Concept of Use for Service-Based Reference Architecture

Version 1.2.1

ConUse Team

26 August 2022

Version Control

Version Number	Change by	Date	Description
0.1	MITRE	6/16/2020	Draft with sections 1 and 2 completed for review.
0.2	MITRE	7/14/2021	Draft with section 3 completed for review.
0.3	MITRE	8/13/2021	Draft with comment adjudications for section 3.
0.4	MITRE	9/2/2021	Completed first two scenarios in Section 4.
1.0	MITRE	9/30/2021	Incorporated comments on version 0.4.
1.1.0	MITRE	1/31/2022	Draft of version 1.1 released for review by ConUse team.
1.1.1	MITRE	3/30/2022	Incorporated comments on version 1.1.0, as well as changes recommended by technical editor.
1.2.0	MITRE	6/27/2022	Completed scenarios – draft for review by ConUse team.
1.2.1	MITRE	8/26/2022	Incorporated comments from FAA review of v1.2.0

Table of Contents

1 Introduction 1-1
1.1 Purpose 1-2
1.2 Scope 1-2
1.3 Assumptions 1-3
1.4 Organization 1-3
2 Reference Architecture Layers: Overview 2-1
2.1 Mission Layer 2-1
2.2 Platform Layer 2-1
2.3 Computing Resources Layer 2-1
2.4 FAA Infrastructure 2-2
3 Roles 3-1
4 Processes 4-1
4.1 Plan 4-5
4.2 Develop 4-6
4.2.1 Developing Mission Applications and Mission Services 4-6
4.2.2 Develop Platform and Computing Resources 4-8
4.3 Deploy 4-9
4.4 Operate 4-10
5 Scenarios 5-1
5.1 Common Example for Scenarios 5-1
5.2 Common Oversight Assumptions for Scenarios 5-4
5.3 Data and Architecture Governance 5-5
5.3.1 Description 5-5
5.3.2 Roles 5-6
5.3.3 Starting State 5-7
5.3.4 Event Flow 5-7
5.3.4.1 Event Flow: Enterprise Engineering and the Definition of a New Solution 5-8
5.3.4.2 Event Flow: Measuring Effectiveness of Data and Architecture Governance 5-11
5.3.5 End State 5-14
5.3.6 Summary 5-14
5.4 Platform and Compute Layer Evolution and Provisioning 5-14
5.4.1 Description 5-14

5.4.2 Roles 5-15
5.4.3 Starting State 5-17
5.4.4 Event Flow 5-17
5.4.5 End State 5-22
5.4.6 Summary 5-22
5.5 Agile Development 5-23
5.5.1 Description 5-23
5.5.2 Roles 5-24
5.5.2.1 Government versus Contractor Roles 5-24
5.5.3 Starting State 5-26
5.5.4 Event Flow 5-27
5.5.5 End State 5-32
5.5.6 Summary 5-33
5.6 Deployment of New Capability to NAS Operations 5-33
5.6.1 Description 5-33
5.6.2 Roles 5-33
5.6.3 Starting State 5-35
5.6.4 Event Flow 5-36
5.6.5 End State 5-39
5.6.6 Summary 5-39
5.7 Monitoring and Management Responsibilities 5-40
5.7.1 Description/Background 5-40
5.7.2 Roles 5-41
5.7.3 Starting State 5-42
5.7.4 Event Flow 5-43
5.7.4.1 Nominal Event Flow 5-43
5.7.4.2 Off-Nominal Event Flow 5-43
5.7.4.3 Parallel Activities 5-44
5.7.5 End State 5-45
5.7.6 Summary 5-45
5.8 Cybersecurity Operations 5-45
5.8.1 Description 5-45
5.8.2 Assumptions 5-46
5.8.3 Roles 5-46
5.8.4 Starting State 5-47

5.8.5 Event Flow 5-47
5.8.6 End State 5-50
5.8.7 Summary 5-50
5.9 Integration of Cloud Cost Management 5-50
5.9.1 Description/Background 5-50
5.9.2 Roles 5-51
5.9.3 Starting State 5-51
5.9.4 Event Flow 5-52
5.9.4.1 Nominal 5-52
5.9.4.2 Off-Nominal Event Flow 5-52
5.9.5 End State 5-52
5.9.6 Summary 5-53
6 Transition of Roles 6-1
7 References 7-1
Appendix A Agile Processes A-1
Appendix B Acronyms B-1

List of Figures

Figure 1-1. Automation Evolution Vision 1-1
Figure 3-1. Reference Architecture Roles 3-2
Figure 4-1. Process Overview 4-1
Figure 4-2. Enterprise Agile 4-2
Figure 4-3 Reference Architecture High Level Concept of Use 4-4
Figure 4-4. Deploy Rules 4-9
Figure 5-1. Electronic Flight Strip Application 5-2
Figure 5-2. Delay Monitor Application 5-2
Figure 5-3. Flight Information Service Concept 5-3
Figure 5-4. Products Integrated to Provide Solutions 5-4
Figure 5-5. Roles for Data and Architecture 5-6
Figure 5-6. Enterprise Engineering and the Definition of a New Solution 5-8
Figure 5-7 Measuring Effectiveness of Data and Architecture Governance 5-12
Figure 5-8. Example Platform and Compute Roles with Single Point of Contact 5-15
Figure 5-9. Computing and Platform Layer Provisioning Roles and Processes 5-16
Figure 5-10. Event Flow for Platform and Compute Layer Evolution and Provisioning 5-17
Figure 5-11. Requirements Flow for Platform and Compute Resources 5-18
Figure 5-12. Roles and Categories of Authority 5-25
Figure 5-13. Agile Development Event Flow 5-27
Figure 5-14. Program Increment Planning and Sprint Planning Processes 5-28
Figure 5-15. Addressing Safety, Security, and Operational Concerns 5-29
Figure 5-16. Portfolio Solution Planning and Scrum of Scrums Processes 5-32
Figure 5-17. Overview of Deployment Event Flow 5-36
Figure 5-18. Event Flow for Deployment to NAS Staging and Integration Environment 5-37
Figure 5-19. Event Flow for OT&E 5-38
Figure 5-20. Facility Level Support Access Privileges 5-41
Figure 5-21. Enterprise Monitoring and Management Access Privileges 5-42
Figure 5-22. Monitoring and Management Scenario 5-43
Figure 5-23. Abnormal Conditions Event Sequence 5-44
Figure 5-24. Security Event Flow: Initial Detection and Quarantine 5-47
Figure 5-25. Part A of Security Event Flow: Library Patch 5-48
Figure 5-26. Part B of Security Event Flow: Policy Mitigation 5-48
Figure 5-27. Part C of Security Event Flow: Wrap Up 5-49

List of Tables

Table 3-1. AES Reference Architecture Role Descriptions.. 3-2
Table 5-1. Baseline and Planning Architecture 5-9
Table 6-1. AES Reference Architecture Role Transition Description.. 6-1

This page intentionally left blank.

1 Introduction

In Fiscal Year (FY) 2020, the Federal Aviation Administration (FAA) developed an Automation Evolution Strategy (AES) [1] to promote consensus on:

- its characteristics and principles,
- strategic outcomes, and
- work plans.

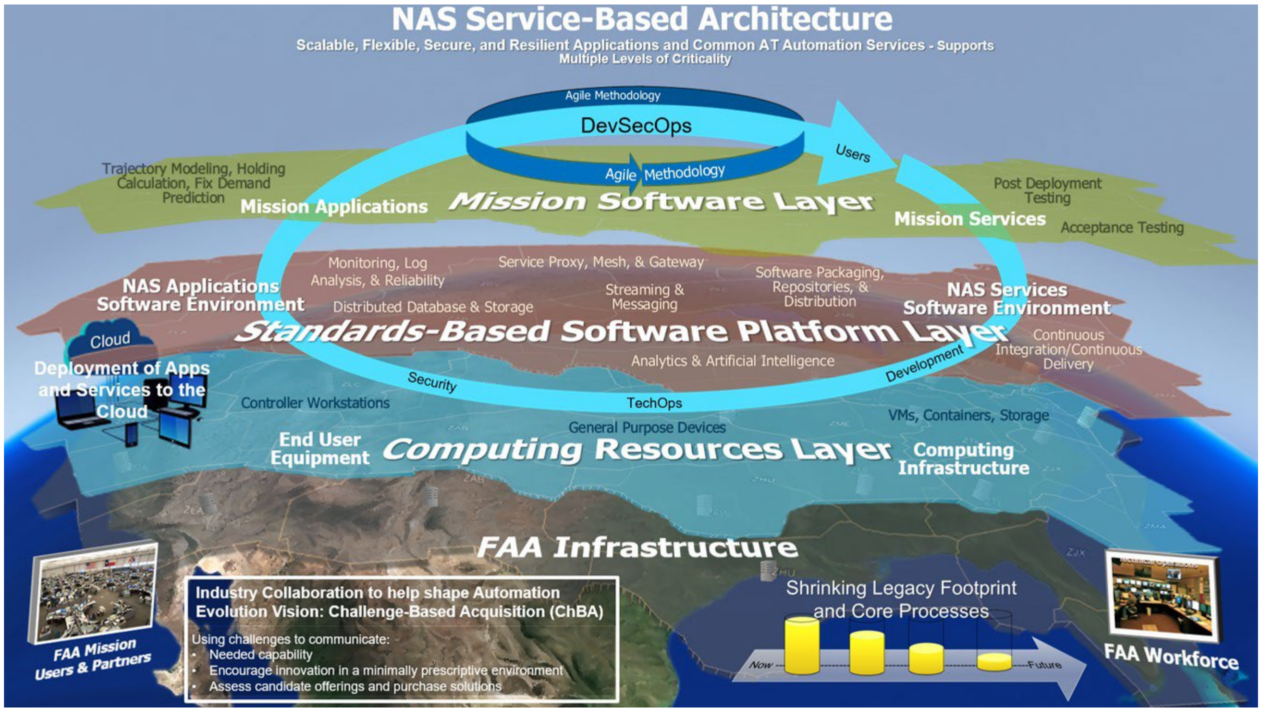

NAS Service-Based Architecture
Scalable, Flexible, Secure, and Resilient Applications and Common AT Automation Services - Supports Multiple Levels of Criticality
Agile Methodology
DevSecOps
Agile Methodology
Users
Trajectory Modeling, Holding Calculation, Fix Demand Prediction
Mission Applications
Mission Software Layer
Mission Services
Post Deployment Testing
Acceptance Testing
NAS Applications Software Environment
Monitoring, Log Analysis, & Reliability
Service Proxy, Mesh, & Gateway
Distributed Database & Storage
Streaming & Messaging
Software Packaging, Repositories, & Distribution
NAS Services Software Environment
Cloud
Standards-Based Software Platform Layer
Continuous Integration/Continuous Delivery
Deployment of Apps and Services to the Cloud
Analytics & Artificial Intelligence
Security
Development
Controller Workstations
TechOps
General Purpose Devices
VMs, Containers, Storage
End User Equipment
Computing Resources Layer
Computing Infrastructure
FAA Infrastructure
Industry Collaboration to help shape Automation Evolution Vision: Challenge-Based Acquisition (ChBA)
Using challenges to communicate:
• Needed capability
• Encourage innovation in a minimally prescriptive environment
• Assess candidate offerings and purchase solutions
Shrinking Legacy Footprint and Core Processes
Now
Future
FAA Mission Users & Partners
FAA Workforce

Figure 1-1. Automation Evolution Vision[1]

Figure 1-1 provides a high-level AES overview. The AES focuses on a set of strategic outcomes including seeking efficiencies to develop and sustain automation systems while also increasing the speed with which the FAA can make new functionality operational. The strategy leverages and applies industry best practices for software development, delivery, and management. It advances the goals by breaking the National Airspace System (NAS) into loosely coupled components that can be developed and sustained independently, hence increasing opportunities for safe incremental improvement, reuse, competition, and parallel development. Agile and DevSecOps[2] methods are key aspects of the AES. Another aspect of the AES is the layered architecture approach that facilitates these methods and provides a robust and secure NAS environment.

To provide the next level of detail on how the FAA will architect the NAS to support the automation evolution vision and achieve the goals, the FAA has developed a Service-Based Reference Architecture for NAS Automation [2], referred to here as the Reference Architecture

[1] The term "Standards-Based" refers to the idea that the Platform Layer will be built on industry standard components that are widely available and used, based on existing commercial-off-the-shelf (COTS) as well as open-source products and tools.

[2] DevSecOps is a methodology that integrates development, security, and operations.

(RA). While the RA focuses on technical aspects of the Service-Based Architecture, this Concept of Use (ConUse) document focuses on the architecture's operational aspects, describing how the FAA, vendors, and other stakeholders will operate and use the RA features.

This ConUse draws on guidance from the following references:

- NAS Automation Evolution Strategy [1],
- Service-Based Reference Architecture for NAS Automation [2],
- Agile Program Management Practices for the FAA Version 1.0 [3],
- Scaled Agile Framework (SAFe) [4], and
- FAA Agile Acquisition Principles and Practices [5]

1.1 Purpose

The purpose of this document is to promote consensus among the FAA and other key stakeholders regarding how the proposed RA as well as the associated Agile and DevSecOps environment are envisioned to operate.[3]

The ConUse describes how the elements of the RA are intended to be used in the context of the NAS and provides a basis for follow-on transition planning by helping to identify future changes to the existing agency capabilities, processes, and policies.

The RA describes the target technical state for the AES. It will serve as a reference that can be used to align evolution initiatives with the overarching technical and operational objectives of the strategy. The RA focuses on exposing and maximizing the benefits of containerized, reusable software services and technology through layered components. It will serve as an important tool to inform the NAS Enterprise Architecture, which describes the NAS evolution through roadmaps and models.

In addition, the AES and the RA will enable future operational capabilities that are not feasible or affordable in current designs (e.g., dynamic airspace in support of efficient use of operational resources, resiliency and contingency operations, and allocation of airspace to Extensible Traffic Management (xTM) service providers, as well as Trajectory Based Operations (TBO) such as gate to gate management of flight schedule and optimized decision support to maintain or minimize schedule changes). [6]

1.2 Scope

The ConUse document includes a high-level description of the roles and the processes followed, as well as scenarios to describe the roles and processes in more detail. This version includes the following scenarios:

1. Data and Architecture Governance,
2. Platform and Compute Layer Evolution and Provisioning,
3. Agile Development,

[3] The ConUse informs the RA, and vice versa; both documents are expected to serve as ongoing references throughout the AES development process. Therefore, this ConUse will be revised as changes are made to the RA and as we develop improved understanding of roles, processes, and scenarios.

4. Deployment of New Capability to NAS Operations,
5. Monitoring and Management Responsibilities
6. Cybersecurity Operations, and
7. Integration of Cloud Cost Management.

The primary target audience for this ConUse includes the following stakeholders:

- FAA leadership,
- Development teams,
- FAA programs,
- FAA security and operations groups,
- FAA enterprise engineering,
- FAA planning and budget, and
- FAA end users.

This document provides a concept of use for the future target architecture described in the RA, with potential new roles and responsibilities and new processes. Section 6 begins to identify how the existing FAA organizations may map into these new roles and responsibilities.

1.3 Assumptions

The ConUse makes the following assumptions:

- Appropriate development methodologies will be applied to software development lifecycles based on unique needs. These include agile, incremental, waterfall, spiral, or hybrid development approaches as appropriate to each service provided. This document assumes an Agile methodology is preferred.
- The FAA will make changes as needed across organizational roles and responsibilities, policies, associated orders, governance, and workforce skills based on a Service-Based Architecture and associated methodologies (e.g., Agile and DevSecOps).
- Technologies (e.g., cloud computing) to provide computing resources and platform layer(s) for NAS operations will be approved and available.
- Acquisition processes and contracting artifacts will be tailored to enable the implementation of Agile and DevSecOps methodologies.
- The FAA will govern the data and architecture for the three service layers (Mission, Platform, and Computing) including but not limited to interfaces, standards, processes, best practices, policies and principles, data rights, security and privacy management, and change management.

1.4 Organization

This ConUse is organized as follows:

- Section 2 provides a high-level concept overview for each of the RA layers.

- Section 3 describes the roles involved in governing, implementing, operating, securing, and evolving the RA.
- Section 4 describes the activities performed by the roles identified in Section 3.
- Section 5 provides scenarios that elaborate on how the actors described in Section 3 will engage in the processes described in Section 4.
- Section 6 maps existing FAA organizations and roles to the new roles identified in Section 3.

2 Reference Architecture Layers: Overview

A key aspect of the AES is the organization of the RA into layers, as illustrated in Figure 1-1. This section provides a brief overview for each layer as these will be referenced throughout this document.

2.1 Mission Layer

The Mission Layer provides the unique functionality needed to support the NAS's mission. The RA is service based, which means software components in the Mission Layer interact by exposing and accessing Mission Services.[4] The FAA will provide governance and architectural oversight that organizes the Mission Services into a taxonomy of Service Groups. The FAA will provide enterprise level architectural oversight and guidance to define which services each Service Group makes available to all other mission services and applications via the Platform Layer.

This ConUse describes how the Mission Layer's functionality will be planned, developed, deployed, and operated by various teams. The scenarios in Section 5 describe various aspects in the lifecycle of two Mission Layer applications (an electronic flight strip application and a delay monitoring application) and a single Mission Layer service (a flight information service).

2.2 Platform Layer

The Platform Layer provides the containerized (or similar) environments that allow Mission Layer software to run on the Computing Resources Layer resources. Examples of elements in the Platform Layer may include a service mesh with sidecar containers that provide security (Zero Trust Architecture[5]) for all communications via application programming interfaces (APIs) for access to required services. (See the RA for more details on the Platform Layer.)

Governance will be applied to ensure that programs developing NAS functionality use the Platform Layer, rather than individual program investments in platform capabilities. However, as different teams are likely to have different needs and preferences and as Platform Layer software evolves, the organization responsible for the Platform Layer will need to provide a broad and evolving (but not unconstrained) range of Platform Layer options to accommodate Mission Layer development team needs.

2.3 Computing Resources Layer

The Computing Resources Layer provides the necessary computing, storage, networking, and other resources to satisfy the requirements for all the functionality in the Mission Software and Platform layers. Those requirements include security, reliability, maintainability and availability, latency, threaded latency, and operating environments. The Computing Resources Layer supports live operations, development, test, and training environments for different levels of criticality. The elements that make up this layer can be provided using a range of different

[4] A Mission Service is a software component that provides the FAA with mission-specific data and computation functions. A Common Mission Service is a Mission Service that is intended for multiple consumers.

[5] Zero Trust Architecture (ZTA) is an approach to information security that improves upon perimeter-based security by requiring all resource requests to be authenticated and authorized on a per session basis regardless of their position with respect to enterprise infrastructure. As a result, breaches are more difficult to propagate, defensive operations are more effective, and granting access can be more flexible.

implementation options, including on premises equipment and cloud computing services. Differences between those options are hidden from the Mission Layer by a generic Platform Layer, described above.

The FAA will exercise governance to ensure that the Computing Resources Layer meets the needs of programs from an enterprise perspective, and that those programs use this layer, rather than making individual program investments in computing resources.

Different organizational models are possible, however, the scenarios in this ConUse describe a single organization that has overall responsibility for the Platform and Computing Resources Layers. The scenarios illustrate how enterprise tooling, services, and compute environments will be selected, configured, and provisioned to meet the needs of Mission Layer development teams.

2.4 FAA Infrastructure

This layer includes specialized equipment and physical infrastructure components that are included in the NAS, but do not fit clearly into the layers described above. Examples include radars, radio equipment, buildings, landing systems, runway lights, and so on. Because of the specialized nature of these items, they are largely not addressed in the RA and in this ConUse.

3 Roles

Figure 3-1 illustrates roles related to developing, sustaining, and using the RA elements. These roles are grouped into the following categories:

- Planning and Budget,
- Enterprise Engineering,
- Enterprise Security and Operations,
- Customer Organization,
- Solution Teams, and
- Product Teams.

The RA requires a variety of government and contractor[6] roles to govern, implement, operate, secure, and evolve the services that fulfill the FAA mission in a safe, and cost-effective manner. Some of these roles may map to existing FAA roles, however, others do not. One of the goals of the RA is to facilitate modern methodologies like Agile and DevSecOps. Therefore, rather than using existing FAA organization names or titles for naming and describing roles, this ConUse adopts terminology from the Scaled Agile Framework (SAFe).[7] This does not imply that the FAA should adopt SAFe or fully map the SAFe framework, with all its complexity, to FAA roles and processes. The SAFe terminology is adopted here only to provide standard terminology for the concepts described in this document. Section 6 begins to address how the existing FAA roles map into the roles described here.

[6] More detailed consideration of government roles versus contractor roles are addressed in the Agile Development scenario, below.

[7] The Scaled Agile Framework, described at https://www.scaledagileframework.com/, has multiple different models, at different levels of complexity. We are using elements of the "Full" SAFe model, but by no means the entire model. It remains to be seen whether the FAA decides to adopt some or all of SAFe or chooses to adopt a different model for development at enterprise scale. Other possible models for scaled include Scrum of Scrums, Scrum@Scale, Large Scale Scrum, Nexus, Disciplined Agile, and Enterprise Kanban (aka Portfolio Kanban). The Rapid Design and Development (RDD) approach, used in the FAA's Project Elroy, could also be expanded and adopted at a larger scale.

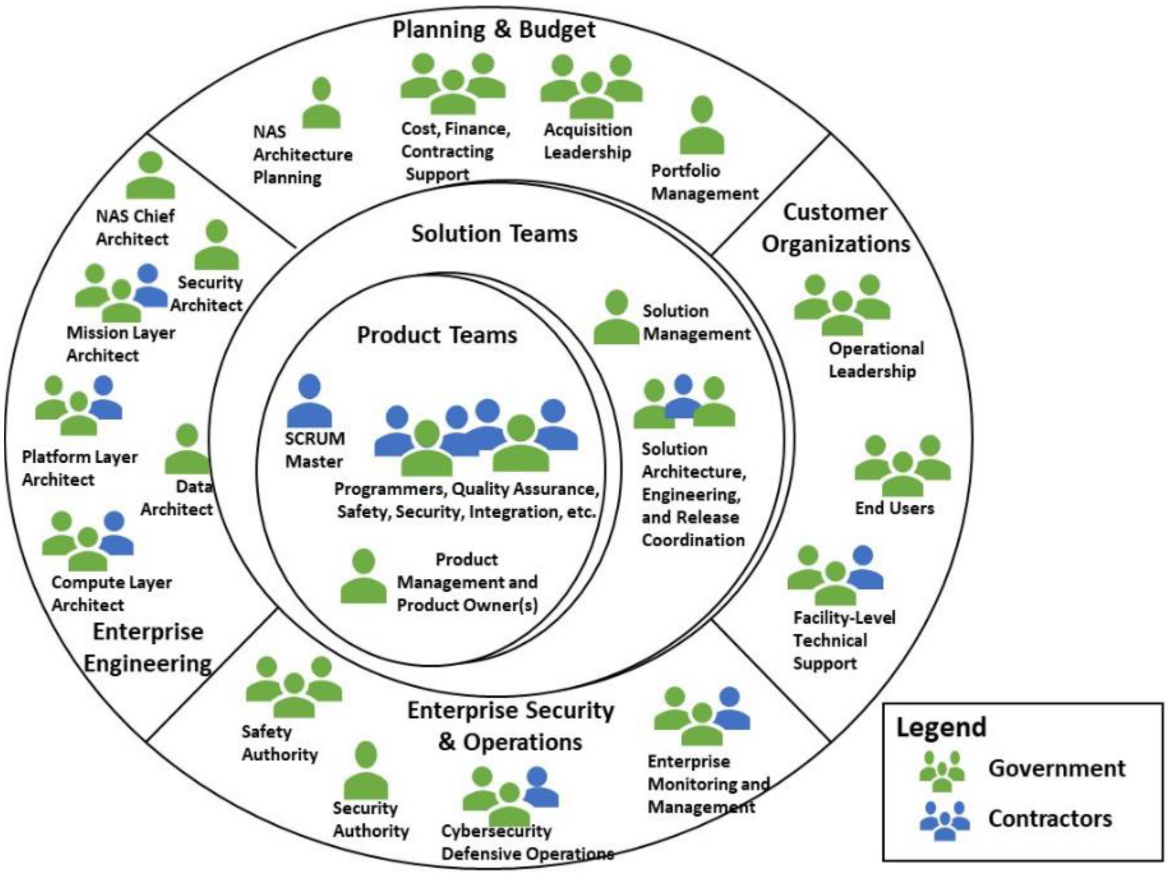


Figure 3-1. Reference Architecture Roles

Table 3-1 lists the roles within the six categories previously mentioned. This is not intended to be a comprehensive account of all the roles, but to establish a baseline to build upon as needed. Roles are collaborative. Open communications and coordination channels are critical to achieve the desired AES outcomes.

Table 3-1. AES Reference Architecture Role Descriptions

Category	Role	Description
Planning & Budget This category includes roles related to high-level agency vision, investment planning, and financial oversight.	Cost, Finance and Contracting Support	Roles involved in the overall management of the solicitation, award, and execution of Agile development contract(s).
	Acquisition Leadership	The agency's investment authority that makes the financial decision on new NAS capabilities investments, technical refreshes, and/or end-of-system lifecycles.
	NAS Architecture Planning	Maintains the NAS Enterprise Architecture, which documents the agency roadmaps, major investment plans, and overall NAS architecture, primarily for planning and management purposes.
	Portfolio Management	Responsible for aligning agency strategic goals and operational priorities with execution for a specific business domain in the Enterprise. Responsible for governance, compliance, and return on investment for a collection of solutions. Creates and maintains the portfolio vision and roadmap in coordination with portfolio stakeholders.

Category	Role	Description
Enterprise Engineering This category includes systems engineering roles that define and oversee the NAS architecture.	NAS Chief Architect	Provides overall NAS-wide architecture design, oversight, and technical guidance. Provides comprehensive overarching view of all layers and across all applications and services. Maintains the Automation Evolution Reference Architecture.
	Mission Layer Architect	Ensures that the right set of Mission Layer services and applications are in place and provides overarching NAS-wide design guidance for the mission layer to ensure interoperability and efficiency. Engineers the performance profiling of services and coordinates with teams to identify bottlenecks and inefficient activities. Looks for opportunities to refactor processes to facilitate streamlining and automation. Has knowledge of the business, collaborates with the community (internal/external), identifies candidate mission/common services, and specifies requirements, including rules. Ensures that the Solution Teams understand and implement the architectural vision.
	Platform Layer Architect	Oversees the selection and evolution of the NAS-wide suite of tools and software components that make up the Platform Layer. Ensures that the needs of Mission Layer developers are met. Certifies and maintains the security of standard container images, ensures availability, and maintains documentation.
	Compute Layer Architect	Engineers fundamental compute, network, and storage resources for Platform and Mission engineering on-demand, NAS-wide. Compute Architecture Engineering enables Mission and Platform users to scale and shrink resources on an as-needed basis, reducing the need for high, up-front capital expenditures or unnecessary on-premises infrastructure.
	Data Architect	Ensure all data across the enterprise is accessible and accurate. The data architect translates business requirements into technology requirements and defines data standards, i.e., the rules that define how data is described and recorded. To meet this responsibility, the data architect specifies a data management framework for reviewing, specifying, refining, acquiring, archiving, and purging data and the associated schemas, and ensures that appropriate coordination occurs among information stakeholders.
	Security Architect	Works with other architects to design the security controls for each layer and their subsystems. The security architect ensures that the operation of the security executed by the Security Authority will be effective by mandating that the measures necessary for effective authentication, authorization and policy enforcement be in the architecture and implementation of the Reference Architecture. This role also develops and maintains a security risk management plan.
Enterprise Security Engineering & Operations This category provisions and operates platform and compute layer services and monitors and manages mission layer software enterprise wide. This category is also responsible	Safety Authority	Ensures that any software deployed into the NAS production environment has been properly assessed for safety and that all safety risks have been adequately mitigated. Ensures that safety processes and standards are integrated into the Architecture including the DevSecOps chain to ensure the safety of services developed in software factories deployed on the Platform Layer.
	Security Authority	Accountable for NAS Enterprise Information Security Governance. Sets or delegates policy for security including Zero Trust, defensive capabilities, compliance, audit, training, and enforcement. Responsible for crafting and implementing processes to ensure policy is followed. Responsible for reviewing and updating policy as conditions, capabilities, and technology evolve. Responsible for working with other roles to understand and make tradeoffs with security policy as is best for the overall FAA mission.

Category	Role	Description
for security and safety oversight and operations.	Cybersecurity Defensive Operations	Actively defends the NAS against cybersecurity threats. This includes operations centers that monitor NAS software and networks respond to incidents. Investigates potential attacks, data breaches and system compromises.
	Enterprise Monitoring and Management	Provisions, monitors, manages, and operates common mission and platform services that are used by multiple NAS stakeholders. This role ensures that quality of service objectives is met in operation and responds appropriately to changes in service or application status.
	Cloud Service Provider Cybersecurity Defensive Operations	Actively defends the vendor Cloud against cybersecurity threats. This includes operations centers that monitor networks, respond to incidents, investigate potential attacks, data breaches and system compromises, and coordinate with FAA Cybersecurity Defensive Operations when needed.
Customer Organizations This category includes the end user organizations that are responsible for providing NAS services (separation services, flow management services, etc.) to airspace users.	Operational Leadership	This role is comprised of FAA managers responsible for different areas of NAS mission operations (e.g., En Route and Terminal air traffic control (ATC), system operations, aeronautical information) as well as technical operations at facilities.
	End Users (FAA and external entities)	This role is comprised of controllers, traffic managers, and coordinators. They conduct operations using Mission Application front ends running on workstations at the facility level as part of the Computing Resources Layer (e.g., workstations and tablets and if not safety-critical, possibly implemented as web applications running in a browser) to access data and computation services needed to perform their job functions. External entities may also be end users of FAA-provided services, for example air carriers might be end users of flight planning and filing services; Department of Defense (DoD) and Department of Homeland Security (DHS) might be end users of a flight data service.
	Facility Level Technical Support	This role includes staff that maintain NAS facilities, systems, and equipment at Air Route Traffic Control Centers (ARTCCs), terminal area traffic control facilities, flight service stations, regional centers, and other FAA facilities supporting operations. They provide support for the daily operation of facilities and offices under the jurisdiction of the FAA's Air Traffic Organization (ATO). Facility Level Technical Support provides technical support to End Users at the facilities who may experience problems or issues with using mission layer software. They coordinate and collaborate with Enterprise Monitoring and Management as well as Product Teams, when necessary, for technical support that is within the purview of these other roles. They are also responsible for hardware at the compute layer in the field, including generic IT equipment (e.g., printers, workstations, local area networks, phones) as well as FAA Infrastructure equipment that is unique to the NAS (e.g., radios, radars, other sensors).
Solution Teams This category includes the roles that are responsible for	Solution Management	Solution management has overall responsibility for the creation and sustainment of a solution that meets customer needs. A solution is provided by an integrated set of products, which may include mission applications and services, as well as platform and computing resource enablers.

Category	Role	Description
developing and sustaining solutions that integrate multiple products to meet FAA needs.	Solution Architecture, Engineering, and Release Coordination	Solution team members support solution management, providing technical and architectural vision that spans a set of mission applications and services, as well as platform layer and computing layer technical enablers that make up a solution. This team ensures that these components are built to be interoperable and coordinates the release trains of multiple Product Teams to synchronize development timelines. The solution team has overall responsibility for deploying solution increments (capabilities) into the production environment for use by Customer Organizations. This includes ensuring that quality control, security, and safety assurance processes have been followed and that development, integration, and operational testing and evaluation have been successfully completed. This team is also responsible for coordinating with Enterprise Security and Operations and Customer Organizations on issues such as support, training, and deployment schedule.
Product Teams A Product Team consists of one or more Agile Teams responsible for developing and sustaining a product, following Agile principles.	Product Management and Product Owner	The Product Owner is an Agile Team member with primary responsibility for defining and prioritizing the backlog of work and organizing the development process into time-boxed iterations that lead to a release of a capability. The Product Owner has the business and operational knowledge to represent both internal and external user and stakeholder Customer Organizations. For large software development efforts, there may be multiple Product Owners, each responsible for one portion of the overall product, with overall coordination provided by a Product Management team.
	Scrum Master	Facilitates the processes, enforces the team's rules, and keeps the team focused on tasks.
	Other Agile Team roles	The team comprises multiple roles, including software developers, software and security engineers, data specialists, testers, quality assurance, release train engineers, and configuration managers. Typically, this would be staffed primarily by contractors, but would also include FAA personnel.

4 Processes

This section relates the RA roles as described in Section 3 to the Agile processes and activities involved in developing and implementing the NAS RA. It provides an overview of how the roles interact and the sequence of steps they perform.

Processes are described in four phases: Plan, Develop, Deploy and Operate.[8] An overview of those phases is provided here, followed by a subsection describing each phase. Throughout this document, italic font is used whenever a role or category of roles identified in Section 3 is mentioned, for example *Enterprise Engineering* or *End User*.

Figure 4-1 provides an overview of the Agile processes and activities.

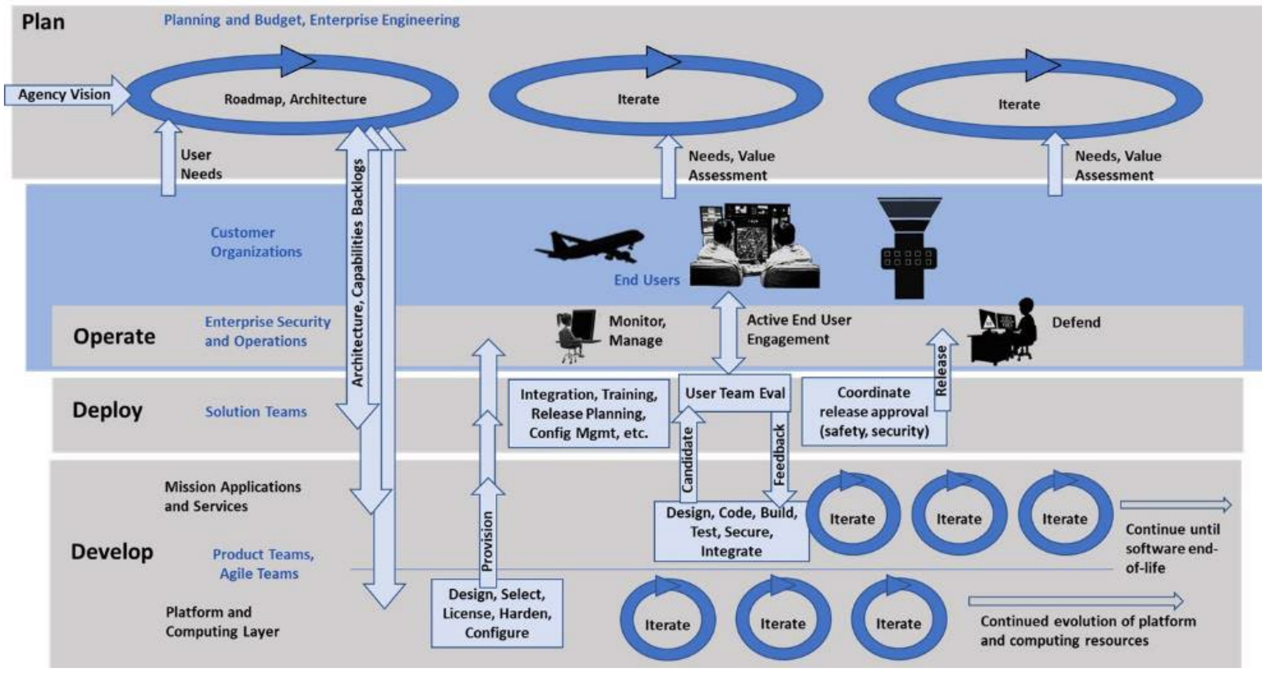

Plan
Planning and Budget, Enterprise Engineering
Agency Vision
Roadmap, Architecture
Iterate
Iterate
User Needs
Architecture, Capabilities Backlogs
Needs, Value Assessment
Needs, Value Assessment
Customer Organizations
End Users
Operate
Enterprise Security and Operations
Monitor, Manage
Active End User Engagement
Defend
Deploy
Solution Teams
Integration, Training, Release Planning, Config Mgmt, etc.
User Team Eval
Coordinate release approval (safety, security)
Release
Candidate
Feedback
Provision
Develop
Mission Applications and Services
Product Teams, Agile Teams
Platform and Computing Layer
Design, Code, Build, Test, Secure, Integrate
Iterate
Iterate
Iterate
Continue until software end-of-life
Design, Select, License, Harden, Configure
Iterate
Iterate
Iterate
Continued evolution of platform and computing resources

Figure 4-1. Process Overview

The Plan phase includes activities by which the FAA determines what capabilities are needed, allocates budget, and determines high level strategy. This includes responding to, and perhaps helping to define, the FAA's high-level vision, roadmap, technical policy, and responding to needs defined by the *End Users*. This phase also includes assessing products being sustained in the field to determine if they are continuing to provide value. These planning activities result in the definition and funding of value streams to be created, sustained, and evolved in the Develop phase.

The Develop phase leverages *Product Team* and *Agile Team* roles and addresses development of mission, platform, and compute layers, with architectural and technical guidance provided by *Enterprise Engineering* roles.

[8] The different phases will be happening in parallel, while a given version of a product is in operation, the next version is being prepared for deployment, while the next version beyond that is being developed, and future versions are being planned. The figure is arranged with the users in the middle to emphasize the importance of active user engagement throughout all stages.

The Deploy phase includes verifying and validating new versions of software products, ensuring that they meet user needs and do not create safety or security risks to the NAS, and coordinating deployment into the production environment at an operationally appropriate cadence.

The Operate phase includes monitoring of products in operational use, detecting any problems with proper functioning and performance, and responding accordingly. Detecting and responding to cybersecurity threats is also included in this phase.

Each of these phases includes iterative processes to converge on target goals. *End User* engagement yields inputs on assessed value and defined needs that can spur another iteration cycle of Plan, Develop, Deploy, and Operate.

The vertical arrows on Figure 4-1 indicate in general terms how the results of one activity rely on, or provide elements needed by, another activity. The Data and Architecture Governance scenario in section 5.3 describes in more detail the engineering artifacts and processes that support a requirements and acquisition flow.

Enterprise Agile Concepts

Agile processes span all activities, not just the activities of individual *Product Teams*. The RA will support a more rapid development approach, with frequent delivery of incremental updates. Each capability set will first be implemented in the form of a Minimum Viable Product (MVP) or Minimum Viable Capability Release (MVCR) that may not include all the desired features but will allow early user evaluation, providing feedback that can inform future iterations. That approach will allow the FAA to field new capabilities faster than they are today, reducing the likelihood that operational needs will change while a capability is in development.

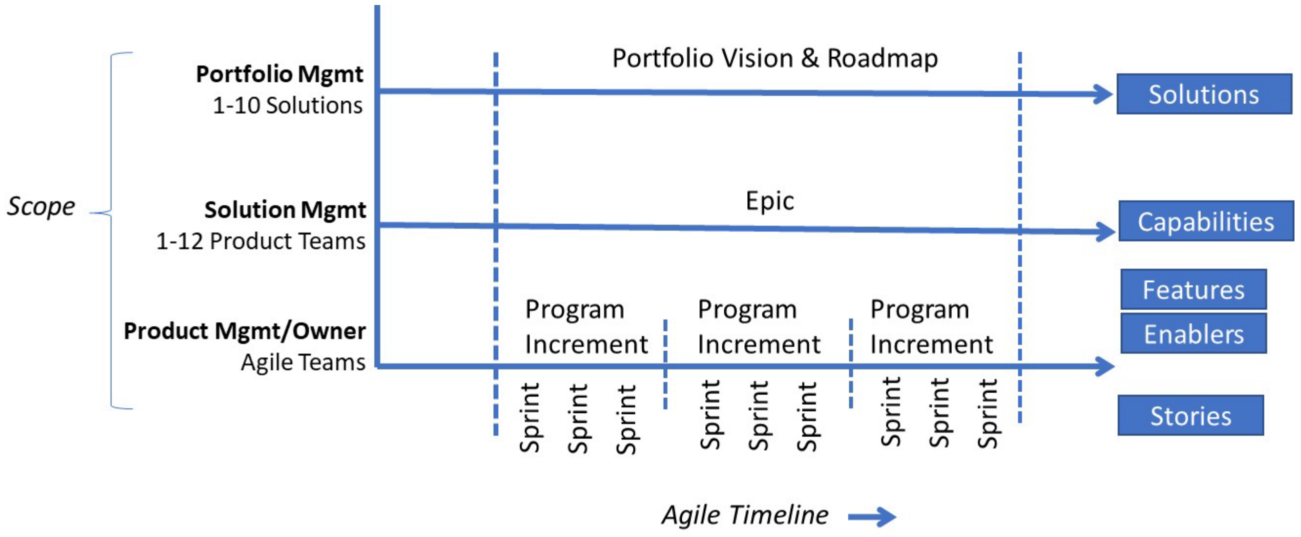


Figure 4-2. Enterprise Agile

As illustrated in Figure 4-2, Agile processes are used at multiple development scopes: Portfolio, Solution, and Product. Appendix A provides more details on these processes. Some of the key terms and concepts are:

- Portfolio Management: At the highest level, the FAA manages, prioritizes, and invests in Portfolios of Solutions to meet the agency's needs.
- Solutions: Each Solution provides a set of Capabilities that improve the FAA's ability to perform its mission.
- Capabilities: Those Capabilities are created by integrating and using a set of Products.
- Products: Products are software components (applications and services) which are developed and evolved incrementally.
- Sprints and User Stories: Agile development teams conduct Sprints to implement User Stories.
- Program Increment: Multiple development team Sprint cycles constitute a Program Increment, which provides a set of Features and Enablers.
- Features and Enablers: The functionality of a software product is defined in terms of a set of Features and Enablers. Loosely speaking, Features provide functionality that is visible to *End Users*, while Enablers provide underlying technical functionality that may not be visible to users but is necessary to enable the Features.
- Epics: Multiple Program Increments comprise an Epic, which provide the capabilities that make up a Solution.

Product Team Concepts

Another overview of the roles and processes, with an emphasis on *Product Teams*, is provided in Figure 4-3. The figure shows multiple *Product Teams* developing new mission applications or mission services. One or more *Product Teams* working on the Platform Layer and Compute Layer provide the necessary software factory tools, computing resources, and other components that make up the information technology (IT) infrastructure. This IT infrastructure is used by other *Product Teams* working on Mission Layer applications and services.

Development, test, staging and integration, and production environments are provisioned as needed. Each team develops their products in an iterative manner, as described above. Data (made available via APIs and/or messaging services mediated by the platform layer) is shared as needed for development, testing, integration, and operational evaluations. The data is also shared among applications and services in the operational environment. *Facility Level Technical Support* staff provides first-level field support at NAS facilities where operations are being conducted (e.g., ARTCCs, TRACONs) or at regional support facilities. *Enterprise Monitoring and Management* provides first and second level enterprise level support, with second and third level engineering support also provided by *Product Teams* staff. *Cybersecurity Defensive Operations* staff provide, across all environments, active cybersecurity monitoring and defense.

For large and complex software development efforts, a *Product Team* may consist of multiple *Agile Teams* coordinated by a single *Product Management* team that includes roles such as *Product Manager*, *Release Train Engineer*, and *System Architect*. This level of detail is not shown in Figure 4-3, and throughout this document the term *Product Team* is often used without distinguishing whether it refers to a single *Agile Team* with a one *Product Owner*, or multiple *Agile Teams*, each with a *Product Owner*, coordinated by a *Product Management* team.

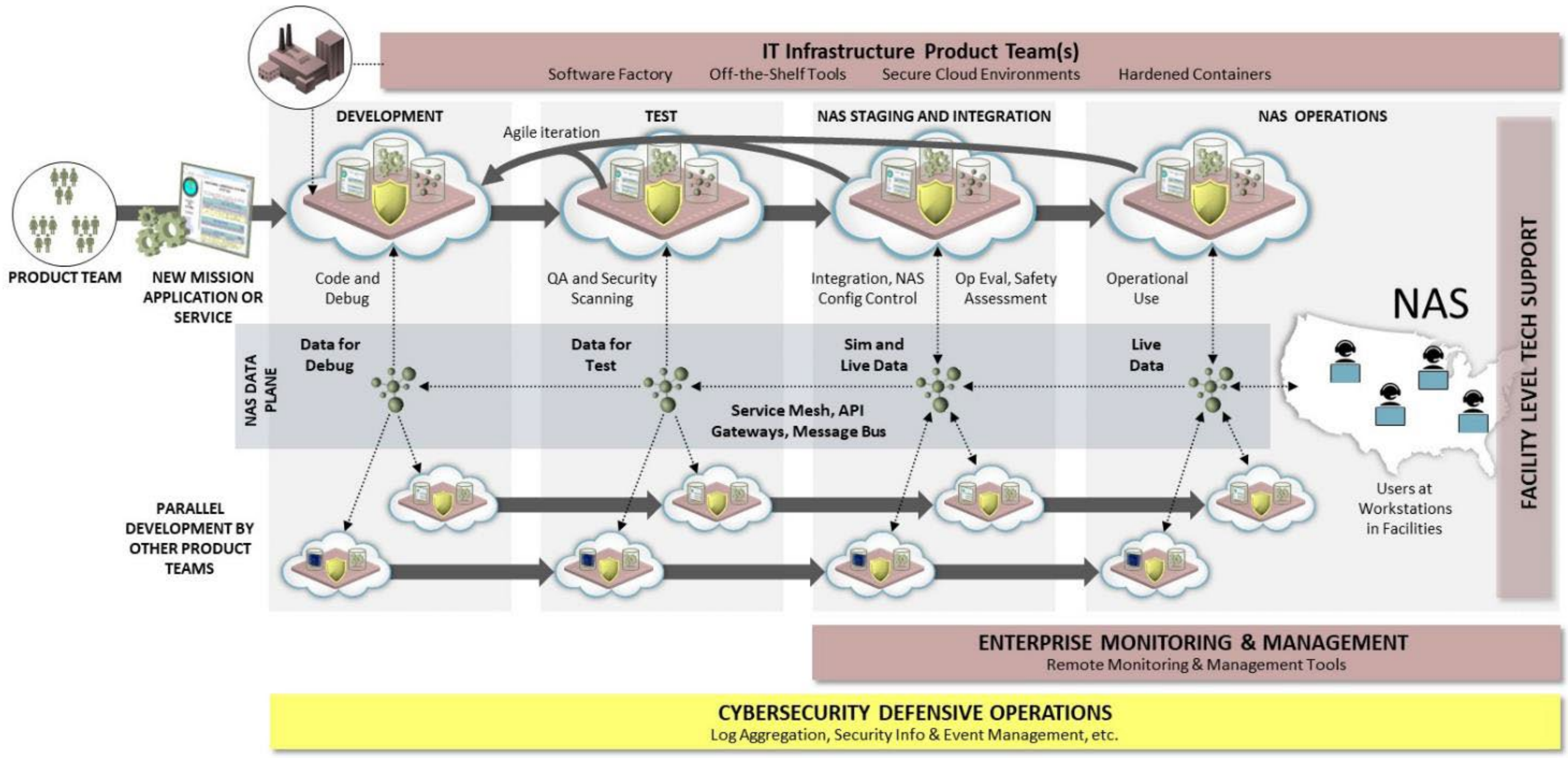

IT Infrastructure Product Team(s)
Software Factory
Off-the-Shelf Tools
Secure Cloud Environments
Hardened Containers
DEVELOPMENT
TEST
NAS STAGING AND INTEGRATION
NAS OPERATIONS
Agile iteration
PRODUCT TEAM
NEW MISSION APPLICATION OR SERVICE
Code and Debug
QA and Security Scanning
Integration, NAS Config Control
Op Eval, Safety Assessment
Operational Use
NAS
NAS DATA PLANE
Data for Debug
Data for Test
Sim and Live Data
Live Data
Service Mesh, API Gateways, Message Bus
PARALLEL DEVELOPMENT BY OTHER PRODUCT TEAMS
Users at Workstations in Facilities
FACILITY LEVEL TECH SUPPORT
ENTERPRISE MONITORING & MANAGEMENT
Remote Monitoring & Management Tools
CYBERSECURITY DEFENSIVE OPERATIONS
Log Aggregation, Security Info & Event Management, etc.

Figure 4-3 Reference Architecture High Level Concept of Use

4.1 Plan

Planning encompasses activities that take in user and business needs and user requests/requirements, prioritizes them, and initiates DevSecOps projects. Planning activities occur at the enterprise level as well as the solution and product level.

At the enterprise level *Acquisition Leaders*, *NAS Architecture Planning*, and *Operational Leadership* organize portfolios to deliver solutions the enterprise needs to accomplish its business and operational goals. Cross organizational planning takes place to review, integrate, and prioritize resource allocation to portfolios. *Portfolio Managers* work with *Acquisition Leaders*, *Operational Leadership*, and *NAS Architecture Planning* to determine near-term priorities and the solutions to fund within the portfolio.

Portfolio Managers create and maintain the Portfolio Vision to describe the future state of the Portfolio's solutions and express how they will cooperate to achieve the portfolio's objectives, the agency strategic goals, and operational priorities. *Portfolio Managers* govern the solutions and can make the decision to incorporate changes (e.g., user requests/requirements) if they meet the vision and are within the bounds of the solution objectives. If the impact of a change exceeds the authority of the *Portfolio Manager*, the decision to accept the change must gain concurrence from the *Acquisition Leadership*, *Operational Leadership* and *NAS Architecture Planning.*

The *NAS Chief Architect* creates and maintains a technology strategy and roadmap for current and future business capabilities that enable solution development within portfolios. The Architects within *Enterprise Engineering* work with the *NAS Chief Architect* to establish the technical guidance used to develop, deploy, and operate mission applications and services. They drive engineering and reuse and foster adoption of the shared technical vision. The roles in *Enterprise Engineering* will be accountable for the engineering work and tooling used to maintain and keep the FAA's architectures traceable and in sync. Architects from the Enterprise Engineering org/group will be assigned to *Planning & Budget*, *Solution & Product Teams*, *Enterprise Security & Ops*, and *Customer Organizations*. *Enterprise Engineering* architect activities include evaluation of the current architecture and design to identify technical opportunities and needs. These architects would do the work of developing a high-level architecture and designs to be used by *Solution* and *Product Teams*, applying governance and change management practices, and then synchronizing specific solution architectures with the high-level enterprise architecture. Solution Teams and Products Team may provide feedback to Enterprise Engineering on the architecture for consideration in planning.

Portfolio Managers, *Solution Management*, and *End Users* collaborate to prioritize capabilities to meet portfolio solution goals and performance targets. They work with *Cost, Finance, and Contract Support* to allocate budgets for capabilities within their portfolio. The *Portfolio Manager* creates and maintains the portfolio roadmap.

Solution Management has overall responsibility for the creation and sustainment of a solution, within the portfolio. *Solution Management* is supported by a team that provides *Solution Architecture, Engineering, and Release Coordination*. This team helps cultivate and apply use of Agile practices across the portfolio, communicate the shared technical and architectural vision, and evaluate and guide execution of portfolio capabilities. Each capability may include multiple products.

At the *Product Team* level, for a large or complex product, there may be a *Product Management* role and multiple *Agile Teams*, with a *Product Owner* for each *Agile Team.* For a relatively small simple product, there may just be one *Agile Team* with one *Product Owner*. In either case, *Product Management* and/or *Product Owner(s)* lead the efforts of one or more development teams focused on the Product. *Product Management* and/or *Product Owner(s)* coordinate with the *Solution Managers* to define the product roadmap, acquisition strategy (includes acquisition approach, risk management, business strategy, contract strategy, IP strategy), and cost estimation (includes user engagement and training costs). With *Portfolio Management* approval, *Solution Management* allocates resources for the *Product Team* to begin development.

Product Management and/or *Product Owner(s)* prioritize the product features and enablers and execute development to provide the capabilities needed by the *Solution Team*. These features and enablers define high level requirements. As these features and enablers are designed and implemented, the *Product Management* and/or *Product Owner(s)* are responsible for ensuring that the *End User*s needs are accurately reflected in the form of User Stories. These User Stories define the backlog of detailed requirements to be implemented in each sprint. To allow *Product Teams* to be Agile, if changes are requested to existing features and enablers, this team is empowered to make decisions to accept or reject the requests within the scope of their authority. *Product Teams* have the best visibility into the backlog and the greatest understanding of a change's impact. If a change's impact exceeds the authority of this team, the decision to accept the change must gain approval from Solution Management. As the features and enablers mature, the *Product Management* and/or *Product Owner(s)* are responsible for periodically updating planning and reporting documentation (e.g., product roadmap, cost estimation, strategies). The *Product Owner/Product Manager* and *Solution Management* initiate the decision to retire a feature or enabler based on key metrics and user value assessments. A feature or enabler is retired with the approval of the *Portfolio Manager* and *NAS Chief Architect*.

Solution Teams and Products Team may provide feedback to Enterprise Engineering on the architecture for consideration in planning.

4.2 Develop

This activity includes development of specific Mission Layer applications and services, as well as the enabling Platform Layer and Computing Layer resources. These activities are described in the subsections below.

Entrance criteria for both these development activities is similar. Funding must have been procured and initial planning must have been done to the point where high-level design has been done and appropriate *Solution Teams* and *Product Teams* have been staffed.

Development activities encompass more than just the development of software. For example, the same processes are appropriate for developing end user documentation, training materials, deployment plans, and adaptation.

4.2.1 Developing Mission Applications and Mission Services

This phase includes activities necessary to create and evolve Mission Layer functionality. At a minimum, it should include:

- Plan specific functionality to be included in the next iteration of a product (as described above).
- Perform detailed design that conforms with enterprise-level architecture and design created by the *Enterprise Engineering* role (as described above) as needed to accomplish

the next iteration of the product. In particular, the *Data Architects* design/engineer the data plane to identify what data needs to be stored, distributed, archived, etc. The data plane manages service invocations and event driven information exchanges using technologies such as API gateways and proxies, message buses, and service meshes.

- Decide how to best leverage the platform layer to best accomplish the work using architectural guidance from enterprise engineering.
- Develop the necessary Mission Layer software.
- Mitigate any safety, security, or operation concerns with the work. Those concerns should be brought up by the *Safety Authority*, *Security Authority, Cybersecurity Defensive Operations,* and *Enterprise Monitoring and Management* staff assigned to the *Product Team* as part of the DevSecOps process.

The build process in the RA uses a software factory that is part of the platform layer. Building the software should happen automatically when new code is added to the software repository or when existing code is changed. The built code should also be automatically added to an artifact repository.

Automated testing is also part of the software factory. The quality assurance team oversees testing. It is responsible for verification and validation, making sure the developed code meets relevant specifications and requirements. While the intent is to automate most testing, some manual testing is still likely to occur as part of the development process.

Securing the software must be part of the other development activities. For example, security must be designed into a system (perhaps encrypting all communications). At the same time, the build process must ensure that only security hardened platform elements are used in building the system. Likewise, testing should run security checks for common problems on the developed code (such as array bound errors or buffer overruns).

The artifacts needed to authorize each deployable increment are defined in the planning phase and updated as the system's capabilities progress. They must be automated as much as possible and can be limited to the Mission Layer software being developed, inheriting security controls from hardened Platform Layer and Computing Resources Layer (e.g., access control enforcement provided by Service Mesh in the Platform Layer). The goal is to shift from a separate process that renews Authority to Operate (ATO) every time major changes are made to the system to a posture of continuous ATO. That will allow authorization to be given for mature DevSecOps Continuous Integration/Continuous Deployment (CI/CD) pipelines that can support software releases at the cadence of Agile development. The current (FY21) FAA Security Authorization Handbook [7] refers to this as "ongoing authorization." The handbook states that the FAA is not yet ready to declare NAS systems ready for ongoing authorization but notes that progress is being made in the FAA's Information System Continuous Monitoring (ISCM) implementation, which is one of the elements necessary for ongoing authorization.

Like security, ensuring safety must be considered early in the development process. Safety should be considered at the Solution level, since it is at this level that the operational impacts of any outages, failures, or errors can be understood. Therefore, it is the job of *Solution Management* and the *Solution Architecture, Engineering and Release Coordination* team to ensure that potential hazards are identified, and appropriate mitigations are included in the feature backlogs, to be implemented and tested by the *Product Teams*.

The *Solution Management* and supporting *Solution Team* must also ensure that, in addition to the *Product Teams* assigned to create the core functionality, additional *Product Teams* are created as needed to create and sustain training tools and materials, simulation and testing tools, and so on.

Integration testing checks how code that has been developed as part of a sprint works in the context of the larger system. Quality assurance personnel oversee the integration testing.

4.2.2 Develop Platform and Computing Resources

One or more *Product Teams* will need to be created to obtain and configure the necessary Platform Layer and Computing Resources Layer elements needed to support the *Solution Teams*, according to the priorities and resources provided by *Portfolio Management*.

The *Product Teams* have considerable flexibility in how the platform and computing resources are developed. Compute infrastructure can be acquired and configured on-premises, from a Cloud Service Provider (CSP), or a mix of the two (a hybrid environment). Platform elements can be:

- Used as provided by a CSP,
- Selected by a Platform Layer *Product Team* (PLPT), or
- Acquired by purchasing or leasing a platform (such as VMware Tanzu or Red Hat OpenShift).

Enterprise Engineering will be involved in determining the architecture of these layers to ensure they meet the FAA's needs and requirements (technical and non-technical). *Platform Layer Architects* will be involved in determining what platform layer elements are needed, and *Compute Layer Architects* will be involved in decisions about how computing resources are provided. *Data Architects* will be involved in decisions regarding any data stores or data distribution mechanisms that are part of the platform. Data Architects also design/engineer the data plane – what data we need to store, distribute, archive, etc. *Mission Layer Architects* involved in the planning and design of the mission applications and services must also be involved to make sure that the platform meets the needs of Mission Layer software and users. *Security Architects* will be involved in ensuring the security controls necessary for the cybersecurity of each aspect of the Reference Architecture are in place.

The following activities may or may not be necessary depending on how the computing resources and platform are provisioned:

- Design: Design is needed to ensure that selected computing resources and platform elements work well with each other to the level of expected performance, standards implementation, and other technical requirements. For example, some service meshes depend on other platform elements such as Kubernetes, a container orchestrator.
- Select: Select a particular platform technology or computing resource.
- License: Identify licensing options, dependent on how a particular platform or computing element is acquired.
- Harden: Ensure that selected computing resources and platform technologies are secure and reliable.
- Configure: Configuring and tuning computing resources and platform elements is typically required to get optimal performance, reliability, and interoperability of those

elements. The Infrastructure as Code (IaC) approach wherein such configurations are captured and maintained as declarative files (e.g., YAML Ain't Markup Language [YAML][9] files) aids in the configuration management of the platform.

Product Teams will have at least some computing resources and perhaps some platform elements deployed in FAA facilities. If so, coordination should also be done with *Facility Level Technical Support.*

4.3 Deploy

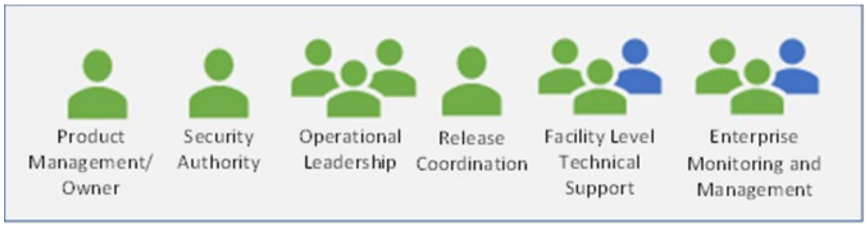

Product Management/ Owner
Security Authority
Operational Leadership
Release Coordination
Facility Level Technical Support
Enterprise Monitoring and Management

Figure 4-4. Deploy Rules

Deploy is the last stage in the pipeline to install/deploy software changes (i.e., a Release) to the operational automation environment. Activities at this stage are associated with roles identified in Figure 4-4, and involve operational evaluation and then coordinating the release of executable software from development to operations.

Before releasing the deployment(s) to the *End Users* in operations, additional testing to ensure operational acceptability of the released software may be needed. This testing differs from the testing that is built into the development pipeline, described in section 4.2.1, which is primarily automated and limited in scope. For example, User Stories associated with the Sprints have been successfully tested prior to the software being released to the Staging/Integration environment. Testing, within this Staging/Integration environment, prior to release to *End Users* is broader in scope and tests how the entire system behaves in an operationally realistic context and may include performance/load testing and evaluation by *End Users*. This type of testing does not need to wait until after a release is complete; software can be deployed to a staging environment for testing at any point in the development process so that early feedback can be received and, if necessary, changes can be made. Operational evaluations for NAS software may be more comprehensive and extensive than what is traditionally done within Agile methodologies. FAA operational evaluations will often involve Human-in-the-Loop testing requiring manual hands-on participation from representative *End Users*. FAA operational evaluations may be broad in scope and include comprehensive end-to-end testing integrating multiple other NAS applications and services. Operational evaluations will require simulation tools and a staging environment that replicates the NAS, allowing the evaluators to fully exercise mission applications and services with live or simulated data and a Platform Layer that replicates the production environment. The replicated test environment can be in the cloud computing, virtual server, or on the actual fielded operational system, depending on its redundancy.

[9] https://en.wikipedia.org/wiki/YAML

Chaos engineering[10] can be performed in the staging environment to ensure resiliency, and *Facility Level Technical Support* staff as well as *Enterprise Monitoring and Management* staff may be involved in making sure that new capabilities can be properly operated.

Configuration management is also a key concern in the Deploy phase. Dependencies among services and applications will be noted and tracked in the software factory. A particular product may be part of multiple different operational solutions, and changes to a single product may impact training, performance, or interoperability with other products. For these reasons, product deployment will generally need to be planned and coordinated at the Solution level.

Results from *End User* evaluations in the staging environment can be fed back into the development cycle or can result in a decision to deploy into production.

Primary activities in the deployment stage are performed by the *Release Coordination, Enterprise Monitoring and Management*, *Security Authority*, and *Facility Level Technical Support* teams. The *Release Coordination* role is responsible for gathering the input of the other roles before deployment is authorized. The intention is that the list of concerns that must be signed off for deployment today will still be signed off in this process, although the role may be named differently. This is the point at which a service or application is certified for operation. Ideally, the teams will plan and execute reconfiguration of operations based on automated deployment scripts and procedures.

In a complex operational environment such as the NAS, the release of the new software to the *End Users* can vary in its scope and timeline. There are several techniques that can be used to control the level of releases (e.g., dark launches, feature toggles, and canary releases). These techniques will be used to ensure that software is deployed in a controlled manner as approved by *Customer Organizations*, including *Operational Leadership, End Users*, and *Facility Level Technical Support.*

The Respond and Recover stage in deployment aims to detect potential operational issues or failures that may arise after having deployed software changes into production and quickly recover to a stable state – thereby minimizing the Mean Time to Recovery (MTTR) metric. To achieve fast recovery, the production environment must be able to roll back to the previous state/version quickly and the version control mechanism of the production environment needs to be highly automated and streamlined.

4.4 Operate

Operations include all activities that ensure the hardware and software needed by NAS specialists to do their jobs is properly configured, performing smoothly, working as intended, and has not been compromised by any cybersecurity incident. Operations activities are performed at both a local (facility) level and at an enterprise level.

Facility Level Technical Support staff provide first-level engineering support, including maintaining hardware at the site (workstations, dedicated systems, power, telecommunications equipment, etc.). These staff coordinate with enterprise security and operations entities to maintain local awareness of any software or security problems that may affect users at the facility and provide a local focal point to coordinate between the facility level and enterprise level support organizations. *Facility Level Technical Support* staff may also have a role in

[10] Chaos engineering is a practice for ensuring reliability and availability that involves purposely injecting faults or disabling parts of a large, complex system to ensure that the system can cope with such outages.

maintaining any facility-specific data that is used by Mission Layer software to allow it to adapt to local conditions.

At the enterprise level, operations support for each Mission Layer software product (which may include one or more applications and mission services) is provided by *Release Coordination* staff. They are part of the DevSecOps team responsible for the product and monitor and control the software to ensure that it continues to function properly and is meeting the needs of users throughout the NAS. They troubleshoot and correct any problems that arise with their product line in the production environment, either by scheduling bug fixes to be included in the next release, or coordinating an emergency release, including regression testing, if necessary.

Also at the enterprise level, *Enterprise Monitoring and Management* staff are responsible for the correct functioning of Platform Layer services, as well as monitoring underlying computing resources. In the case of externally provided cloud computing, visibility into the hardware that provides the compute layer will be limited. However, *Enterprise Monitoring and Management* staff will monitor conformance of the service provider with service level agreements (SLAs) and will take corrective action if SLAs are not being met.

Also at the enterprise level, *Cybersecurity Defensive Operations* staff actively defend the operational and development environments. That includes proactively detecting and securing vulnerabilities, monitoring for indications of compromise, and responding when necessary.

5 Scenarios

This section provides scenarios that elaborate and elucidate how the actors described in Section 3 will engage in the processes described in Section 4. The scenarios are:

- Data and Architecture Governance
- Platform and Compute Layer Evolution and Provisioning
- Agile Development
- Deployment of New Capability to NAS Operations
- Monitoring and Management Responsibilities
- Cybersecurity Operations
- Integration of Cloud Cost Management

The scenarios are not exhaustive but are intended to provide examples of how the AES will manifest in development, sustainment, and operational activities.

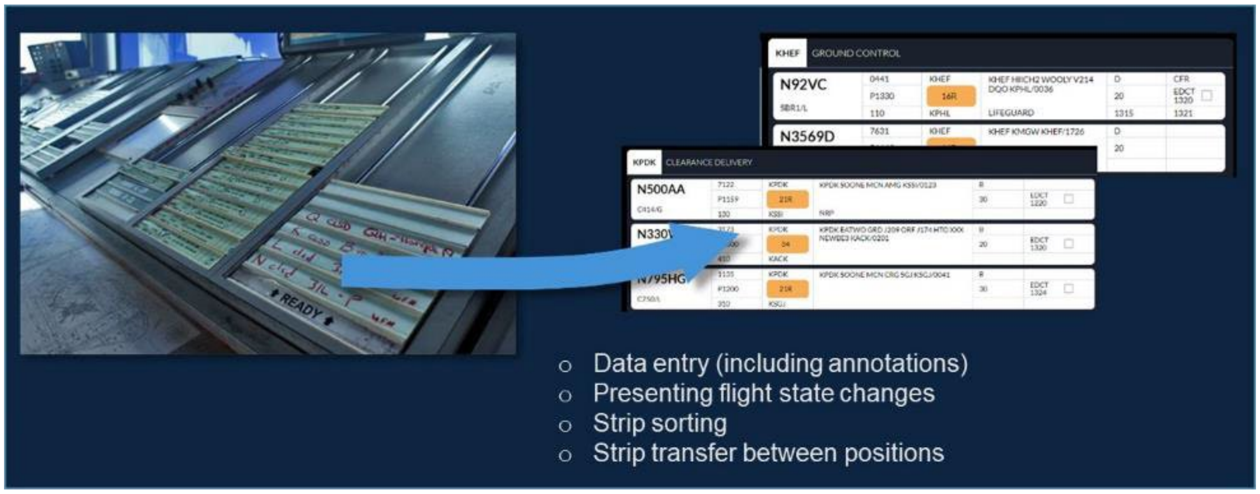
5.1 Common Example for Scenarios

To provide a common example that runs throughout all the scenarios, we introduce three notional Products consisting of two mission applications and one mission service. These are:

- Electronic Flight Strip (EFS) Application. This application presents flight information to airport traffic control tower (ATCT) and TRACON controllers and allows the controllers to modify and enter information. This is analogous to, but not limited to, the way flight information is managed on legacy paper flight strips. Figure 5-1 illustrates a prototype EFS application that was developed by the Application Based Capability Development (ABCD) project.[11]
- Delay Monitor Application. This application monitors flows of flights through the NAS, allowing Traffic Management Coordinators to view the delays being experienced in each sector. Figure 5-2 illustrates a prototype Delay Monitor application developed by the ABCD project. (The Delay Monitor application could be part of a larger Time Based Flow Management (TBFM) system, but for the purposes of this document we are focusing only on a single application.)
- Flight Information Service (FIS). This is a notional NAS Mission Service that maintains highly available, high integrity, distributed stores of flight information, and enforces business rules to ensure that the data is valid and accessed and updated appropriately. The FIS itself does not have a graphical user interface (GUI) to display the flight data, but instead makes the data available via one or more well defined APIs for use in

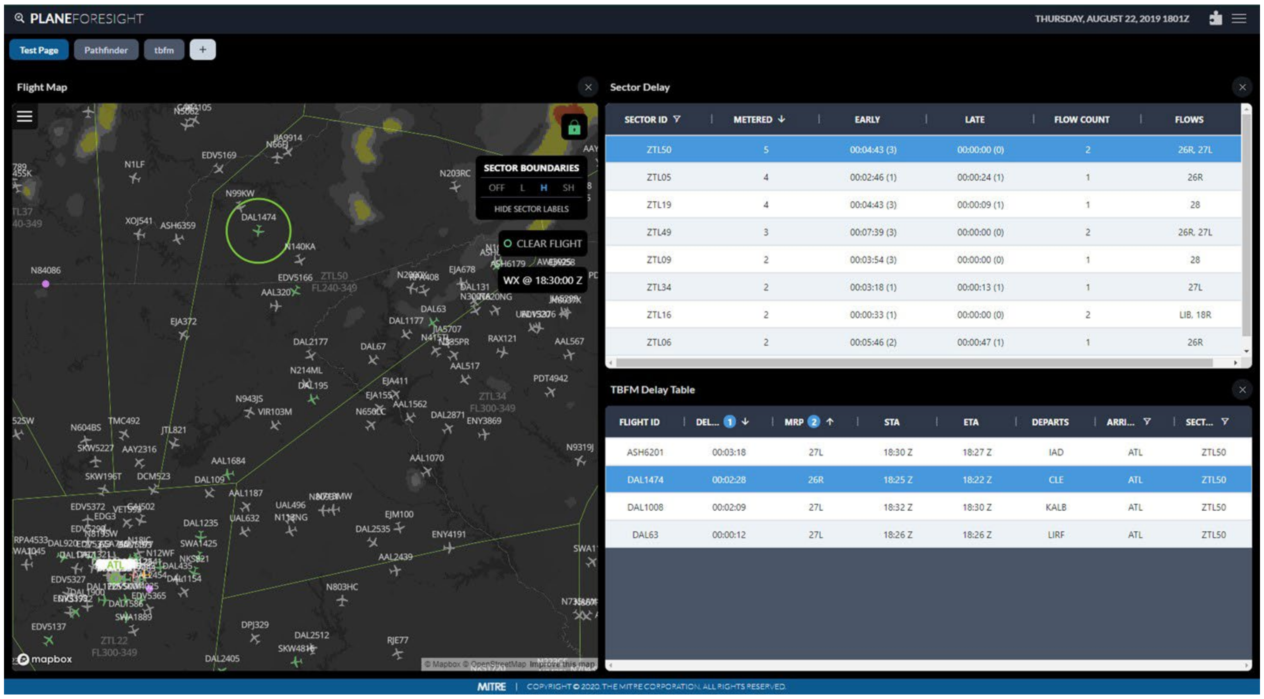
[11] The ABCD framework combines modern software development tools and architectures with Agile processes to enable rapid prototyping of composable software applications for air traffic management. The framework includes the use of common services, lightweight applications, DevSecOps tooling, and cloud deployment environments to continuously develop and deploy traffic flow management services and applications. ABCD is fully documented, including platform design and application enhancements, in [20].

applications. A FIS concept using a schema based on the Flight Information Exchange Model (FIXM) is illustrated in Figure 5-3.



- Data entry (including annotations)
- Presenting flight state changes
- Strip sorting
- Strip transfer between positions

Figure 5-1. Electronic Flight Strip Application


PLANEFORESIGHT
THURSDAY, AUGUST 22, 2019 1801Z
Test Page
Pathfinder
tbfm
Flight Map
SECTOR BOUNDARIES
OFF L H SH
HIDE SECTOR LABELS
CLEAR FLIGHT
WX @ 18:30:00 Z
mapbox
© Mapbox © OpenStreetMap Improve this map
MITRE | COPYRIGHT © 2020 THE MITRE CORPORATION. ALL RIGHTS RESERVED.

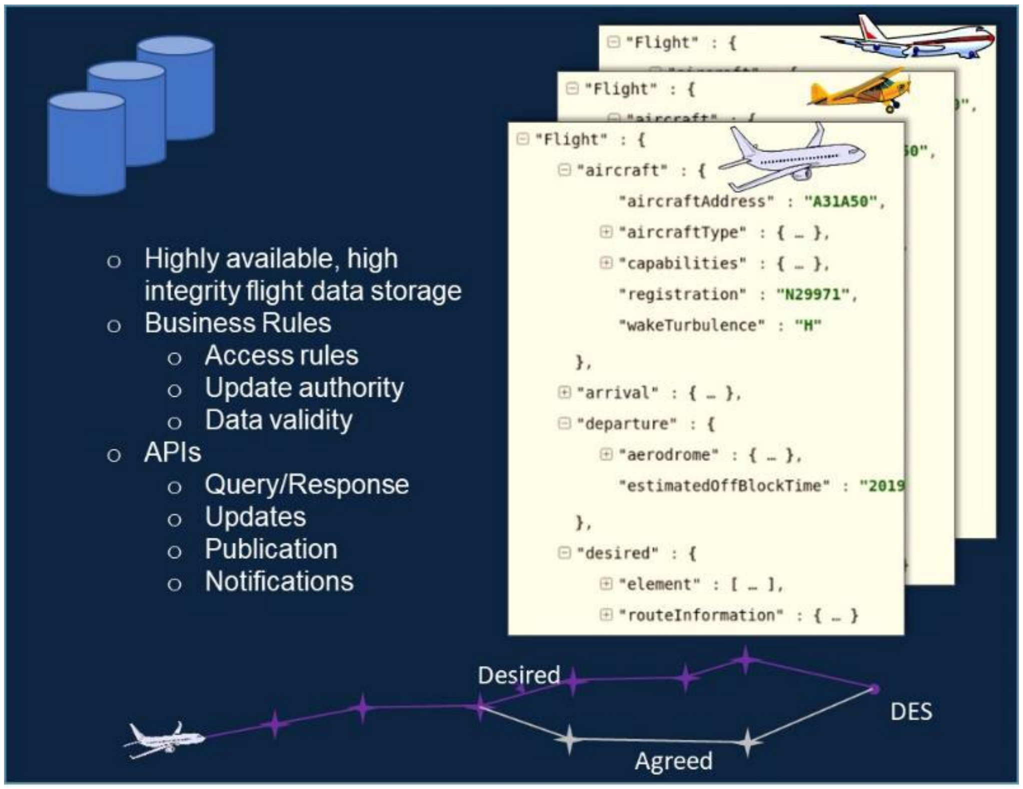
Sector Delay

SECTOR ID	METERED	EARLY	LATE	FLOW COUNT	FLOWS
ZTL50	5	00:04:43 (3)	00:00:00 (0)	2	26R, 27L
ZTL05	4	00:02:46 (1)	00:00:24 (1)	1	26R
ZTL19	4	00:04:43 (3)	00:00:09 (1)	1	28
ZTL49	3	00:07:39 (3)	00:00:00 (0)	2	26R, 27L
ZTL09	2	00:03:54 (3)	00:00:00 (0)	1	28
ZTL34	2	00:03:18 (1)	00:00:13 (1)	1	27L
ZTL16	2	00:00:33 (1)	00:00:00 (0)	2	LIB, 18R
ZTL06	2	00:05:46 (2)	00:00:47 (1)	1	26R

TBFM Delay Table

FLIGHT ID	DEL... 1	MRP 2	STA	ETA	DEPARTS	ARRI...	SECT...
ASH6201	00:03:18	27L	18:30 Z	18:27 Z	IAD	ATL	ZTL50
DAL1474	00:02:28	26R	18:25 Z	18:22 Z	CLE	ATL	ZTL50
DAL1008	00:02:09	27L	18:32 Z	18:30 Z	KALB	ATL	ZTL50
DAL63	00:00:12	27L	18:26 Z	18:26 Z	LIRF	ATL	ZTL50

Figure 5-2. Delay Monitor Application



- Highly available, high integrity flight data storage
- Business Rules
  - Access rules
  - Update authority
  - Data validity
- APIs
  - Query/Response
  - Updates
  - Publication
  - Notifications

```
"Flight" : {
    "aircraft" : {
        "aircraftAddress" : "A31A50",
        "aircraftType" : { … },
        "capabilities" : { … },
        "registration" : "N29971",
        "wakeTurbulence" : "H"
    },
    "arrival" : { … },
    "departure" : {
        "aerodrome" : { … },
        "estimatedOffBlockTime" : "2019
    },
    "desired" : {
        "element" : [ … ],
        "routeInformation" : { … }
```

Figure 5-3. Flight Information Service Concept

The EFS application, the Delay Monitor application, and the FIS are all products, with a *Product Owner*, one or more *Agile Teams*, and so on. The EFS application and Delay Monitor application are independent of one another, but both applications depend on the FIS.

Solutions to the FAA's business/mission needs are created by integrating these products. The scenarios in this document focus on two solutions to FAA mission needs:

- Improved flight data in small towers.
- Improved flow management.

Those solutions are illustrated in Figure 5-4. Note that the Platform Layer and Computing Resource layer elements necessary to support these Mission Layer applications and services are also part of these solutions.

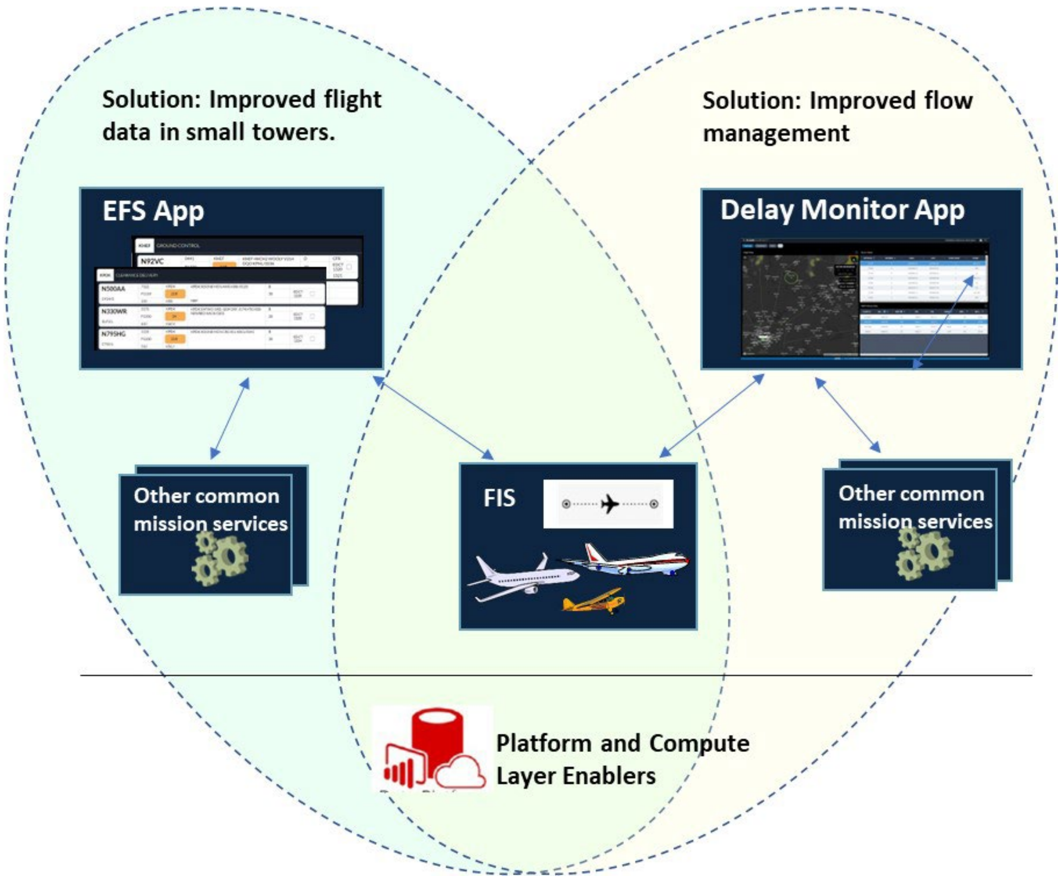

Solution: Improved flight data in small towers.
Solution: Improved flow management
EFS App
Delay Monitor App
Other common mission services
FIS
Other common mission services
Platform and Compute Layer Enablers

Figure 5-4. Products Integrated to Provide Solutions

This example is a simplified view. The Delay Monitor application will depend on additional common mission services to obtain airspace information, status of traffic management initiatives, and so on. The FIS will depend on other data sources, including perhaps interfaces with legacy ATC systems to provide the authoritative source of some elements of flight information. However, to keep the scenarios simple, this document focuses on these three products and the interdependencies among them.

The FIS is an example of a shared service. As in this example, shared services are products. When a solution requires the use of a shared service and changes are needed to that shared service then the shared service product becomes one of the parts that make up a solution.

5.2 Common Oversight Assumptions for Scenarios

Scenarios within this section assume that FAA leadership has defined the NAS Vision, Administrator priorities, the Enterprise Architecture, and technical policies to inform development, maintenance, and evolution of prioritized solutions. With these artifacts, field needs, and stakeholder value assessments, the FAA will prioritize solutions for deployment. In the following scenarios, FAA leadership has selected the solutions “Improved flight data in small towers” and “Improve flow management” for near-term release.

Throughout solution planning, development, deployment and operations, the FAA governance structure aims to facilitate collaboration, synchronization, and transparency across the Agency using DevSecOps and Agile principles. As part of acquisition processes, during initial planning, FAA leadership will evaluate key artifacts and high-level capability design to inform decision-making, solution prioritization, and authorize funding for incremental development.

The following scenarios assume that initial planning has been approved based on the submission of key artifacts and high-level design for the capabilities within the two solutions. FAA leadership and acquisition processes are assumed to result in staffing and funding *Product Teams* to develop the EFS application and enhance the FIS common mission service and Delay Monitor application. FAA leadership and acquisition processes are also assumed to result in staffing and funding *Solution Teams* to address coordination of release schedules of these products, interoperability and integration testing, training, operations support, and any other cross-product issues that must be addressed to realize useful capabilities in the field. After initial *Product Team* funding, strategic budgetary planning will include funding streams to continue provision for *Product Teams* to maintain expertise to sustain and enhance valued products. When *Planning and Budget*, *Enterprise Engineering*, *Solution Management* and *Product Management* determine that a product is no longer providing value, a decision will be made to deprecate it, retire it, and then end funding for its sustainment.

5.3 Data and Architecture Governance

5.3.1 Description

This scenario describes the process and governance that *Enterprise Engineering* and *Solution Teams* follow to define a new solution [8] and measure the effectiveness of governance as the new solution is developed, deployed, and used. This scenario is centered around using the governance process when defining, implementing, and fielding a new solution to improve flight data at small towers.

This scenario explores the following questions:

1. Who are the key players in governance (for this scenario)?
2. What is the governance for the interactions between *Enterprise Engineering* and other areas (i.e., *Solution Teams*, *Product Teams*, *Agile Teams*) to ensure that the envisioned architecture is being followed when solutions are defined by *Solution Teams* and implemented by *Product Teams*?
3. How will governance be measured and updated to address any deficiencies identified in the governance process?

This scenario is related to the Agile Development (Section 5.5) and the Platform and Compute Layer Evolution and Provisioning scenario (Section 5.4). The Agile Development scenario focuses on how *Product Teams* collaboratively work together to develop a new solution including modifying the existing FIS service to support the EFS application whereas this scenario describes the governance process for defining a new solution. The Platform and Computing Layer Creation and Provisioning scenario describes a process for adding a desired product to the existing Platform and Computing Layers to meet the needs of the EFS product team. This scenario, Data and Architecture Governance, focuses on a process for developing a solution that primarily impacts the Mission Layer.

This scenario assumes that the authoritative systems data and models are being maintained using Digital Engineering [9]. This scenario assumes that the artifacts identified in Table 5-1, and described throughout the scenario event flow, will be updated and maintained using Digital Engineering.

5.3.2 Roles

This section describes the involvement of key roles (as identified in Figure 3-1) and how they work together to evolve solution(s).

Enterprise Engineering will play a significant role in developing and evolving the architecture and working with *Solution Teams* to define solutions, given that common services are anticipated to be modified. Enterprise Engineering will ensure that Solution Teams understand and follow the architectural vision. Governance is defined to encourage decisions to be made by the team that is closest to outcome of the decision (i.e., decisions are made at the lowest level possible). The benefits and effectiveness of governance should be measured regularly using defined operational measurements. Examples of measurements include customer metrics such as improved reliability, quality, and efficiency; quick delivery of products; reuse of capabilities; reuse of models and data; and issues caught early-on reducing risk to architecture assurance to name a few [8].

Figure 5-5 identifies the roles for Data and Architecture Governance.

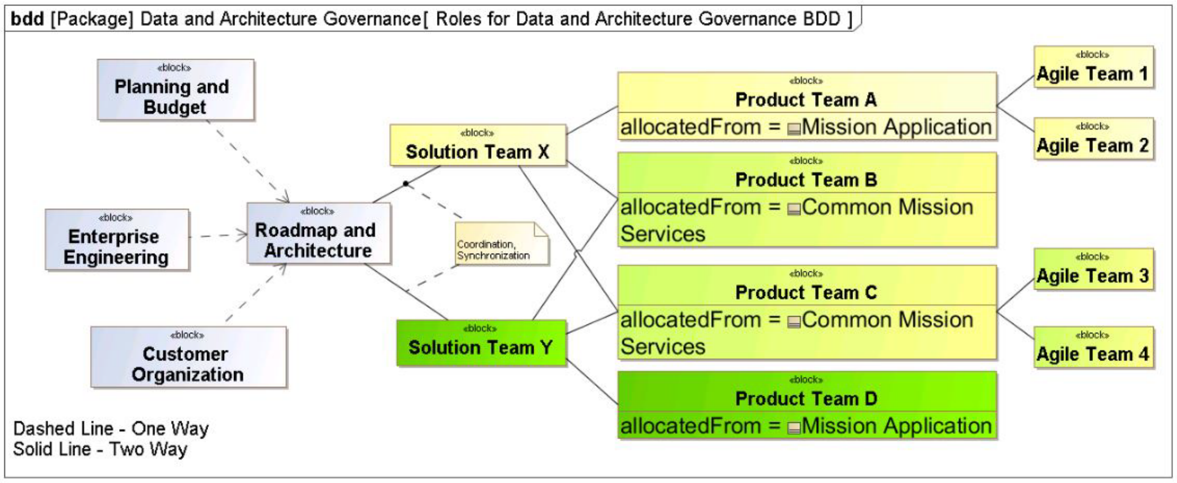

bdd [Package] Data and Architecture Governance[Roles for Data and Architecture Governance BDD]
«block» Planning and Budget
«block» Enterprise Engineering
«block» Customer Organization
«block» Roadmap and Architecture
«block» Solution Team X
«block» Solution Team Y
Coordination, Synchronization
«block» Product Team A allocatedFrom = Mission Application
«block» Product Team B allocatedFrom = Common Mission Services
«block» Product Team C allocatedFrom = Common Mission Services
«block» Product Team D allocatedFrom = Mission Application
«block» Agile Team 1
«block» Agile Team 2
«block» Agile Team 3
«block» Agile Team 4
Dashed Line - One Way
Solid Line - Two Way

Figure 5-5. Roles for Data and Architecture

Enterprise Engineering

Enterprise Engineering will collaborate with *Planning and Budget* and *Customer Organizations* to use and develop a roadmap and architecture that is used to plan for and evolve capabilities, features, and enablers. *Enterprise Engineering* will collaborate with *Solution Teams* (and *Product Teams* and *Agile Teams* as needed) to review and refine the architecture and to provide input on epics.

The *Mission Layer Architect* and *Data Architect* are the primary *Enterprise Engineering* roles in this scenario. The *Mission Layer Architect* collaborates with other teams, identifies requirements and any updates to the common FIS service and common System Analysis and Recording (SAR) service. The *Data Layer Architect* provides the data management framework and ensures that the updates to FIS and SAR are accessible and accurate and compliant with standards. *Enterprise Engineering* may also specify how something is done when the nature of the change has enterprise implications.

Solution Team

Solution Teams in collaboration with *Enterprise Engineering* will review the existing architecture/models, designs, and requirements and perform detailed analysis to assess and refine the architecture and specify features. Solution *Teams* can receive input from *Customer Organization* that may require modest changes. As noted above, *Customer Organizations* will provide significant needs as input to the development of the roadmap. The *Solution Teams* (along with *Product Teams*) will also identify the capabilities and features to be developed.

Product Team

Product Teams develop features according to the updated architecture (provided by *Enterprise Engineering*) and based on a prioritization provided by the *Solution Team*. *Product Teams* will update registries for services, applications, and APIs. In this scenario, the *Product Teams* will develop the new EFS application and modify two common services (FIS, SAR).

Agile Team

Agile Teams develop features, provided by the *Product Teams*, via stories. Recall that a *Product Team* may consist of a single *Agile Team* with a *Product Owner*, or multiple *Agile Teams* each with a *Product Owner* and coordinated by a *Product Management* team. In the example in Figure 5-5, products A and C are relatively complex and require two *Agile Teams* each, whereas products B and D are relatively simple and the *Product Team* for each comprises a single *Agile Team*.

5.3.3 Starting State

The starting state is that *Planning and Budget* and *Enterprise Engineering* have provided *Solution Teams* and *Product Teams* with a roadmap, vision, and architecture to use as a basis to develop the solution.

5.3.4 Event Flow

This section discusses two event flows:

1. The first event flow describes the interactions between *Enterprise Engineering Solution Teams* to define a new solution.
2. The second event flow describes the process for evaluating the effectiveness of Data and Architecture Governance from defining new solutions through delivery and maintenance of the solution. Metrics, including those based on user feedback, are defined and used to assess and improve the governance.

The event flows for this scenario are based on the following guiding principles:

- Identify and develop solutions and products that meet FAA goals and needs.
- Determine what services to acquire based on an enterprise view so that similar or duplicate services are not acquired. In addition, ensure that developed services (in conjunction with existing services) work together, deliver value, and support the vision.

- Ensure that the features that are developed integrate and work with other features. In addition, ensure that features that can be re-used are identified and utilized.

5.3.4.1 Event Flow: Enterprise Engineering and the Definition of a New Solution

Figure 5-6 provides the event flow to define a new solution.

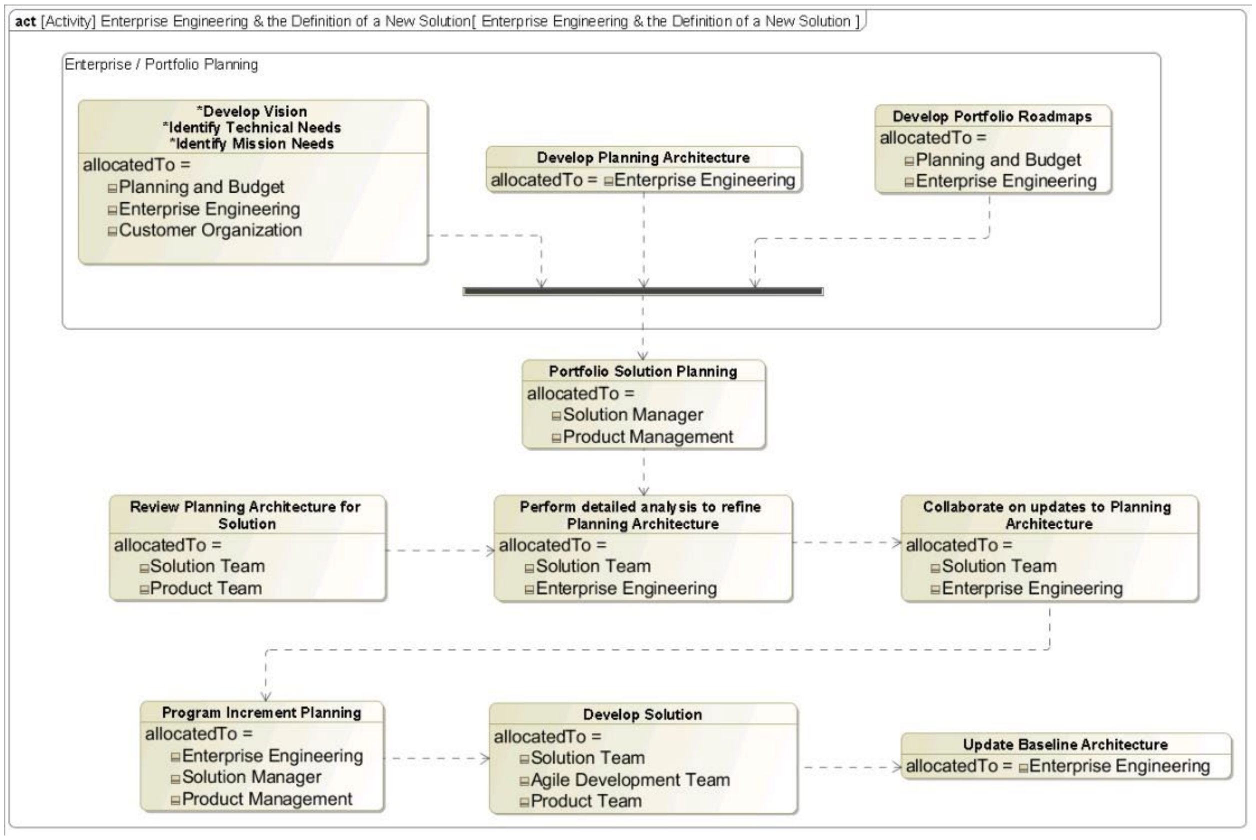

act [Activity] Enterprise Engineering & the Definition of a New Solution[Enterprise Engineering & the Definition of a New Solution]
Enterprise / Portfolio Planning
*Develop Vision *Identify Technical Needs *Identify Mission Needs
allocatedTo = Planning and Budget, Enterprise Engineering, Customer Organization
Develop Planning Architecture
allocatedTo = Enterprise Engineering
Develop Portfolio Roadmaps
allocatedTo = Planning and Budget, Enterprise Engineering
Portfolio Solution Planning
allocatedTo = Solution Manager, Product Management
Review Planning Architecture for Solution
allocatedTo = Solution Team, Product Team
Perform detailed analysis to refine Planning Architecture
allocatedTo = Solution Team, Enterprise Engineering
Collaborate on updates to Planning Architecture
allocatedTo = Solution Team, Enterprise Engineering
Program Increment Planning
allocatedTo = Enterprise Engineering, Solution Manager, Product Management
Develop Solution
allocatedTo = Solution Team, Agile Development Team, Product Team
Update Baseline Architecture
allocatedTo = Enterprise Engineering

Figure 5-6. Enterprise Engineering and the Definition of a New Solution

<u>Enterprise/Portfolio Planning</u>

For an outlook up to several years or more, *Planning and Budget*, *Customer Organizations*, *and Enterprise Engineering* will develop a vision and identify mission needs. Based on a technical strategy and an assessment of mission needs, *Enterprise Engineering* will identify technical needs. *Enterprise Engineerin*g will use the Baseline Architecture (the implemented architecture) to develop a Planning Architecture that meets the vision, mission needs, and technical needs. *Enterprise Engineering* will also leverage other work that may be available such as concepts based on research efforts. Table 5-1 identifies the components of the Baseline and Planning Architectures along with who is delegated the authority to update the architecture component.

Note that *Enterprise Engineering* develops all the Planning Architecture components identified in Table 5-1 during this activity. For example, *Enterprise Engineering* will define the needed common services, applications, needed messaging and initial APIs based on functional decomposition and domain analysis. The Planning Architecture is developed in sufficient detail to enable initial cost estimates and prioritization and to support the development of Portfolio Roadmaps as well as address transition challenges that may apply within scope.

Table 5-1. Baseline and Planning Architecture

Architecture Components	Baseline Architecture	Planning Architecture	Delegation
Architecture Overview			
Discussion of major architectural decisions and rationale. Technical strategy to support evolution	X	X	*Enterprise Engineering*
Mission Layer: Common Services, Applications, and Data			
Definition of Common Services and Applications. Definition of the interactions between services and applications (via messages or APIs). (Functional decomposition, data inputs, and data outputs)	X	X	*Enterprise Engineering*
Logical Data Model (Defines the structure and relationships between data classes for common services and applications)	X	X	*Enterprise Engineering*[12]
Functional and Performance Requirements for Common Services and Applications	X	X	*Enterprise Engineering*
Data Exchange	X		*Solution Team*
Physical Data Model	X		*Solution Team*
Registries (Services, Applications, and APIs)	X		*Product Team* *Agile Team*
Platform Layer			
Platform Layer Requirements	X	X	*Enterprise Engineering*
Catalog of selected Platform Layer Tools and Software Components	X		*IT Infrastructure Product Management*[13]
Computing Resources Layer			
Computing Resources Layer Requirements	X	X	*Enterprise Engineering*
Catalog of selected Computing Resources Layer network, storage, and computing resources	X		*IT Infrastructure Product Management*
Security			
Security Requirements	X	X	*Enterprise Engineering*
Guidance			
Best Practices, Standards, and Guidance	X	X	*Enterprise Engineering*

[12] The Stewardship Communities of Practice (SCoP) is responsible for updating the Logical Data Model. The SCoP understands and stays abreast of Operational Improvements (OIs) needed within their domain. The SCoP is integral to designing the data model.

[13] See section 5.4 for the Platform and Computing Layer Creation and Provisioning scenario.

Portfolio Solution Planning

Solution Manager and *Product Management* prioritize solutions in the portfolio. The *Solution Team* is tasked to define a solution to improve flight data at small towers.

Review Planning Architecture

The *Solution Team* reviews the vision, mission needs, technical needs, and the Planning Architecture as related to Improve Flight Data at Small Towers. Note that the Planning Architecture was developed by *Enterprise Engineering* during the Enterprise/Portfolio Planning activity. The Planning Architecture identifies that a new application, Electronic Flight Strip (EFS) is needed along with changes to two existing commons services, FIS, and SAR. The specific architecture components include:

- Definition of Common Services, Applications, and initial APIs (includes functional decomposition, inputs, and outputs)
    - Changes to FIS common service
    - Changes to SAR common service
    - Definition of EFS Application
- Logical Data Model and changes for flight data (for additional surface data) and SAR data
- Functional and Performance Requirements for EFS application and changes to requirements for FIS and SAR.
- No changes are needed for the Platform Layer, Computing Resources Layer, or Security Requirements (for this scenario)

The Planning Architecture also identifies that FIS will need to follow the FIXM exchange standard when information is exchanged.

Perform Detailed Analysis to Refine Planning Architecture

The *Solution Team*, the existing FIS Product Team, SAR Product Team, along with *Mission Layer Architect* and *Data Architect* will perform a more detailed analysis of the functional decomposition and logical data model (from the Planned Architecture) which can include a data domain analysis[14]. This analysis may inform, for instance, the need to further decompose a service into microservices. The *Solution Team* follows guidance from *Enterprise Engineering* (as captured in Best Practices, Standards, and Guidance) to refine the FIS and SAR services and the EFS application. The *Solution Team* also does not identify any changes to platform, computing, or security requirements.

Collaborate on Proposed Changes to Planning Architecture

Based on the more detailed analysis, the *Solution Team* proposes minor changes to the Planning Architecture, specifically:

[14] The Solution Team has access to the Baseline Architecture components and can, for example, access the relevant Physical Data Model if desired.

- Changes to the definition of EFS Application
- Changes to the Logical Data Model for flight data

Enterprise Engineering reviews the proposed changes and collaborates with the *Solution Team* until agreement on the proposed changes is reached. *Enterprise Engineering* updates the Planning Architecture based on the agreed changes.

For changes to the Logical Data Model, the Stewardship Communities of Practice (SCoP) will need to review and approve the proposed changes. The SCoPs are communities that are specific to a data subject area and are focused on stewardship and data architecture across systems, programs, and organizations. The SCoP will need to implement the approved changes to the Logical Data Model.

<u>Program Increment Planning</u>

Solution Manager and *Product Management* identify a plan for developing features and enablers. Some features for EFS Application, for example, include:

- Data entry (including annotations)
- Presenting flight state changes
- Strip sorting

<u>Develop Solution</u>

The *Solution Team* will use the updated Planning Architecture (related to improving flight data at small towers) to develop:

- The Physical Data Model
- The Data Exchange Matrix

For the Physical Data Model, for example, the *Solution Team* will start with the Logical Data Model and define the data schemas considering the characteristics of the database to be used. The FIS, EFS, and SAR *Product Teams* and *Agile Teams* will develop prioritized capabilities, features and enablers and will update registries for services, applications, and APIs as these are developed. The *Product Teams* and *Agile Teams* will develop features and enablers using Best Practices, Standards, and Guidance.

<u>Update Baseline Architecture</u>

Enterprise Engineering updates the Baseline Architecture to reflect the changes needed to implement Improved Flight Data at Small Towers. *Enterprise Engineering* integrates the revised Planning Architecture into the Baseline Architecture. *Solution Teams*, *Product Teams*, and *Agile Teams* will update the other components of the Baseline Architecture as noted in Table 5-1. This approach ensures that the Baseline Architecture reflects the as built system and hence provides an accurate foundation for subsequent planning by *Enterprise Engineering*.

5.3.4.2 Event Flow: Measuring Effectiveness of Data and Architecture Governance

Figure 5-7 depicts a high-level event flow diagram of evaluating the effectiveness of Data and Architecture Governance by quantifying a demonstrable working system (capabilities, features) and reducing risk throughout the capabilities implementation. The enterprise architecture framework (i.e., best practices, RA, roadmaps), as well as demonstrable capabilities and value assessment of the operational capability, drives the governing process, which is then measured and used as input to optimize the Data and Architecture Governance processes.

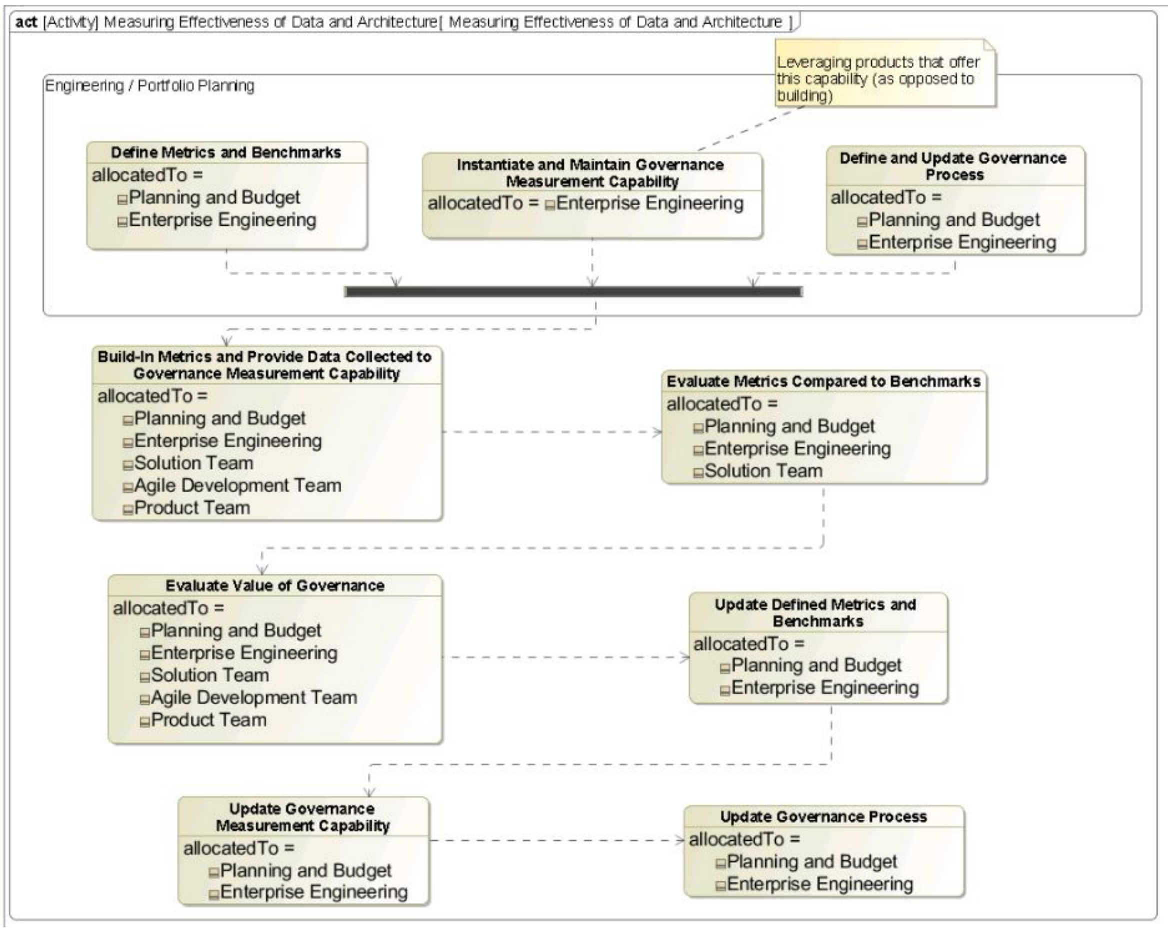

act [Activity] Measuring Effectiveness of Data and Architecture[Measuring Effectiveness of Data and Architecture]
Engineering / Portfolio Planning
Leveraging products that offer this capability (as opposed to building)
Define Metrics and Benchmarks
allocatedTo = Planning and Budget, Enterprise Engineering
Instantiate and Maintain Governance Measurement Capability
allocatedTo = Enterprise Engineering
Define and Update Governance Process
allocatedTo = Planning and Budget, Enterprise Engineering
Build-In Metrics and Provide Data Collected to Governance Measurement Capability
allocatedTo = Planning and Budget, Enterprise Engineering, Solution Team, Agile Development Team, Product Team
Evaluate Metrics Compared to Benchmarks
allocatedTo = Planning and Budget, Enterprise Engineering, Solution Team
Evaluate Value of Governance
allocatedTo = Planning and Budget, Enterprise Engineering, Solution Team, Agile Development Team, Product Team
Update Defined Metrics and Benchmarks
allocatedTo = Planning and Budget, Enterprise Engineering
Update Governance Measurement Capability
allocatedTo = Planning and Budget, Enterprise Engineering
Update Governance Process
allocatedTo = Planning and Budget, Enterprise Engineering

Figure 5-7 Measuring Effectiveness of Data and Architecture Governance

Define Metrics and Benchmarks

Enterprise Engineering and *Planning and Budget* define metrics (e.g., value of operations and demonstrable capabilities [10], quality of capabilities, Agile development, user feedback) and benchmarks that will be used to evaluate the effectiveness of the Data and Architecture and Governance. These metrics will show whether governance is bringing value and where to improve [11]. *Enterprise Engineering* will use these metrics as an aid to improve the governance process. It is anticipated that new metrics and benchmarks will not be needed for this scenario and those already defined will be used (given that other common services have already been deployed and updated and new applications have been deployed).

Develop and Maintain Governance Measurement Capability

Enterprise Engineering implements and updates the governance measurement capability (when needed) using the metrics and benchmarks defined. It is anticipated that the governance measurement capability will not need to be updated given this scenario.

Define and Update Governance Process

Planning and Budget and *Enterprise Engineering* update the data and architecture governance process based on findings provided by the governance measurement capability. It is anticipated that the existing governance process will not be updated given this scenario.

Build-in Metrics and Provide Data Collected to Measurement Capability

Planning and Budget, Enterprise Engineering, Solution Teams, Product Teams, and *Agile Teams* build-in indicators and provide data and metrics to the governance measurement capability to provide a holistic view of the effectiveness of governance (i.e., from vision and strategy to the development of capabilities to the fielding and use of capabilities by operations). For this scenario, metrics will be collected during the Agile development and use of the capability. It is anticipated that changes will not be needed to collect the metrics during Agile development for the two common services FIS and SAR and that existing indicators will be used. Indicators will need to be built-in to collect metrics from the new EFS application. Agile development, demonstrable, and value metrics will be collected on the use of updated common services and the new application. All metrics data collected will be aligned with metrics data collected on the vision, strategy, and the architecture

Evaluate Metrics Compared to Benchmarks

The governance measurement capability auto-analyzes the collected data and metrics relative to benchmarks. The *Planning and Budget*, *Enterprise Engineering*, and *Solution Teams* also evaluate the metrics via human-in-the-loop. It is anticipated that the existing benchmarks will not be updated given this scenario.

Evaluate Value of Governance

Planning and Budget, *Enterprise Engineering*, *Solution Teams*, *Product Teams*, and *Agile Teams* use output from the governance measurement capability to identify any updates to the data and architecture governance such as where thresholds are exceeded indicating that governance is not as effective as it could be. This includes evaluating user feedback. Given that this scenario is a nominal case, governance is effective.

Update Defined Metrics and Benchmarks

Planning and Budget and *Enterprise Engineering* update (add, delete) defined metrics and benchmarks as needed based on the evaluation of metrics collected and determined value of governance. Given that this scenario is a nominal case where governance was effective, metrics and benchmarks will not need to be updated

Update Measurement Capability

Enterprise Engineering updates the governance measurement capability to reflect the updates to governance and metrics. This scenario does not result in updates to the measurement capability.

Update Governance Process

Planning and Budget and *Enterprise Engineering* update the governance process to reflect any resultant changes in the governance process. The updated governance process is available to all teams (*Planning and Budget*, *Enterprise Engineering*, *Solution Teams*, *Product Teams*, and *Agile Teams*). Given that this scenario is a nominal case where governance was effective, governance will not need to be updated.

5.3.5 End State

The end state of this scenario is that 1) the Baseline Architecture is updated to reflect the changes needed to improve flight data at small towers, and 2) the governance measurement capability provided information on the effectiveness of governance (and was updated to address any identified governance inefficiencies).

5.3.6 Summary

This scenario examined the governance for adding a new solution. More specifically, reuse and modification of two existing common services (FIS, SAR), and one new application (EFS). This scenario was intentionally focused where *Enterprise Engineering* has a significant role. This scenario is related to the Agile Development (Section 5.5).

This scenario addressed the three questions posed in Section 5.3.1. These questions focused on identifying the key players in governance (for this scenario); the governance for the interactions between *Enterprise Engineering* and other areas (i.e., *Solution Teams*, *Product Teams*, *Agile Teams*) to ensure that the envisioned architecture is being followed when solutions are defined by *Solution Teams* and implemented by *Product Teams*; and measuring the effectiveness of governance and updating the governance process to maintain effectiveness.

This scenario assumes that the artifacts identified in this scenario will be developed and maintained as digital engineering models.

5.4 Platform and Compute Layer Evolution and Provisioning

5.4.1 Description

This scenario illustrates how the FAA will acquire, configure, and provision platform and computing resources to meet the needs of Mission Layer *Product Teams* (MLPTs).

In this scenario, the FIS product is ready to be developed and it is now necessary to stand up the computing resources as well as the development and runtime environments that will be used for development. These will be instantiated using the platform that already exists as a basis. The platform consists of security hardened containers that implement a software factory, as well as various run-time components such as message busses, a container orchestrator, a service mesh and other run time platform elements identified in the RA. In addition, the high-level design for FIS calls for the use of an event mesh. This runtime element is not presently contained within the platform. This need is identified to the platform team for potential inclusion in the platform.

The Platform Layer abstracts aspects of the implementation from the development teams that are using it. For example, geographic information about the implementation (where things are hosted) may be abstracted. Non-functional requirements, such as latency and availability requirements, will determine those abstracted characteristics (e.g., geographic locations and failover mechanisms). These abstractions allow the MLPTs doing development of mission services and applications to focus on implementing NAS capabilities instead of having to write code implementing many non-functional requirements. These abstractions also allow the underlying functionalities for things like failover to be developed in a consistent way across the mission services and applications, which simplifies monitoring and managing the NAS. These abstractions also enable the platform development teams to provide that functionality in a way that may evolve over time with minimal impacts on the mission services and applications.

This scenario answers the following questions:

1. How will the set of teams be organized to create, manage, and sustain the platform/compute layer?
2. How do requirements flow into the selection of platform services and compute resources?
3. How are requirements tested and validated?
4. What is the onboarding process and how does the platform/compute layer support this process? How does the platform and/or compute layer support each development activity and independent product teams?
5. To what extent are product teams required to use what is in the platform? Are we taking a recommendation or directive approach?
6. What is the concept for the FAA to evolve the platform?

5.4.2 Roles

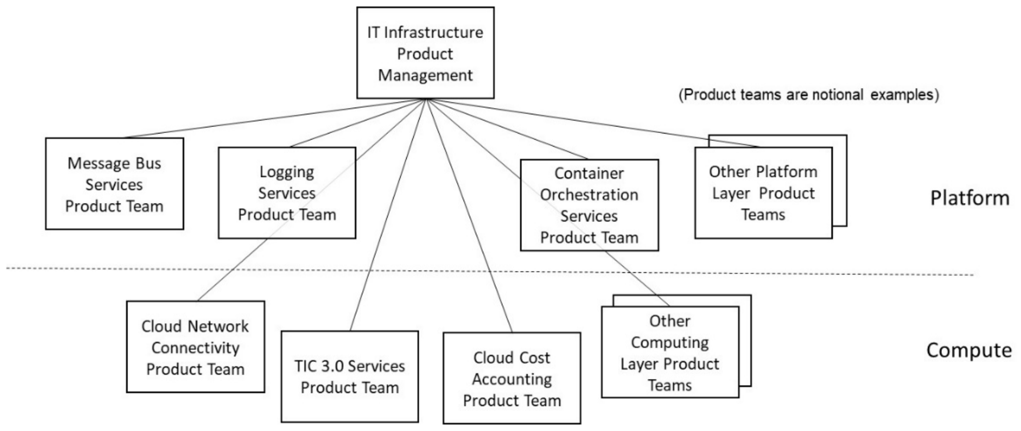

IT Infrastructure Product Management
(Product teams are notional examples)
Message Bus Services Product Team
Logging Services Product Team
Container Orchestration Services Product Team
Other Platform Layer Product Teams
Platform
Cloud Network Connectivity Product Team
TIC 3.0 Services Product Team
Cloud Cost Accounting Product Team
Other Computing Layer Product Teams
Compute

Figure 5-8. Example Platform and Compute Roles with Single Point of Contact

As shown in Figure 1-1, IT infrastructure related layers in the RA include the Platform Layer and an underlying Compute Layer. These layers make possible the efficient development and deployment of the Mission Layer applications and services. While the Platform and Compute Layers are separate in the architecture, there are advantages to having a single focal point responsible for both. This provides a means to ensure integration of the elements within these layers. Also, it provides MLPTs a single point of contact responsible for coordinating provisioning of both compute resources and platform elements. As shown in Figure 5-8, that single point of contact is the IT Infrastructure *Product Management* team (ITIPMT), which includes a single *Product Manager* as well as roles such as *Release Train Engineer* and *System Engineer*. The ITIPMT team coordinates and integrates the work of individual *Product Teams* responsible for specific elements of the Platform Layer and Compute Layer, using scaled Agile methodologies as described in the following scenarios.

Figure 5-8 shows a single instance of the ITIPMT organization. An open question is whether there should be multiple instances, each in charge of a separate operating environment. For example, there could be one instance of an IT Infrastructure organization supporting the Mission Critical Operating Environment and another supporting the Mission Essential Operating Environment. The key question for deciding between those options is whether there are sufficient differences in the specific knowledge and expertise needed to support each environment to justify the duplication of effort and increased complexity of having different organizations in charge of each environment. The rest of this scenario will be written as if a single instance will be used. If a single instance is used, there would be specific *Product Teams* devoted to understanding and supporting each operating environment.

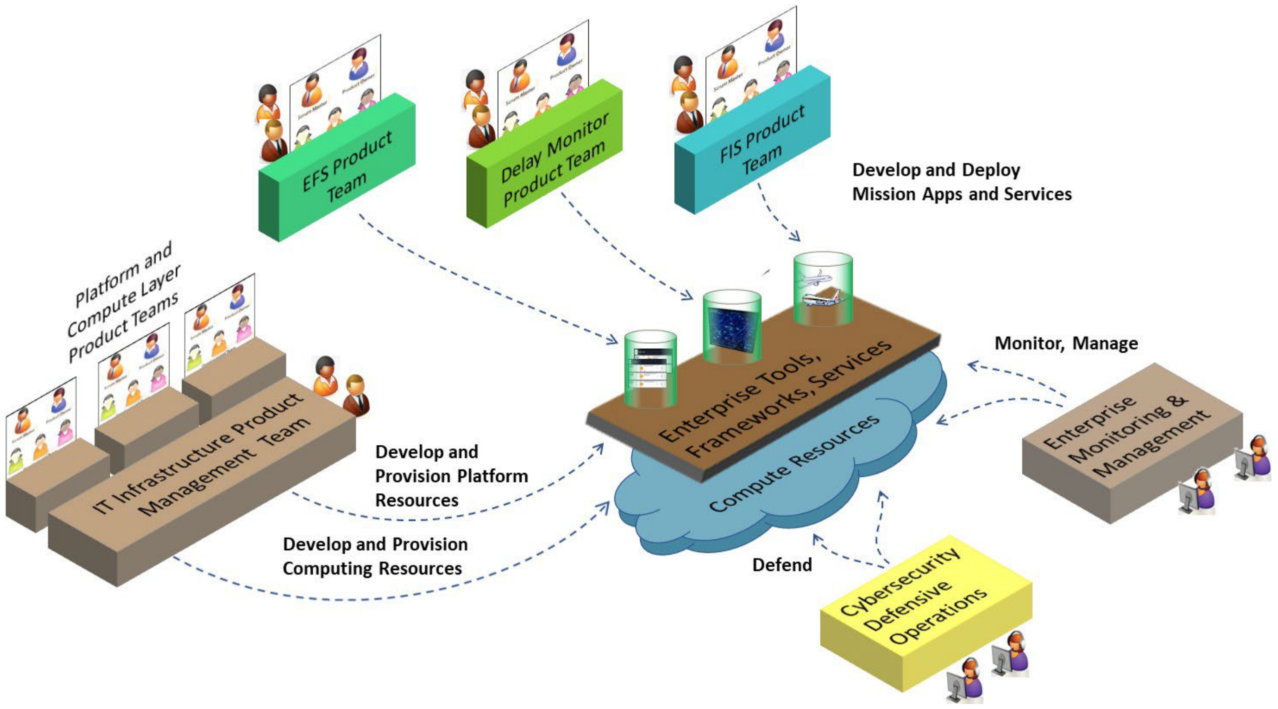

EFS Product Team
Delay Monitor Product Team
FIS Product Team
Develop and Deploy Mission Apps and Services
Platform and Compute Layer Product Teams
IT Infrastructure Product Management Team
Enterprise Tools, Frameworks, Services
Compute Resources
Monitor, Manage
Enterprise Monitoring & Management
Develop and Provision Platform Resources
Develop and Provision Computing Resources
Defend
Cybersecurity Defensive Operations

Figure 5-9. Computing and Platform Layer Provisioning Roles and Processes

Figure 5-9 shows Platform and Compute Layer *Product Teams* developing and sustaining the elements that make up these layers. MLPTs such as the FIS Product Team come to the ITIPMT when they need platform and compute layer resources. ITIPMT would then assign a *System Architect* under their purview to gain an understanding of the needs of the MLPTs and then assist in provisioning the appropriate compute and Platform Layer resources.

It is recognized that the platform is likely to have certain approved configurations. For example, a set of the platform services may be approved for use by applications and services expected to run in a Mission Critical Operating Environment. A different set of platform services may be approved for use in Mission Essential Operating Environments. There will be *Product Teams* organized to create these approved configurations and test them to make sure they meet the requirements of those operating environments. The ITIPMT will have some *Product Teams* devoted to supporting individual platform services or compute resources, and other *Product Teams* devoted to creating the sets of platform services and compute resources approved for use in the different operating environments.

Note that for this scenario we are assuming that all three *Product Teams* developing the NAS applications (EFS and Delay Monitor) and common services (FIS) are using instances of a common Platform Layer, but other models are possible. *Product Teams* may have unique requirements or situations that lead to different platforms being used, especially during development. However, since applications and mission services need to interoperate, some aspects of the platform (e.g., message busses, API gateways) will need to be common, at least in the production environment.

5.4.3 Starting State

This scenario starts with multiple *Product Teams* under one ITIPMT with the responsibility for evolving and sustaining platform and compute resources to be used by the various Mission Service and Mission Application *Product Teams*. The *Product Teams* under ITIPMT also have responsibility for assisting the FIS *Product Teams* in provisioning the platform and compute resources for FIS use. The following assumptions are made for the purpose of this scenario:

- The FIS product has been funded and initial high-level planning has been done.
- FIS *Product Management* and *Product Teams* have been created.
- An initial set of Platform Layer products has been created to meet the needs of an initial set of Mission Layer *Product Teams*.
- Suitable cloud-based compute resources have been acquired for use by NAS products.
- The high-level design for FIS calls for the use of an event mesh that is not currently part of the platform.

5.4.4 Event Flow

As described in the starting state, a platform already exists, and suitable cloud-based compute resources have been acquired. Figure 5-10 illustrates the overall nominal event flow followed in this scenario and the roles involved in each step of that flow. Figure 5-12 provides detail on some off nominal cases that can occur during the event flow.

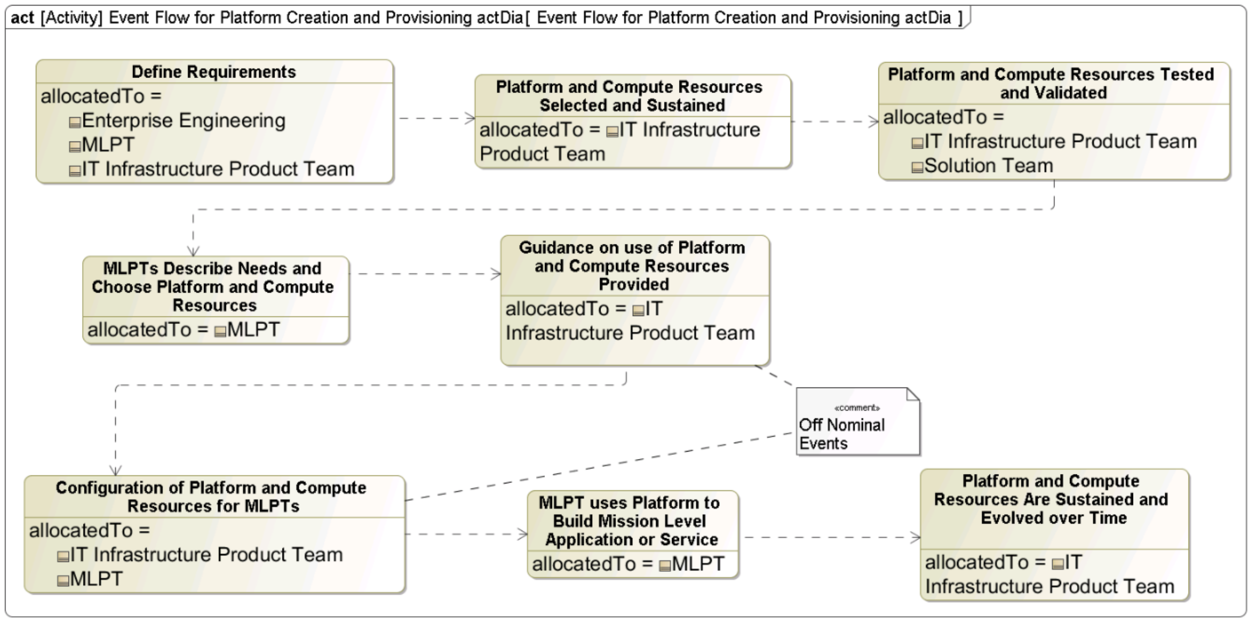

act [Activity] Event Flow for Platform Creation and Provisioning actDia[Event Flow for Platform Creation and Provisioning actDia]
Define Requirements
allocatedTo =
Enterprise Engineering
MLPT
IT Infrastructure Product Team
Platform and Compute Resources Selected and Sustained
allocatedTo = IT Infrastructure Product Team
Platform and Compute Resources Tested and Validated
allocatedTo =
IT Infrastructure Product Team
Solution Team
MLPTs Describe Needs and Choose Platform and Compute Resources
allocatedTo = MLPT
Guidance on use of Platform and Compute Resources Provided
allocatedTo = IT Infrastructure Product Team
«comment»
Off Nominal Events
Configuration of Platform and Compute Resources for MLPTs
allocatedTo =
IT Infrastructure Product Team
MLPT
MLPT uses Platform to Build Mission Level Application or Service
allocatedTo = MLPT
Platform and Compute Resources Are Sustained and Evolved over Time
allocatedTo = IT Infrastructure Product Team

Figure 5-10. Event Flow for Platform and Compute Layer Evolution and Provisioning

illustrates how requirements flow into the Platform and Compute Resources Layers.

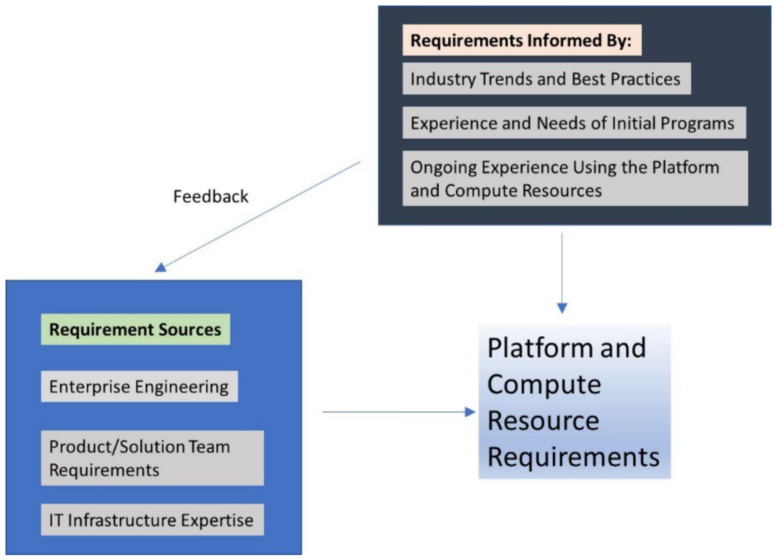

Requirements Informed By:
Industry Trends and Best Practices
Experience and Needs of Initial Programs
Ongoing Experience Using the Platform and Compute Resources
Feedback
Requirement Sources
Enterprise Engineering
Product/Solution Team Requirements
IT Infrastructure Expertise
Platform and Compute Resource Requirements

Figure 5-11. Requirements Flow for Platform and Compute Resources

<u>Requirements Development and Service Adoption</u>

There are three major sources from which requirements can flow into the selection of platform service and compute resource elements. Those ways are:

- *Enterprise Engineering* generates high level requirements for platform services or compute resources. This can be based on high level requirements of the NAS operational infrastructure or other architectural concerns under their purview. An example might be the need for fail-over capability and performance requirements associated with that capability. The *Security Architect* within *Enterprise Engineering* is responsible for ensuring that security requirements are included.
- *Product Teams* and *Solution teams* using the platform and compute resources may define requirements based on their needs. In this scenario FIS defines a requirement for a possible new platform service as part of the product's design process.
- Platform and Compute Layer *Product Teams* could generate requirements based on their expertise and industry trends/best practices.

Any of these alternatives could lead to the adoption of a new platform service or compute resource. Ongoing funding to support evolution of the platform and compute layer in a timely manner must be provided, but how that is done is beyond the scope of this document. Fortunately, the process for determining whether to add a new platform service or compute resource is the same, regardless of the origin of the requirement.

While there are three possible origins for platform requirements, requirements that govern whether a platform service is appropriate to an operating environment will usually come from *Enterprise Engineering*. For example, they would be the likely source of a requirement that states platform services used in a mission critical environment will meet DO 278 software assurance standards.

Platform Layer and Computing Resources Layer *Product Teams* select, configure, license, and test platform elements that meet those requirements. While doing so, they are informed by:

- Industry trends/best practices
- Experience and needs of initial programs
- Ongoing experience using the platform and compute resources by Platform Layer and Computing Resources Layer *Product Teams* as well as MLPTs.

Any of these influences can also end up feeding back into the requirements. For example, there may have been an initial requirement that dealt with acceptable latency for a platform service to respond to a service request. With experience, it is found that this requirement is too lax and that a stricter latency time is required to achieve acceptable system performance. This feedback is then incorporated into the requirements and a stricter service latency response standard is adopted.

Requirements Validation and Testing

Testing of platform and compute resource requirements is very similar to the testing of requirements for any development project as discussed in section 4.2.1. The use of automated testing is expected when possible. There are two broad categories of tests that will need to be frequently run:

- Tests to ensure that the platform or compute resources are built, configured, and operating properly within the specifications of the requirements.
- Tests to ensure that the platform is operating properly with mission layer applications and services that make use of the platform.

When a change is made to the compute resources or the Platform Layer software, it is important that both kinds of tests be run. Clearly it is important to run tests checking that mission layer applications and services run properly after some change has been made to the APIs between the Platform Layer and the Mission layer. But experience has shown that even when those interfaces have not changed, unexpected side effects can sometimes prevent mission layer software from operating properly. Therefore, it is critical that regression tests of mission layer software be automated in such a way that the PLPTs can run them. When APIs have changed or if it is necessary for the mission layer software to be rebuilt, it will still be necessary to get the Solution Teams developing mission layer software involved as they may need to change or rebuild their software. Version control will be essential so that these dependencies are known, and the correct set of software will be released for deployment. An essential part of this version control is that every mission layer application and service should have a Software Bill of Materials (SBOM) that details dependencies. This SBOM should be created through automated processes to ensure that it is accurate and up to date.[15]

MLPTs come to the ITIPMT and describe their needs. The ITIPMP will provide a customer service interface for MLPTs, which will be automated where possible. ITIPMT then assigns *Product Teams* to assist in provisioning needed platform and compute resources. There will be times where there will be more interaction required between a MLPT such as FIS and IT Infrastructure. In those cases, ITIMPT will assign a *Product Team* to work with the MLPT.

The Platform Layer *Product Teams* provide information on the various platform elements that make up the Platform Layer and provide guidance how they are intended to be used. (This is

[15] This software bill of materials will also be key to maintaining security of the system and will play a role in the security scenario in section 5.8.

analogous to the way the FTI program currently provides guidance to programs on what kind of network services they should order for different needs.)

While the platform will come preconfigured for use, there will be some aspects of the configuration that will need adjustment by the MLPTs such as FIS. This is because some aspects of the configuration are better known by the MLPTs. For example, MLPTs will have a much better idea about how scalable their product needs to be and what are appropriate limits for the number of instances of a service that would be instantiated by default. Where this is the case, ITIPMT should document these configuration items, and how MLPTs should adjust them. Note that the MLPTs will not be directly changing configurations or parameters in the operational environment. Rather, these parameters will be set in automated build configuration files. Actual deployment into the production environment is assumed to be under control of staff in the Enterprise Security and Operations category, working closely with the *Product Teams* following DevSecOps principles.

Cloud computing is designed to provide multitenancy. This is true both at the compute resources layer and at the Platform Layer running on a cloud. For the purposes of this scenario, that is important because it allows better utilization of resources while at the same time providing necessary security and workload protections. In practice, this means that *Product Teams* do not need to concern themselves with what products run where and what compute resources are used. It also means that it is easy to setup multiple independent environments for product development, testing, product staging and integration, and NAS operations (production).

The FIS makes use of the Platform Layer elements. *Product Teams* using the platform to develop NAS applications and services are expected to use software that is part of the platform. This applies both at the level of individual platform services, and to sets of services that have been configured for use in particular operating environments. For example, if Solace and Kafka are the supported message bus platform elements, the FIS *Product Teams* should not use some other message bus software such as RabbitMQ. If only Kafka has been rated as acceptable for use in a mission critical operating environment, and FIS is expected to operate in that environment, the FIS *Product Teams* are not free to choose Solace even though it is part of the platform. In the nominal event flow, MLPTs use the existing products within the platform layer.

<u>Platform/Compute Layer Evolution</u>

It is recognized that occasionally there may be a good reason why a MLPT cannot use some portion of the platform. In other cases, a MLPT may want a new type of platform element added. These off nominal cases are covered in Figure 5-12. If a *Product Team* feels there is a compelling reason that they cannot use an existing capability in the platform and they need to use some alternative that provides that same functionality, then they must complete a waiver process that will need approval from *Enterprise Engineering*. Because this will almost inevitably also involve questions that impact overall funding, planning, and timing, the *Solution Team* and *Planning and Budget* will also need to be involved in the waiver process.

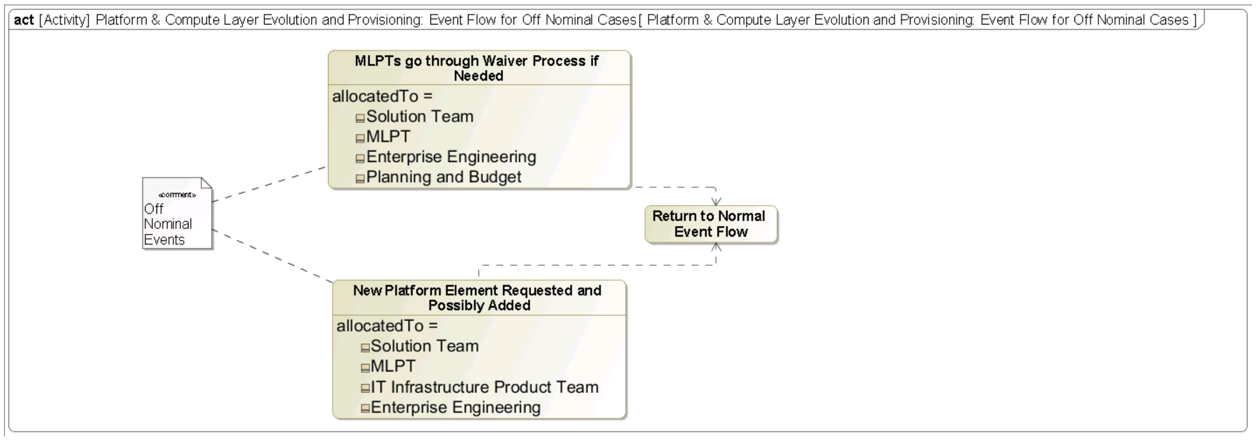

act [Activity] Platform & Compute Layer Evolution and Provisioning: Event Flow for Off Nominal Cases [Platform & Compute Layer Evolution and Provisioning: Event Flow for Off Nominal Cases]
MLPTs go through Waiver Process if Needed
allocatedTo =
Solution Team
MLPT
Enterprise Engineering
Planning and Budget
«comment» Off Nominal Events
Return to Normal Event Flow
New Platform Element Requested and Possibly Added
allocatedTo =
Solution Team
MLPT
IT Infrastructure Product Team
Enterprise Engineering

Figure 5-12. Platform and Compute Layer Evolution and Provisioning: Event Flow for Off Nominal Cases

While at first this may seem like a restrictive approach, it is important to accrue various benefits of having a platform. These benefits accrue to both the NAS and the products such as FIS using the platform. Among these benefits are:

- Lower costs to FIS because of enterprise licensing of platform elements.
- Less vendor lock-in since many vendors have experience using the platform.
- Potentially lower development costs for FIS because the platform has already been acquired and configured for differing Operating Environments (OEs) in the NAS.
- Less need for FIS to worry about things like DO-278 compliance for the platform portions of the software since that is handled at the platform level and the FIS inherits that compliance.
- From a security perspective, an easier path towards getting ATO, again because the platform has already received ATO. Therefore, FIS only needs to show the code built on top of the platform should receive an ATO.

A second off nominal case is when a MLPT identifies the need for a new type of platform element. In this scenario a FIS *Product Team* identifies the need for a potential platform element type, an event mesh, that is not part of the platform. As with the waiver process, *Planning and Budget* and the *Solution Team* over the FIS *Product Teams* will also need to be involved because of likely impacts.

ITIPMT agrees with the MLPT for FIS and decides to add an event mesh to the Platform. Other stakeholders will be consulted as necessary. For example, feedback might be requested from *Enterprise Engineering* in an advisory role. A Platform Layer *Product Team* is assigned to select the best product and add it to the menu of platform elements. They do so, coordinating with FIS as needed. *Enterprise Engineering* is made aware of the new capability.

The FIS MLPT makes use of the event mesh that has become part of the platform.

While the above gives one example of how the platform will evolve, it is only one possible scenario. It is expected that over time new technologies will be developed and others deprecated. Likewise new compute resources will become available, while others may no longer be available. ITIPMT should be actively researching what is available. This may be in the form of a new entrant to fill an existing platform element role such as a new message broker. Or it may be in the form of an entirely new category of platform element or compute resource. The event mesh in this scenario is an example of an entirely new category of platform element. When a new technology clearly fills a need within the NAS infrastructure while doing so at acceptable cost and within acceptable security constraints, it may be adopted. Technologies may be dropped for lack of use, inability to maintain an acceptable security posture, cost reasons, or similar issues. As described in the Data and Architecture Governance scenario, ITIPMT works with *Enterprise Engineering* to ensure the configuration of the deployed applications and infrastructure components is maintained in the baseline architecture.

5.4.5 End State

The scenario ends at the point where the FIS *Product Teams* have the Platform and Computing Resources Layer resources they need. Those resources include both tools and infrastructure.[16] Those resources include what is needed to both develop their mission layer software products and to deploy them into an operational environment. Finally, those resources include what is needed to monitor and defend what has been deployed. Furthermore, the *Product Teams* developing the platform have configured versions of the platform appropriate to different operating environments such as mission critical or mission essential.

5.4.6 Summary

This scenario has described how the FAA will acquire, configure, and provision platform and computing resources.

The question about how teams will be organized to create and evolve the Platform and Compute Layers was addressed in the Roles description. Both the Platform and Compute layers are supported by individual *Product Teams* devoted to the development and support of services and resources for their associated layers. All Platform and Compute *Product Teams* activities are coordinated by a *Product Management* team, the ITIPMT, which integrates the products and provides a central point of contact for the Mission Layer *Product Teams* using the Platform.

The question about requirements flow and testing was addressed in the event flow. The resources and services that the Platform and Compute layers contain is driven by a set of requirement sources that themselves are informed by industry best practices, trends, and *Product Team* experiences. Using an Agile process, the Platform and Compute requirements continue a controlled evolution in meeting the needs of the Mission Layer and Enterprise. Also, in an iterative, Agile manner, PLPTs select, configure, license, and test platform elements that meet those requirements. The use of automated testing is promoted throughout the development lifecycle.

Questions about onboarding and support for Mission Layer Product Teams were also addressed in the event flow. The AES Platform contains tools that help facilitate software development in a more consistent way. Standardized CI/CD pipelines that ensure all software performs steps required to validate new and updated software is included. The platform also includes a collection of commonly reusable frameworks and libraries that add consistency in the approaches

[16] In other words, the Product Teams will have the software development tools they need such as a DevSecOps software factory. They will also have the runtime infrastructure needed to build mission services and applications, e.g. message buses, and container orchestration. And they will have the compute infrastructure needed to host development through eventual deployment.

and design taken by the software developers. It also reduces the time spent and the costs involved in developing the software.

The question about whether we are taking a directive or a recommendation approach was also addressed in the event flow. MLPTs are directed to use available software services, frameworks, and resources that the Platform and Compute layer provides. However, if the MLPTs require the use of a component that is not part of the platform or is not approved for use in the operating environment, they can complete a waiver process.

The question about evolution was also addressed in the event flow. The addition of new technologies and the removal of existing services is under the direction of the ITIPMT. Various stakeholders (e.g., *Enterprise Engineering*, MLPT, ITIPMT) can propose the addition of new or alternative service offerings into the Platform and/or Compute layer. The ITIPMT, after coordinating with MLPT, has the authority to approve any changes to the Platform and Compute layer services. Creation of the initial Platform and Compute layer capabilities was not described in the scenario, but it is assumed to occur in the same way that evolution occurs, which was described in the "off nominal" case in the event flow.

5.5 Agile Development

5.5.1 Description

This scenario describes the activities of two *Product Teams*, working collaboratively to create a solution for improving flight management in small towers. In this scenario, one team is developing the EFS App, and the other team is modifying an existing FIS to add additional information that will be used by the EFS application. The EFS application and supporting changes to the FIS together make up a solution that supports improved flight data in small towers, as shown in Figure 5-4. Other common mission services might also be needed for the overall solution but in this scenario, we assume that only the FIS needs to change to support the EFS application.

This scenario addresses the following questions:

- What is the composition (government vs. contractor staffing) of the various teams at each level of the Enterprise Agile framework (Figure 4-2)?
- How will the efforts of multiple different development efforts be coordinated?
- How are safety hazards identified and mitigated?
- How are security vulnerabilities avoided?
- How are training needs identified and met?
- How is deployment planning conducted?

While this scenario focuses on the *Product Teams* doing software development, there will be other *Product Teams* working on other aspects of the solution. For example, there will be one or more *Product Teams* focused on creating materials and tools that will be used to train tower controllers on how to use the EFS application. There may also be *Product Teams* focused entirely on deployment planning for rollout of the EFS App and new version of the FIS service to small towers. These *Product Teams* will follow methodologies and processes like those of the *Product Teams* that are developing the software. It is the responsibility of the *Solution*

Management to make sure that *Product Teams* have been assigned to these non-functional requirements that will make up part of the solution backlog.

5.5.2 Roles

This section describes the key roles involved in this scenario, following the overall structure of high-level roles from Figure 3-1.

Solution Teams

Solution Management – Responsible for the entire lifecycle of the *Improved flight data in small towers solution*, which includes both the EFS application, and the changes required to the FIS to support the EFS application.

Enterprise Security and Operations

Safety Authority – Provides expertise and guidance supporting the *Solution Teams* and *Product Teams* to identify hazards and ensure mitigations are proposed, implemented, and tested.

Security Authority – Provides expertise and guidance for ensuring security policies have been followed and security requirements have been met by the EFS solution.

Enterprise Monitoring and Management – Ensure capabilities and features needed for enterprise-wide monitoring are identified and provided and provides feedback on operational suitability of how these features are implemented.

Customer Organizations

Operational Leadership – Provides input to *Solution Teams* and *Product Teams* as capabilities and features are identified and prioritized.

End Users – Provide feedback to developers on suitability of features for operational use.

Facility Level Technical Support – Provided input on features needed for field support.

Product Teams

The *Product Teams* participate in planning but are primarily responsible for the development of software as well as other products such as training materials and monitoring tools.

5.5.2.1 Government versus Contractor Roles

This section addresses how the FAA will allocate roles to government and contractors by examining the roles from the perspective of Content Authority, Technical Authority, and Execution Authority. As seen in Figure 5-12, Content Authority roles define what work the team performs. Technical Authority roles determine how to best get the work done. Finally, Execution Authority roles facilitate how the team can execute the work better.

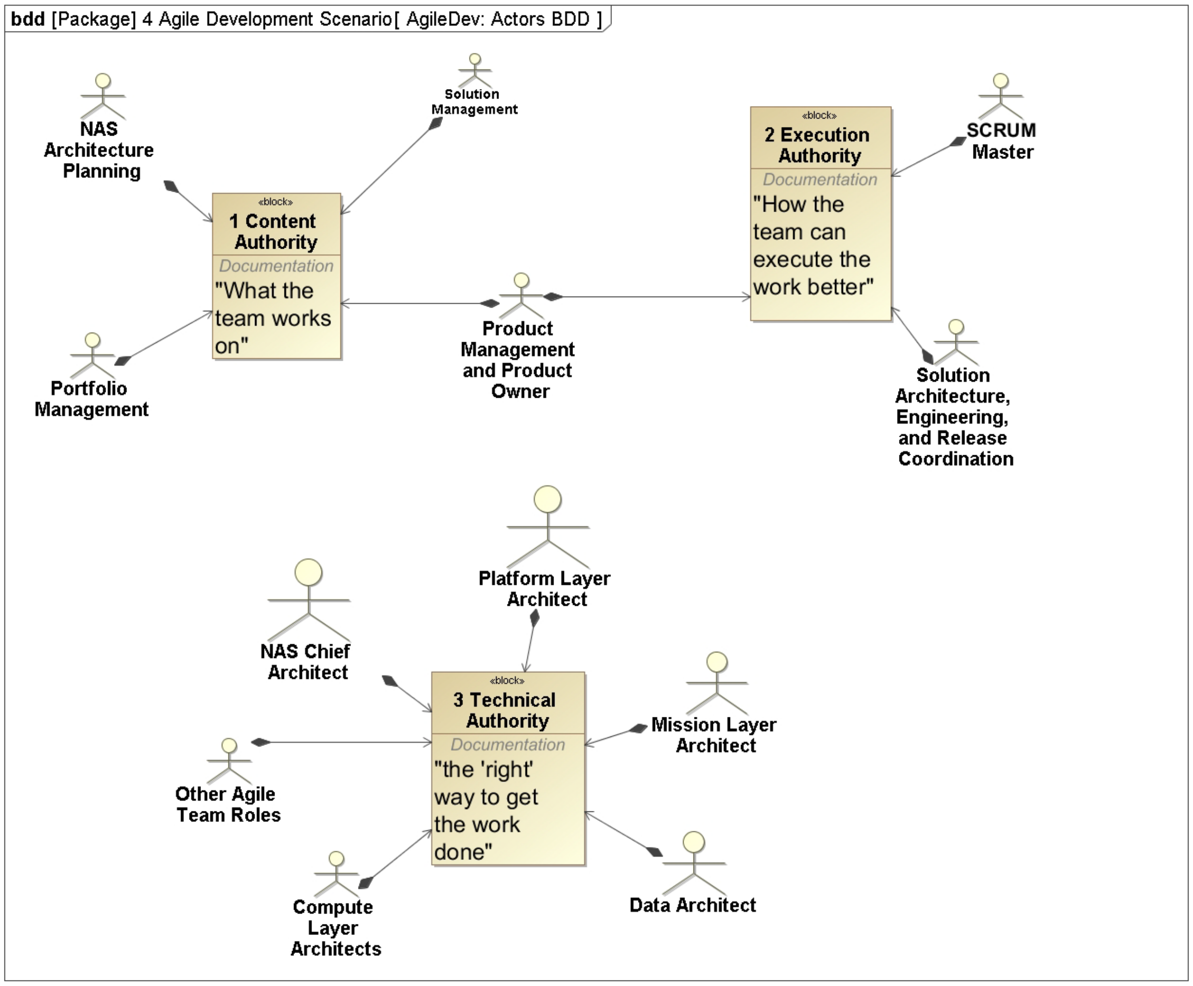

bdd [Package] 4 Agile Development Scenario[AgileDev: Actors BDD]
NAS Architecture Planning
Solution Management
«block» 1 Content Authority
Documentation
"What the team works on"
Portfolio Management
Product Management and Product Owner
«block» 2 Execution Authority
Documentation
"How the team can execute the work better"
SCRUM Master
Solution Architecture, Engineering, and Release Coordination
Platform Layer Architect
NAS Chief Architect
«block» 3 Technical Authority
Documentation
"the 'right' way to get the work done"
Mission Layer Architect
Other Agile Team Roles
Compute Layer Architects
Data Architect

Figure 5-12. Roles and Categories of Authority

Figure 5-12 can be used to visualize what roles are inherently governmental, what roles would be filled by contractors, and which could be filled by either FAA or contractor personnel. This scenario assumes a contracting model that allows for a government-led team to define the functionality to be built in small increments. Contractors provide ongoing support to build that functionality under government supervision.

Those roles that fall under Content Authority decide what capabilities are to be developed, which commits the government to a course of action. Therefore, those roles are inherently governmental and must be filled by a government employee. The government must decide what work the teams perform.

Under Execution Authority, the *Scrum Masters* that facilitate the work of the EFS application and FIS *Product Teams* can be contractor personnel. They do not commit the government to a course of action and would facilitate the development effort. The *Release Train Engineer* can also be a contractor that can facilitate the Scrum of Scrums. While they do have a role communicating with FAA personnel by working closely with *Product Owners* and *Solution Management*, they do not commit the government to a course of action. They may be involved with meeting with *End Users*, but that would be alongside the *Product Owner* or *Solution Management*, which would commit the Government to a course of action.

The roles under Technical Authority could largely be done by contractors. Those roles involve the design and execution of capabilities. In this scenario that would include designing and developing the EFS application and the modifying of the FIS. Because of the overarching role of the *NAS Chief Architect* and their oversight responsibilities, that role should be an FAA position.

The government can use contractors to help support governmental personnel in exercising their roles. Care must be taken to ensure conflict of interest is avoided. For example, the maintenance of a backlog associated with the EFS application at the solution level could be done by a contractor if what was on the backlog was controlled by a government employee. At the Product level, backlogs might be managed by contractors since at that level, functionality has already largely been determined at the Solution level. Work at the Product level largely focuses on the most expeditious way to develop the functionality specified at higher levels.

5.5.3 Starting State

This scenario focuses on developing the EFS application and associated changes to the FIS. The scenario begins at the point where a project has already been funded and at the Enterprise level, portfolios have been organized and cross organizational planning has taken place to review, integrate, and prioritize resource allocation to portfolios.

This scenario assumes the compute and platform resources the EFS application and FIS will use have already been acquired and the Platform Layer being used has been developed, and the necessary instances and infrastructure have been provisioned for the EFS application and FIS teams. The development of the Platform Layer was conducted by a different set of *Product Teams*.

5.5.4 Event Flow

An overview of the event flow for the development scenario is provided in Figure 5-13, and is described below.

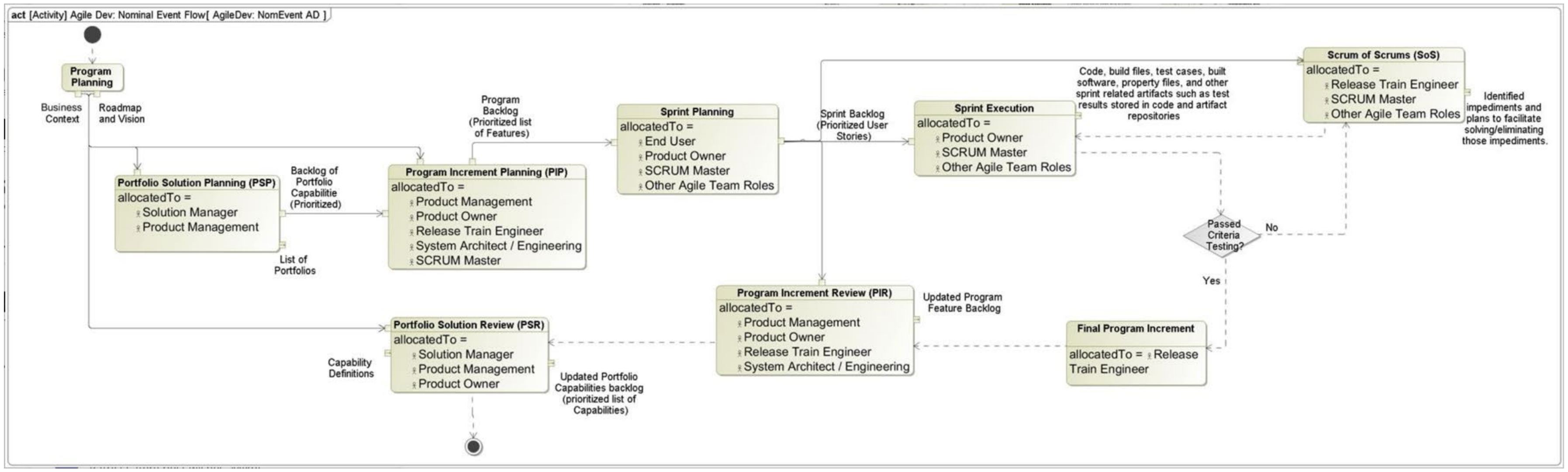

act [Activity] Agile Dev: Nominal Event Flow[AgileDev: NomEvent AD]
Program Planning
Business Context
Roadmap and Vision
Portfolio Solution Planning (PSP)
allocatedTo =
Solution Manager
Product Management
Backlog of Portfolio Capabilitie (Prioritized)
List of Portfolios
Program Backlog (Prioritized list of Features)
Program Increment Planning (PIP)
allocatedTo =
Product Management
Product Owner
Release Train Engineer
System Architect / Engineering
SCRUM Master
Sprint Planning
allocatedTo =
End User
Product Owner
SCRUM Master
Other Agile Team Roles
Sprint Backlog (Prioritized User Stories)
Sprint Execution
allocatedTo =
Product Owner
SCRUM Master
Other Agile Team Roles
Code, build files, test cases, built software, property files, and other sprint related artifacts such as test results stored in code and artifact repositories
Scrum of Scrums (SoS)
allocatedTo =
Release Train Engineer
SCRUM Master
Other Agile Team Roles
Identified impediments and plans to facilitate solving/eliminating those impediments.
Passed Criteria Testing?
No
Yes
Final Program Increment
allocatedTo = Release Train Engineer
Program Increment Review (PIR)
allocatedTo =
Product Management
Product Owner
Release Train Engineer
System Architect / Engineering
Updated Program Feature Backlog
Portfolio Solution Review (PSR)
allocatedTo =
Solution Manager
Product Management
Product Owner
Capability Definitions
Updated Portfolio Capabilities backlog (prioritized list of Capabilities)

Figure 5-13. Agile Development Event Flow

Solution and Program Increment Planning

The *Solution Team* for Improved Flight Data in Small Towers works with architects in *Enterprise Engineering* roles to create a high-level design and an architecture of the overall solution, identifying which products and product improvements will be needed. For each Program Increment, the *Solution Team* works with the *Product Management and Product Owners* for relevant products to identify the capabilities that will be provided at the end of the Epic, and for each product, to further decompose the capabilities into a prioritized list of features.

An overview of the planning activities is provided in Figure 5-14 and described in more detail below.

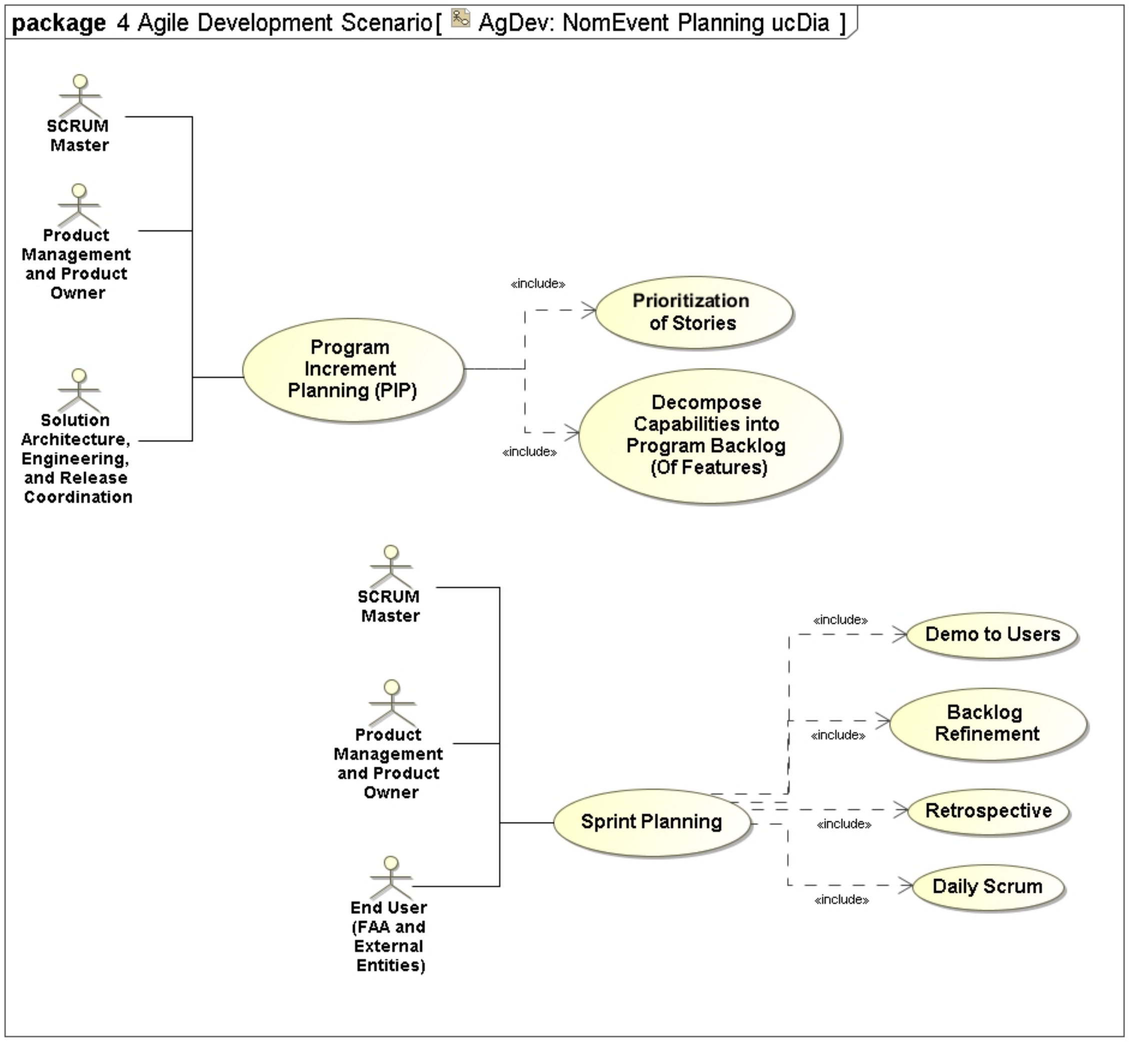

package 4 Agile Development Scenario[AgDev: NomEvent Planning ucDia]
SCRUM Master
Product Management and Product Owner
Solution Architecture, Engineering, and Release Coordination
Program Increment Planning (PIP)
«include»
Prioritization of Stories
«include»
Decompose Capabilities into Program Backlog (Of Features)
SCRUM Master
Product Management and Product Owner
End User (FAA and External Entities)
Sprint Planning
«include»
Demo to Users
«include»
Backlog Refinement
«include»
Retrospective
«include»
Daily Scrum

Figure 5-14. Program Increment Planning and Sprint Planning Processes

Planning the features to be included in an increment is done with input from *End Users* of the system. The prioritized list of features is referred to as a Feature Backlog, and the Scrum of Scrums process is used to coordinate the development of these features across multiple products that must interoperate to provide the feature. An example feature might be "Display flight data retrieved from the Flight Information Service as Electronic Flight Strips in the following format…" The statement of need should be detailed enough to support a Safety Review Board for the Solution. This will lead to additional features or User Stories in the Solution feature backlog or User Stories in the product backlog.

As illustrated in Figure 5-15, an important aspect of the envisioned *Solution Teams* and *Product Teams* is that they include not only software experts, but also representatives from operations, security, and safety. That way safety, operation, and security concerns are identified and mitigated as development proceeds. For example, during a program increment focused on the Display of the Electronic Flight Strips, the safety representative might identify a hazard that would occur if a controller does not realize that data is stale and has not been refreshed properly. A mitigation is proposed that the display should clearly indicate if the data has not been refreshed within the last minute and when the data was last refreshed. The team agrees on the recommended mitigation and implements it. This safety requirement is documented, and an automated test of this mitigation is also implemented to ensure the safety requirement is met.

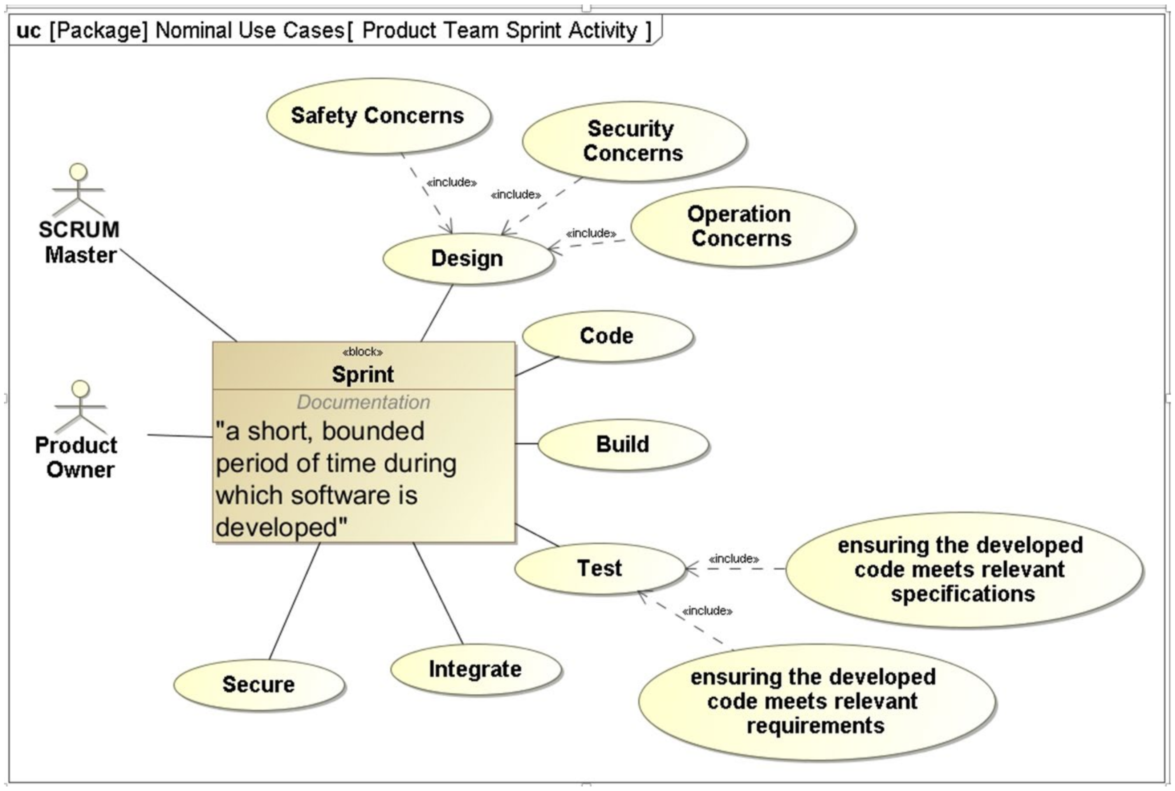

uc [Package] Nominal Use Cases[Product Team Sprint Activity]
Safety Concerns
Security Concerns
Operation Concerns
«include»
«include»
«include»
Design
SCRUM Master
Product Owner
«block»
Sprint
Documentation
"a short, bounded period of time during which software is developed"
Code
Build
Test
«include»
«include»
ensuring the developed code meets relevant specifications
ensuring the developed code meets relevant requirements
Secure
Integrate

Figure 5-15. Addressing Safety, Security, and Operational Concerns

While safety concerns can be surfaced at the *Product Team* level as in the example above, *Solution Management* will also take a more proactive role in making sure safety concerns are mitigated. For each program increment, the Safety Authority performs a safety assessment and needed mitigations that are surfaced are added to the solution level backlog. It is important that these safety assessments be made at the solution level because safety concerns often depend on product use. For example, a feed of aircraft positions that updates every 30 seconds might be perfectly acceptable for the Delay Monitor application but may be totally inadequate for an application intended for use in the Terminal or En Route environment.

The Operations members of the *Product Team* can surface concerns similarly. For example, the operations member of the *Product Team* indicates that the lack of data refresh is something that should also be brought to the attention of operations through their existing monitoring and management software. A requirement for this is added by the *Product Team*. The *Product Team* meets the requirement by notifying the monitoring and management software of the lack of data refresh by using an existing API. The *Product Team* also creates an automated test of this requirement. When it comes to later certification of the software, the successful passing of the automated test will be part of the certification process.

The Security members of the *Product Team* engage in the continuous authorization process in coordination with the Security Authority. When developing the backlog of initial security controls or adjusting them due to an evolving threat environment, the Security members determine which controls are can be inherited from validated and authorized Compute and Platform Layer controls. Inherited controls may also include components of the CI/CD chain such as hardened sidecars or hardened containers from the Platform Layer repository. When changes that might have a security concern are made (e.g., a new interface is added) the *Product Team* works with the security authority to develop control backlog items and automatic continuous authorization artifacts.

Sprint Planning

The *Product Owner* is responsible for creating a prioritized list of User Stories that comprise a Product Backlog. Again, that should be done in consultation with the users, and the User Stories should serve to implement the features in the Feature backlog created by the *Solution Team*. Product backlogs will be created for both the new EFS application and modifications to the existing FIS. User Stories for the EFS application would include the ability to:

- Present flight data.
- Facilitate data entry (including annotations).
- Capture and present flight state changes.
- Allow strip sorting.
- Streamline the flight progress strip transfer between positions.

A backlog has also been defined for the FIS that includes:

- Adding information to the FIS to be used by the EFS application.
- Modifying the API to be used between the FIS and EFS application.

Full decomposition of Features into User Stories does not necessarily have to happen all at once. For example, the first program increment may be to just present flight data but not include the capability of facilitating data entry. It would be imperative going into the first program increment to have a full breakdown of the User Stories based on the feature for presenting flight data. But the User Stories for facilitating data entry could be worked on while the first program increment proceeds.

Non-functional requirement User Stories (e.g., stories around the development and/or execution of training tools and written materials, unit test and integration test, security verification and validation software, monitoring and control, deployment planning, and Operational Test and Evaluation [OT&E] plans) are also defined. It is *Solution Management's* responsibility to make sure that *Product Teams* are assigned to support those non-functional requirements. Some of the non-functional User Stories are based on requirements levied on all solutions. For example, any solution is going to need OT&E planning and deployment planning. Also, some of the non-functional requirements are a result of features in the Feature backlog. Features for displaying the electronic flight strips and for allowing data entry will clearly need training materials to explain the usage of those features.

Sprint Execution

The *Product Teams* focus on the User Stories to implement the application.

The *Product Teams* implement User Stories over an interval of time known as a sprint. The work is facilitated through daily Scrums by a *Scrum Master*. At the end of the sprint there is a sprint review, and any User Stories not meeting the definition of done or requiring revision per the user/management direction will be placed back into the Product Backlog and revisited in the next sprint or Program Increment (PI) development cycle.[17]

During development, safety and security members of the *Software Team* ensure that safety and security mitigations have been implemented and tested.

Another critical aspect of Agile development is getting early user feedback. As the *Software Team* develops and tests the software, it is checked into an artifact repository. The artifacts can be pulled into integration or testing environments where users can test what has been developed and provide feedback. As an example, the requirement that stale data must be clearly indicated has been implemented. The software is pulled into a testing area and prospective users of the EFS application test it. They find that the indicator of the stale data is not obvious enough and is easy to overlook. The users propose a different way of indicating that. The *Product Team* receives the feedback and in a later sprint, the feedback is incorporated.

The *Program Team* will solicit feedback after a program increment has been completed. The program increments are associated with the solution level and consist of a series of sprints during which a set of the features is implemented. Any features not meeting the PI definition of done or requiring revision per the user/management direction will be placed back into the Feature Backlog and revisited in the next PI development cycle.[18]

Feedback in the form of trouble reports is handled in much the same way. The trouble reports are added into the product or solution backlog (depending on the nature of the problem). They are then prioritized for inclusion in a sprint along with the other items contained in the product or solution backlog.

Cross-Product Coordination

Figure 5-16 provides an overview of some of the activities necessary to coordinate the combined efforts of multiple teams to create an overall solution. At high levels, portfolio planning includes identifying the needed products and solutions, and creating a roadmap of their development.

[17] Details of how Scrums are run and various activities such as backlog refinement and sprint reviews are not specific to the FAA and are covered extensively by others. These details are eliminated here for brevity.

[18] Again, the activities during program increments are widely documented elsewhere and have been eliminated for brevity.

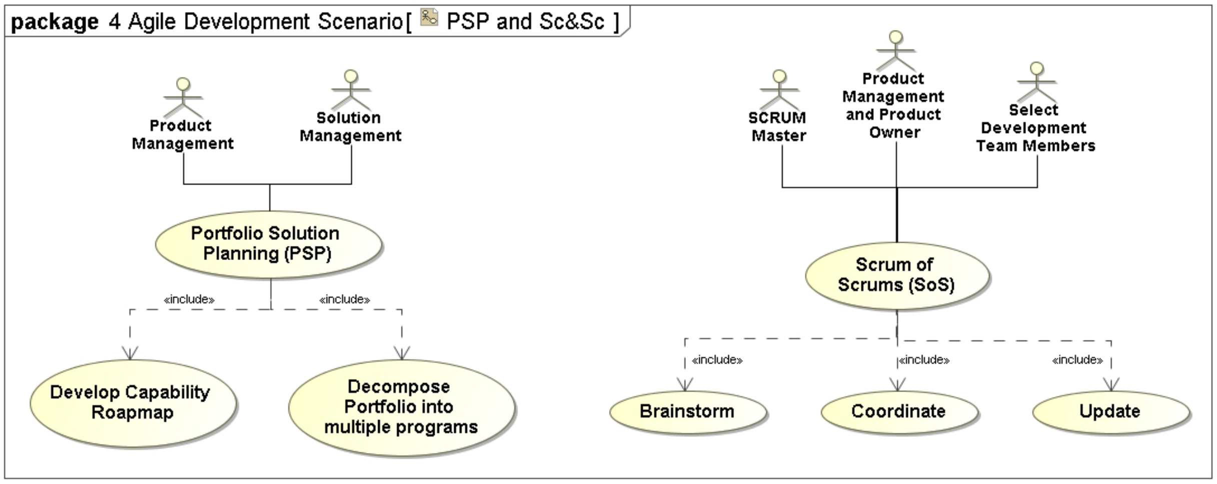

package 4 Agile Development Scenario[PSP and Sc&Sc]
Product Management
Solution Management
Portfolio Solution Planning (PSP)
«include»
«include»
Develop Capability Roapmap
Decompose Portfolio into multiple programs
SCRUM Master
Product Management and Product Owner
Select Development Team Members
Scrum of Scrums (SoS)
«include»
«include»
«include»
Brainstorm
Coordinate
Update

Figure 5-16. Portfolio Solution Planning and Scrum of Scrums Processes

At the solution level, a Scrum of Scrums (or similar Agile ceremony) is held daily or weekly, as needed. This is a short meeting held to align work between *Product Teams*, improve processes and identify impediments. Select representatives of each *Product Team* or the *Product Owner* (no more than two per *Product Team*) should discuss team impediments, risks to achieving the sprint goal or dependencies on other *Product Teams* followed by discovered improvements that can be leveraged by other teams.

Where there are separate *Product Teams* for the EFS application and the FIS, the Scrum of Scrums serves to coordinate when each team's portion of the software will be ready for integration testing.

If members of a solution team need to coordinate work, for example to agree on an API to be used between the FIS and EFS App, they would do so at the Scrum of Scrums. But in such cases, *Product Teams* are encouraged to reach out to each other directly. The Scrum of Scrums aids in resolving conflict and facilitates convergence on design decisions.

<u>Portfolio and Solution Reviews and Feedback</u>

Upon completion of a set of PIs, a portfolio review and/or solution review is held to assess the success of the associated PI implementations. Feedback is again solicited from the user community. Again, items needing revision will be incorporated back into the backlogs and another implementation iteration will proceed.

5.5.5 End State

The end state is the creation of candidate releases of the EFS application and the modified FIS. These candidate releases will then go through the deployment process detailed in the Agile Deployment scenario in Section 5.6.

In addition to a release candidate, the end state also includes the training materials, deployment plans, and OT&E plans that have been developed and that are needed in the starting state for the Agile Deployment Scenario.

By following the Agile methodology, in some sense, the process doesn't end because it is iterated continually until the solution being developed is end-of-life.

5.5.6 Summary

This scenario examined the development of a solution for improved flight data availability and accuracy in small towers.

This scenario addressed the questions posed in section 5.5.1 as follows:

This scenario was used to examine what roles were best filled by government personnel, and what roles could be filled by contractors, by applying the concept of Content Authority, Execution Authority, and Technical Authority.

The discussion of the event flow for this scenario described how solution management will be responsible for ensuring that safety hazards and security vulnerabilities will be surfaced, and mitigations included in the capabilities and features that are included in the products. The discussion also described how *Product Team*s will implement and test these safety and security features as part of the development process.

The event flow discussion also described how training and operational monitoring and management needs would also be identified, implemented, and tested.

This scenario also examined how the work of the *Product Team* working on the EFS application could be coordinated with the work of the *Product Team* working on modifying the FIS, as part of a larger overall solution.

5.6 Deployment of New Capability to NAS Operations

5.6.1 Description

This scenario describes the deployment of capabilities developed using the Agile Development processes described in section 5.4 into the operational NAS.

In this scenario, a new version of the EFS application is deployed for production use. In addition, a new version of the FIS is concurrently deployed that incorporates new features needed by the new EFS application. The Delay Monitor application, already in production use, also relies on the FIS. The new FIS is designed to be backwards compatible with the Delay Monitor application.

This scenario will answer the following questions:

1. How will safety be assured prior to deploying new software for operational use?
2. What testing needs to be done as part of the deployment process?
3. What is the cadence for deploying new software for use in operations?
4. Will we still have staged deployments (such as Key sites used today)?
5. How is customization for each facility's unique operational requirements accomplished?

5.6.2 Roles

Solution Delivery and Sustainment

Solution Management – This role is responsible for the entire lifecycle of the "Improved flight data in small towers" solution, which includes both the EFS application, and the changes required to the FIS to support the EFS application. That responsibility includes guiding the solution through the deployment process (including activities such as OT&E). They are also responsible for making sure products training materials, OT&E planning, and deployment planning are ready prior to commencing deployment. *Solution Management* may need to prioritize problems found during OT&E that are placed into *Solution Team* backlog for correction.

Release Coordination –This is an oversight position that coordinates with key players and has sign-off on whether the EFS application and FIS changes get deployed to the NAS. The decision is made based on operational effectiveness and operational suitability testing that are both part of OT&E and testing that occurred earlier in the Agile development process.[19] One possible model that has been used at the FAA is for *Release Coordination* to have an advisor from the user community at each facility that will use the software. Those advisors then indicate whether they feel the software is ready to be fielded and whether any proposed mitigations of problems found during OT&E are sufficient. This role also considers the input of the *Safety Authority* and the *Security Authority*.

Enterprise Security and Operations

Safety Authority –Responsible for ensuring safety of the EFS application and the FIS changes being deployed and that safety requirements have been met. Must give approval for deployment to occur.

Security Authority –Responsible for ensuring security policies have been followed and security requirements have been met by the EFS solution. Must give approval for deployment of the EFS application and for FIS changes to occur.

Enterprise Monitoring and Management – Since the FIS is used by multiple NAS applications, those other applications should be involved in OT&E to make sure their needs have been met. In this scenario, the Delay Monitor is one such application. *Enterprise Monitoring and Management* must also ensure enterprise-wide monitoring requirements of the EFS application have also been met.

Customer Organizations

Operational Leadership – The leadership responsible for any NAS operational areas impacted by the applications and services to be deployed must sign off on the deployment. In this scenario, that would be leadership in charge of ATCT and TRACON operations, as well as Traffic Flow Management (TFM) since FIS changes could impact them. If there are other users impacted by the FIS changes, they will also need to sign off.

End Users – In this case the end users are the controllers at ATCTs and TRACONs that will be using the EFS application. The end users of the system are needed to participate in OT&E.

Facility Level Technical Support – If the capability to be deployed requires facility level technical support, (and the EFS Application might), then they should be included in OT&E to make sure their monitoring and training needs have been met.

[19] Operational Effectiveness is a measure of how well a system enables a mission to be accomplished by the systems expected users in the expected operational environment. Operational suitability is a measure of whether a system can be put into operation considering various “ilities” such as availability, maintainability, safety, security, and similar factors. A system could be highly operationally effective but so unreliable that it is unsuitable for deployment. Likewise, a system may be highly stable, maintainable, and have great security, but have terrible operational effectiveness because users cannot readily accomplish the mission when using it.

Product Teams

The *Product Teams* may need to help deploy the EFS Application and FIS to the NAS Staging and Integration environment. They may also need to support the smoke testing of the EFS application and FIS and correcting any problems found with the deployment scripts.

The *Product Owner* may need to prioritize problems found during OT&E that are placed into *Product Team* backlog for correction.

5.6.3 Starting State

The starting state for capability deployment into the NAS Operations environment will normally occur at the end of a Program Increment or an Epic. There could be exceptions if a particularly urgent fix is required for security or operational reasons.

The capability to be deployed has been declared a release candidate by the Improved Flight Data for Small Towers *Solution Team*. That capability includes the EFS application and a new version of the FIS with new functionality needed by the EFS application. Both products have gone through successful Development Test and Evaluation (DT&E). User feedback has been solicited and incorporated into the products during development sprints.

The release candidate is made up of artifacts that have been stored in the artifact repository (e.g., executable files, scripts, property files, and Infrastructure as Code [IaC]).

OT&E planning has been performed, and any simulation or data generation capabilities needed to conduct operational evaluations have been prepared by the *Product Team*s during development.

- Training needs have been identified and training materials have been prepared, including:
- Training for the controllers that will be using the EFS application,
- Training for *Enterprise Monitoring and Management*, *Facility Level Technical Support* (if needed), and
- Training for others who are expected to use or interact with the EFS application in some capacity.

If the FIS changes impact how it is monitored, a FIS *Product Team* will prepare training materials that address those changes.

The *Solution Team*, during the development process, has completed deployment planning. Training preparation and deployment plans should be done in parallel with the software development using the same Agile development methods detailed in section 5.4 and 5.5.[20] The expertise required to produce these materials is different and should be reflected in the development teams preparing them.

The process for moving the developed software from the testing environment to and from the NAS Staging and Integration environment to the NAS Operations environment should be as automated as possible. The code needed to move the software from the Staging and Integration environment to the Operational environment must be completed in parallel while developing the

[20] While the same methods may be used, there are alternative development methods that could be used instead. For example, the use of Kanban boards could be used instead of sprints to track the development of the training materials

functional requirements. This code is assumed to be part of the starting state. Furthermore, the code used for application or service deployment into the NAS should be nearly identical to that used for deploying into the NAS Staging and Integration environment where OT&E will take place. That way the deployment code should be robust by the time it is used to deploy an application or service into the NAS Operations environment.

It may be necessary to use different feeds than those used in the NAS Operations environment. In general, the APIs used to ingest those feeds should not change, nor should the underlying code that uses the data. That software should be deployable to the NAS Staging and Integration environment in the same way it will be deployed to the NAS Operations environment. What will change is the source of the data. For example, data may normally be placed onto a topic on a message bus by a live feed. For OT&E purposes, it may be desirable to use recorded data instead. An application for playing back that recorded data would need to be deployed for OT&E. It should play back that data and place it on the same message bus and topic as used by the live data. To put this a different way, the application being tested should not change and should be deployed in the same manner as when deploying to NAS Operations. However, the source of a feed may change, and a different application may be needed to act as the source of that data.

5.6.4 Event Flow

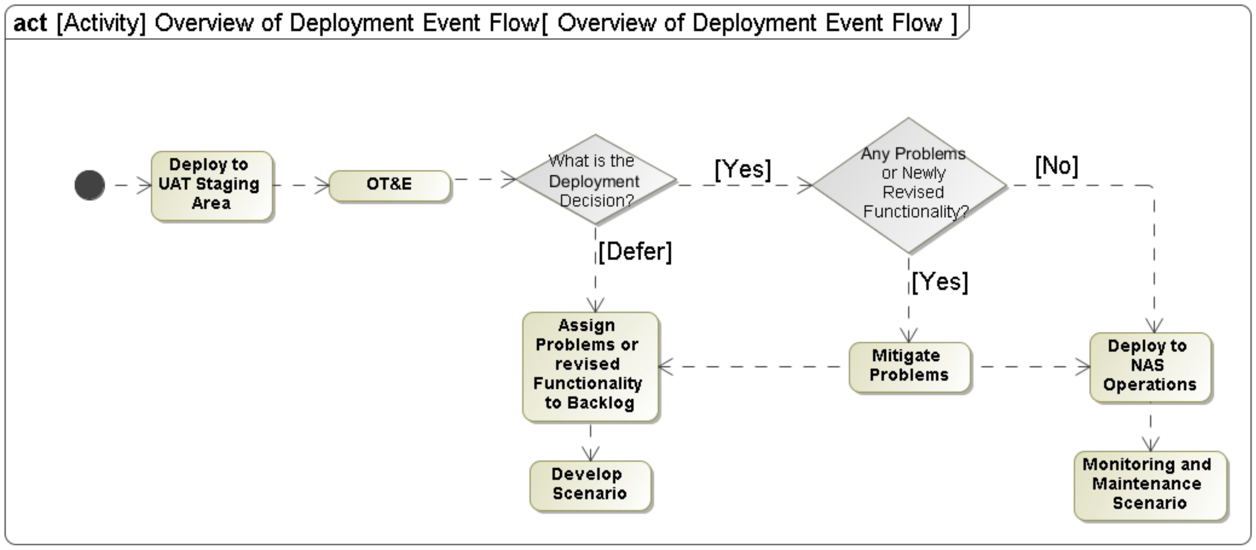

act [Activity] Overview of Deployment Event Flow[Overview of Deployment Event Flow]
Deploy to UAT Staging Area
OT&E
What is the Deployment Decision?
[Yes]
[Defer]
Any Problems or Newly Revised Functionality?
[No]
[Yes]
Assign Problems or revised Functionality to Backlog
Mitigate Problems
Deploy to NAS Operations
Develop Scenario
Monitoring and Maintenance Scenario

Figure 5-17. Overview of Deployment Event Flow

Figure 5-17 is an overview of the event flow for this deployment scenario. Portions of this event flow are further decomposed in Figure 5-18 and Figure 5-19. Those figures go into detail on deployment to the NAS Staging and Integration environment area and OT&E respectively. A quick overview of the overall flow is that deployment is first done to a NAS Staging and Integration environment. This staging area is where the software will be run for OT&E. After successful deployment to this staging area, *End Users,* and technical operations staff (*Enterprise Monitoring and Management* and *Facility Level Technical Support*) perform OT&E. Following the OT&E, *Release Coordination* makes the deployment decision. *Release Coordination* can defer deployment if there are problems that cannot be immediately mitigated. A deferral or mitigation decision will result in problems being assigned to the *Product Team* or *Solution Team* backlog. Either the *Product Owner* or *Solution Team Management* will prioritize the issue and either a *Product Team* or *Solution Team* will deal with that backlog per the Agile Development Scenario (section 5.5). If no problems were found, or *Release Coordination* made the decision to deploy anyway while mitigating any problems, then deployment to NAS operations occurs. That leads to the Monitoring and Maintenance Scenario (section 5.7).

One of the AES goals is to accelerate getting new software to the field. However, given the important safety and security aspects of operating the NAS, continuous deployment of releases to the operational NAS are not expected.[21] Instead the goal is to release frequently but in a planned and coordinated manner. In today's NAS, facilities have up to 180 days to install releases. Unless that rule is modified, it might put a limit on the release cadence to no more than twice a year.[22] CI/CD is still practiced, but only from the Development environment into the Test environment, or possibly into the Staging environment, if that can be done without impacting OT&E. Those concepts are shown in Figure 4-3.

More detailed information on parts of this event flow follows.

Deployment to NAS Staging and Integration Environment

Automated tools are used to deploy the EFS application and the changed version of the FIS into the NAS Staging and Integration environment where OT&E will take place. Once it has been deployed, preliminary smoke testing should be done to ensure that OT&E can proceed. Much of the automated testing that was created during the development of the EFS application and FIS service can be used. Figure 5-18 illustrates this part of the event flow and its associated roles.

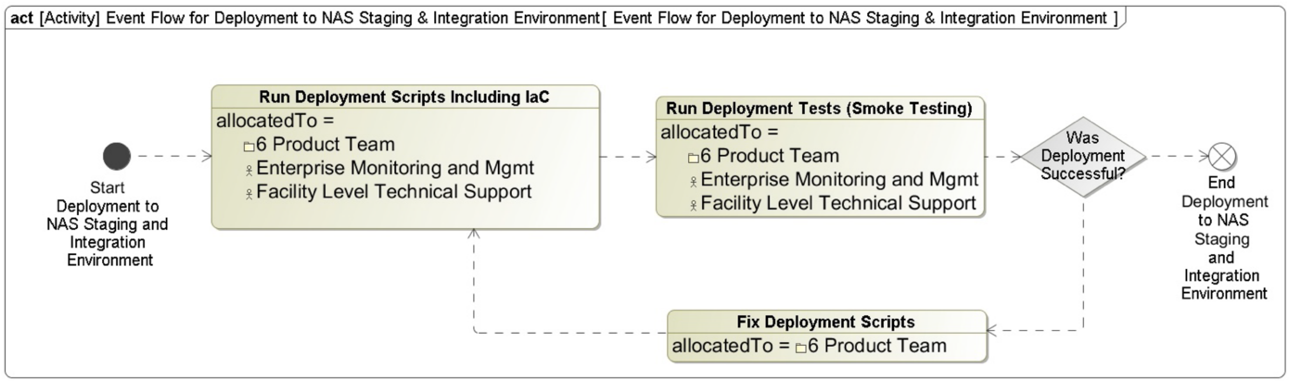

act [Activity] Event Flow for Deployment to NAS Staging & Integration Environment[Event Flow for Deployment to NAS Staging & Integration Environment]
Start Deployment to NAS Staging and Integration Environment
Run Deployment Scripts Including IaC
allocatedTo =
6 Product Team
Enterprise Monitoring and Mgmt
Facility Level Technical Support
Run Deployment Tests (Smoke Testing)
allocatedTo =
6 Product Team
Enterprise Monitoring and Mgmt
Facility Level Technical Support
Was Deployment Successful?
End Deployment to NAS Staging and Integration Environment
Fix Deployment Scripts
allocatedTo = 6 Product Team

Figure 5-18. Event Flow for Deployment to NAS Staging and Integration Environment

Operational Test and Evaluation

Before the software is deployed to NAS Operations, *End Users* and technical operations staff (*Enterprise Monitoring and Management* and *Facility Level Technical Support*) perform OT&E using the release candidate. This test should occur in as operationally realistic environment as possible. OT&E for a release candidate that interoperates with other software should use the versions of the other software that are expected to be deployed with the release candidate. In this

[21] Continuous Integration/Continuous Deployment is an important aspect of DevSecOps. This will still be occurring, but the deployment will be occurring to an artifact repository. Software can then be pulled from the artifact repository into staging environments. How often it is pulled will depend on the purpose of the staging environment. For example, an integration environment might have the software pulled daily. But a staging environment used for training purposes might pull the latest software only after the software has passed integration tests.

[22] The idea of coordinating the release schedule across different applications has been discussed. This could simplify the release process at facilities and enable a somewhat faster cadence.

case, the EFS application gets its information from the FIS. But the FIS also feeds the Delay Monitor application. Therefore, testing must also be performed to ensure that the new version of FIS does not impact the Delay Monitor application. Ideally, this would be an automated test, but may also need to be included in OT&E. Also, the FIS must get its information from other systems. Those other systems, or suitable simulations of those systems, will need to be running as well to provide the necessary data. The objective is to digitally twin the NAS Operations environment as far as practical. This includes using real data feeds, if possible, or recorded or simulated data feeds. The purpose is to make sure the software functions as designed when integrated with the other software. Likewise, the adaptation data used during OT&E should be very close to that used to run the EFS application and FIS upon deployment. Figure 5-19 details the event flow and roles for OT&E.

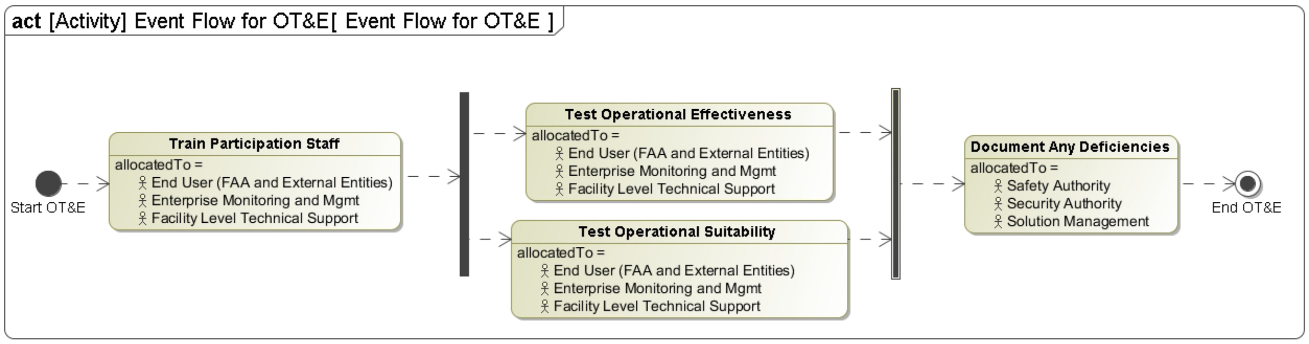

act [Activity] Event Flow for OT&E[Event Flow for OT&E]
Start OT&E
Train Participation Staff
allocatedTo =
End User (FAA and External Entities)
Enterprise Monitoring and Mgmt
Facility Level Technical Support
Test Operational Effectiveness
allocatedTo =
End User (FAA and External Entities)
Enterprise Monitoring and Mgmt
Facility Level Technical Support
Test Operational Suitability
allocatedTo =
End User (FAA and External Entities)
Enterprise Monitoring and Mgmt
Facility Level Technical Support
Document Any Deficiencies
allocatedTo =
Safety Authority
Security Authority
Solution Management
End OT&E

Figure 5-19. Event Flow for OT&E

User feedback on any new features that have been implemented should have already been received as part of the Agile development process before the software became a release candidate. Prior to OT&E execution, all users taking part in the OT&E should have been trained using the training materials assumed in the starting state.

Testing during OT&E should include those functions needed by *Enterprise Monitoring and Management*. A DevSecOps development process is assumed for the EFS application and FIS changes. Therefore, *Enterprise Monitoring and Management* should have been part of the development process and ensured that their needs for monitoring the application are met. If *Facility Level Technical Support* has needs, they should also have been part of the DevSecOps process.

As more of the software making up the NAS gets hosted in the cloud, it may no longer be necessary to run OT&E at the William J. Hughes Technical Center in New Jersey. And, even in the interim, it may be possible for those people running parts of the NAS operational software that are hosted in the cloud to participate from elsewhere. This has the potential to save travel costs and time when people can participate from their usual facility. These savings will need to be balanced against the ease of getting feedback when participants are not all in one location.

Safety requirement development and any safety concern mitigations should be part of the Agile development process as detailed in section 5.4. However, as part of the release candidate's safety certification, all safety mitigations should be verified before deployment. Any mitigations requiring user interactions should be verified during OT&E. Since verifying safety mitigations may require off-nominal conditions, it may be necessary to build scenarios for OT&E that are specifically designed to exercise the mitigations. Verifying and documenting that safety concerns have been met is necessary for the release candidate to be safety certified for operational use in the NAS. Even non-safety critical systems may have safety mitigations and need safety certification. Final testing and safety certification is done by those delegated by the *Safety Authority* as part of OT&E.

While the OT&E is an important double check to make sure the release candidate is functioning as intended, if the Agile development process is working properly, significant problems should rarely surface during this testing.

Deployment to Operations

The nominal case is that the system is approved for release and operational use. However, there will be times where faults are noted. When that occurs, the severity of the fault should be considered when choosing one of the following options:

- Delay the release. The noted issues would go into onto one of the backlogs depending on the fault. Most such faults would go into the *Product Team* backlog or the *Solution Team* backlog. As usual, the *Product Owner* is responsible for prioritizing the backlog. Since the issues are preventing release, the *Product Owner* would classify them as high priority and assign them to one or more *Product Teams* for resolution.
- Deploy, but turn off the features that are causing problems. It is good practice to design features in a way that they can be turned off with software switches (e.g., feature flags), if necessary. Sometimes it is highly desirable to continue with a release but to turn off problematic features. The backlog would be updated to include fixing the problematic features.
- Deploy, but still put the user issues into the backlog to be fixed (subject to prioritization) in next version.

Since software may be developed and released in the cloud, some changes to how adaptation data is handled may be needed. Instead of having a national adaptation sent out for customization by individual facilities, all adaptation for a software application would be available via a national adaptation data service. The need for customization by facilities could be handled by using a facility tag in the adaptation database that would control what local adaptation a facility should use. The same is true about software configuration parameters, which can have national, facility, role, and even individual or workstation customization. Configuration parameters of any level could be stored in the cloud, and then used based on tags that control what is used where and by whom. Facilities would still be responsible for any customization of the adaptation needed. The difference would be in how those customizations are stored and accessed.

5.6.5 End State

The end state is the introduction of software into the NAS operational environment. By using adaptation data, the software has been configured specifically for operational environment use for a specific set of users. Depending on the deployment strategy [12], multiple versions of services can coexist in the cloud for various communities to use, including key sites. The ability to support multiple versions of services aids in legacy transition and acceptance by the user community.

5.6.6 Summary

This scenario addressed capabilities deployed into the operational NAS that were developed using the Agile Development processes described in section 5.5.

This scenario addressed the questions posed in section 5.6.1 as follows:

Throughout the development of the software, both safety and security concerns have been evaluated and addressed. Regarding how safety will be assured prior to deploying new software for operational use, final testing and safety certification is done by those delegated by the *Safety Authority* as part of OT&E. Given the important safety and security aspects of operating the NAS, CI/CD of releases to the operational NAS are not expected. CI/CD is still used but only from the Development environment into the Test environment, or possibly into the Staging environment if that can be done without impacting OT&E. Final approval from both Safety and Security Authorities are required for deployment into the operational environment.

Regarding what testing needs to be done as part of the deployment process, the principal criteria for transitioning new/updated software to an operational environment is successful completion of OT&E. Prior to performing OT&E, a successful DT&E has been performed in the Test Integration environment. Preparation for the OT&E event includes the development of test procedures and training materials along with successful transition of the software from the Test environment to the Staging environment. Required data and feeds have been introduced into the staging environment that mimic the data and feeds available in the operational environment. A smoke test is performed to verify that successful transition into the Staging environment has occurred.

In today's NAS, facilities have up to 180 days to install releases. Unless this rule is modified, this might put a limit on the cadence of releases to no more than twice a year.

With the advent of the cloud environment, per Figure 4-3, multiple staged sub-environment enclaves representing specific environmental requirements of key sites are supported for both the Staging and Operational environments. To manage the customization of a site's environment, the use of a National Adaptation Data Management capability is proposed that facilitates configuration of each NAS site based on the site's unique operational requirements.

5.7 Monitoring and Management Responsibilities

5.7.1 Description/Background

This scenario focuses on Monitoring and Management of the automation applications, microservices, message bus, and APIs in the NAS production environment.

The objective is to notionally prioritize the services that need to be monitored and understand the roles and responsibilities of *Facility Level Technical Support*, *Enterprise Monitoring and Management*, and *Operational Leadership*.

Monitoring and management activities occur from the beginning of software development to the production environment. As NAS automation applications and mission services transition to a cloud-based microservices architecture, monitoring and management functions will be different than how they are performed today.

This scenario addresses the following question:

- What business boundaries, responsibilities, and associated access privileges within the technical operations organization are needed to adapt to future NAS automation and services as they are migrated to the cloud?

5.7.2 Roles

In reference to Section 3, each of the following play a significant role to monitor and manage NAS automation in a cloud production platform:

- *Facility Level Technical Support*
- *Enterprise Monitoring and Management*
- *Operational Leadership*

Figure 5-20 notionally defines *Facility Level Technical Support* responsibility in the form of read and write access privileges for monitoring and anomaly alert detection response of the automation application and service dependencies. This role will have limited access privileges to reboot (stop/start) virtual server instances and access their local physical networks and virtual networks to modify configuration settings (start/stop/modify).

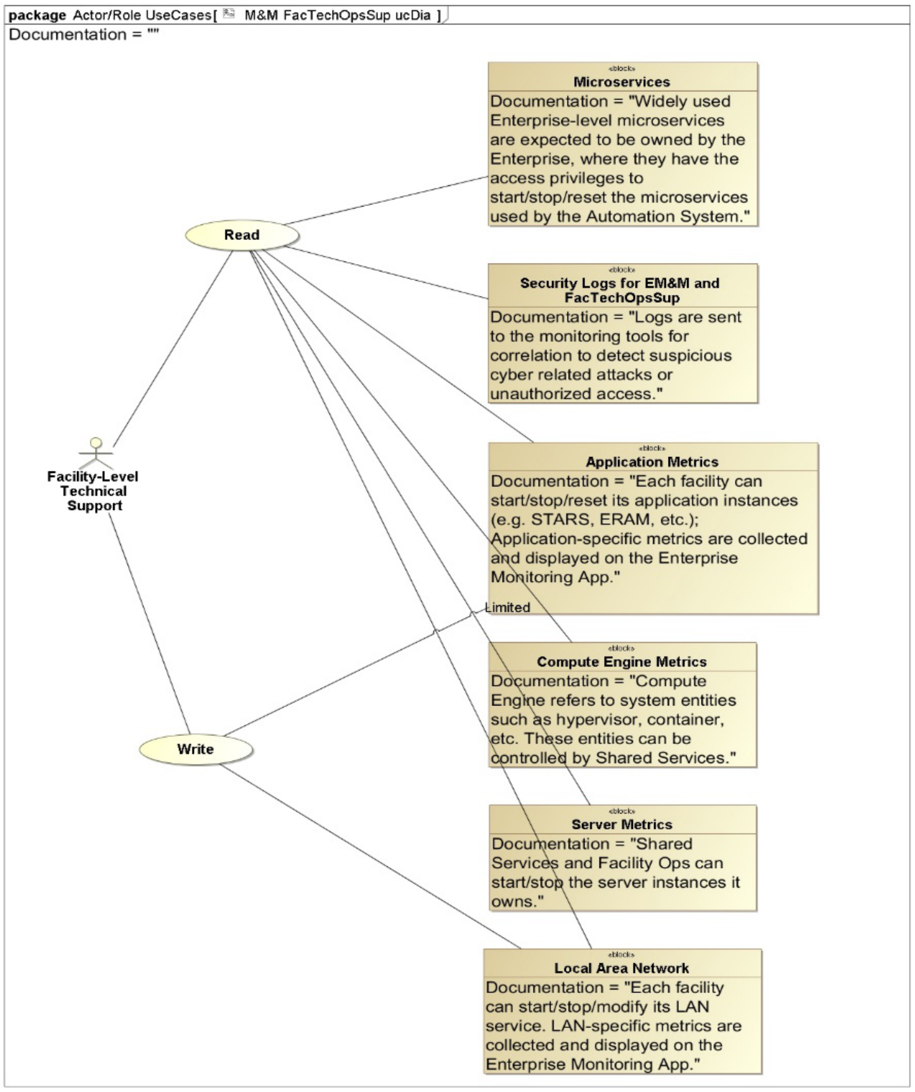

package Actor/Role UseCases[M&M FacTechOpsSup ucDia]
Documentation = ""
«block»
Microservices
Documentation = "Widely used Enterprise-level microservices are expected to be owned by the Enterprise, where they have the access privileges to start/stop/reset the microservices used by the Automation System."
Read
«block»
Security Logs for EM&M and FacTechOpsSup
Documentation = "Logs are sent to the monitoring tools for correlation to detect suspicious cyber related attacks or unauthorized access."
Facility-Level Technical Support
«block»
Application Metrics
Documentation = "Each facility can start/stop/reset its application instances (e.g. STARS, ERAM, etc.); Application-specific metrics are collected and displayed on the Enterprise Monitoring App."
Limited
«block»
Compute Engine Metrics
Documentation = "Compute Engine refers to system entities such as hypervisor, container, etc. These entities can be controlled by Shared Services."
Write
«block»
Server Metrics
Documentation = "Shared Services and Facility Ops can start/stop the server instances it owns."
«block»
Local Area Network
Documentation = "Each facility can start/stop/modify its LAN service. LAN-specific metrics are collected and displayed on the Enterprise Monitoring App."

Figure 5-20. Facility Level Support Access Privileges

Figure 5-21 illustrates the *Enterprise Monitoring and Management* roles based on notional read and write access privileges of the end automation applications and services. In alignment with the RA, the author of this scenario anticipates that *Enterprise Monitoring and Management* will have more access privileges than the local facility technical support level to effectively respond to automation anomalies, such as errors, latency, oversubscriptions, and outage situations.

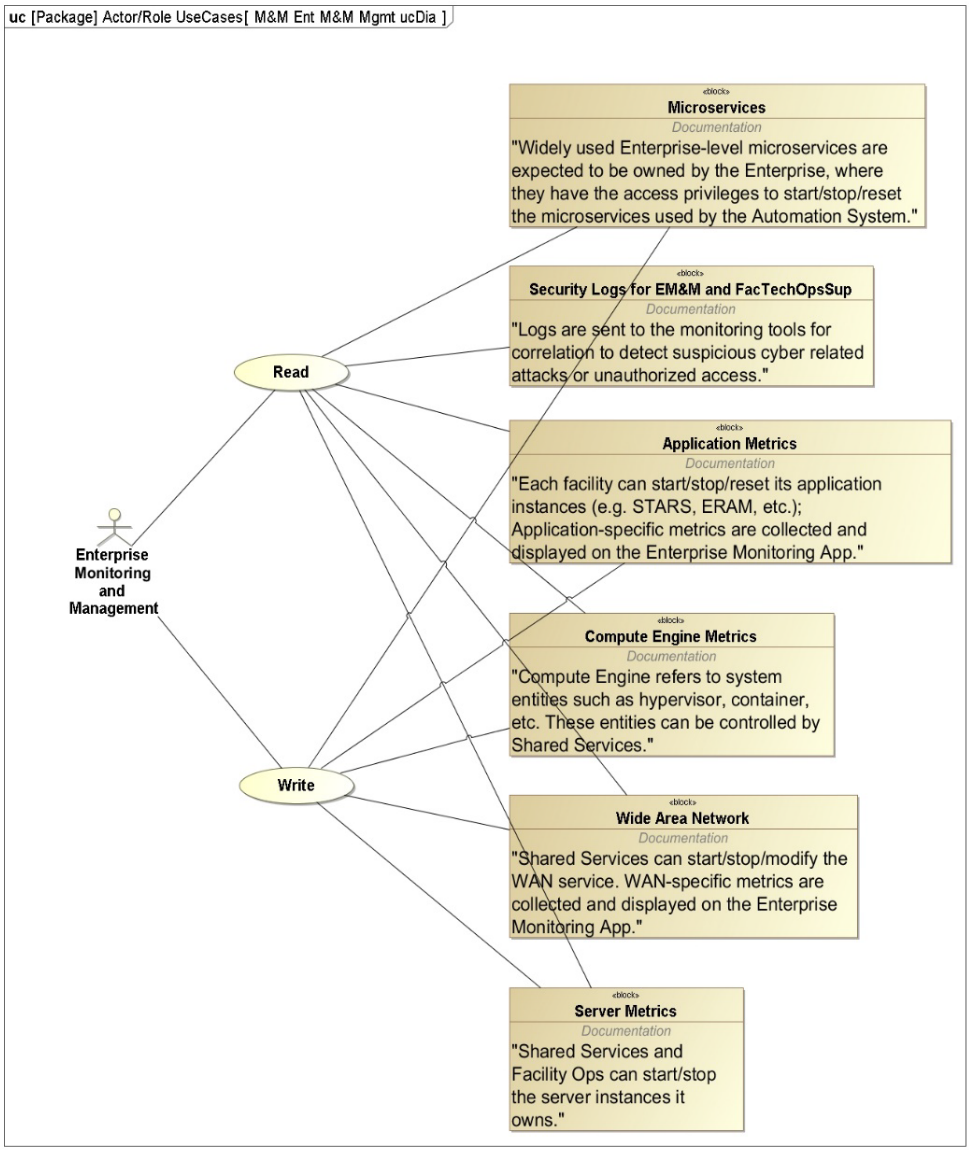

uc [Package] Actor/Role UseCases[M&M Ent M&M Mgmt ucDia]
Enterprise Monitoring and Management
Read
Write
«block» Microservices
Documentation
"Widely used Enterprise-level microservices are expected to be owned by the Enterprise, where they have the access privileges to start/stop/reset the microservices used by the Automation System."
«block» Security Logs for EM&M and FacTechOpsSup
Documentation
"Logs are sent to the monitoring tools for correlation to detect suspicious cyber related attacks or unauthorized access."
«block» Application Metrics
Documentation
"Each facility can start/stop/reset its application instances (e.g. STARS, ERAM, etc.); Application-specific metrics are collected and displayed on the Enterprise Monitoring App."
«block» Compute Engine Metrics
Documentation
"Compute Engine refers to system entities such as hypervisor, container, etc. These entities can be controlled by Shared Services."
«block» Wide Area Network
Documentation
"Shared Services can start/stop/modify the WAN service. WAN-specific metrics are collected and displayed on the Enterprise Monitoring App."
«block» Server Metrics
Documentation
"Shared Services and Facility Ops can start/stop the server instances it owns."

Figure 5-21. Enterprise Monitoring and Management Access Privileges

5.7.3 Starting State

The scenario begins with a state in which operations staff have visibility into application, microservice dependencies, message bus, and underlying cloud resource health status. Monitoring tools and services have been created and deployed for use to allow effective and timely response to performance and operational health status changes. Those tools can detect anomalous behavior, generating alarms based on threshold settings, and providing logs and statistical metrics to aide in troubleshooting. The tools provide situational awareness of the NAS automation system that is relied on by ATC end-user community. The ConUse authors envision that cloud-based monitoring services will be accessed through a dashboard that is customizable to the *Enterprise Monitoring and Management* remote operations centers and local facility technical support organizations. Monitoring services and tools for NAS automation will be redundant, highly available, and out-of-band (OOB) to display status information in the form of logs, metrics, and alerts unitarily, but customized to the operations to observe the applications, services, and underlying resources that are running in the cloud.

5.7.4 Event Flow

5.7.4.1 Nominal Event Flow

A cloud-based monitoring dashboard will be configured to present the health status of the Mission Applications, Mission Services, Platform Layer services, and cloud resources. The Enterprise and *Facility Level Technical Support* role will monitor the dashboard 24/7. The nominal monitoring and management scenario diagram is shown in Figure 5-22.

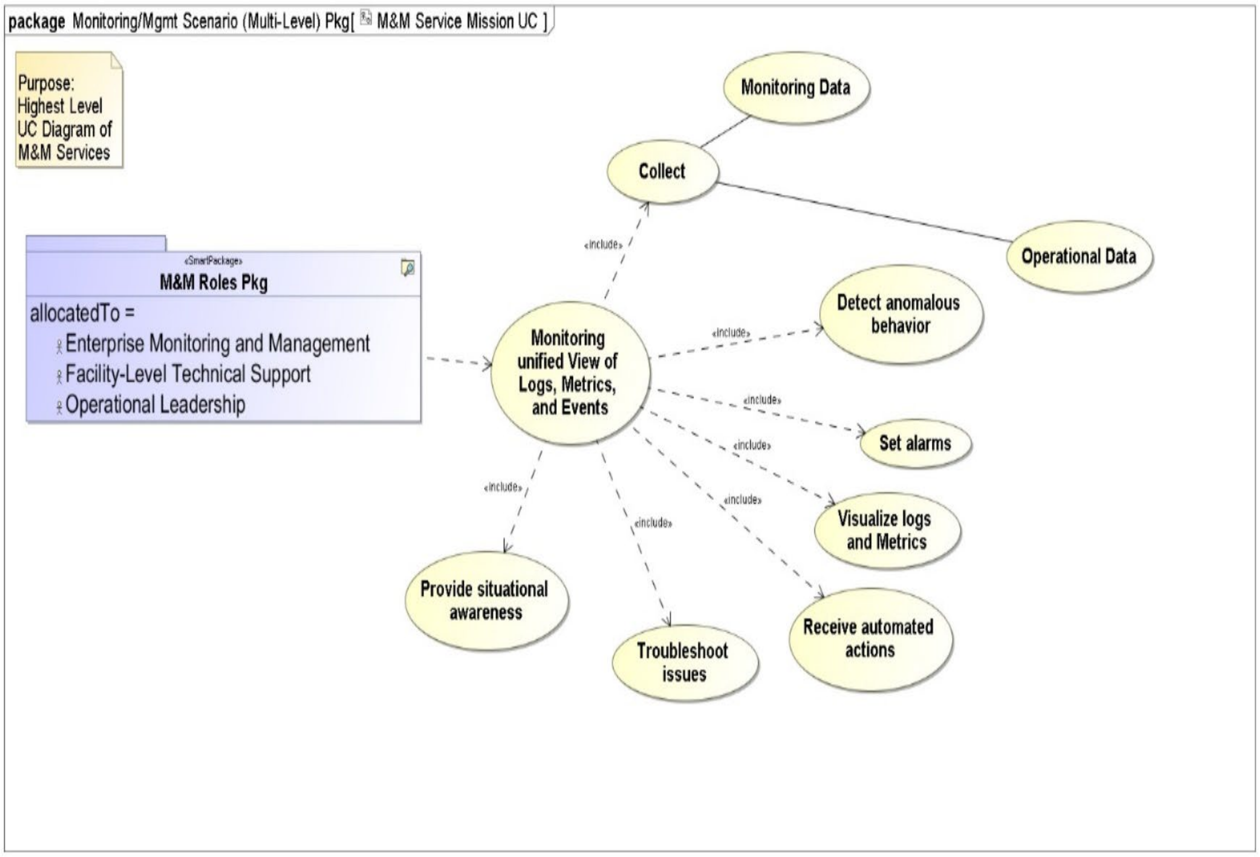

package Monitoring/Mgmt Scenario (Multi-Level) Pkg[M&M Service Mission UC]
Purpose: Highest Level UC Diagram of M&M Services
«SmartPackage»
M&M Roles Pkg
allocatedTo =
Enterprise Monitoring and Management
Facility-Level Technical Support
Operational Leadership
Monitoring unified View of Logs, Metrics, and Events
«include»
Collect
Monitoring Data
Operational Data
Detect anomalous behavior
Set alarms
Visualize logs and Metrics
Receive automated actions
Troubleshoot issues
Provide situational awareness

Figure 5-22. Monitoring and Management Scenario

5.7.4.2 Off-Nominal Event Flow

When abnormal behaviors, faults and/or alerts are displayed on the monitoring dashboard it initiates a sequence of events between *Facility Level Technical Support*, *Operational Leadership* and *Enterprise Monitoring and Management*. Figure 5-23 depicts the sequence of events based on an event that alerts on an overutilization of the virtual server's Central Processing Unit (vCPU) that host the automation application when a new FIS software release has been automatically deployed to the production cloud.

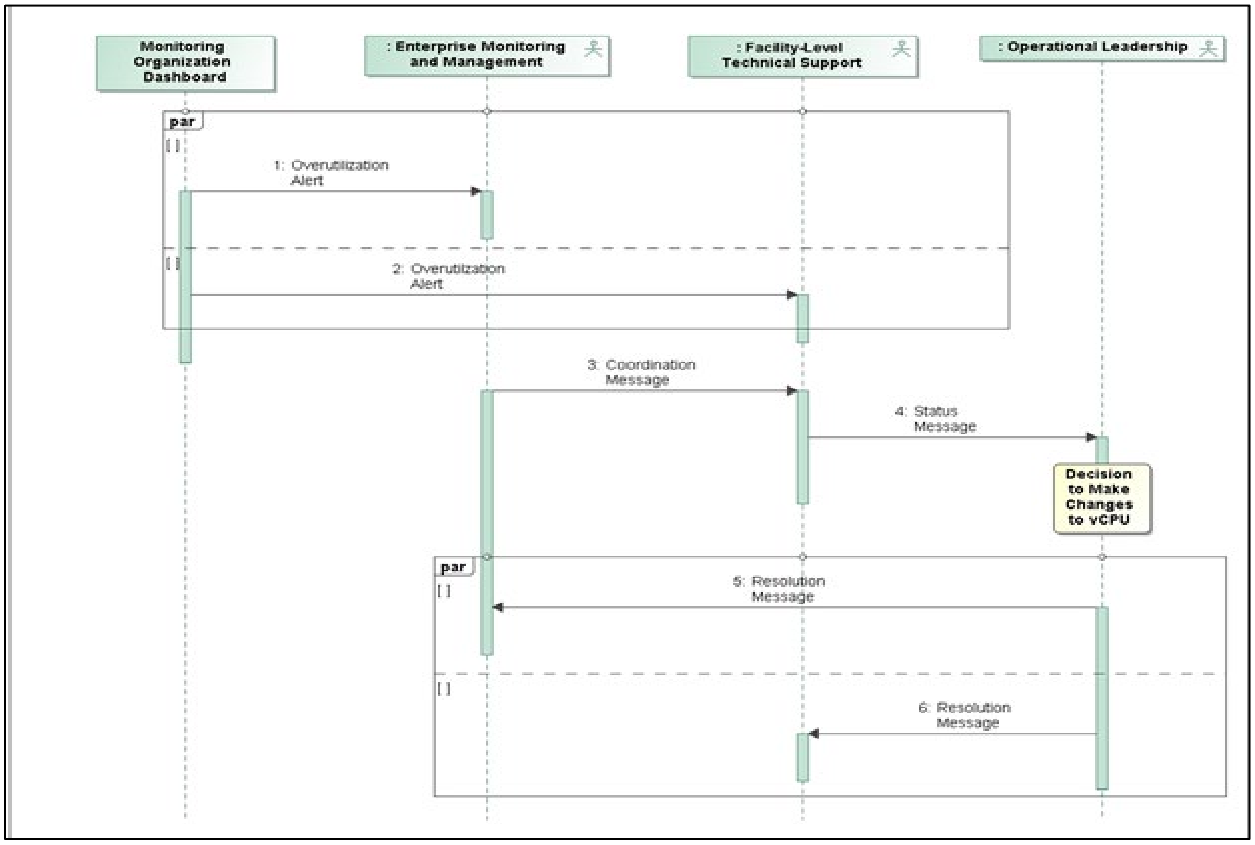


Figure 5-23. Abnormal Conditions Event Sequence

The following sequence of events occurs as depicted in Figure 5-23.

1. Alert triggered after a new FIS software release has been deployed in the production automation environment.
2. The alert triggered that vCPU utilization exceeded the threshold of 90 percent.
3. *Enterprise Monitoring and Management* coordinates with *Facility Technical Support* with a solution to increase vCPUs to resolve the issue.
4. *Facility Level Technical Support* gets the authorization from the *Operational Leadership* to enable *Enterprise Monitor and Management* to proceed in implementing the solution to increase the vCPU from four to eight cores.
5. The solution end state delivers a stable server with acceptable vCPU use that the automation application resides on.

5.7.4.3 Parallel Activities

When an abnormal event that can potentially impact and degrade NAS ATC automation operations occurs, *Operational Leadership* is notified by the local *Facility Level Technical Support*. Once the issue that caused the off-nominal event has been isolated and a solution to the problem has been determined, *Operational Leadership* authorizes *Enterprise Monitoring and Management* to apply the solution to fix the issue and restore the automation service to a normal state. Close monitoring at the Facility and Enterprise Level will be required to ensure the applied fix corrected the problem.

5.7.5 End State

Enterprise Monitoring and Management and *Facility Level Technical Support* under normal conditions will likely access the monitoring dashboard and tools deployed in the cloud. The dashboard will provide unified metrics, logs, and alerts to ensure all roles are observing the same information, situational awareness, and health status of the automation environment. The dashboard will be customizable to the operations role with responsibility at the *Enterprise Monitoring and Management* and *Facility Level Technical Support* levels.

Under abnormal conditions it's presumed that the *Enterprise Monitoring and Management* functions will have the access control (read/write) privileges to make changes to automation applications, and cloud resources and services (e.g., containers, microservices, message bus, etc.). *Facility Level Technical Support* could have access privileges with limited read/write access to only be able to stop/start virtual servers and network cloud compute services.

Enterprise Monitoring and Management should have read/write access privileges to isolate, troubleshoot, modify, and restore the automation applications, services, and cloud resources to a normal operational state. The authors of this ConUse anticipate that *Facility Level Technical Support*, both local and remote, will have limited access (start/stop) to the automation application server in the cloud resource that supports their facility operations. *Enterprise Monitoring and Management* will coordinate with *Facility Level Technical Support* when anomalous behaviors are detected prior to taking corrective action to resolve the automation application issue in the cloud. *Operation Leadership* knowledge and situation awareness of the airspace will be required to give the authorization to *Enterprise Monitoring and Management* to fix and restore the NAS automation service from a degraded to a normal state.

5.7.6 Summary

The scenario establishes a baseline of monitoring, maintenance, and response action responsibilities and business boundaries at the enterprise and local facility levels.

This scenario addressed the question posed in section 5.7.1 as follows:

The question on business boundaries, responsibilities, and associated access privileges were addressed by describing the privileges that *Facility Level Technical Support* and *Enterprise Monitoring and Management* roles would have to view the status and performance metrics of applications and services (read access) and to control and configure those applications and services (write access).

5.8 Cybersecurity Operations

5.8.1 Description

This scenario focuses on the detection and remediation of the Delay Monitor application after an exploitation is detected.

The objective is to understand the roles and responsibilities of *Cybersecurity Defensive Operations, Security Authority, Safety Authority, Solution/Product Teams, Operational Leadership,* and *Enterprise Monitoring and Management* in various branches of a cybersecurity incident.

As NAS automation applications and mission services transition to a layered cloud-based microservices architecture, cybersecurity incident detection and response functions will be different than how they are performed today.

This scenario addresses the following questions:

- How will the different roles work together to remediate a cybersecurity incident given no vertical program integration?
- How will Zero Trust be used during an incident?
- How many Security Operations Centers (SOCs) are needed for the NAS? Will some mission teams have one? Will RA layers have one?
- How will operational risk be managed if automation applications have been exploited?
- How will other teams be notified and respond to the risk of contagion and degraded operations?

5.8.2 Assumptions

We make the following assumptions:

- Zero Trust is fully implemented in the RA.
    - All containers have hardened sidecars for configuration and communication mediation.
- Any security controls as a service from the CSP are fully integrated into the environment e.g., Web Application Firewall (WAF) logs.
- The part of the Delay Monitor application that is compromised is deployed and operated centrally on the cloud platform.

5.8.3 Roles

In reference to Section 3, each of the following play a significant role to monitor and manage NAS automation in a cloud production platform:

- *Cybersecurity Defensive Operations* – Actively defends the NAS against cybersecurity threats. This includes operations centers that monitor NAS software and networks respond to incidents. Investigates potential attacks, data breaches and system compromises. This includes all “SOC-like” organizations.
- *Security Authority* – Sets security policy and works with *Product Teams* for continuous authorization. Works with Platform *Enterprise Monitoring and Management* Zero Trust Operational Maintenance for Zero Trust policy decision and enforcement point configuration.
- *Operational Leadership* – This role is comprised of FAA managers responsible for different areas of NAS mission operations (e.g., En Route and Terminal ATC), system operations, aeronautical information) as well as technical operations at facilities.
- *Solution/Product Teams* – Develop mission applications and work with *Security Authority* and *Enterprise Monitoring and Management* to configure and deploy applications.

- *Enterprise Monitoring and Management* – Deploys and maintains services,

5.8.4 Starting State

The scenario begins with a state in which *Cybersecurity Defensive Operations* (CDO) detects an anomaly in the Delay Monitor application based on Zero Trust violations and begins to investigate. Soon afterward a 0-day exploit for a library used to build the application is announced. The Delay Monitor application is efficiency critical, but not safety critical and is centralized, running operationally on two FedRAMP High government clouds in the Mission Essential Operating Environment.

5.8.5 Event Flow

The event flow is shown in the below figures as follows:

- Figure 5-25 shows the beginning actions,
- Figure 5-26 expands upon process A depicted in Figure 5-25, a thread where a permanent fix is implemented,
- Figure 5-27 expands upon process B depicted in Figure 5-25 where temporary mitigations are implemented, and
- Figure 5-28 expands upon process C depicted in Figure 5-25 which shows wrap up activities.

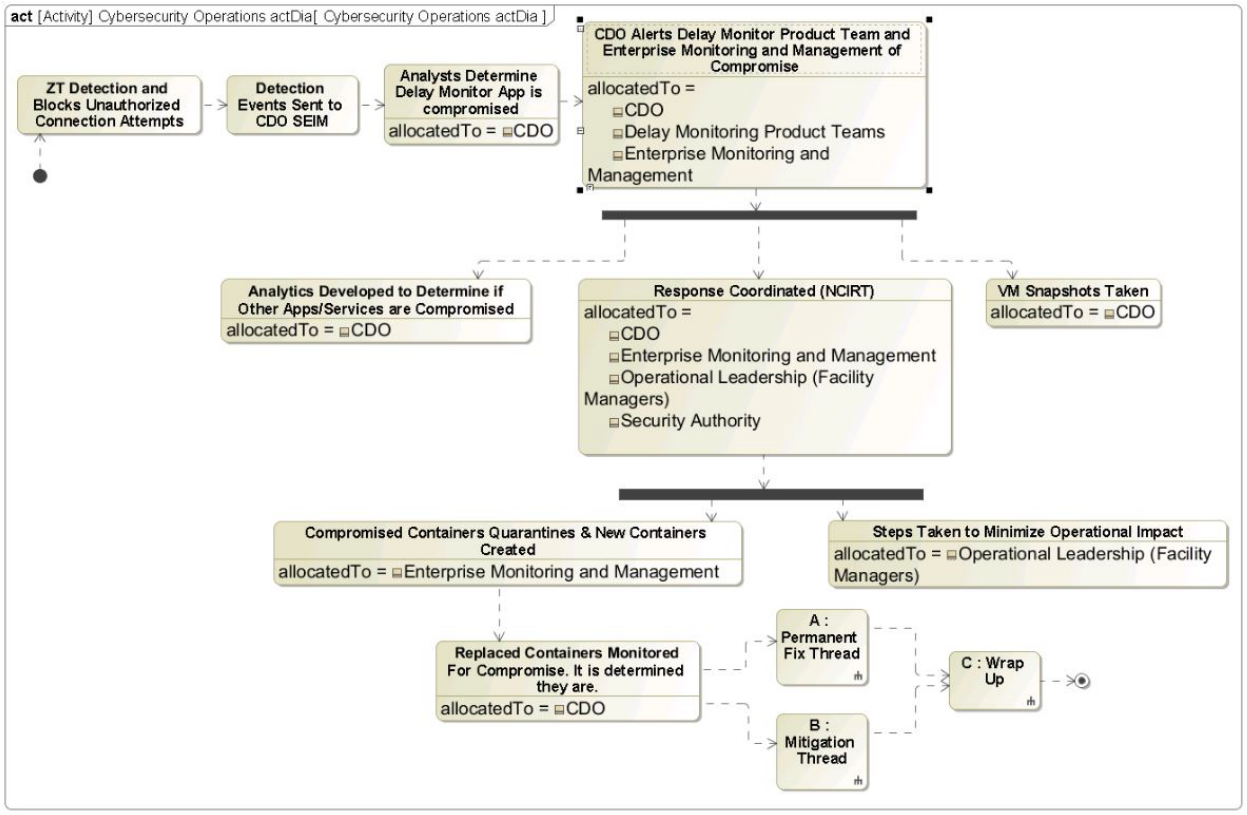

act [Activity] Cybersecurity Operations actDia[Cybersecurity Operations actDia]
ZT Detection and Blocks Unauthorized Connection Attempts
Detection Events Sent to CDO SEIM
Analysts Determine Delay Monitor App is compromised
allocatedTo = CDO
CDO Alerts Delay Monitor Product Team and Enterprise Monitoring and Management of Compromise
allocatedTo = CDO, Delay Monitoring Product Teams, Enterprise Monitoring and Management
Analytics Developed to Determine if Other Apps/Services are Compromised
allocatedTo = CDO
Response Coordinated (NCIRT)
allocatedTo = CDO, Enterprise Monitoring and Management, Operational Leadership (Facility Managers), Security Authority
VM Snapshots Taken
allocatedTo = CDO
Compromised Containers Quarantines & New Containers Created
allocatedTo = Enterprise Monitoring and Management
Steps Taken to Minimize Operational Impact
allocatedTo = Operational Leadership (Facility Managers)
Replaced Containers Monitored For Compromise. It is determined they are.
allocatedTo = CDO
A : Permanent Fix Thread
B : Mitigation Thread
C : Wrap Up

Figure 5-24. Security Event Flow: Initial Detection and Quarantine

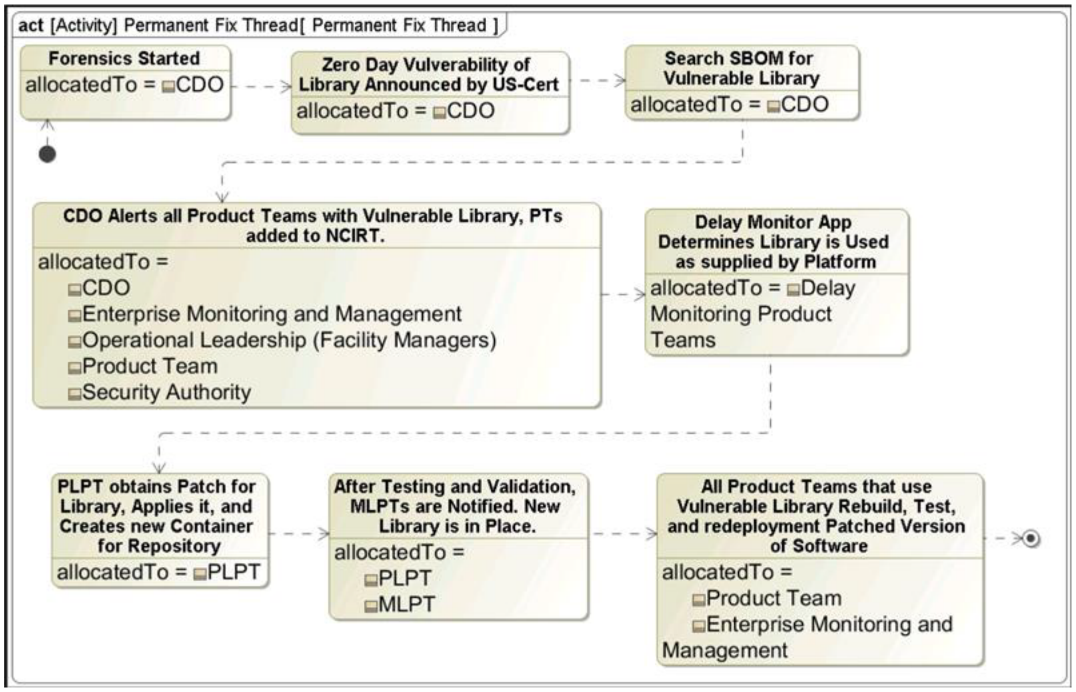

act [Activity] Permanent Fix Thread[Permanent Fix Thread]
Forensics Started
allocatedTo = CDO
Zero Day Vulverability of Library Announced by US-Cert
allocatedTo = CDO
Search SBOM for Vulnerable Library
allocatedTo = CDO
CDO Alerts all Product Teams with Vulnerable Library, PTs added to NCIRT.
allocatedTo = CDO, Enterprise Monitoring and Management, Operational Leadership (Facility Managers), Product Team, Security Authority
Delay Monitor App Determines Library is Used as supplied by Platform
allocatedTo = Delay Monitoring Product Teams
PLPT obtains Patch for Library, Applies it, and Creates new Container for Repository
allocatedTo = PLPT
After Testing and Validation, MLPTs are Notified. New Library is in Place.
allocatedTo = PLPT, MLPT
All Product Teams that use Vulnerable Library Rebuild, Test, and redeployment Patched Version of Software
allocatedTo = Product Team, Enterprise Monitoring and Management

Figure 5-25. Part A of Security Event Flow: Library Patch

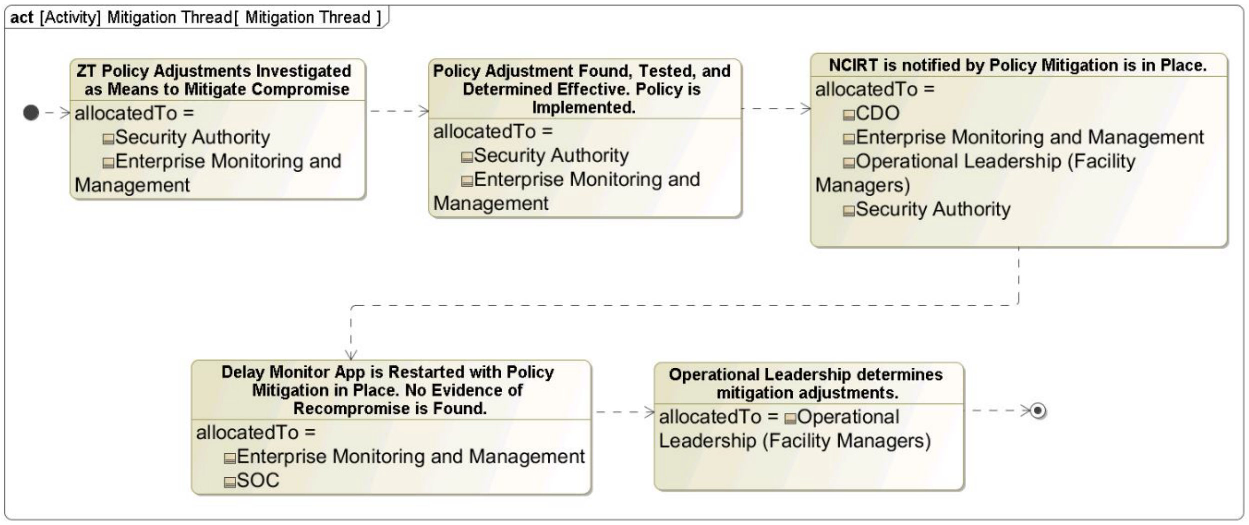

act [Activity] Mitigation Thread[Mitigation Thread]
ZT Policy Adjustments Investigated as Means to Mitigate Compromise
allocatedTo = Security Authority, Enterprise Monitoring and Management
Policy Adjustment Found, Tested, and Determined Effective. Policy is Implemented.
allocatedTo = Security Authority, Enterprise Monitoring and Management
NCIRT is notified by Policy Mitigation is in Place.
allocatedTo = CDO, Enterprise Monitoring and Management, Operational Leadership (Facility Managers), Security Authority
Delay Monitor App is Restarted with Policy Mitigation in Place. No Evidence of Recompromise is Found.
allocatedTo = Enterprise Monitoring and Management, SOC
Operational Leadership determines mitigation adjustments.
allocatedTo = Operational Leadership (Facility Managers)

Figure 5-26. Part B of Security Event Flow: Policy Mitigation

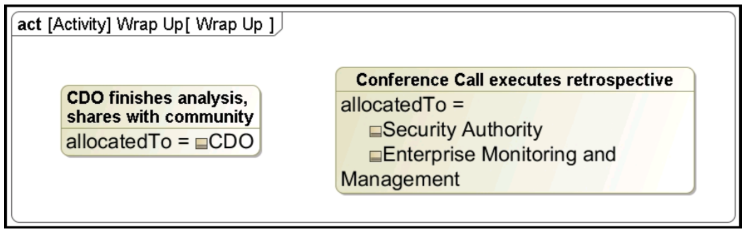

act [Activity] Wrap Up[Wrap Up]
CDO finishes analysis, shares with community
allocatedTo = CDO
Conference Call executes retrospective
allocatedTo = Security Authority, Enterprise Monitoring and Management

Figure 5-27. Part C of Security Event Flow: Wrap Up

The Zero Trust authorization system detects multiple attempts by the Delay Monitor application to establish connections with several unknown systems. Zero Trust policy enforcement does not allow the connections and these events are sent to the *Cybersecurity Defensive Operations* Security Information and Event Management (SIEM) system which generates an alert. *Cybersecurity Defensive Operations* analysts determine that the service is likely compromised.

Cybersecurity Defensive Operations contacts the Delay Monitor *Product Team* and alerts them of the probable compromise. The *Cybersecurity Defensive Operations* also contacts the *Enterprise Monitoring and Management Team,* alerts them of the probable compromise. *Cybersecurity Defensive Operations* takes Virtual Machine (VM) snapshots for the system(s) hosting the compromised microservice container for digital forensics.

Cybersecurity Defensive Operations develops analytics to ascertain if other systems have been compromised by exploit or lateral movement but finds no such indication.

Cybersecurity Defensive Operations establishes the NAS Cyber Incident Response Team (NCIRT) "conference call" with the *Solutions/Product Team, Enterprise Monitoring and Management Team, Security Authority* and *Operational Leadership* to coordinate the response. *Operational Leadership* confirms that there is no safety risk but implements appropriate procedures to mitigate reduced efficiency.

The *Enterprise Monitoring and Management Team* quarantines the exploited containers using platform facilities and manually induces a routine container recreation, creating unexploited Delay Monitor containers from the image repository. *Cybersecurity Defensive Operations* determines that over time the replaced containers are compromised.

Cybersecurity Defensive Operations begins forensics on the quarantined application and VM snapshot.

The United States Computer Emergency Readiness Team (US-CERT) announces a 0-day vulnerability for a 3rd party library. *Cybersecurity Defensive Operations* searches the software bill of materials for all applications for applications that use the vulnerable code in the library. *Cybersecurity Defensive Operations* determines that the library is supported by the Platform Layer and is in an image in the platform hardened authoritative repository. *Cybersecurity Defensive Operations* contacts all *Product Teams* that use the library to alert them to this vulnerability. *Product Teams* including Delay Monitor prepare to rebuild the software when a patched library is available from the platform layer. *Product Teams* that use the vulnerable code in the library are added to the NCIRT.

The Platform *Product Team* obtains the patched library using supply chain validation procedures and their DevSecOps team begins to develop a patched version for the platform container repository.

The *Security Authority* and *Enterprise Monitoring* and *Management Team* begin investigating what Zero Trust policy adjustments might mitigate the vulnerability. They identify a policy adjustment that will mitigate the vulnerability and work thorough the testing procedures to verify its effectiveness and non-interference with other systems. Finding that it is effective and benign, they implement it. This fix is coordinated through the NCIRT.

The *Enterprise Monitoring and Management Team* deploys the Delay Monitor application to production, from container images using platform orchestration facilities. The compromised containers are quarantined and maintained for forensic purposes. *Cybersecurity Defensive*

Operations no longer finds evidence of re-compromise and advises NCIRT of this finding. *Operational Leadership* determines if the operational mitigations in place due to the compromise will be changed or removed given the ZT mitigations appear to be effective.

After going through testing and validation, The Platform *Product Team* publishes the patched container to the repository and advises all Mission *Product Teams* of its availability. All *Product Teams* and *Enterprise Monitoring and Management Team* using the vulnerable library rebuild, test, and deploy updated versions of their software.

All roles involved go through a retrospective on the incident noting things that went well and things that could have been done better.

Cybersecurity Defensive Operations completes forensic activities with quarantined and snapshotted containers and reports their findings to the government cybersecurity community.

5.8.6 End State

Delay Monitor application is fully remediated by rebuilding with the patched library. Zero Trust enforces against the vulnerability. Normal operations are restored.

5.8.7 Summary

This scenario focused on the detection and remediation of the Delay Monitor application after an exploitation is detected.

The question of how different roles work together was answered by illustrating the cooperation of different roles in the event flow of detection and remediation.

The question of how Zero Trust is used was addressed by showing how Zero Trust elements are used to detect and remediate incidents by forwarding policy violations to *Cybersecurity Defensive Operations* and adjusting policy to prevent exploitation.

The question regarding SOCs was addressed in the event flow. *Mission Teams* will not have their own SOCs, but work with product DevSecOps teams and *Cybersecurity Defensive Operations* during an incident.

The question about management of operational risk was also addressed in the event flow. Operational risk will be managed by a combination of technical controls, including Zero Trust policy adjustments and operational procedures, invoked by *Operational Management* and *Mission Teams.*

The question of how other teams will be notified and respond to incidents was also addressed in the event flow.

5.9 Integration of Cloud Cost Management

5.9.1 Description/Background

This scenario introduces cloud cost management methods and tools as part of an overall operational monitoring and management strategy for NAS automation. Without cloud cost management methods, tools, and processes, programs could be at risk of exceeding operational budgets. The monitoring and management of automation application containers and underlying cloud computing, networking, and storage resources is essential both technically and from a program management perspective to manage and sustain operational costs. Cloud cost management tools, in conjunction with Automation cloud monitoring tools, will aid in understanding which automation applications and workloads are suitable for autoscaling (up and down), scheduling shutdown of idle resources, and detect forgotten resources not used, to continuously mange and contain operational costs.

This scenario attempts to answer the following question:

- How are the costs of cloud computing resources governed, monitored, and managed in a production environment?

5.9.2 Roles

The following roles are involved in this scenario:

- *Enterprise Monitoring and Management*
- *Product Management Team*
- *Cost, Finance and Contacting Support*
- *Enterprise Architecture*

5.9.3 Starting State

A new FIS software update has been released into the production cloud environment creating an oversubscription of the automation application container. End-user air traffic controllers have also reported significant aircraft track lag (latency) projected on their display. *Enterprise Monitoring and Management* observes on their cloud-based monitoring and maintenance dashboard the oversubscription of the containers and the auto-scaling of cloud computing resources to relieve the oversubscription and normalize the performance.

Although the cloud resources assigned to the application containers have been auto-scaled (virtual cores, memory, and storage) to alleviate the performance problems, *Enterprise Monitoring and Event Management* is unclear whether the operational budget in the long-term can be sustained overtime. *Enterprise Monitoring and Event Management* defers this operational budget sustainment question to the Automation Application *Product Management team. The Product Management Team* initiates a response and investigates and learns they do not have visibility or the access to collect relevant metrics into the resources assigned to the automation containers that were auto scaled.

Product Management Team proceeds to conduct a market study. The *Product Management team* learns that there are many commercial vendors[23], including cloud providers like Amazon and Azure that offer products in the cloud cost management space. Because there are many vendor products, *Product Management team* has defined criteria based on a tool that monitors, correlates metrics, improves resource visibility, alerts overconsumption of resources, discovers areas of optimization, provides key performance indicators (KPIs) to measure performance and provide intelligence to inform decisions across the responsible actors.

[23] Potential list of cost management products for cloud: CloudZero, Amazon CloudWatch, Azure Cost Management + Billing, Densify, Virtana Optimize and ParkMyCloud, Harness Cloudability, Flexera, CloudHealth, and CloudCheckr.

5.9.4 Event Flow

5.9.4.1 Nominal

Enterprise Monitoring and Management has confirmed the under-provisioning of the containers and associated cloud resources in real-time after the new FIS software release. Containers and cloud resource dependencies were auto-scaled to remediate the latency of tracks via the controllers thin-client ATC display. Cloud cost management tools are integrated in the *Enterprise Monitoring and Management* tool suite of capabilities to measure the costs of auto-scaling of resources to ensure that it can be sustained over the life of the automation application.

5.9.4.2 Off-Nominal Event Flow

Enterprise Monitoring and Management routinely optimizes the use of cloud resources with by integrating and using cloud cost management tools to discovers idle, unused, or forgotten container application and dependent services resources used in the cloud. This activity is performed on a routine basis to sustain costs and extract savings over the life of the automation program application containers and dependent services in the cloud.

5.9.5 End State

The cloud cost management tool has been selected and deployed in the cloud, where *Product Management* realizes that there is a need to provide access to *Cost, Finance and Contacting Support*, *Enterprise Architecture* and *Enterprise Monitoring and Management*. The discovered benefit to providing the cloud cost management tools to these roles are as follows:

- Enables *Enterprise Architects* and *Enterprise Monitoring and Management* to become more cost conscious with their architecture and resource configuration decisions.
- Helps *Cost, Finance and Contracting Support* align application development and cloud engineering, enterprise architecture, development, and operations with the financial budgets.

The cloud cost management tool collects logs and metrics from cloud application services and resources in real-time and displayed on a dashboard for resource optimization and cost monitoring purposes. The tool has demonstrated to be useful to the *Cost, Finance and Contracting Support* to better forecast plan and establish budgets. Cost management tools can enable *Enterprise Architecture* and *Product Management* teams to identify the best system design to extract cost savings and seek areas that need to be reduced or optimized. It delivers insight to the *Enterprise Monitoring and Management* role to detect and identify unused, idle or forgotten cloud resources, and provides better insight into assigning the right cloud resources to remediate technical troubles.

All four roles (*Enterprise Monitoring and Management, Product Management team, Cost, Finance and Contracting Support* and *Enterprise Architecture*) have the responsibility and instrumental in controlling costs, both from a technical operation and financial management perspective. With the integration of cloud cost management tools these roles are better equipped to align the monitoring and management activities with finance and the business. It further enables all roles with the visibility to identify automation application architecture to be right sized and optimized routinely.

5.9.6 Summary

This scenario aids in forming an important aspect of cloud cost management integration into an Enterprise M&M future concept. The integration of cloud cost management is an essential function to ensure cloud resources are continuously being optimized and operational budgets are sustained over the life of the automation application and its dependencies.

This scenario addressed the questions posed in section 5.6.1 as follows:

The question about how costs of cloud computing resources are governed, monitored, and managed in a production environment was answered by providing an example that illustrates how cloud cost tools can be used to obtain visibility and control over cloud costs.

6 Transition of Roles

Table 6-1 expands on the role description provided in Table 3-1 by providing a mapping between the described role and the candidate expertise that fulfills that role. This table is not meant to imply that all the required expertise exists, only that there are some roles and organizations that may provide some expertise and have similar functions. Also, like Table 3-1, this is not intended to be a comprehensive account of all the roles but provides a preliminary basis for follow-on analysis of how roles might transition to the future state described in this ConUse.

Table 6-1. AES Reference Architecture Role Transition Description

Category	Role	Description	Candidate Expertise
Planning & Budget This category includes roles related to high-level agency vision, investment planning, and financial oversight.	Cost, Finance and Contracting Support	Roles involved in the overall management of the solicitation, award, and execution of Agile development contract(s).	Office of Finance & Management (Office of Finance & Management [AFN], Acquisition & Business Services [ACQ], Acquisition & Contracting [AAQ]); Capital Investment Team
	Acquisition Leadership	The agency's investment authority that makes the financial decision on new NAS capabilities investments, technical refreshes, and/or end-of-system lifecycles.	Capital Investment Team; Joint Resources Council; AFN; ACQ; AAQ
	NAS Architecture Planning	Maintains the NAS Enterprise Architecture, which documents the agency roadmaps, major investment plans, and overall NAS architecture, primarily for planning and management purposes.	NAS Enterprise Planning and Analysis Division (ANG-B2); Chief Data Officer (ADO-001)
	Portfolio Management	Responsible for aligning agency strategic goals and operational priorities with execution for a specific business domain in the Enterprise. Responsible for governance, compliance, and return on investment for a collection of solutions. Creates and maintains the portfolio vision and roadmap in coordination with portfolio stakeholders.	NAS Lifecycle Planning Division (ANG-C7), Portfolio Management Activities within the Strategy Directorate (AJV-S000);
Enterprise Engineering This category includes systems engineering roles that define and oversee the NAS architecture.	NAS Chief Architect	Provides overall NAS-wide architecture design, oversight, and technical guidance. Provides comprehensive overarching view of all layers and across all applications and services. Maintains the Automation Evolution Reference Architecture.	Chief Scientist for NextGen (ANG-3); NAS Enterprise Architecture & Requirements Services Division (ANG-B1)
	Mission Layer Architect	Ensures that the right set of Mission Layer services and applications are in place and provides overarching NAS-wide design guidance for the mission layer to ensure interoperability and efficiency. Engineers the performance profiling of services; coordinates with teams to identify bottlenecks and inefficient activities. Looks for opportunities to refactor processes to facilitate streamlining and automation. Has knowledge of the business, collaborates with the community (internal/external), identifies candidate mission/common services, and specifies requirements, including rules.	ANG-B1

Category	Role	Description	Candidate Expertise
	Platform Layer Architect	Oversees the selection and evolution of the NAS-wide suite of tools and software components that make up the Platform Layer. Ensures that the needs of Mission Layer developers are met. Certifies and maintains the security of standard container images, ensures availability, and maintains documentation.	Communication, Information, and Network Programs (AJM-3100)
	Compute Layer Architect	Engineers fundamental compute, network, and storage resources for Platform and Mission engineering on-demand, NAS-wide. Compute Architecture Engineering enables Mission and Platform users to scale and shrink resources on an as-needed basis, reducing the need for high, up-front capital expenditures or unnecessary on-premises infrastructure.	Communication, Information, and Network Programs (AJM-3100); NAS Cloud Integration Service (NCIS); National Enterprise Management Center (Integrated Enterprise Services Platform) (AJW-B11); FAA Cloud Services (FCS) Program Manager (AIF-001)
	Data Architect	Ensure all data across the enterprise is accessible and accurate. The data architect translates business requirements into technology requirements and defines data standards, i.e., the rules that define how data is described and recorded. To meet this responsibility, the data architect specifies a data management framework for reviewing, specifying, refining, acquiring, archiving, and purging data and the associated schemas, and ensures that appropriate coordination occurs among information stakeholders.	Chief Data Officer (ADO-001); Enterprise Information Management (EIM) Governance; NAS Enterprise Planning and Analysis Division (ANG-B2); ANG-B1
	Security Architect	Works with other architects to design the security controls for each layer and their subsystems. The security architect ensures that the operation of the security executed by the Security Authority will be effective by mandating that the measures necessary for effective authentication, authorization and policy enforcement be in the architecture and implementation of the RA. This role also develops and maintains a security risk management plan.	Authorizing Official (AO) and Authorizing Official Designated Representative (AODR) – Security
Enterprise Security Engineering & Operations This category provisions and operates platform and compute layer services and	Safety Authority	Ensures that any software deployed into the NAS production environment has been properly assessed for safety and that all safety risks have been adequately mitigated. Ensures that safety processes and standards are integrated into the Architecture including the DevSecOps chain to ensure the safety of services developed in software factories deployed on the Platform Layer.	Office of Aviation Safety (AVS); Air Traffic Safety Oversight (AOV-2); Safety (AJI-1000)

Category	Role	Description	Candidate Expertise
monitors and manages mission layer software enterprise wide. This category is also responsible for security and safety oversight and operations.	Security Authority	Accountable for NAS Enterprise Information Security Governance. Sets or delegates policy for security including Zero Trust, defensive capabilities, compliance, audit, training, and enforcement. Responsible for crafting and implementing processes to ensure policy is followed. Responsible for reviewing and updating policy as conditions, capabilities, and technology evolve. Responsible for working with other roles to understand and make tradeoffs with security policy as is best for the overall FAA mission.	Information System Security Officers (ISSOs); FAA/DoT SOC; Vendor SOCs (Harris for FTI, GDIT for FCS); NAS Cybersecurity Operations (NCO); Security and Hazardous Materials (ASH) – Physical and Personnel Security; Air Traffic Organization (ATO); NAS Information Security (AJW-B400); NAS Cyber Operations (AJW-B340)
	Cybersecurity Defensive Operations	Actively defends the NAS against cybersecurity threats. This includes operations centers that monitor NAS software and networks respond to incidents. Investigates potential attacks, data breaches and system compromises.	AJW-B340; AJW-B400; Security Operations Division (AIS-310, AIS-320, AIS-330); Cybersecurity Services Branch (AMK-230);
	Enterprise Monitoring and Management	Monitors, manages, and operates common mission and platform services that are used by multiple NAS stakeholders. This role ensures that quality of service objectives are met in operation and responds appropriately to changes in service or application status.	National Enterprise Ops (AJW-B000); Enterprise Operations Group (AJW-B100); NAS Enterprise Operations (NASEO) and NAS Enterprise Monitoring Centers (NEMCs); Operational Support Facilities (Regions); FAA Telecommunications Infrastructure (FTI) Network Operations Center (NOC)
	Cloud Service Provider Cybersecurity Defensive Operations	Actively defends the vendor Cloud against cybersecurity threats. This includes operations centers that monitor networks respond to incidents. Investigates potential attacks, data breaches and system compromises. Coordinates with FAA Cybersecurity Defensive Operations when needed.	
Customer Organizations This category includes the end user organizations that are responsible for providing NAS services (separation services, flow management services, etc.) to airspace users.	Operational Leadership	This role is comprised of FAA managers responsible for different areas of NAS mission operations (e.g., En Route and Terminal air traffic control (ATC), system operations, aeronautical information) as well as technical operations at facilities.	Air Traffic Services (AJT-0000)
	End Users (FAA and External entities)	This role is comprised of controllers, traffic managers, and coordinators. They conduct operations using Mission Application front ends running on their devices as part of the Computing Resources Layer (e.g., workstations and tablets and if not safety-critical, possibly implemented as web applications running in a browser) to access data and computation services needed to perform their job functions. External entities may also be end users of FAA-provided services, for example air carriers might be end users of flight planning and filing services, Department of Defense (DoD) and Department of Homeland Security (DHS) might be end users of a flight data service.	National Air Traffic Controllers Association (NATCA) reps; Professional Aviation Safety Specialists (PASS) representatives; Specialists (controllers, flow managers, etc.);

Category	Role	Description	Candidate Expertise
	Facility Level Technical Support	This role includes staff that maintain NAS facilities, systems, and equipment at Air Route Traffic Control Centers (ARTCCs), terminal area traffic control facilities, flight service stations, regional centers, and other FAA facilities supporting operations. They provide support for the daily operation of facilities and offices under the jurisdiction of the FAA's Air Traffic Organization (ATO). Facility Level Technical Support provides technical support to End Users in the facilities who may experience problems or issues with using mission layer software. They coordinate and collaborate with Enterprise Monitoring and Management as well as Product Teams, when necessary, for technical support that is within the purview of these other roles. They are also responsible for hardware at the compute layer in the field, including generic IT equipment (e.g., printers, workstations, local area networks, phones) as well as FAA Infrastructure equipment that is unique to the NAS (e.g., radios, radars, and other sensors).	Technical Operations Central Service Area (CSA), Eastern Service Area (ESA), Western Service Area (WSA) (AJW-C000, E000, W000);
Solution Teams This category includes the roles that are responsible for developing and sustaining solutions that integrate multiple products to meet FAA needs.	Solution Management	Solution management has overall responsibility for the creation and sustainment of a solution that meets customer needs. A solution is provided by and integrated set of products, which may include mission applications and services, as well as platform and computing resource enablers.	Portfolio Management Activities within the Strategy Directorate (AJV-S000); For solutions that are implemented within a single program: Program Managers; AJM-0
	Solution Architecture, Engineering, and Release Coordination	Solution team members support solution management, providing technical and architectural vision that spans a set of mission applications and services, as well as platform layer and computing layer technical enablers that make up a solution. This team ensures that these components are built to be interoperable and coordinates the release trains of multiple Product Teams to synchronize development timelines. The solution team has overall responsibility for deploying solution increments (capabilities) into the production environment for use by Customer Organizations. This includes ensuring that quality control, security, and safety assurance processes have been followed and that development, integration, and operational testing and evaluation have been successfully completed. This team is also responsible for coordinating with Enterprise Security and Operations and Customer Organizations on issues such as support, training, and deployment schedule.	Portfolio Area Leads within the Strategy Directorate (AJV-S000); Program Office staff, including: Systems Engineering Lead; Second Level Engineering (SLE); For large solutions that involve multiple programs, Program Office teams must collaborate. Guidance and coordination may be provided by groups such as the Architecture Review Board and Technical Review Board. Also, guidance and input from TechOps (Support and Security [ATO Cybersecurity Group (ACG)] staff); Safety & Technical Training (AJI-3).

Category	Role	Description	Candidate Expertise
Product Teams A Product Team consists of one or more Agile Teams responsible for developing and sustaining a product, following Agile principles.	Product Management and Product Owner	Facilitates and coordinates participation by the end users within the Agile Development Team. The Product Owner has the business and operational knowledge to represent both internal and external user and stakeholder Customer Organizations. The role organizes the development process in terms of time-boxed iterations that lead to a release of a capability. For large software development efforts, there may be multiple Product Owners, each responsible for one portion of the overall product, with overall coordination provided by Product Management.	Product Management would be provided by a Program Manager. Product Owner role may be provided by a contractor. It is possibly a future government role.
	Scrum Master	Facilitates the processes, enforces the team's rules, and keeps the team focused on tasks.	This occurs on a limited basis within the contractor organization and is not an enterprise activity.
	Other Agile Team roles	The team comprises multiple roles, including software developers, software and security engineers, data specialists, testers, quality assurance, release train engineers, and configuration managers. Typically, this would be staffed primarily by contractors, but would also include FAA personnel.	Within the Contractor Organization; Program Office and Safety Panel Members; InfoSec (ACG assigns Information Systems Security Officer(s) [ISSO]); Second Level Engineering (AJM-2000); Test and Evaluation staff at the Tech Center (ANG-E5/6)

7 References

[1] C. R. Andrews, D. A. Chaloux, W. S. Heagy, M. E. Liggan, O. Olmos and D. Thomson, "National Airspace System Automation Evolution Strategy," The MITRE Corporation, McLean, VA, 2020.

[2] The Federal Aviation Administration, "Service-Based Reference Architecture for NAS Automation (Version 1.0)," FAA, Washington D.C., 2021.

[3] Federal Aviation Administration, "Agile Program Management Practices for the Federal Aviation Administration Version 1.0," Federal Aviation Administration, Washington D.C., 2016.

[4] Scaled Agile, Inc., "SAFe 5 for Lean Enterprises," [Online]. Available: https://www.scaledagileframework.com/. [Accessed 14 July 2021].

[5] Federal Aviation Administration, "Federal Aviation Administration Agile Acquisition Principles and Practices," Federal Aviation Administration, Washington D.C., 2016.

[6] Federal Aviation Administration, "Mission and Common Services - Transition Planning," Federal Aviation Administration, Washington, DC, 2021.

[7] The Federal Aviation Administration, Fiscal Year 2021 Security Authorization Handbook, Version 1, 2020.

[8] G. K. Behara, "Enterprise Architecture Governance: A Holistic View," 9 January 2022. [Online]. Available: https://dzone.com/articles/enterprise-architecture-governance-a-holistic-view. [Accessed 22 June 2022].

[9] D. Shepard and J. Scherb, "What Is Digital Engineering and How Is It Related to DevSecOps?," 16 November 2020. [Online]. Available: https://insights.sei.cmu.edu/blog/what-digital-engineering-and-how-it-related-devsecops/. [Accessed 22 June 2022].

[10] M. Hayes, S. Miller, M. A. Lapham, E. Wrubel and T. Chick, "Agile Metrics: Progress Monitoring of Agile Contractors," January 2014. [Online]. Available: https://resources.sei.cmu.edu/asset_files/TechnicalNote/2014_004_001_77799.pdf. [Accessed June 22 2022].

[11] S. Wolpers, "Agile Metrics — The Good, the Bad, and the Ugly," 24 August 2020. [Online]. Available: https://www.scrum.org/resources/blog/agile-metrics-good-bad-and-ugly. [Accessed 22 June 2022].

[12] E. Tremel, "Kubernetes Deployment Strategies," 25 September 2017. [Online]. Available: https://blog.container-solutions.com/kubernetes-deployment-strategies.

[13] C. Andrews, D. Chaloux, W. Heagy, M. Liggan, O. Olmos and D. Thomson, "National Airspace System Automation Evolution Strategy: Coordination Draft," The MITRE Corporation, McLean, VA, 2020.

[14] "Beyond Agility," [Online]. Available: https://beyond-agility.com/deployment-vs-release/.

[15] D. Parzych, "What are Feature Flags," 10 June 2020. [Online]. Available: https://launchdarkly.com/blog/what-are-feature-flags/.

[16] D. Des, "How to handle stories that were not completed," 20 September 2018. [Online]. Available: https://www.scrum.org/forum/scrum-forum/25587/how-handle-stories-were-not-completed.

[17] KnowledgeHut, "Agile Conflict Resolution Hacks You Should Master," 28 April 2021. [Online]. Available: https://www.knowledgehut.com/blog/agile/agile-conflict-resolution-hacks-you-should-master.

[18] Kubernetes.io, "Namespaces," 21 August 2021. [Online]. Available: https://kubernetes.io/docs/concepts/overview/working-with-objects/namespaces/.

[19] DoDCIO, "DoD Enterprise DevSecOps Reference Design," 12 August 2019. [Online]. Available: https://dodcio.defense.gov/Portals/0/Documents/DoD Enterprise DevSecOps Reference Design v1.0_Public Release.pdf.

[20] K. Long, D. Greenbaum, T. Stewart, Park, Y. Park, Ellis, C. Bodoh, L. Ellis, S. Zobell and D. Blumenthal, "Application-Based Capability Development Framework - Building, Connecting, and Using Software Platforms for Rapid Implementation of Air Traffic Management Capabilities," MITRE, McLean, VA, 2021.

Appendix A Agile Processes

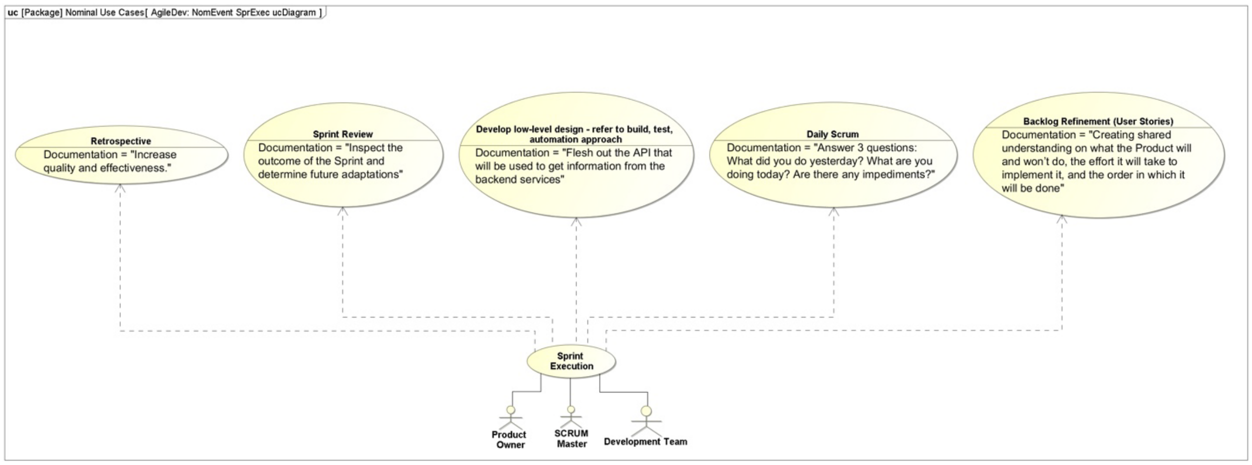
This appendix provides more detail on the Agile ceremonies that are mentioned, in simplified form, in the body of the document.

- <u>Portfolio Solution Planning (PSP)</u>
    - Follows: Enterprise Decomposition into Portfolios/Solutions
    - Precedes: Program Increment Planning
    - Frequency: Once per Program Increment
    - Inputs:
        - Business context
        - Roadmap and vision
    - Activity:
        - Decompose portfolio into multiple programs
        - High level requirements specification
        - Develop Capability roadmap
    - Participants:
        - *Solution Manager*, *Product Managers*
    - Outputs:
        - Backlog of Portfolio Capabilities (Prioritized)
        - List of Portfolio Programs
- <u>Program Increment Planning (PIP)</u>
    - Follows: PSP
    - Precedes: Release Train Engineer, Sprint Planning
    - Frequency: Once a Program Increment
    - Inputs:
        - Business context
        - Roadmap and vision
        - High level program requirements
        - Portfolio Capability Backlog
    - Activity:
        - Decompose Capabilities into program backlog (of features)
        - Detailed requirements specification (functional, non-functional, domain)
        - Testing guidance
        - In this scenario, this planning is where it would be identified that an API needs to be created that will allow the frontend to get the information it needs from the backend service.
    - Participants:
        - *Product Manager*, *Product Owners*, *Engineering and Release Coordination*, *Enterprise Engineering*, *Scrum Masters*

- Outputs:
    - Program backlog (prioritized list of features)
    - Detailed program requirements (functional, non-functional, domain)

- <u>Scrum of Scrums (SoS)</u>
    - Follows: Sprint Execution
    - Precedes: Program Increment Review
    - Frequency: Daily to Weekly
    - Inputs:
        - Sprint backlogs (prioritized User Stories)
        - Program Requirements (functional, non-functional, domain)
    - Activity:
        - 15 minutes scaled scrum as a key meet-up to align, improve, and tackle impediments. A representative of each team or the product owner should discuss team impediments, risks to achieving the sprint goal or dependencies on other teams followed by discovered improvements that can be leveraged by other teams.
        - In this scenario with a backend and frontend *Product Teams*, this is where coordinating when their portion of the software will be ready for integration testing will happen.
        - Process compliance verification.
    - Participants:
        - *Engineering and Release Coordination*, *Scrum Masters*, Select *Product Team* members
    - Outputs:
        - Identified impediments and plans to facilitate solving/eliminating those impediments.
- <u>Sprint Planning</u>
    - Follows: Program Increment Planning
    - Precedes: Sprint Execution
    - Frequency: Once per Sprint
    - Inputs:
        - Program backlog (prioritized features)
        - Program Requirements (functional, non-functional, domain)
    - Activity:
        - Decompose prioritized features into User Stories
        - Prioritize the User Stories
        - Estimate effort to implement stories and determine which ones make it into the sprint
    - Participants:
        - *Product Owner* (representing End Users), *Scrum Master*, *Product Team* members

- Outputs:
    - Sprint backlog (prioritized User Stories) – what can realistically be delivered


uc [Package] Nominal Use Cases[AgileDev: NomEvent SprExec ucDiagram]
Retrospective
Documentation = "Increase quality and effectiveness."
Sprint Review
Documentation = "Inspect the outcome of the Sprint and determine future adaptations"
Develop low-level design - refer to build, test, automation approach
Documentation = "Flesh out the API that will be used to get information from the backend services"
Daily Scrum
Documentation = "Answer 3 questions: What did you do yesterday? What are you doing today? Are there any impediments?"
Backlog Refinement (User Stories)
Documentation = "Creating shared understanding on what the Product will and won't do, the effort it will take to implement it, and the order in which it will be done"
Sprint Execution
Product Owner
SCRUM Master
Development Team

- <u>Sprint Execution</u> (shown in the figure above)
    - Follows: Sprint Planning
    - Precedes: Program Backlog Refinement (prioritized features)
    - Frequency: Multiple times per Program Increment
    - Inputs:
        - Sprint backlog (prioritized User Stories)
        - Program Requirements (functional, non-functional, domain)
    - Activity:
        - Develop low-level design – refer to build, test, automation approach
            - In this scenario this is where representatives of the two *Product Teams* would meet to flesh out the API that will be used by the backend to get information from the backend service.
        - Daily Scrum
            - 15 minutes daily meet-up to provide status and tackle impediments. Development *Product Team* members discuss impediments and risks to achieving the sprint goal. Answer three questions: 1) What did you do yesterday? 2) What are you doing today? 3) Are there any impediments?
        - Backlog refinement (User Stories)
            - Is about creating shared understanding on what the Product will, and won't, do, on the effort it will take to implement it, and the order in which it will be done.
        - Sprint Review
            - The purpose of the Sprint Review is to inspect the outcome of the Sprint and determine future adaptations. The *Product Team* presents the results of their work (e.g., demo) to key stakeholders and progress toward the Product Goal is discussed.

- Retrospective
  - The purpose of the Sprint Retrospective is to plan ways to increase quality and effectiveness.
  - The *Product Team* inspects how the last Sprint went with regards to individuals, interactions, processes, tools, and their Definition of Done.
  - The *Product Team* discusses what went well during the Sprint, what problems it encountered, and how those problems were (or were not) solved.

- Participants:
  - *Product Owner*, *Scrum Master*, *Product Team* members
- Outputs:
  - Code, build files, test cases, built software, property files, and other sprint related artifacts such as test results stored in code and artifact repositories.

- <u>Program Increment Review (PIR)</u>
  - Follows: Program Increment
  - Precedes: Portfolio Solution Review (PSR)
  - Frequency: Once per Program Increment
  - Inputs:
    - Sprint backlog (prioritized User Stories)
    - Program Requirements (functional, non-functional, domain)
  - Activity:
    - Program Backlog feature refinement
    - Program Increment Review
    - Program Increment Retrospective
  - Participants:
    - *Product Manager*, *Product Owners*, *Engineering and Release Coordination*, *Enterprise Engineering*, *Scrum Masters*
  - Outputs:
    - Updated Program Feature Backlog
- <u>Portfolio Solution Review (PSR)</u>
  - Follows: Final Program Increment Review
  - Precedes: ---
  - Frequency: Once per Program Increment
  - Inputs:
    - Business context
    - Roadmap and vision
    - High Level Requirements
    - Capability Definitions
  - Activity:

    - Program Backlog Refinement (of features)
    - Program Increment Review
    - Program Increment Retrospective
- Participants:
    - *Solution Manager*, *Product Managers*, *Product Owners*
- Outputs:
    - Updated Portfolio Capabilities backlog (prioritized list of Capabilities)

Appendix B Acronyms

Acronym	Definition
AAQ	Acquisition & Contracting
ABCD	Application Based Capability Development
ACQ	Acquisition & Business Services
ADO-001	Chief Data Officer
AES	Automation Evolution Strategy
AFN	Office of Finance & Management
AIF-001	FAA Cloud Services Program Manager
AJI-1000	Safety Division
AJI-3	Safety & Technical Training
AJM-3100	Communication, Information, and Network Programs
AJT-0000	Air Traffic Services
AJW-B000	National Enterprise Operations
AJW-B100	NAS Enterprise Operations Group
AJW-B11	Enterprise Management Center
AJW-B340	NAS Cyber Operations
AJW-B400	NAS Information Security
AMK-230	Cybersecurity Services Branch
ANG-3	Chief Scientist for NextGen
ANG-B1	NAS Enterprise Architecture & Requirements Services Division
ANG-B2	NAS Enterprise Planning and Analysis Division
ANG-C7	NAS Lifecycle Planning Division
AO	Authorizing Official
AODR	Authorizing Official Designated Representative
AOV-2	Air Traffic Safety Oversight Division
API	Application Programming Interface
ARTCC	Air Route Traffic Control Center
ASH	Security and Hazardous Materials
ATC	Air Traffic Control
ATCT	Airport Traffic Control Tower
ATO	Air Traffic Organization
ATO	Authority to Operate
AVS	Office of Aviation Safety
CDO	Cybersecurity Defensive Operations
CI/CD	Continuous Integration/Continuous Deployment
ConUse	Concept of Use

Acronym	Definition
CSA	Central Service Area
CSP	Cloud Service Provider
DevSecOps	Development, Security, and Operations
DT&E	Development Test and Evaluation
EFS	Electronic Flight Strip
EIM	Enterprise Information Management
ESA	Eastern Service Area
FAA	Federal Aviation Administration
FCS	FAA Cloud Services
FedRAMP	Federal Risk and Authorization Management Program
FIS	Flight Information Service
FIXM	Flight Information Exchange Model
FTI	FAA Telecommunications Infrastructure
FY	Fiscal Year
GUI	Graphical User Interface
IaC	Infrastructure as Code
ISCM	Information System Continuous Monitoring
ISSO	Information System Security Officers
KPI	Key Performance Indicator
MLPT	Mission Layer Product Team
MTTR	Mean Time to Recovery
MVCR	Minimum Viable Capability Release
MVP	Minimum Viable Product
NAS	National Airspace System
NASEO	NAS Enterprise Operations
NATCA	National Air Traffic Controllers Association
NCIRT	NAS Cyber Incident Response Team
NCIS	NAS Cloud Integration Service
NCO	NAS Cybersecurity Operations
NEMC	NAS Enterprise Monitoring Center
NOC	Network Operations Center
OE	Operating Environment
OOB	Out-of-Band
OT&E	Operational Test and Evaluation
PASS	Professional Aviation Safety Specialist
PI	Program Increment
PIP	Program Increment Planning

Acronym	Definition
PIR	Program Increment Review
PLPT	Platform Layer Product Team
PSP	Portfolio Solution Planning
PSR	Portfolio Solution Review
RA	Reference Architecture
SAFe	Scaled Agile Framework
SAR	System Analysis and Recording
SBOM	Software Bill of Materials
SCoP	Stewardship Communities of Practice
SIEM	Security Information and Event Management
SLA	Service Level Agreements
SLE	Second Level Engineering
SOC	Security Operations Center
SoS	Scrum of Scrums
TBFM	Time Based Flow Management
TBO	Trajectory Based Operations
TRACON	Terminal Radar Approach Control Facility
US-CERT	United States Computer Emergency Readiness Team
vCPU	Virtual Server Central Process Unit
VM	Virtual Machine
WAF	Web Application Firewall
WSA	Western Service Area
xTM	Extensible Traffic Management
ZT	Zero Trust
ZTA	Zero Trust Architecture