Digital Twin configuration change management to be more broadly applicable to all product classes across The Company. The purpose was to ensure that the migration process maintained and improved DT fidelity from older software as The Company moved to state-of-the-art software better suited to

utilize DT features in the Digital Thread.

The data gathered through the surveys and interviews of the broad stakeholder network of multiple product classes allowed The Team to improve their understanding of The Company's DT configuration definition needs from a narrow, Commercial-specific implementation to one that would serve a larger swath of product classes.

Beginning with the Commercial product class, evaluating the PDO and Update service DT features against the DT framework clearly demonstrates the value of maintaining this DT functionality within the Commercial product class. By creating change records of all engineering and manufacturing data and regularly running the Update service, each product's design iteration speed is increased. Moreover, the complete record contained within the PDO is routinely referenced and required during the certification process for Commercial products, improving product effectiveness through demonstration and inspection of the PDO within the DT. The manufacturing benefits of these DT features are also abundant in the Commercial product class. The array of customizability and product individualization offered by these DT features is a competitive advantage maintained by The Company specifically enabled by this novel implementation of DT functionality. Moreover, the digitalization of design and manufacturing instruction edits within the software are incorporated in near-real-time due to the Update service, ensuring the production process is always working with the most accurate DT representation.

Comparing the Military product class's DT product representation needs against the DT framework shows that while the more simplistic use case may not benefit greatly in the near-term from functionality such as the PDO and the Update service, there is significant option value in maintaining the functionality for potential incorporation in future Military product designs. While the Military product chosen here had completed its conceptual design review and thus has been mostly finalized, other Military products earlier in the design process would be able to make use of the ability

to rapidly iterate and test design changes and capture that value through the PDO. Additionally, the manufacturing anomalies captured in each product instance's PDO are capable of informing optimization algorithms should they ever be implemented in the future. Between the basic DT functionality native to The Company's next generation software and the option value preserved by maintaining advanced DT capability through the PDO and Update service, DT use cases for Military-class products are covered by solutions within the PLM ART.

The most specific piece of feedback from the SAIS team dealt with the manner in which clarifying data was represented in the engineering and design software systems. Because SAIS-type products were incorporated across many product classes and use cases, tailoring the SAIS engineering and manufacturing data for all use cases was time consuming. Attaching additional detailed data to individual parts for various use cases across multiple production software environments was a tedious and manual process. While The Team deemed this feedback out-of-scope when discussing the PDO and Update service, with respect to the DT framework, it represents a challenge to the individualization of production. Properly digitalizing the methods for including additional detail would alleviate the manual tedium prevalent in this product class, as well as improve design iteration speeds. Furthermore, moving to a single environment in which DT configurations are represented would reduce the number of environment-specific tailoring required, improving DT alignment. The single design engineering environment is precisely the solution proposed by The Company and having add-on software in lieu of a completely separate, additional environment is in furtherance of that effort.

In viewing R&C product class DT alignment from a product homogeneity lens, the R&C product class is similar to the Military class of products, where a "cookie-cutter" end state was desired. In the case of conversions, it is precisely the deep levels of customization that are enabled by the Option selection process that R&C

was trying to strip out. The depth and complexity captured by the PDO carried no utility to a product line that cares only about the end-state and works hard to ensure no variation exists after the conversion process.

However, in departure from the Military-class of product, the application of the DT framework to R&C products is more nuanced. While this product class is the least likely to benefit from the capabilities offered by the PDO and Update service in the short-term, the long-term evolution of DT technology should serve the R&C product class's needs well. As DT connectivity and Digital Thread mature, product configurations like the PDO will be able to be maintained throughout the product's service life. Upon repurchase, the R&C team will have better data to inform the operations required to retrofit and refurbish, affecting and improving the unique R&C "production" process. Therefore, continuing to pursue complex configuration representation technologies like the PDO remain aligned with the R&C product class's long-term interests.

Application of the DT framework to the feedback gathered from the MD experts is where The Team saw the greatest improvement in understanding and alignment of DT use cases. The modifications to both the PDO and the Update service that allowed for uncertified parts to be included on MD but not Commercial configurations was instrumental in allowing for individualized production. Furthermore, aligning both the production systems and the DT representations allowed for in-line manufacturing of MD products alongside Commercial ones. A single manufacturing process is significantly closer to optimal and was a significant source of cost savings for the MD team. These cost savings were cited by engineers familiar with the project's inception as being a major reason why the contract was awarded, and the program remained successful.

The Team's working set of requirements did not previously include an additional object for or explicit tracking of departures from Commercial specifications included

on an MD product. Without this feedback, The Team's proposed PDO functionality would not have indicated the certification status of individual parts unique to MD product. This feedback and the stakeholder mapping resulted in the formation of a working relationship with the MD team to continue discussions on how the requirements must continue to evolve to fully encompass the changes made to enable production of MD products in-line with Commercial ones. Applying the DT framework here, the effect of this feedback was ultimately to improve the digital representation of individual parts and their nuances, which in turn would increase the speed at which designs can be implemented and iterated upon, as well as improve the production process.

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 5

Conclusions, Recommendations, and Future Work

Taking a step back, this research was presented as a case study in migrating from legacy software systems in a large corporation towards new software that enables greater utilization of state-of-the-art manufacturing technologies such as Digital Twin and Digital Thread. The research focused on a single aspect of a generational migration effort, presenting a case study on preserving a core legacy DT functionality for Commercial products and expanding it to become product agnostic and compatible with the new software environment. While Agile is commonly used in development of software, the framework presented here broadened the definition to a set of principles to guide work and generate requirements. The insights generated informed The Team's understanding of The Company's DT business needs across use cases, as compared to a framework built to measure how specific DT functionalities contribute to advantages made available by the Digital Thread.

Application of the Agile framework proposed here clearly improved The Team's understanding of and applicability to DT use cases across The Company. The network analysis provided academic and business justification for The Team's realignment

from their former Commercial-only reporting structure into the product-agnostic PLM ART, alongside other teams working on configuration change management solutions. By migrating their statement-of-work under PLM, The Team was able to access a greater amount of enterprise resources within the ART, increasing their self-sufficiency. An analysis of the other teams within PLM evidenced commonalities in mission statements that allowed the distinct teams to define new functionality in a collaborative ecosystem. This move reflected support in literature by Knaster and Leffingwell, Podeswa, and Highsmith.

The network analysis step of the Agile framework also provided a base understanding of The Company's enterprise structure confirmed the presence of additional stakeholders. The stakeholder value mapping stage of the Agile framework identified key personnel on those teams whose input would be valuable in improving the breadth of applicability of The Team's proposed functionality. Because it was not possible to bring knowledgeable engineers from those programs to work full-time in the PLM ART, their input had to be solicited via alternate means. The Agile framework offered here proposed the use of surveys that introduced The Team's basic functionality at a high level to gather expert opinions on their applicability or how they might be improved to widen their usage. These surveys and interviews served to gather, distill, and disposition data for incorporation into The Team's product requirements, as well as provide contacts for conversations about feature development going into the future.

Finally, application of the Digital Twin framework against the gathered data provided an improvement in both the understanding of DT use cases as well as their alignment across product classes. The information gained through the network analysis, stakeholder mapping, and structure provided by the DT framework can be used to inform future functionality as The Team's DT use cases and supporting technology grow in capacity and complexity. DT use case feedback can be shared amongst the experts within the Product Lifecycle Management ART, which is facilitated by The

Team's realignment and reaffirmed by the network analysis conducted here. Equally as importantly, feedback designated for follow-on discussion to the current iteration of the product acted as a catalyst for additional conversations both internally and with the additional stakeholders who provided the feedback. This lent credence to the idea that the value in these feedback loops is in their iterative nature. By conducting extensive network analysis and identifying additional stakeholders, that work can be used as a foundation for subsequent outreach efforts. Repetition of this process will ensure that important configuration change management capabilities that remain in-use by teams across the enterprise on legacy software will be replicated and improved with the next generation of solutions.

This research's applications could be limited across a very large organization such as The Company. For instance, this Agile framework was effective at improving the day-to-day activities of a single team, as well as their horizontal alignment and immediate reporting structure. However, this research did not explore application of these frameworks outside of the Milestone 2/Concept/Visioning stage or when attempting to bridge to other teams in The Company responsible for the later stages. The researcher anticipates that The Team may encounter resistance to providing iterative, Agile requirements and milestones to The Company's software development organization, which is accustomed to a waterfall-style approach to product development. This represents an area for future study, particularly in the area of migrating from Waterfall to Agile systems engineering methods across a large organization.

Moreover, the DT features and the Digital Thread advantages they enable as presented in the DT framework are reliant on the arrival of technology and data at the very nascency of its use within manufacturing. By properly creating the DT model definitions within a computing environment designed to leverage the models and feed the Digital Thread, The Company is laying the foundational digital infrastructure for Industry 4.0 technologies. What remains to be seen is the degree to which the

physical world delivers on providing the data. Further technical work must be done to implement data gathering sensors and tools in the design state, on the factory floor, and in the products' service lives.

Appendix A

Abstracted Version of Feedback Solicitation Surveys and Interview Questions

The Team Introduction:

The Team exists under The Company's Next Generation Engineering unit to generate additional capabilities as The Company incorporates new software into the product lifecycle. The Team exists alongside several other Product Lifecycle Management (PLM) teams, whose goal is to ensure that The Company maintains agility throughout the design and manufacture phases by providing configuration change management services. The complex configuration change management services proposed by The Team would be external to the new software and would represent a significant competitive advantage as compared to relying on the new software's current or proposed native functionality. The Team is responsible for generating requirements and a prototype for complex configuration change management that will work with current systems and programs as well as the next-generation software suite.

What is Configuration Change Management (50,000-foot view)?

The Company allows customization on its products. That customization presents itself in the form of Options to the customer. A product's "configuration" in this case refers to the Options that the customer selects (and the options that they cannot select but are vital in the definition of the product).

From a documentation perspective, everything on a product can be broken down to individual parts, each of which has an engineering drawing and manufacturing instructions. Assemblies of parts become modules/installations, which are put on the product. When a module is generated, its owner defines the Options to which it applies/fulfills. When a design engineer edits a module because the engineering drawing for one of its parts has changed, or when the manufacturing engineer edits a module to modify the manufacturing instructions or control code, the editor defines when in the current line of products being manufactured those changes are going to take place. These two capabilities of a module (changing options and availability) compose the change management.

Configuration change management is a combination of all of the above, including how different configurations (customer Option selections), and engineering and manufacturing changes affect the final configuration of the product and documentation thereof.

There are several key enabling concepts that The Team proposes to include in its configuration change management solution. This survey aims to introduce those concepts and the added complexity that they would facilitate to additional (non-Commercial) teams across The Company enterprise, assess their relevancy, and solicit feedback.

Definition of terms

-Product Definition Object (PDO)

The Product Definition Object is a persistent object unique to each individual instance of a product that contains all of the installations/modules (assemblies of parts) that have ever been associated with that instance of product. This includes all installations that satisfy the customer's selected Options, as well as installations that have been force-included or –excluded by Design or Manufacturing Engineers. Because this object is persistent, it is a complete record of all design and manufacturing changes that a particular instance of a product has undergone during its design and build lifecycle, from customer order until product delivery. Historic iterations of the PDO contain a relationship to installations/modules that are potentially not part of the final product but were associated with the product at some point in its history. The Team's solution proposes a service that creates a PDO for any product, agnostic of design platform, a capability that does not exist in the current design of the new software platform. From the PDO, a report can create the Engineering and Manufacturing Bill of Materials (EBOM and MBOM).

-Update service

The Update service is a nightly software algorithm that ingests input data to update the PDO for all Commercial products. The Update algorithm updates PDOs in certain phases of production by evaluating engineering and manufacturing drawings based on their state, which options they fulfill, and whether it is available for a specific product instance.

The product of the Update algorithm is an identification of which installations are associated to the PDO. The Team proposes a similar service that will run regularly, receiving input from any canonical engineering/manufacturing data source, to create the aforementioned PDO module associations. Additionally, The Team's Update service proposes far greater complexity, including evaluation of a greater variety of module states and the ability for an EBOM installation to map to more than one MBOM installation (for more detailed information about advanced capabilities, please contact the researcher, Jeff Miller).

The surveys and interviews subsequently asked each of the respondents the following set of questions separately for both the PDO and the Update service.

Questions for both of the above capabilities:

How familiar are you with this capability?
Scale 1-5
(1 - not at all familiar, 5 - very familiar)

To what degree does your team already implement this solution?
Scale 1-5
(1 - No implementation, 5 – already implemented exactly as-is)

Without changing this capability as described, to what degree do you think this capability could be implemented in your program's design?
Scale 1-5
(1 – not implementable even with significant changes or input, 5 – implementable without any changes or input)

What changes could be made to this capability to better suit your program's design and configuration management needs?
(Short answer)

Given what you now know about this capability, to what degree would you change your product offering or design strategy in the future if it is offered as-is?
Scale 1-5
(1 – could not change product or design strategy at all, 5 – could adapt product or design strategy to wholly implement the solution as-is)

Given what you now know about this capability, to what degree would you change your product offering or design strategy in the future if it is offered after implementing the suggested changes?
Scale 1-5
(1 – could not change product or design strategy at all, 5 – could adapt product or design strategy to wholly implement the solution with the recommended changes)

THIS PAGE INTENTIONALLY LEFT BLANK

Bibliography

[1] Barbara Rita Barricelli, Elena Casiraghi, and Daniela Fogli. A Survey on Digital Twin: Definitions, Characteristics, Applications, and Design Implications. *IEEE Access*, 7:167653–167671, 2019.

[2] John Borky and Thomas Bradley. *Effective model-based systems engineering.* Springer Science+Business Media, New York, NY, 2018.

[3] Bruce Douglass. *Agile Model-Based Systems Engineering Cookbook.* Packt Publishing, 2021.

[4] James A. Highsmith. *Agile software development ecosystems.* The Agile software development series. Addison-Wesley, Boston, 2002.

[5] Richard Knaster and Dean Leffingwell. *SAFe 4.0 distilled: applying the Scaled Agile Framework for Lean software and systems engineering.* Addison-Wesley, Boston, MA, 2017.

[6] Werner Kritzinger, Matthias Karner, Georg Traar, Jan Henjes, and Wilfried Sihn. Digital Twin in manufacturing: A categorical literature review and classification. *IFAC-PapersOnLine*, 51(11):1016–1022, January 2018.

[7] Matt LeMay. *Implementing and scaling Agile in the enterprise: a goals-first approach.* O'Reilly Media, Inc., 2018.

[8] Kendrik Yan Hong Lim, Pai Zheng, and Chun-Hsien Chen. A state-of-the-art survey of Digital Twin: techniques, engineering product lifecycle management and business innovation perspectives. *Journal of Intelligent Manufacturing*, 31(6):1313–1337, August 2020.

[9] Patrice Micouin. *Model-based systems engineering: fundamentals and methods.* Control, systems and industrial engineering series. iSTE ; Wiley, London, UK : Hoboken, NJ, USA, 2014.

[10] National Aeronautics and Space Administration, editor. *NASA Systems Engineering Handbook.* Verlag nicht ermittlebar, Erscheinungsort nicht ermittelbar, 2007.

[11] Surjya Pal, Debasish Mishra, Arpan Pal, Samik Dutta, Debashish Chakravarty, and Srikanta Pal. *Digital twin – fundamental concepts to applications in advanced manufacturing*. Springer, Cham, Switzerland, 2022.

[12] Howard Podeswa. *The agile guide to business analysis and planning: from strategic plan to detailed requirements*. Addison-Wesley, Boston, first edition, 2021.

[13] Klaus Pohl, Harald Hönninger, Reinhold Achatz, and Manfred Broy, editors. *Model-Based Engineering of Embedded Systems*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[14] Joachim Rossberg. *Agile project management with Azure DevOps: concepts, templates, and metrics*. Apress, New York, NY, 2019.

[15] Navin Sabharwal, Raminder Rathore, and Udita Agrawal. Introduction to Agile and DevOps. In Navin Sabharwal, Raminder Rathore, and Udita Agrawal, editors, *Hands-On Guide to AgileOps: A Guide to Implementing Agile, DevOps, and SRE for Cloud Operations*, pages 29–40. Apress, Berkeley, CA, 2022.

[16] Michael Schluse, Marc Priggemeyer, Linus Atorf, and Juergen Rossmann. Experimentable Digital Twins—Streamlining Simulation-Based Systems Engineering for Industry 4.0. *IEEE Transactions on Industrial Informatics*, 14(4):1722–1731, April 2018.

[17] Mike Shafto, Mike Conroy, Rich Doyle, Ed Glassegen, Chris Kemp, Jacqueline LeMoigne, and Lui Wang. Modeling, Simulation, Information Technology & Processing Roadmap. DRAFT, National Aeronautics and Space Administration, November 2010.

[18] Sumit Singh, Essam Shehab, Nigel Higgins, Kevin Fowler, John A. Erkoyuncu, and Peter Gadd. Towards Information Management Framework for Digital Twin in Aircraft Manufacturing. *Procedia CIRP*, 96:163–168, January 2021.

[19] David D. Walden, Garry J. Roedler, Kevin Forsberg, R. Douglas Hamelin, Thomas M. Shortell, and International Council on Systems Engineering, editors. *Systems engineering handbook: a guide for system life cycle processes and activities*. Wiley, Hoboken, New Jersey, 4th edition edition, 2015.

[20] Timothy D. West and Mark Blackburn. Is Digital Thread/Digital Twin Affordable? A Systemic Assessment of the Cost of DoD's Latest Manhattan Project. *Procedia Computer Science*, 114:47–56, January 2017.