TECHNICAL MANUAL

SL 4J

USAREUR-AF

TM-40J — MAVEN SMART SYSTEM (MSS)

Specialist Course Manual

HEADQUARTERS
UNITED STATES ARMY EUROPE AND AFRICA
(USAREUR-AF)
Wiesbaden, Germany

26 MARCH 2026

DRAFT — UNOFFICIAL — NOT FOR OPERATIONAL USE

TM-40J — MAVEN SMART SYSTEM (MSS)

Forward: SL 4J qualifies Technical Project Managers, Product Owners, and Team Leads to plan, execute, and govern data, AI, and software capability builds on the Maven Smart System (MSS). This track bridges technical execution (SL 4G through SL 4O developers) and operational requirements (commanders, staff, end users). **Prereqs:** SL 1, Maven User; SL 2, Builder; SL 3, Advanced Builder (required); Data Literacy Technical Reference (required); CONCEPTS_GUIDE_TM40J_PROGRAM_MANAGER (read before this manual). *HQ USAREUR-AF · v1.0 · 2026 · DISTRIB: USG only · AUTH: C2DAO/UDRA v1.1*

WARNING

MSS data products built under SL 4J oversight directly support commander decision

TABLE OF CONTENTS

- CHAPTER 1 — INTRODUCTION: THE TECHNICAL PM ROLE
- CHAPTER 2 — AGILE PROJECT MANAGEMENT FOR DATA AND AI PROJECTS
- CHAPTER 3 — MANAGING ML AND AI PROJECT LIFECYCLES
- CHAPTER 4 — STAKEHOLDER MANAGEMENT AND REQUIREMENTS TRANSLATION
- CHAPTER 5 — BUILDING PROJECT TRACKING SYSTEMS ON MSS
- CHAPTER 6 — RISK AND DEPENDENCY MANAGEMENT
- CHAPTER 7 — DELIVERY PLANNING AND PRODUCTION READINESS
- CHAPTER 8 — CHANGE MANAGEMENT AND USER ADOPTION
- APPENDIX A — PROJECT KICKOFF CHECKLIST
- APPENDIX B — DEFINITION OF DONE — DATA PRODUCT STANDARDS
- APPENDIX C — PROFESSIONAL READING LIST
- GLOSSARY

CHAPTER 1 — INTRODUCTION: THE TECHNICAL PM ROLE

1-1. Program Manager Specialist Manual

BLUF: SL 4J qualifies Technical Project Managers, Product Owners, and Team Leads to plan, execute, and govern data, AI, and software capability builds on the Maven Smart System (MSS). This track bridges technical execution (SL 4G through SL 4O developers) and operational requirements (commanders, staff, end users).

This manual provides task-level instruction for managing the full lifecycle of a data or AI project on MSS — from initial stakeholder requirements through sprint planning, execution, release, and sustainment. SL 4J graduates serve as the connective tissue between technical build teams and the operational units they support.

SL 4J covers Agile project management methods adapted for data and AI projects in a military operational context; backlog management: writing user stories and acceptance criteria for data products and dashboards; managing ML and AI project lifecycles from research through production and sustainment; translating operational requirements from commanders and staff into actionable technical tasks; building live project tracking systems on MSS using Workshop, Ontology, and Pipeline Builder; risk and dependency management specific to data availability, model performance, and technical debt; delivery planning: scope, timeline, and quality tradeoffs for data product releases; coordinating across technical tracks: when to pull in SL 4H (AI Engineer), SL 4L (Software Engineer), SL 4M (ML Engineer), SL 4K (Knowledge Manager), and SL 4L (SWE); change management: deploying new MSS capabilities to operational users who resist change; and platform governance from a PM perspective: Foundry project permissions, resource allocation, and access stewardship.

SL 4J does NOT cover raw coding of Foundry transforms, Functions on Objects, or API integrations — see SL 4L; machine learning model development and validation — see SL 4M (ML Engineer); AI agent and workflow automation build — see SL 4H (AI Engineer); advanced statistical analysis and modeling — see SL 4G (ORSA); or knowledge management architecture and data dictionary design — see SL 4K (KM).

NOTE

SL 4J is a management and coordination track. The PM does not need to write code. The PM must understand enough about each technical discipline to write accurate user stories, identify blockers, assess risk, and make scope tradeoff decisions. Technical depth comes from the SL 4 specialists the PM leads.

1-2. The Technical PM in the MSS Ecosystem

USAREUR-AF data and AI capability builds operate at speed. Theater requirements generate demands for new dashboards, new data pipelines, new AI-assisted tools, and new integrations faster than any single developer can absorb. Technical PMs manage the backlog, prioritize the work, protect the team from scope sprawl, and ensure that what ships is operationally sound.

The MSS ecosystem involves multiple technical disciplines that must be coordinated:

Track	Role	What They Build
SL 4G	ORSA	Quantitative models, statistical analysis, analytical products
SL 4H	AI Engineer	AI agents, AIP Logic, automated workflows, LLM integrations
SL 4M	ML Engineer	ML models, training pipelines, model evaluation and deployment
SL 4J	Program Manager (Technical)	Project tracking, stakeholder mgmt, delivery coordination
SL 4K	Knowledge Manager	Data dictionaries, ontology governance, data stewardship
SL 4L	Software Engineer	Custom Foundry code, Functions, API integrations, services
SL 4N	UI/UX Designer	Workshop layouts, Slate apps, accessibility, design systems
SL 4O	Platform Engineer	Infrastructure, CI/CD, Kubernetes, monitoring, compliance

The PM does not manage people's performance in the HR sense. The PM manages scope, timeline, risk, and stakeholder communication. The PM shields the technical team from requirements churn and ensures the team delivers the right thing to the right standard at the right time.

1-2a. DDOF Roles and PM Oversight Responsibilities

The Data and Digital Operations Framework (DDOF) defines six roles that participate in the data product lifecycle. The PM interacts with all six and must understand the oversight relationship for each.

DDOF Role	Typical Echelon	PM Oversight Responsibility
Decision Maker (DM)	O-6+	PM ensures DM requirements are SMART-compliant (Specific, Measurable, Achievable, Relevant, Time-bound) before they enter the backlog
C2DAO	O-4/O-5	PM coordinates gate approvals with C2DAO at each DDOF phase transition
Functional Data Manager (FDM)	O-3/O-4	PM tracks FDM product lifecycle health and ensures FDM sign-off on data quality

DDOF Role	Typical Echelon	PM Oversight Responsibility
Data Engineer	Technical	PM manages sprint deliverables, pipeline schedules, and cross-team dependencies
Data Scientist / ORSA	Technical	PM aligns analytical output to DM requirements and validates evaluation methodology
Knowledge Manager	Staff	PM ensures documentation, ontology governance, and training compliance are current

NOTE

The PM does not replace any DDOF role. The PM coordinates across roles and ensures that handoffs between roles do not stall. When a DDOF gate decision is pending, the PM is responsible for assembling the gate package, scheduling the review, and documenting the outcome. The gate authority remains with the C2DAO and DM.

Source: DDOF Playbook v2.2, T2COM C2DAO, December 2025.

1-2b. The XVIII ABC ODT Model — PM Role in Operational Data Teams

XVIII Airborne Corps published its experience building and employing Operational Data Teams in *Military Review* (February 2026, "Fighting with Live Data"). Their pilot — one of the first in the Army — places a Product Manager (5F ASI) as one of five core roles in each ODT, alongside a UI/UX Designer, Software Engineer, Data Engineer, and Data Scientist. This mirrors the SL 4J through SL 4O specialist track structure. The XVIII ABC model is a useful reference point from one corps' pilot, not a universal template — different commands will adapt roles and governance to their own context. The lessons below are drawn from their specific experience at Corps level.

Key PM-relevant lessons from XVIII ABC:

XVIII ABC Practice	SL 4J Application
Technical Innovation Objectives (TIOs) — staff working groups develop TIOs aligned to CAMPLAN operational problems; ODT + G-5 + CTO assess operational value and development feasibility	PM leads the TIO-equivalent scoping process: translating commander requirements into SMART-compliant problem statements, not desired solutions
Program Increment (PI) cycles — 12-week development cycles aligned to Corps exercise cycles; CG approves the "cut line" and retains authority to retask	PM manages sprint cadence within the PI framework: delivery milestones aligned to training calendar and exercise windows
CG governance policy — signed priorities governance prevents ODT overextension; adjudication process for	PM enforces the governance boundary: no work enters the backlog without going through the

XVIII ABC Practice	SL 4J Application
competing requirements	prioritization process; PM assembles the gate package
Problem-solution methodology — scoping (2 wk) → discovery (4 wk) → framing (4 wk) → development (8 wk) → handoff (2 wk); invest/divest/pivot gates	PM owns the process cadence and gate documentation: each decision point requires a structured recommendation with operational value assessment
Exercise integration — MVP products get bug fixes/minor adjustments during exercises; scoping-phase products get user research only; discovery/framing held stable	PM manages exercise scope: shields team from pressure to do major adjustments during exercises while ensuring user feedback is captured for next PI

NOTE

XVIII ABC's early experience without a governance policy resulted in too many customers requesting ODT resources simultaneously, with no adjudication process. The PM role is the primary defense against this failure mode. SL 4J graduates must understand that the governance process is not optional — it is a prerequisite for ODT effectiveness.

Source: Forney, Herrmann, and Steele, "Fighting with Live Data," Military Review Online Exclusive, February 2026.

Supplementary reading — Adkins' proposal: Adkins ("Achieving Decision Dominance," *Military Review*, January-February 2025) is a thought piece by one officer proposing terminology and concepts for data team employment. Useful shorthand for SL 4J PMs:

- **Automated Fighting Products (AFP):** Adkins' term for visualization tools connected to live data that reduce staff burden and inform commander decisions. The products your team builds fit this description. The PM manages the AFP lifecycle from requirements through sustainment.
- **Echeloned reach-back:** Adkins proposes a triage system where tactical units reach back to higher-echelon data teams for complex problems. PMs should understand their team's echelon and the reach-back relationships above and below them.
- **MSS reference:** Adkins names the Maven Smart System as an ASCC-level COP platform.

NOTE

The Forney "Fighting with Live Data" article above documents the most relevant pilot experience for ODT/PM employment — authored by a COL, MAJ, and CPT who led the XVIII ABC ODT pilot. Adkins provides supplementary context.

Source: Adkins, "Achieving Decision Dominance," Military Review 105, no. 1 (January-February 2025).

1-3. Prerequisites and Entry Standards

Conditions

Personnel seeking SL 4J qualification have completed SL 3 (Advanced Builder) and can operate MSS at the advanced no-code level. They are assigned or designated as a project lead, product owner, or team lead for a data or AI capability build.

Standards

Upon completion of SL 4J, the Technical PM can:

1. Stand up an Agile project structure (backlog, sprint cadence, ceremonies) for a data/AI project
2. Write user stories and acceptance criteria that SL 4G through SL 4O developers can execute without ambiguity
3. Manage an ML/AI project from research brief through production release using the MSS lifecycle model
4. Translate a commander or staff requirement into a structured requirements document and prioritized backlog
5. Build a live project tracking dashboard on MSS Workshop visible to all project stakeholders
6. Identify and track top five project risks and dependency blockers at any point in the project
7. Conduct a production readiness review against the Definition of Done (Appendix B)
8. Execute a change management plan for a new MSS capability deployment to operational users

Equipment

- MSS account with Builder access (SL 3 level)
- Foundry Workshop, Pipeline Builder, and Ontology UI access
- Access to the USAREUR-AF C2DAO project space for coordination
- Connectivity to project communication channels (Teams, SharePoint, or MSS-hosted)

1-4. SL 4J in the Curriculum Architecture

SL 4J sits at the top of the no-code track and the entry point of the management track.

```
SL 1 (User) → SL 2 (Builder) → SL 3 (Advanced Builder) → SL 4J (Technical PM)
                                                          ↘ SL 4G (ORSA)
                                                          ↘ SL 4H (AI Engineer)
                                                          ↘ SL 4M (ML Engineer)
                                                          ↘ SL 4K (KM)
```

```
                                        ↘ SL 4L (SWE)
                                        ↘ SL 4N (UI/UX Designer)
                                        ↘ SL 4O (Platform Engineer)
```

The PM must have completed SL 3 to understand what each specialist track builds and why decisions have technical consequences. A PM who has never built on MSS cannot accurately estimate effort, identify blockers, or write meaningful acceptance criteria for MSS-based deliverables.

Prerequisite (explicit): SL 3 (Advanced Builder) is REQUIRED — not recommended.

Advanced track: Upon completing SL 4J, qualified Technical PMs should pursue **SL 5J (Advanced Program Manager)** for advanced topics including multi-program portfolio management, enterprise data governance leadership, cross-command integration coordination, and MSS capability roadmap ownership.

WFF awareness: The Technical PM coordinates delivery of data products consumed by WFF-qualified users (SL 4A through SL 4F — Intelligence, Fires, Movement and Maneuver, Sustainment, Protection, and Mission Command). Understanding which WFF function a product serves is not optional context — it determines the operational consequence of delivery delays, quality failures, and scope changes. A readiness dashboard degrading for a Sustainment (SL 4D) staff section has a different operational risk profile than a training schedule tool.

1-5. Governing References

The following publications establish the policy, architectural, and governance framework within which SL 4J program managers operate. Familiarity with these references is required — not recommended.

Publication	Title	Relevance
Army CIO Memorandum	Data and Analytics Policy (April 2024)	Data governance authority
USAREUR-AF C2DAO Guidance	Command governance for data operations	Operational governance
Army DIR 2024-03	Digital Engineering Policy	Army digital transformation directive
AR 25-1	Army Information Technology	IT governance and data management policy
DDOF Playbook v2.2	Data and Digital Operations Framework (T2COM C2DAO, December 2025)	DDOF roles, phases, gates, and Friction Matrix
learn-data.armydev.com	CDA Portal	Training platform reference

1-5a. Strategic Guidance

The following are strategic guidance documents — not doctrine — that inform MSS training design and operational context.

Document	Authority	Relevance
UDRA v1.1	Unified Data Reference Architecture (February 2025)	Technical reference architecture
DoD Data Strategy	DoD Data Strategy (2020)	Enterprise data management framework

CHAPTER 2 — AGILE PROJECT MANAGEMENT FOR DATA AND AI PROJECTS

2-1. Why Agile for Data Projects

BLUF: Traditional waterfall project management fails for data and AI projects because requirements evolve as the data is understood. Agile methods — particularly Scrum and Kanban — accommodate the iterative, discovery-driven nature of data work.

Data and AI projects have characteristics that make fixed-scope, fixed-timeline planning unreliable:

- **Data availability is unknown until the data is examined.** A pipeline that looks straightforward may hit schema changes, missing records, or access barriers that were not visible during scoping.
- **Model performance is uncertain.** An ML model may require three iterations to reach acceptable accuracy. The PM cannot know the final timeline until the data is cleaned, features are engineered, and baselines are established.
- **Operational requirements change.** Commanders get new missions. Staff priorities shift. A dashboard built for one reporting cycle may need to change format before the next cycle.
- **Technical dependencies are interdependent.** A data pipeline feeds a model that feeds a dashboard. A block at any layer propagates up. Fixed waterfall schedules cannot absorb these cascading delays without scope cuts or deadline slips.

Agile methods address these realities by delivering value in short increments (sprints), surfacing blockers early (daily standups), and recalibrating scope and priorities based on what the team learns each sprint.

NOTE

Agile does not mean undisciplined. It means structured iteration with explicit decisions about scope and priority at each cycle boundary. The PM is responsible for that discipline. Undisciplined "agile" is just disorganized waterfall.

2-2. Scrum Framework for MSS Projects

Conditions

The PM is standing up a new data or AI project on MSS with a team of two to eight personnel across SL 4 tracks.

Standards

The PM establishes a Scrum structure that includes: defined roles (PM as Scrum Master/Product Owner), a groomed product backlog, a sprint cadence (one or two weeks), and the four core ceremonies (sprint planning, daily standup, sprint review, retrospective).

Procedure — Standing Up a Scrum Project

1. **Define the product.** Write a one-paragraph product vision statement: what the product does, who uses it, and what operational outcome it enables. Post this in the project space. All backlog prioritization decisions reference this statement.
2. **Identify team members and tracks.** List each team member and their SL 4 track. Clarify who contributes to what layer of the product (data pipeline, model, dashboard, API).
3. **Set sprint length.** Recommend: two-week sprints for projects with six or more team members or complex ML components. One-week sprints for small teams or dashboard-only projects requiring rapid stakeholder feedback.
4. **Establish sprint cadence calendar.** Block recurring times for:
5. Sprint Planning: first day of sprint, 60-90 minutes
6. Daily Standup: every working day, 15 minutes (async acceptable if team is distributed)
7. Sprint Review: last day of sprint, 30-45 minutes with stakeholder attendance
8. Retrospective: last day of sprint, 30 minutes (team only)
9. **Build the initial product backlog.** See task 2-3 for user story format. Initial backlog should contain enough stories to fill two to three sprints before the first sprint starts.

10. **Define the sprint goal format.** Each sprint has one sentence stating what the team will demonstrate at the sprint review. Example: "At the end of this sprint, stakeholders can view live equipment readiness by unit on the Workshop dashboard."
11. **Create the project tracking workspace on MSS.** See Chapter 5 for build instructions. The Scrum board must be live on MSS before sprint 1 begins — not in a spreadsheet.

> **CAUTION**
>
> Do not run sprints without a defined sprint goal. Teams without a sprint goal default to ticket-grinding with no coherent output for the stakeholder review. The sprint goal is the PM's tool for maintaining delivery focus.

2-3. Backlog Management and User Stories

2-3a. User Story Format

User stories for data products follow the standard Agile format adapted for operational context:

```
AS A [type of user — commander, analyst, logistics officer, data engineer]
I WANT [a specific capability or information need]
SO THAT [the operational or analytical outcome it enables]
```

Examples — good user stories for MSS data products:

Story	Track Owner	Size
As a G4 officer, I want a live equipment readiness dashboard filtered by subordinate unit so that I can identify maintenance bottlenecks before the morning update.	SL 4L / SL 3	M
As a data scientist, I want a cleaned and deduplicated personnel roster dataset refreshed daily so that my models have current input data.	SL 4L	S
As a G2 analyst, I want the anomaly detection model to flag records scoring above 0.85 probability with an explanation of contributing features so that I can triage alerts without reviewing all raw data.	SL 4M	L
As a battalion S6, I want a Workshop form that submits equipment status updates directly to the MSS ontology so that I no longer re-key data from paper forms.	SL 4L / SL 3	M

Bad user story patterns to reject:

- "Build the data pipeline." (No user, no outcome — this is a task, not a story)
- "Improve model performance." (No specific acceptance criteria — unmeasurable)

- "Fix the dashboard." (No description of what is broken or what done looks like)

2-3b. Acceptance Criteria

Every user story requires explicit acceptance criteria (AC). AC are the conditions the PM uses to determine whether a story is complete. Write AC as testable statements.

Format:

```
GIVEN [the system state or precondition]
WHEN [the user takes a specific action]
THEN [the expected result that confirms the story is done]
```

Example — acceptance criteria for the G4 readiness dashboard story:

```
GIVEN I am logged into MSS Workshop with G4 role
WHEN I open the Equipment Readiness Dashboard
THEN I see a filterable table of readiness rates by subordinate unit, updated within
the last
24 hours, with drill-down to individual equipment records
AND records with readiness below 70% are flagged in red
AND I can export the current view to PDF without requesting developer assistance
```

NOTE

The PM writes acceptance criteria in collaboration with the stakeholder — not in isolation. AC written without stakeholder input often miss the actual operational requirement. Always walk the stakeholder through the AC before the story enters the sprint.

2-3c. Story Sizing and Velocity

Size stories using T-shirt sizing (S/M/L/XL) or story points (Fibonacci: 1/2/3/5/8/13). For MSS projects, T-shirt sizing is recommended for teams new to Agile:

Size	Description	Typical Duration
S	Single-layer change: one dataset, one Workshop widget, one configuration	<1 day
M	Two-layer change: pipeline + dashboard, or model update + evaluation	1-3 days
L	Multi-layer: new ontology type + pipeline + dashboard + access control	3-7 days
XL	New capability end-to-end: new data source, model, product, governance	>1 sprint — decompose

CAUTION

Stories sized XL should be decomposed before entering the sprint. An undecomposed XL story in a sprint almost always carries over unfinished. Decompose into L or M stories with their own acceptance criteria.

Velocity: Track how many story points or T-shirt sizes the team completes each sprint. After three sprints, use average velocity to forecast delivery dates. Do not commit to external stakeholder deadlines based on sprint 1 velocity alone — it takes three to four sprints for a team to establish a reliable throughput baseline.

2-4. Kanban for Operational Support Work

Kanban is appropriate for MSS teams performing ongoing support and enhancement work rather than building a defined new product. Use Kanban when:

- The team is in sustainment mode for a production product
- Work arrives continuously from multiple stakeholders at unpredictable rates
- There is no defined end state or launch date

Kanban Board Structure for MSS Support Teams

Column	Definition
Backlog	All requested work, unscheduled
Ready	Groomed, sized, assigned, has acceptance criteria — ready to start
In Progress	Actively being worked — WIP limit: 2 per developer
In Review	Built, pending PM or peer review before promoting to production
Done	Accepted by stakeholder and promoted to production (or closed as won't do)

WIP Limits. Work-in-progress (WIP) limits are the single most important Kanban discipline. A developer with five items In Progress finishes nothing. WIP limit of two per developer forces completion before starting new work. Enforce WIP limits. Resist requests to bypass them.

2-5. Sprint Ceremonies — Execution Standards

2-5a. Sprint Planning

Duration: 60 minutes for a one-week sprint; 90 minutes for a two-week sprint.

Procedure:

1. PM presents the sprint goal (one sentence).
2. PM walks the top backlog items in priority order. Team asks clarifying questions.
3. Each team member pulls stories they are committing to deliver by sprint end.
4. Team confirms: does the sum of committed stories match the sprint goal?
5. PM records the committed sprint scope in the MSS tracking board (Chapter 5).
6. Close: team and PM confirm the sprint start date, standup time, and review date.

> **NOTE**
>
> Sprint planning is not a negotiation session. The PM holds final authority on priority order. Developers hold final authority on technical feasibility within the sprint. If a developer says a story cannot be completed in the sprint as scoped, the PM either rescopes the story or moves it to the next sprint. Do not override developer effort estimates.

2-5b. Daily Standup

Duration: 15 minutes. Timebox strictly.

Format: Each team member answers three questions: 1. What did I complete since last standup? 2. What will I complete before next standup? 3. What is blocking me?

The PM records blockers in the risk/dependency tracker (Chapter 6). The standup is not a status meeting for the PM — it is a synchronization mechanism for the team. Take problem-solving offline. Do not let standup run over 15 minutes regardless of how many blockers surface.

2-5c. Sprint Review

Duration: 30-45 minutes. Stakeholders attend.

Procedure:

1. PM presents the sprint goal and whether it was achieved.
2. Each developer demonstrates what they built — live in MSS, not PowerPoint slides.
3. Stakeholders provide direct feedback.
4. PM notes any stories not completed and explains whether they carry forward or are rescoped.
5. PM presents the updated product roadmap showing projected delivery for remaining features.

WARNING

Do not demo a feature that is not deployed to the review environment. Demoing locally and saying "it will work the same way in production" is a PM failure mode. The sprint review demo must use the actual deployed product.

2-5d. Retrospective

Duration: 30 minutes. Team only — no stakeholders.

Format: Three questions, time-boxed discussion, one to two action items maximum:

1. What went well this sprint? (Keep doing)
2. What did not go well? (Stop doing or change)
3. What will we try differently next sprint? (One to two specific, ownable changes)

The PM owns tracking retrospective action items. If a retro action item is not tracked and followed up, retrospectives become a venting session with no effect on team performance.

2-6. DDOF Configuration Management Friction Matrix

BLUF: The DDOF Friction Matrix is the PM's primary diagnostic tool for identifying where a data product's progression through DDOF phases is blocked. The PM completes this matrix at each gate review and uses it to focus mitigation efforts on the specific assessment area and wedge causing delay.

The DDOF Playbook defines three execution wedges, each containing two phases:

Wedge	Phase 1	Phase 2
Wedge 1 — Define	Problem Framing	Data Provisioning
Wedge 2 — Build	Data Wrangling	Development
Wedge 3 — Deliver	Test & Evaluation	Operations

The Friction Matrix assesses six areas against these three wedges. Each cell receives one rating:

- **OK** — No impediments. Work proceeds as planned.
- **FRICTION** — Impediment identified but contained. Mitigation underway. Does not block adjacent phases.
- **CASCADING FRICTION** — Impediment propagates to downstream phases or adjacent assessment areas. Requires immediate PM escalation and cross-functional coordination to resolve.

DDOF Friction Matrix Template

Assessment Area	Wedge 1 (Define)	Wedge 2 (Build)	Wedge 3 (Deliver)
Overall Phase Status			
People			
Process / Policy			
Products			
Resources			
Technology			

Procedure — Completing the Friction Matrix

1. **Complete the matrix at each DDOF gate review.** Rate every cell based on current conditions. Do not leave cells blank — an empty cell is an unassessed risk.
2. **Mark FRICTION cells with a one-line root cause.** Example: "People / Wedge 2 — FRICTION: SL 4M vacancy; no trained ML engineer available until backfill arrives."
3. **Escalate all CASCADING FRICTION cells immediately.** A cascading friction in Wedge 1 blocks Wedge 2 and Wedge 3 downstream. The PM presents cascading items to the C2DAO with a recommended mitigation and a timeline for resolution.
4. **Track friction trends across sprints.** A cell that stays at FRICTION for three or more consecutive reviews is functionally cascading — escalate it regardless of the current rating.

WARNING

Do not use the Friction Matrix as a static reporting artifact. The matrix is an active management tool. If the PM completes it but takes no action on FRICTION or CASCADING FRICTION cells, the matrix provides no value. Every non-OK cell requires a documented mitigation action and an owner.

Source: DDOF Playbook v2.2 (p.15), T2COM C2DAO, December 2025.

CHAPTER 3 — MANAGING ML AND AI PROJECT LIFECYCLES

3-1. The ML/AI Project Lifecycle

BLUF: ML and AI projects do not follow the same lifecycle as dashboard or pipeline builds. They have a research-to-production funnel that the PM must manage explicitly. Most ML projects fail not because the model is bad — they fail because the PM did not manage the funnel correctly.

The MSS ML/AI lifecycle has six phases:

```
[1] PROBLEM DEFINITION → [2] DATA AUDIT → [3] PROTOTYPE → [4] EVALUATION → [5] PRODUCTION → [6] SUSTAINMENT
```

Each phase has a PM gate — a decision point where the PM reviews outputs and decides whether to proceed, iterate, or stop.

3-2. Phase 1 — Problem Definition

Conditions

The PM has received a stakeholder requirement for an AI or ML capability and must determine whether it is technically feasible and operationally scoped correctly before committing development resources.

Standards

The PM produces a one-page Problem Definition document containing: the operational question the model must answer, the input data sources, the output format, the success metric, and the recommended SL 4 tracks to assign.

Procedure

1. **State the operational question precisely.** "Use AI to help G2" is not a problem definition. "Predict which equipment records are most likely to require unscheduled maintenance in the next 30 days, ranked by probability, so the G4 can pre-position maintenance teams" is a problem definition.
2. **Identify the input data.** What datasets does the model need? Where do they live on MSS? What is the refresh rate? Who owns them? Pull in SL 4K (KM) to validate data availability and quality before committing to a model approach.
3. **Define the output format.** A ranked list? A probability score per record? A binary flag? A natural language explanation? The output format drives the model architecture and the Workshop display design. Define it before building.
4. **Define the success metric.** How will the PM know the model is good enough to release? Examples: precision/recall thresholds, false positive rate limits, agreement rate with subject matter expert review, stakeholder acceptance rate in evaluation. Write this down before modeling begins — do not let SL 4M define done post hoc.
5. **Assign tracks.** Typical ML project assignment:
6. SL 4M: model development, training pipeline, evaluation

7. SL 4L: data pipeline, feature engineering code
8. SL 4G: baseline statistical analysis, performance benchmarking
9. SL 4H: AIP integration if model output feeds an AI agent or workflow
10. SL 4K: data dictionary, ontology objects for model inputs/outputs
11. SL 4J (PM): project tracking, stakeholder communication, gate reviews

> **NOTE**
>
> Engage SL 4K at problem definition, not after the model is built. Data quality issues surface early if the KM is involved in assessing input sources. Discovering a critical data quality problem after two sprints of model development is a PM failure.

3-3. Phase 2 — Data Audit

Conditions

The team has identified input data sources for the ML project and must assess whether the data is sufficient to train, validate, and run the model in production.

Standards

SL 4M and SL 4K deliver a Data Audit Report. The PM reviews the report and makes a go/no-go decision before authorizing prototype development.

Procedure

1. **Pull SL 4M and SL 4K to audit each data source.** For each source, document:

Attribute	Question to Answer
Completeness	What percentage of records have the key fields populated?
Timeliness	How current is the data? What is the refresh lag?
Historical depth	How far back does history go? Is it sufficient for training?
Label availability	For supervised models: are ground truth labels available?
Class balance	For classification: are rare events represented in sufficient volume?
Access	Does the team have read access to the source dataset on MSS?
Schema stability	Has the schema changed in the last 12 months? Will it change again?

1. **PM gate — Data Audit Review.** Review the completed Data Audit Report. Gate criteria:
2. If critical fields are less than 80% complete: stop. Require data remediation before proceeding. Assign SL 4L to build a data quality pipeline.
3. If historical depth is insufficient for training: stop. Present the constraint to the stakeholder. Offer alternative: rule-based heuristics instead of ML, or a simpler statistical model that requires less history.
4. If labels are unavailable for a supervised approach: reassess model type. Unsupervised or semi-supervised methods may apply; engage SL 4M for recommendation.
5. If access is blocked: escalate to C2DAO immediately. Do not let access blockers sit.
6. **Document the data audit decision.** Record in the project tracker: data sources approved, known quality limitations, and any data remediation tasks added to the backlog.

WARNING

Skipping the data audit and proceeding directly to prototype is the most common cause of ML project failure. A model trained on incomplete or misunderstood data will not generalize. The PM who authorizes prototype work without a data audit owns the subsequent failure.

3-4. Phase 3 — Prototype

Conditions

The data audit is complete. Data sources are accessible, sufficiently complete, and historically deep enough to support model training. The prototype sprint begins.

Standards

At the end of the prototype phase, SL 4M delivers a working model that can run on the approved input data and produce output in the defined format, with preliminary performance metrics against the defined success criteria.

Procedure — PM Actions During Prototype Phase

1. **Scope the prototype sprint(s) tightly.** The prototype is not the production model. It is a proof of feasibility. Resist stakeholder requests to add features, expand the data scope, or integrate the prototype with production Workshop before evaluation is complete.
2. **Protect SL 4M from distraction.** Model development requires sustained focus. Do not pull SL 4M into stakeholder demonstrations, unrelated tasks, or data pipeline work during prototype sprints unless it is directly required for the model.

3. **Check in daily on blockers — not on model progress.** Ask: "Are you blocked on anything?" not "How accurate is the model today?" Premature accuracy questions push developers toward overfitting and rushed evaluation.
4. **Track prototype milestones:**
5. Milestone 1: First model version runs end-to-end on training data
6. Milestone 2: Initial performance metrics generated against holdout dataset
7. Milestone 3: Model output presented to PM in defined output format
8. **Do not show prototype outputs to operational stakeholders.** A prototype is not a product. Showing prototype model outputs to a commander before evaluation is complete risks anchoring the stakeholder on incomplete results.

3-5. Phase 4 — Evaluation

Conditions

The prototype model produces output. The PM must now determine whether the model meets the defined success metric and is ready to enter a production readiness review.

Standards

The PM conducts an Evaluation Review with SL 4M, SL 4G (for independent statistical assessment), and at least one operational subject matter expert (SME). The review produces a documented pass/fail/iterate decision.

Procedure

1. **Assemble the evaluation package.** SL 4M delivers:
2. Performance metrics (precision, recall, F1, AUC, or task-appropriate metrics)
3. Confusion matrix or error analysis
4. Feature importance or model explanation summary
5. Examples of correct predictions and failure cases
6. **Assign SL 4G to conduct independent validation.** SL 4G (ORSA) independently reviews the methodology and performance claims. The ORSA does not re-train the model — they validate the evaluation process and flag any statistical concerns.

7. **Conduct SME review.** Present the model outputs to an operational SME (the stakeholder's representative who understands the domain). Ask: Do these outputs make operational sense? Are the false positives operationally tolerable? Are the false negatives operationally tolerable?
8. **PM Evaluation Gate decision:**

Outcome	Condition	PM Action
Pass	Metrics meet threshold; SME accepts output quality	Proceed to Phase 5 (Production)
Iterate	Metrics near threshold; specific failure mode identified and fixable	Return to prototype with scoped improvement task
Stop — Redesign	Fundamental data or approach problem; metrics far from threshold	Restart at Phase 1 or 2 with revised approach
Stop — Descope	Model not feasible given data; simpler heuristic approach better	Descope ML; proceed with rule-based product

CAUTION

Do not iterate indefinitely. Set a maximum of two iteration cycles before making a Stop decision. Unlimited iteration on a failing model is a resource drain and delays delivery of an alternative solution that might actually work.

3-6. Phase 5 — Production

NOTE — Palantir Developers reference: *Lennar: Scaling from Pilot to Production with Foundry* — A real case study of scaling a Foundry deployment from early prototype through production, covering the organizational and technical gates required at each transition. Directly reinforces the pilot-to-production lifecycle gate content in this section. Available on the Palantir Developers YouTube channel (@PalantirDevelopers).

Conditions

The model passes evaluation. The PM must now manage the production deployment process.

Standards

The PM completes the Definition of Done checklist (Appendix B), coordinates a production release plan with SL 4L and SL 4M, and communicates the release to operational users before deployment.

Procedure

1. **Complete the Definition of Done checklist.** Do not abbreviate or skip items. If any DoD item is not satisfied, the release does not proceed.
2. **Build the production monitoring plan.** Define:
3. How will model performance be monitored in production? (Input data distribution shifts, output confidence score distribution, user override rate)
4. Who is the model owner responsible for monitoring? (Typically SL 4M)
5. What threshold triggers a model review? (Define numerically — not "if it seems off")
6. What is the rollback procedure if the model degrades?
7. **Coordinate the deployment window.** Schedule the production deployment during a low- operational-tempo window. Notify downstream users 48 hours in advance per the Safety Summary.
8. **Execute the deployment with SL 4L and SL 4M present.** Run the deployment against the production environment. Validate model outputs after deployment before declaring success.
9. **Brief the operational stakeholder.** Walk the stakeholder through the production product: how to read the outputs, what confidence thresholds mean, how to report anomalies, and who to contact if the model produces results that seem wrong.

3-7. Phase 6 — Sustainment

NOTE — Palantir Developers reference: *Code in Production: HCA Deploying AI in Healthcare at Scale | DevCon 3* — Documents a large-scale AI deployment in a production environment, covering the monitoring, governance, and sustainment considerations that apply at production scale. Reinforces the PM's sustainment responsibilities for AI models in this phase. Available on the Palantir Developers YouTube channel (@PalantirDevelopers).

BLUF: A model in production is not done — it is a system that requires active management. Model performance degrades as input data drifts, operational contexts change, and upstream data sources change schema or refresh rates.

PM Sustainment Responsibilities

1. **Schedule quarterly model reviews.** SL 4M reviews performance metrics quarterly. PM reviews the report and decides: sustain, retrain, or retire.
2. **Track upstream data source changes.** If a SL 4L pipeline engineer or a SL 4K KM flags that an input data source has changed schema or refresh rate, escalate immediately. A changed input can silently degrade model performance without triggering any obvious error.

3. **Maintain model version tracking on MSS.** Model deployment in Foundry follows this pattern: batch inference transforms write predictions to a Foundry dataset, which then serves as the source for computed properties on Object Types, or models are integrated into AIP Logic workflows. The SL 4M model owner must keep the model card and associated dataset documentation current — reflecting version, training date, performance metrics, and owner — after every retrain.
4. **Manage model retirement.** When a model is retired (superseded, no longer operationally relevant, or failing sustainment review), the PM coordinates: stakeholder notification, shutdown of the Workshop display, archival of the model artifacts, and documentation of the retirement decision.

CHAPTER 4 — STAKEHOLDER MANAGEMENT AND REQUIREMENTS TRANSLATION

4-1. The Translation Problem

BLUF: The PM's most critical skill is translating between two languages: the operational language of commanders and staff and the technical language of developers and data scientists. Failure to translate accurately in both directions is the leading cause of delivering the wrong product.

The translation problem manifests in two directions:

Direction 1 — Operational to Technical: A commander says "I need better situational awareness on logistics." The PM must translate this into: a specific dataset, a refresh cadence, a display format, a user story, and acceptance criteria that SL 4L can build.

Direction 2 — Technical to Operational: A SL 4M says "the model has an AUC of 0.87 but the recall on the minority class is 0.61 at the 0.5 threshold." The PM must translate this into: "The model correctly identifies roughly six out of every ten high-risk events, which means it misses four. For this use case, missing four out of ten is operationally acceptable because the cost of a false alarm is low."

Neither translation is automatic. Both require the PM to understand enough of both domains to bridge them accurately.

4-2. Requirements Elicitation

Conditions

The PM has been assigned to scope a new data or AI capability. The operational stakeholder has expressed a requirement in vague or general terms.

Standards

The PM produces a Requirements Document containing: the stakeholder's operational need, the current workaround or pain point, the proposed MSS solution, the input data required, the output format, the users and roles, the success metric, and the out-of-scope boundaries.

Procedure

1. **Conduct a discovery interview with the operational stakeholder.** Ask:
2. Walk me through your current workflow for [the task]. What are you doing today?
3. Where does that process break down or slow down?
4. What would done look like? What would you be able to do that you cannot do today?
5. Who else uses this information? Who are the downstream consumers?
6. How often does this need to update? Real-time, daily, weekly?
7. What would make you not use this product? What would make it wrong?
8. **Resist accepting the stakeholder's proposed solution as the requirement.** Stakeholders often bring a solution ("I need a dashboard") when the PM needs to understand the problem ("I cannot track equipment readiness across all subordinate units without calling each S4"). The PM's job is to validate whether the proposed solution actually solves the problem.
9. **Draft the Requirements Document.** Use the template:

```
REQUIREMENT: [One sentence — what operational outcome the product enables] STAKEHOLDER: [Name, role, unit] CURRENT STATE: [How the stakeholder handles this today — the workaround or manual process] PROPOSED SOLUTION: [What will be built on MSS] INPUT DATA: [Datasets required — names, sources, owners, refresh rates] OUTPUT FORMAT: [Dashboard / alert / export / API / model score — be specific] USERS AND ROLES: [Who will use the product — by role, not by name] SUCCESS METRIC: [How the PM and stakeholder will know it is working] OUT OF SCOPE: [What this product will NOT do — prevents scope creep] PRIORITY: [Must have / Should have / Nice to have]
```

1. **Review the Requirements Document with the stakeholder before starting work.** Read back the requirement, success metric, and out-of-scope section explicitly. Confirm agreement before the document is baselined and backlog items are created.

> **NOTE**
>
> A signed-off Requirements Document does not prevent requirements from changing. It creates a baseline for managing change. When a stakeholder requests a scope change, the PM compares the request to the baselined requirements, estimates impact, and presents options — not just "yes" and not just "no."

4-3. Managing Stakeholder Expectations

4-3a. Setting Delivery Expectations

Do not commit to a delivery date before three sprints of velocity data exist. Use this language with stakeholders:

Stage	What to Say
Project kickoff	"We will have an initial estimate after two sprints. First sprint is scoping and data validation."
After sprint 1	"Still building velocity baseline. Initial sense is [X sprints] but that may change."
After sprint 3	"Based on velocity, we project delivery of [feature set] by [date range] +/- one sprint."
Confirmed forecast	"We are committed to [feature set] by [date]. Scope changes after this point require a tradeoff."

4-3b. Stakeholder Communication Cadence

Stakeholder Type	Recommended Touchpoint	Format
Direct operational user	Sprint review attendance (every sprint)	Live demo on MSS
Commanding officer	Monthly project status update	MSS dashboard (Chapter 5) — not a slide deck
Technical leadership	Sprint review + async update in project channel	MSS board + brief written summary
C2DAO / platform team	As needed for governance events (access, ontology, deploys)	Coordination via C2DAO workflow

NOTE

Commander-level project status should be served from a live MSS Workshop dashboard, not a PowerPoint deck. A static briefing slide is stale the moment it is printed. A live dashboard shows real data. Chapter 5 covers building the commander-facing project status dashboard.

4-4. Managing the Technical Team

4-4a. PM Authority vs. Technical Authority

The PM holds authority over: scope, priority, stakeholder communication, release timing, and project tracking. The PM does not hold authority over: how a developer implements a feature, which algorithm a SL 4M uses, or how SL 4L structures code.

Attempting to micro-manage technical implementation decisions creates friction, reduces team ownership, and produces worse outcomes. Delegate technical decisions to the appropriate SL 4 track and hold them accountable to the acceptance criteria — not to the implementation method.

4-4b. Protecting the Team from Scope Sprawl

Scope sprawl is the most common project killer in operational data environments. Stakeholders who see a working dashboard immediately want to add filters, additional datasets, new metrics, and new features — before the current sprint is complete.

PM procedure for incoming scope requests during a sprint:

1. Record the request in the backlog immediately.
2. Acknowledge the stakeholder: "Logged. We will assess priority in sprint planning."
3. Do not add the request to the current sprint unless it is a blocking defect.
4. At sprint planning, assess priority against current backlog. Let the stakeholder make the tradeoff: "If we add this to the sprint, we move [existing item] to the next sprint. Which do you prefer?"

4-4c. Cross-Track Coordination

When a story requires multiple SL 4 tracks, the PM must sequence the work and manage the hand-off points:

Dependency Pattern	PM Action
SL 4L pipeline must complete before SL 4M can train	Make pipeline story a sprint 1 item; model story a sprint 2 item with explicit dependency noted
SL 4K ontology object must exist before SL 4L can write to it	SL 4K story goes in sprint backlog; SL 4L story blocked until merged
SL 4H agent workflow depends on SL 4M model output format	PM defines the output schema interface in sprint planning; both tracks work to that contract
SL 3 Workshop dashboard depends on SL 4L pipeline	Pipeline completes and is validated before Workshop build begins

CHAPTER 5 — BUILDING PROJECT TRACKING SYSTEMS ON MSS

NOTE — Palantir Developers reference: *Product Launch: Managing Production Applications with Workflow Builder* — Covers Palantir's Workflow Builder tool for managing production data products, including status tracking, handoffs, and operational visibility. Directly relevant to the Workshop-based tracking systems built in this chapter. Available on the Palantir Developers YouTube channel (@PalantirDevelopers).

NOTE — Palantir Developers reference: *Product Launch: Workflow Builder | DevCon 4* — A second launch reference for Workflow Builder with additional implementation depth. Useful companion to the section above for PMs standing up new project tracking infrastructure. Available on the Palantir Developers YouTube channel (@PalantirDevelopers).

5-1. Why Track on MSS, Not Spreadsheets

BLUF: Project tracking on MSS provides live visibility to all stakeholders without the version-control and distribution problems of spreadsheets. A Workshop-based project tracker is the single source of truth, accessible to every team member with the correct Foundry role, updated in real time.

A Jira-style spreadsheet or SharePoint tracker has the following failure modes in USAREUR-AF operations:

- Multiple versions in circulation with no authoritative copy
- Commander-level visibility requires manual compilation into a briefing slide
- Status updates depend on someone remembering to update a cell
- No automated alert when a milestone slips or a blocker exceeds SLA

An MSS-based tracker eliminates all of these. This chapter covers the procedure for building it.

5-2. Designing the Project Tracker Ontology

Conditions

The PM is setting up a new project on MSS and needs a tracking infrastructure. This task is performed in coordination with SL 4K (Knowledge Manager) if a new ontology design is needed, or by the PM using an existing project tracker template if one is available in the USAREUR-AF project tracker library.

Standards

The project tracker Ontology contains the following Object Types at minimum: Project, Sprint, Story (or Task), Team Member, Risk, and Dependency. Links connect Stories to Sprint, Sprint to Project, and Team Member to Stories (as assignee).

Procedure — Ontology Design for Project Tracker

1. **Check the USAREUR-AF project tracker library first.** The C2DAO maintains reusable Ontology templates. Do not rebuild from scratch if a validated template exists.
2. **Define the Object Types.** For each type, define the properties:

Project Object Type

Property	Type	Notes
project_name	String	Short name, used as display label
product_vision	String	One-paragraph product vision statement
pm_owner	String	PM name and SL 4J track
start_date	Date	
target_end_date	Date	Expected delivery — not a commitment
current_status	Enum	On Track / At Risk / Blocked / Complete
stakeholder_org	String	Unit or directorate of primary stakeholder

Sprint Object Type

Property	Type	Notes
sprint_name	String	e.g., "Sprint 4 — 10-21 Mar 2026"
sprint_goal	String	One sentence
start_date	Date	
end_date	Date	
velocity_planned	Integer	Story points or T-shirt size count planned
velocity_actual	Integer	Completed at sprint end
status	Enum	Planning / Active / Review / Complete

Story Object Type

Property	Type	Notes
story_id	String	Unique ID (auto-generated or manual)
story_title	String	Short title for board display
story_text	String	Full "As a / I want / So that" text
acceptance_criteria	String	Full AC text
size	Enum	S / M / L / XL
status	Enum	Backlog / Ready / In Progress / In Review / Done
assignee	String	SL 4 track + name
track	Enum	40A / 40B / 40C / 40D / 40E / 40F / 40G / 40H / 40M / 40J / 40K / 40L / 40N / 40O / 30
blocked	Boolean	True if blocker exists
blocker_description	String	Description of blocker (if blocked = true)
sprint_link	Link	→ Sprint

Risk Object Type — see Chapter 6.

Dependency Object Type — see Chapter 6.

1. **Create the Object Types in the Foundry Ontology UI.** Use the UI to create types and properties. Coordinate with SL 4K if the project tracker will share any Object Types with the enterprise ontology (e.g., linking Team Members to an existing Personnel object).
2. **Configure write access for all team members.** Each team member must have write access to update their own Story records. The PM configures role-based access in the Foundry project settings.

NOTE

If SL 4K has already designed a standard project tracker ontology for USAREUR-AF, use it. Ontology proliferation — every PM designing their own object types — creates governance debt that SL 4K must clean up. Check before designing.

5-3. Building the Sprint Board in Workshop

Conditions

The Ontology is defined and populated with at least one Sprint and its associated Stories. The PM is building the Scrum board view in Foundry Workshop.

Standards

The Workshop application displays: the current sprint goal, a Kanban-style board with Story cards grouped by status column, a blockers panel, and a velocity chart. The board updates in real time as team members change Story status.

Procedure

1. **Create a new Workshop application** in the MSS project space. Name it: `[Project Name] — Sprint Board`.
2. **Add a Sprint Selector widget** at the top of the page. This allows users to switch between viewing the current sprint and historical sprints for reference.
3. **Add a Kanban Board widget** linked to the Story Object Type, filtered to the selected sprint. Configure columns: Backlog | Ready | In Progress | In Review | Done. Each card shows: story_title, assignee, size, and a red indicator if blocked = true.
4. **Add a Sprint Goal banner** at the top: a text widget that displays the sprint_goal property of the selected Sprint object.
5. **Add a Blockers Panel** below the Kanban board. Filter Stories where blocked = true. Display: story_title, assignee, blocker_description. This panel is visible to all stakeholders and creates accountability for blocker resolution.
6. **Add a Velocity Chart** on a second Workshop tab. Display a bar chart of velocity_planned vs. velocity_actual for all completed sprints in the project. This is the PM's forecasting tool.
7. **Configure Workshop publication permissions.** The sprint board should be visible to: the full project team (read/write for own stories), the PM (read/write all), and the stakeholder (read-only). Do not publish to a wider audience than required — story-level tracking data may contain information about resourcing and capability gaps.

CAUTION

Do not use the same Workshop application for internal team tracking and commander-facing status reporting. These are different audiences with different information needs. Build separate applications — see 5-4 for commander-facing status.

5-4. Commander-Facing Project Status Dashboard

Conditions

The PM must provide project status visibility to a commanding officer, directorate chief, or senior leader who does not need sprint-level detail but requires accurate project health information.

Standards

The commander-facing dashboard displays: project name, current status (On Track / At Risk / Blocked), sprint velocity trend, top risks (by severity), and projected delivery milestones. The dashboard is designed for a 90-second read in a senior leader's workspace.

Design Principles for Senior Leader Dashboards

Principle	Application
BLUF first	Status indicator (green/amber/red) visible at top of page, no scrolling required
Numbers over narrative	Show metrics — velocity, risk count, days to milestone — not prose explanations
Drill-down, don't flatten	Summary view on top; detail available on click/expand — not all on one page
Current as of label	Always display "Data as of [timestamp]" — never let staleness go unlabeled
No jargon	Story points, sprint velocity, AUC — not visible on commander dashboard. Translate.

Procedure

1. **Create a new Workshop application** named: `[Project Name] — Status`.
2. **Add a Status Header block** at the top:
3. Project name (large text)
4. Current status indicator: a colored badge (green/amber/red) driven by the `current_status` property on the Project object
5. "As of [last_updated]" timestamp
6. **Add a Milestone Table** showing the next three to five major milestones:
7. Milestone name
8. Target date
9. Status (On Track / At Risk / Complete)

10. Color-coded by status
11. **Add a Progress Summary panel:**
12. Stories completed / total stories (shown as a fraction and progress bar)
13. Current sprint number and goal (one sentence)
14. Projected completion date range (derived from velocity trend)
15. **Add a Top Risks panel** showing the top three risks by severity score (Chapter 6). Display: risk description, severity, and current mitigation status. If no critical risks exist, show "No critical risks currently identified."
16. **Set publication permissions** to read-only for all stakeholder roles. The PM updates the Project object status field — the dashboard reflects the change automatically.

> **NOTE**
>
> The commander dashboard does not replace verbal briefs. It supplements them. When a senior leader asks for a project update, the PM points to the live dashboard rather than producing a new slide. This builds trust in MSS as the authoritative source.

5-5. Automated Status Alerts

Conditions

The PM wants to receive automated notification when a blocker exceeds a defined SLA, a milestone passes its target date without being marked complete, or project status changes.

Standards

The PM configures at least two automated alerts using MSS Pipeline Builder or AIP Logic: a blocker-aging alert (fires when a story has been blocked for more than 48 hours) and a milestone-slip alert (fires when today's date exceeds a milestone target date and status is not Complete).

Procedure

1. **Coordinate with SL 4H** to configure the alert logic using AIP Logic or a lightweight Pipeline Builder scheduled transform.
2. **Define the alert recipients** in the notification configuration. Do not configure alerts to send to commanders or senior leaders directly — route to the PM, who then decides whether to escalate.

3. **Test the alerts** in a non-production sprint before relying on them in a live project. Verify that the trigger condition fires correctly and that the notification reaches the intended recipient.

CHAPTER 6 — RISK AND DEPENDENCY MANAGEMENT

6-1. Risk Management for Data Projects

BLUF: Data and AI projects have a specific risk profile that differs from general software projects. The PM must manage technical risks, data risks, and operational risks simultaneously.

The three risk domains for MSS data/AI projects:

Domain	Examples
Data	Source dataset unavailable, schema changed, quality below threshold, access blocked
Technical	Model accuracy insufficient, pipeline latency too high, platform capacity constraints
Operational	Stakeholder changes requirements, end users resist adoption, organizational priority shifts

6-2. Risk Register

Conditions

The PM is managing an active project and must maintain a risk register that is visible to the project team and relevant stakeholders.

Standards

The risk register is an MSS Ontology Object Type populated with all identified risks. Each risk has: description, domain, likelihood, impact, severity score, owner, mitigation strategy, and current status. The register is reviewed at every sprint planning session.

Risk Object Type Properties

Property	Type	Notes
risk_id	String	Auto or manual unique ID
description	String	Plain-language description of what could go wrong

Property	Type	Notes
domain	Enum	Data / Technical / Operational
likelihood	Enum	Low (1) / Medium (2) / High (3)
impact	Enum	Low (1) / Medium (2) / High (3)
severity_score	Integer	Computed: likelihood × impact (1-9)
owner	String	Team member responsible for monitoring and mitigation
mitigation	String	Specific action being taken to reduce likelihood or impact
contingency	String	What will be done if the risk materializes
status	Enum	Open / Mitigated / Accepted / Closed
last_reviewed	Date	Date of last PM review

Risk Severity Matrix

	Low Impact (1)	Medium Impact (2)	High Impact (3)
High Likely (3)	3	6	**9 — Critical**
Med Likely (2)	2	4	**6 — High**
Low Likely (1)	1	2	**3 — Moderate**

Severity 7-9 (Critical): Escalate to project sponsor/stakeholder leadership immediately. Severity 4-6 (High): Active mitigation required; review weekly. Severity 1-3 (Moderate): Monitor; review at sprint planning.

Procedure — Sprint Risk Review

1. At each sprint planning session, open the risk register in Workshop.
2. Review all Open risks with severity ≥ 4.
3. For each: has likelihood or impact changed? Has mitigation progressed?
4. Update the risk record. Change status to Mitigated if the risk has been addressed.
5. Add any new risks identified during the previous sprint.
6. Present any new Critical risks to the project sponsor before the sprint begins.

6-3. Dependency Management

BLUF: Data projects have more cross-team dependencies than most software projects. A pipeline must exist before a model can train. An Ontology type must be published before a Workshop view can display it. Managing these dependencies is a primary PM function.

Dependency Object Type Properties

Property	Type	Notes
dependency_id	String	Unique ID
description	String	What does Story A need from Story B (or external source)?
blocking_story	Link	→ Story that is blocked
provider	String	Team member or external team providing the dependency
expected_date	Date	When will the dependency be resolved?
actual_date	Date	When was it actually resolved?
status	Enum	Open / At Risk / Resolved

Common Dependency Patterns

Pattern	PM Action
Data pipeline (SL 4L) blocks model training (SL 4M)	Sequence sprints; pipeline in sprint N, model in sprint N+1
External data source access blocked by C2DAO	Escalate at sprint planning; do not let access requests age
SL 4K ontology design blocks SL 4L coding	Start ontology design in sprint 1 even if build is sprint 2 work
SL 4H agent depends on SL 4M model endpoint	Define API contract in sprint planning; both tracks work to spec
MSS platform upgrade changes Workshop component API	Track platform release calendar; coordinate with SL 4L on timing

WARNING

Do not assume external dependencies will resolve on schedule. Treat all external dependencies (C2DAO access requests, data source onboarding, platform upgrades) as risks with a separate risk record. If an external dependency is on the critical path, escalate immediately — do not wait for the blocked sprint to surface it.

6-4. Technical Debt Management

Technical debt accumulates in data projects when shortcuts are taken to meet delivery dates: hardcoded values instead of config files, missing documentation, untested pipeline logic, a model deployed without a monitoring plan, or Workshop widgets that bypass the Ontology.

The PM is responsible for managing the technical debt backlog — not just the feature backlog.

Procedure

1. **Create a "Tech Debt" label or tag in the Story tracker.** Any story addressing technical debt is tagged so it can be tracked separately from feature work.
2. **Reserve 15-20% of each sprint's capacity for tech debt.** Do not let technical debt stories compete with feature stories without a dedicated capacity allocation. Stakeholders will always want features; tech debt will never win if unprotected.
3. **Conduct a tech debt review at each quarterly retrospective.** SL 4L and SL 4M identify the highest-impact debt items. PM prioritizes the top three into the next quarter's capacity.
4. **Never allow tech debt to accumulate in production monitoring.** An ML model in production without a monitoring dashboard is not a debt item — it is an active risk. Treat it as severity-6 risk (Chapter 6-2).

CHAPTER 7 — DELIVERY PLANNING AND PRODUCTION READINESS

NOTE — Palantir Developers reference: *DTN: Beyond Building: Accelerating Value Delivery with Foundry* — A real-world case study on accelerating data product value delivery from build through operational adoption, covering the delivery planning decisions that determine whether a product reaches its intended users. Reinforces the scope, milestone, and release planning content in this chapter. Available on the Palantir Developers YouTube channel (@PalantirDevelopers).

NOTE — Palantir Developers reference: *TWG + Palantir: Minutes Not Months | DevCon 2* — Covers rapid deployment strategies and techniques for compressing the time from requirement to operational capability. Useful context for PMs under timeline pressure planning a production release. Available on the Palantir Developers YouTube channel (@PalantirDevelopers).

7-1. Scope, Timeline, and Quality Tradeoffs

BLUF: The PM cannot simultaneously guarantee scope, timeline, and quality. At any decision point, one of these three is the constraint that others must flex around. Define the constraint explicitly before making delivery tradeoff decisions.

Constraint Mode	Fixed	Can Flex	When to Use
Fixed Deadline	Timeline	Scope (cut features) or Quality risk	Hard operational deadline (exercise, OPORD, command review)
Fixed Scope	Scope	Timeline (slip the date)	Safety-critical product where partial is unacceptable
Fixed Quality	Quality	Timeline and scope	Production ML model or regulated data product

The PM must communicate the constraint mode to both the technical team and the stakeholder. Failing to name the constraint results in the team trying to hold all three — which produces a late, descoped, low-quality product.

7-2. Release Planning

7-2a. Milestone Types

Milestone Type	Definition
Internal Milestone	Checkpoint within the team (data audit complete, model trained)
Alpha Release	First build available for stakeholder feedback — incomplete, unstable
Beta Release	Feature-complete build for operational user testing — not production
Production Release	DoD-complete, validated, deployed to production audience
Patch Release	Bug fix or minor update to a production product
Major Release	Significant new capability added to an existing production product

7-2b. Release Planning Procedure

1. **Define release milestones at project kickoff.** Include them in the commander-facing dashboard from day one. Milestone dates are projections — flag them as projections until three-sprint velocity baseline establishes a credible forecast.
2. **Run a release readiness review before every production release.** The release readiness review is a PM-led meeting with SL 4L, SL 4M (if ML involved), and SL 4K. The agenda:
3. Is the Definition of Done checklist complete?
4. Are all known defects triaged — none outstanding at Critical severity?
5. Is the monitoring plan in place?
6. Is the rollback procedure documented and tested?
7. Have operational users been notified of the release?
8. **Do not release on Fridays or the day before a low-staffing period.** If a production deployment fails or produces bad data, the team must be available to respond.

7-3. Definition of Done — Data Product

The Definition of Done (DoD) is the PM's quality gate for every data product release. See Appendix B for the complete checklist. The DoD has five sections:

Section	What It Covers
Data Quality	Input data validated, lineage documented, refresh pipeline tested
Product Function	All acceptance criteria met and tested, no critical defects open
Access Control	Role-based access configured; least-privilege verified; C2DAO review completed
Documentation	User guide exists, data dictionary updated, model card complete (if ML)
Operability	Monitoring plan active, rollback procedure documented, stakeholder briefed

No section may be skipped. If any DoD item cannot be satisfied, the release is blocked until it is resolved or formally accepted as a known risk with documented rationale.

7-4. What "Production Ready" Means for a Data Product

Production ready is not a feeling. It is a checklist result.

For a dashboard or data pipeline product:

- All source data is connected to the live Ontology (not a static snapshot)

- The pipeline runs on schedule without manual intervention
- Access control is configured to production roles (not the developer's personal account)
- The product has been demonstrated to at least one operational user who confirmed it meets the requirement
- A data steward (SL 4K) has reviewed and accepted the ontology design
- The PM has completed Appendix B and no items are outstanding at Critical severity

For an ML/AI model product (in addition to the above):

- Model performance meets the defined success metric from Phase 1 (3-2)
- The model has been evaluated on data it was not trained on (holdout or production sample)
- A model card documenting training data, performance metrics, known failure modes, and appropriate use cases exists and is stored in the project space
- A monitoring plan is active and the model owner (SL 4M) has accepted monitoring responsibility
- The stakeholder has been briefed on how to interpret model outputs, including what the model does not do and what to do when the output seems wrong

---

7-5. Post-Release Review

Within two weeks of a production release, the PM conducts a Post-Release Review. This is not a retrospective on the project — it is a review of whether the product is performing as intended in production.

Post-Release Review Agenda:

1. Are users accessing the product? (Check Workshop usage analytics)
2. Are users taking the actions the product was designed to enable?
3. Have any data quality issues surfaced in production that were not caught in testing?
4. Have users reported any defects or usability problems?
5. Is the monitoring pipeline running on schedule?
6. For ML products: is the model performance distribution in production consistent with evaluation performance?

PM Action: If answers to questions 1-2 indicate the product is not being used, escalate to Chapter 8 (Change Management). Low adoption in the first two weeks after release is the leading indicator of a change management problem, not a product quality problem.

7-6. Data Product Portfolio Health Metrics

BLUF: PMs managing multiple data products track portfolio health using DDOF-aligned metrics. These metrics identify products that are stalled, degrading, or approaching retirement thresholds before they become operational gaps.

The PM maintains a portfolio health view using the following metric categories:

Products per DDOF Phase. Track how many data products are in each DDOF phase (Problem Framing, Data Provisioning, Data Wrangling, Development, Test & Evaluation, Operations). A healthy portfolio has products distributed across phases. A portfolio with all products in Development and none in Operations indicates a delivery bottleneck.

Gate Pass/Fail Rates. Track the outcome of each DDOF gate review across the portfolio. A high gate-fail rate in a specific phase signals a systemic problem — insufficient data quality processes, unclear requirements, or inadequate evaluation methodology — not just individual product issues.

Time-in-Phase vs. Standard/Maximum. Each DDOF phase has an expected duration (standard) and a maximum allowable duration. The PM tracks actual time-in-phase for each product. Products exceeding the standard duration require a documented explanation. Products exceeding the maximum trigger an escalation to the C2DAO for a continue/stop decision.

VAULTIS-A Quality Scores. Data products in Operations are assessed against the VAULTIS-A quality dimensions (Visible, Accessible, Understandable, Linked, Trusted, Interoperable, Secure, Auditable). The PM tracks these scores at quarterly reviews. A product scoring below 70% on any VAULTIS-A dimension enters a remediation cycle.

Retirement Trigger Monitoring. The PM monitors two retirement triggers for products in the Operations phase:

Trigger	Threshold	PM Action
No-access period	90 days with zero user access	Initiate retirement review with stakeholder
Extended no-access	180 days with zero user access	Recommend retirement to C2DAO; archive product
Quality degradation	VAULTIS-A score below 70% for two consecutive quarters	Initiate remediation or retirement decision

NOTE

Retirement is not failure. Data products have a lifecycle. A product built for a specific exercise, deployment, or reporting cycle may no longer be operationally relevant. The PM's responsibility is to identify products that should be retired and execute the retirement cleanly — freeing platform resources and reducing governance burden — rather than allowing unused products to accumulate.

CHAPTER 8 — CHANGE MANAGEMENT AND USER ADOPTION

8-1. Why Operational Users Resist New Data Products

BLUF: Building a technically excellent data product does not guarantee adoption. Operational users in USAREUR-AF have established workflows, high operational tempo, and limited patience for tools that require learning investment before delivering value. The PM must manage adoption actively — it does not happen automatically.

Resistance to new MSS capabilities is not irrational. It comes from:

Resistance Source	What Users Are Actually Saying
"I don't have time to learn this"	The tool's learning curve is longer than the value delivered per use
"I don't trust the data"	The data has been wrong before, or they cannot verify it
"My current process already works"	The new tool does not improve their workflow enough to justify change
"Nobody told me about this"	The deployment was silent; users discovered it without context
"It's too complicated"	The interface was designed for builders, not operators

Each of these is a PM problem — not a user problem. The PM must address the root cause, not dismiss the resistance.

8-2. Change Management Planning

Conditions

The PM has a production-ready data product that will change the workflow of an operational unit or staff element. The PM must plan and execute the transition from the old workflow to the new product.

Standards

The PM completes a Change Management Plan before the production release date. The plan covers: stakeholder mapping, communication plan, training plan, adoption metrics, and rollback communication plan.

Procedure — Change Management Plan

1. **Map the affected stakeholders.** For each group that will change their workflow:

Stakeholder Group	Current Workflow	New Workflow	Impact Level
G4 officers	Weekly Excel tracker	Live MSS dashboard	High
Unit S4s	Phone calls + email	Workshop form submission	High
G4 staff	Manual compilation	Automated pipeline output	Medium

1. **Identify change sponsors.** For high-impact stakeholder groups, identify a senior person within the group who understands the new product and will champion it internally. A change sponsor is not a trainer — they are a peer voice who signals that the new way is endorsed at the unit level.
2. **Build the communication plan.** Minimum communication events:

Timing	Event	Audience
T-30 days	Announcement: "New product coming"	All affected stakeholders
T-14 days	Demo / preview session	Key users and change sponsors
T-7 days	Training session	End users
T-2 days	Final reminder + link to user guide	All affected stakeholders
T+0 (release day)	Go-live announcement + support contact	All affected stakeholders
T+14 days	Adoption check-in / feedback session	Users + change sponsors

1. **Build the training plan.** For MSS Workshop products, training should be:
2. 30 minutes maximum for a dashboard user
3. Hands-on in MSS — not slides about MSS
4. Recorded for users who cannot attend the live session
5. Followed by a user guide (see 8-3 for user guide standards)
6. **Define adoption metrics.** How will the PM know adoption is occurring?
7. Workshop application open rate (first 30 days)
8. Form submission volume (if the product includes user input)
9. Reduction in manual workaround behavior (e.g., no more weekly Excel distribution)
10. Stakeholder self-reported satisfaction at T+30 days
11. **Define the rollback communication plan.** If the product must be pulled back or a critical defect surfaces post-release, the PM communicates to all affected stakeholders within four hours. The rollback communication includes: what happened, what users should do in the interim, and when the

fix will be deployed.

8-3. User Guide Standards

Every production MSS product requires a user guide. The user guide is not a technical architecture document — it is written for the operational user, not the developer.

User Guide Minimum Contents:

Section	Content
Purpose	One paragraph: what this product does and who it is for
Access	How to access the product on MSS (URL/link + role required)
Navigating the product	Screenshots with labeled callouts of each panel and its function
How to [Task 1]	Step-by-step instructions for the primary user task
How to [Task 2]	Repeat for each additional user task
Interpreting outputs	For ML/AI products: what the scores, flags, or recommendations mean
Known limitations	What the product does NOT do; data quality caveats; refresh lag
How to report a problem	Contact, channel, and what information to include when reporting a defect

Format standards: - Maximum 10 pages for any single user guide - Screenshots from the actual production environment — not mockups - Plain language — define any technical term used - Reviewed by one actual end user before publication

---

8-4. Managing Resistance During Rollout

When Low Adoption Surfaces at the T+14 Check-In

If the T+14 adoption check-in shows low uptake, diagnose before escalating:

1. **Is it an access problem?** Check if affected users have the correct Foundry role. Access provisioning gaps are the most common early-adoption blocker.
2. **Is it an awareness problem?** Check whether the communication reached all intended recipients. USAREUR-AF distribution lists are imperfect. Personal outreach via change sponsors often reaches users that mass distribution does not.

3. **Is it a usability problem?** Observe a user attempting to use the product without assistance. Does the interface make the primary task obvious? If users cannot complete the primary task without asking for help, the interface needs revision — not more training.
4. **Is it a trust problem?** Ask users directly: "Is there anything about the data that seems wrong or that you don't trust?" Data trust problems require evidence. Pull the data lineage, walk the user through the source-to-display chain, and demonstrate that the numbers are correct. A single trust failure that goes unaddressed destroys adoption.
5. **Is it a change sponsor problem?** If the unit-level change sponsor is not using or endorsing the product, junior users will not adopt it. Re-engage the sponsor. If the sponsor has concerns, address them before expecting the team to follow.

8-5. Platform Governance — PM Perspective

The PM is responsible for the governance of the MSS project space associated with their product. This includes:

Access Management: - Review project membership quarterly. Remove accounts of personnel who have PCS'd, completed TDY, or no longer require access. - Coordinate all access additions and removals with SL 4K (KM/Data Steward) and the C2DAO. - Do not grant admin or builder access to users who only need consumer access. Apply least-privilege.

Resource Allocation: - Monitor dataset storage usage in the project. Unmanaged pipeline output accumulation can hit storage quotas and degrade platform performance for other users in the AOR. - Coordinate with SL 4L to implement dataset retention policies on pipeline outputs that generate large volumes. - Report resource constraint concerns to the C2DAO before they become capacity incidents.

Ontology Governance: - Do not modify shared enterprise Ontology types (managed by SL 4K) without C2DAO review. - When a project-local Object Type matures to the point where it should be promoted to the enterprise ontology, coordinate the promotion with SL 4K — do not attempt it directly.

Audit Trails: - MSS maintains activity logs for Object edits, pipeline runs, and Workshop access. - Preserve audit logs for all production data products per USAREUR-AF data retention policy. - If a data quality incident occurs, the audit trail is the PM's first diagnostic resource.

APPENDIX A — PROJECT KICKOFF CHECKLIST

A-1. Purpose

This checklist is completed by the Technical PM before sprint 1 begins. No sprint work starts until all items are complete or formally accepted as known gaps with a plan to close.

A-2. Checklist

Section 1 — Scope and Requirements

- ☐ Product vision statement written and posted in the project space
- ☐ Requirements Document drafted, reviewed with stakeholder, and baselined
- ☐ Out-of-scope boundaries explicitly documented and agreed by stakeholder
- ☐ Success metrics defined and measurable (not "it works" — specific thresholds)
- ☐ Priority tiers established (Must Have / Should Have / Nice to Have)
- ☐ Initial product backlog contains at least 2 sprints of groomed, sized, AC-complete stories

Section 2 — Team Structure

- ☐ All team members identified by name and SL 4 track
- ☐ PM role confirmed (Scrum Master / Product Owner)
- ☐ SL 4K (Knowledge Manager) identified for ontology and data dictionary coordination
- ☐ SL 4L (or appropriate developer track) assigned to pipeline work
- ☐ Cross-track dependencies mapped and initial dependency records created

Section 3 — MSS Project Setup

- ☐ Foundry project space created in the correct USAREUR-AF environment
- ☐ All team members have appropriate Foundry role access (not admin unless required)
- ☐ Project tracker Ontology types created (Project, Sprint, Story, Risk, Dependency)
- ☐ Sprint Board Workshop application built and accessible to all team members
- ☐ Commander-facing Status dashboard built (even if placeholder) and stakeholder has access
- ☐ At least one Sprint object created with sprint goal, start date, and end date

Section 4 — Data and Platform

- ☐ Input data sources identified for each pipeline or model component
- ☐ Data access confirmed — team can read all required source datasets on MSS
- ☐ Data Audit initiated (or scheduled for sprint 1) if ML components are involved

- ☐ No blocking dependencies on external data sources that are not yet on MSS
- ☐ Platform resource constraints assessed — no current storage or compute flags

Section 5 — Governance

- ☐ C2DAO coordination initiated if project involves shared enterprise Ontology types
- ☐ Access control plan documented (who gets which role on production product)
- ☐ Data classification of all input sources confirmed with SL 4K
- ☐ Change management stakeholder map drafted
- ☐ Communication plan for production release drafted (even if T-0 is months away)

Section 6 — Sprint Cadence

- ☐ Sprint length confirmed (1 week or 2 week)
- ☐ Sprint 1 start date and end date confirmed
- ☐ Sprint ceremony calendar blocked for all team members (planning, standup, review, retro)
- ☐ Sprint 1 scope committed and visible on MSS sprint board

APPENDIX B — DEFINITION OF DONE — DATA PRODUCT STANDARDS

B-1. Purpose

The Definition of Done (DoD) is the PM's quality gate for every production release. No data product is released to production without all DoD items satisfied or explicitly waived with documented rationale. The PM signs off on the DoD — not the developer, not the stakeholder.

How to use this checklist: - Run through the checklist at the production readiness review (7-2b) - Each item is PASS, FAIL, or WAIVED (with written rationale) - A FAIL on any Critical item blocks the release - A WAIVED item requires PM signature and a documented plan to close within 30 days

---

B-2. DoD Checklist — All Data Products

Section 1 — Data Quality

#	Item	Priority	Status
1.1	Input data sources are connected to live Foundry datasets — no manual file uploads in the production pipeline	Critical	

#	Item	Priority	Status
1.2	Data lineage is documented from source to display (which dataset flows to which transform to which ontology object to which Workshop widget)	Critical	
1.3	All input datasets have a documented refresh schedule and the pipeline runs on that schedule without manual intervention	Critical	
1.4	Missing value handling is documented and tested — the pipeline does not silently produce incorrect outputs when upstream data is null or delayed	High	
1.5	Schema changes in input datasets have been accounted for — the pipeline handles variations seen in the past 12 months	High	
1.6	A data quality check (SL 4M: @check, or equivalent validation step) runs on critical input fields before outputs are published	High	

Section 2 — Product Function

#	Item	Priority	Status
2.1	All user stories in the production scope have acceptance criteria that have been tested and passed	Critical	
2.2	Zero open defects at Critical severity	Critical	
2.3	Zero open defects at High severity (or all waived with stakeholder sign-off)	High	
2.4	The product has been tested in the production environment (not only in a dev branch)	Critical	
2.5	At least one operational user (not a developer) has tested the product and confirmed it meets the stated requirement	Critical	
2.6	The product degrades gracefully when data is unavailable — it shows a "data unavailable as of [timestamp]" state, not a blank or error screen	High	

Section 3 — Access Control

#	Item	Priority	Status
3.1	Production roles are configured — access is not running under a developer's personal account	Critical	
3.2	Least-privilege applied — users have the minimum access level required for their role	Critical	
3.3	Access control reviewed and signed off by SL 4K (Data Steward)	High	

#	Item	Priority	Status
3.4	C2DAO coordination complete for any ontology types shared with the enterprise	High	
3.5	No test accounts, sandbox credentials, or temporary access grants present in the production configuration	Critical	

Section 4 — Documentation

#	Item	Priority	Status
4.1	User guide complete and reviewed by at least one end user	High	
4.2	Data dictionary entry updated in the project's SL 4K-managed data dictionary for all new datasets or ontology types	High	
4.3	Pipeline documentation updated — transform logic, schedule, and dependencies documented	High	
4.4	Known limitations section in the user guide accurately describes what the product does NOT do	High	

Section 5 — Operability

#	Item	Priority	Status
5.1	Monitoring is active — PM knows how to detect if the product stops working	Critical	
5.2	Rollback procedure is documented and has been tested (or tested in a prior comparable deployment)	Critical	
5.3	Operational users notified of the release at least 48 hours in advance	High	
5.4	Support contact identified and communicated to users — someone to call if the product fails	High	
5.5	Change management plan for post-release adoption is in place (Chapter 8)	Medium	

B-3. Additional DoD Items — ML/AI Products

Complete Section B-2 first. Then complete these additional items for any product that includes a trained ML model, AI agent, or LLM integration.

#	Item	Priority	Status
ML-1	Model card exists and is stored in the project space: documents training data, algorithm, evaluation metrics, known failure modes, and appropriate use cases	Critical	
ML-2	Model was evaluated on held-out data (not the training set). Evaluation metrics meet the defined success threshold from the Problem Definition (3-2)	Critical	
ML-3	SL 4G (ORSA) has conducted independent validation of the evaluation methodology	High	
ML-4	Operational SME has reviewed sample model outputs and accepted output quality	Critical	
ML-5	Model output display includes confidence scores or uncertainty indicators — not bare binary outputs without context	High	
ML-6	The product UI communicates what the model does NOT predict and where it should not be used	High	
ML-7	Model monitoring is configured: SL 4M has accepted model owner responsibility and defined the threshold that triggers a model review	Critical	
ML-8	Model version and training date are documented in the model card and associated Foundry dataset records. Model deployment in Foundry follows this pattern: batch inference transforms write predictions to a Foundry dataset, which then serves as the source for computed properties on Object Types, or models are integrated into AIP Logic workflows.	High	
ML-9	Retraining schedule or trigger is documented (time-based or performance-based)	High	

APPENDIX C — PROFESSIONAL READING LIST

Curated articles from Army professional journals and military publications. These supplement doctrinal references with contemporary operational perspectives.

Source	Title	Date	Relevance
Army AL&T	"Accelerating the Army's AI Strategy"	2024-25	AI program management
Army AL&T	"The Army's Data (Ad)Vantage"	2024	Data program strategy
Army AL&T	"The Software Advantage"	2024-25	Software acquisition modernization
Army AL&T	"Reality Check" (AI/ML implementation)	2024-25	AI/ML program realities
Military Review	"Attaining Readiness by Developing a Data-Centric Culture"	2024	Organizational data culture

GLOSSARY

Acceptance Criteria (AC) — Testable conditions that define when a user story is complete. Written in GIVEN/WHEN/THEN format. The PM uses AC to verify that a story meets the stakeholder's requirement before marking it Done.

Agile — An iterative approach to project management that delivers value in short cycles (sprints), adapts to change, and emphasizes working products over comprehensive plans. Contrast with waterfall (fixed scope, fixed timeline, single delivery).

AIP Logic — The Maven Smart System capability for configuring AI-driven workflow automation and agent behaviors without writing code. Managed by SL 4H (AI Engineer).

Backlog — The prioritized list of all work items (user stories, tasks, defects) for a project. Maintained by the PM. Groomed continuously as new requirements arrive or priorities change.

Blocker — An impediment that prevents a team member from progressing on a story. Identified at daily standup. Tracked as a Boolean flag on the Story object and visible on the sprint board. The PM is responsible for resolving blockers, not just recording them.

C2DAO — Command and Control Data Architecture Office. The USAREUR-AF governance authority for MSS platform configuration, enterprise Ontology design, and access decisions that cross project boundaries.

Change Management — The process of planning and executing the transition of operational users from an existing workflow to a new data product. Includes communication, training, sponsor engagement, and adoption monitoring.

Change Sponsor — A peer-level advocate within an affected stakeholder group who champions adoption of a new data capability. Not a trainer — a credible voice within the organization that signals the change is endorsed.

Daily Standup — A 15-minute daily synchronization ceremony for the sprint team. Three questions: What did I complete? What will I complete? What is blocking me? Not a status meeting — a coordination mechanism.

Data Audit — A structured assessment of an input data source prior to ML model development. Covers completeness, timeliness, historical depth, label availability, and schema stability. PM gate before prototype authorization.

DDOF (Data and Digital Operations Framework) — The T2COM C2DAO framework governing the lifecycle of data products from problem framing through operations. Defines six roles, three execution wedges (Define, Build, Deliver), and phase gates. The PM uses the DDOF Friction Matrix (2-6) to assess execution health across all phases.

Definition of Done (DoD) — The PM-owned quality gate checklist for production releases. All items must pass (or be formally waived) before a data product is deployed to production. See Appendix B.

Dependency — A constraint where one story or deliverable cannot be completed until another task or external input is available. Tracked as a Dependency Object in the project tracker. Managed actively by the PM to prevent sprint-level blocks.

FDM (Functional Data Manager) — The DDOF role (typically O-3/O-4) responsible for managing a data product's lifecycle within a functional area. The PM tracks FDM product health and ensures FDM sign-off on data quality at each gate.

Feature — A discrete, user-facing capability of a data product. Composed of one or more user stories. Features are prioritized in the backlog; stories are scheduled into sprints.

Foundry — The underlying Palantir platform on which the Maven Smart System (MSS) is built. SL 4J personnel interact with Foundry primarily through Workshop, Ontology UI, Pipeline Builder, and Contour.

Friction Matrix — The DDOF Configuration Management assessment template (5x6 grid) that rates six assessment areas across three execution wedges as OK, FRICTION, or CASCADING FRICTION. The PM's primary diagnostic tool for identifying DDOF execution blocks. See 2-6.

Kanban — An Agile method that manages continuous flow of work using a visual board with WIP limits, rather than fixed-length sprints. Used for sustainment and support work where demand is continuous and variable.

Model Card — A documentation artifact for a deployed ML model. Contains: training data description, algorithm, evaluation metrics, known failure modes, intended use cases, and out-of-scope uses. Required for all production ML deployments under the DoD checklist.

Model Drift — Degradation of an ML model's production performance over time as input data distribution shifts away from the training distribution. Detected through production monitoring. Triggers a retraining review.

MSS (Maven Smart System) — The USAREUR-AF operational data and AI platform built on Palantir Foundry. The platform within which all SL 4J project tracking, dashboards, and data products are built and operated.

Ontology — The structured, semantic data model in Foundry that defines Object Types, their properties, and the links between them. The foundation of all MSS Workshop applications and Analytics. Governed by SL 4K (KM/Data Steward) at the enterprise level.

OPDATA — Operational data. Data generated by or supporting theater operations. The primary source of requirements for MSS capability builds in USAREUR-AF.

Product Backlog — See Backlog.

Product Owner — In Scrum, the role responsible for backlog prioritization and stakeholder representation. At USAREUR-AF, SL 4J TMs typically fulfill this role alongside Scrum Master responsibilities.

Production Ready — A state in which a data product meets all Definition of Done criteria and is deployable to operational users. Not a judgment — a checklist result.

Retrospective — A sprint ceremony conducted at the end of each sprint, for the team only. Reviews what went well, what did not, and what to improve next sprint. PM owns tracking and following up on retrospective action items.

Risk Register — The PM-maintained log of identified project risks, including likelihood, impact, severity score, owner, mitigation, and status. Maintained as an MSS Ontology Object and reviewed at every sprint planning session.

Scrum — An Agile framework using fixed-length sprints (typically 1-2 weeks) with four ceremonies: Sprint Planning, Daily Standup, Sprint Review, and Retrospective. Suited for data/AI projects with defined product outcomes and moderate team sizes.

Scrum Master — In Scrum, the role responsible for facilitating ceremonies, removing blockers, and protecting the team from external disruption. At USAREUR-AF, SL 4J TMs typically fulfill this role.

Sprint — A fixed-length development cycle (1 or 2 weeks) in Scrum. Each sprint has a goal, a committed scope, and a deliverable product increment. Managed via the MSS sprint board.

Sprint Goal — A one-sentence statement of what the team will demonstrate at the sprint review. Set at sprint planning. The PM's primary alignment tool during the sprint.

Sprint Review — A sprint ceremony at the end of each sprint where the team demonstrates completed work to stakeholders via live demo in MSS. Not a PowerPoint slide deck.

Stakeholder — Any person or organization with an interest in the outcome of a data product. Includes operational users, commanding officers, and data governance stakeholders (C2DAO, SL 4K).

Story Points — A unit of effort estimation for user stories. Often uses Fibonacci sequence (1/2/3/5/8/13). Used to measure sprint velocity and forecast delivery timelines. SL 4J tracks use T-shirt sizing (S/M/L/XL) as a simpler equivalent.

Technical Debt — Work deferred to meet a delivery deadline that must eventually be completed to maintain product quality and operability. Examples: hardcoded values, missing tests, undocumented transforms. PM manages tech debt as a reserved backlog capacity item.

SL 4 Track — Specialized capability track for data/AI professionals on MSS. Tracks: SL 4G (ORSA), SL 4H (AI Engineer), SL 4M (ML Engineer), SL 4J (Technical PM), SL 4K (KM), SL 4L (SWE), SL 4N (UI/UX Designer), SL 4O (Platform Engineer). SL 4J coordinates across all tracks.

UDRA — Unified Data Reference Architecture, version 1.1 (February 2025). The Army's authoritative data architecture guidance. Informs MSS data product design standards and data governance practices.

VAULTIS-A — Data quality assessment framework with eight dimensions: Visible, Accessible, Understandable, Linked, Trusted, Interoperable, Secure, and Auditable. The PM tracks VAULTIS-A scores at quarterly reviews for all products in Operations phase. Products scoring below 70% on any dimension enter a remediation cycle. See 7-6.

User Story — A brief description of a feature from the perspective of the user who will benefit from it. Format: "As a [user], I want [capability], so that [outcome]." The primary unit of backlog work in Scrum.

Velocity — The average number of story points (or T-shirt size equivalents) a team completes per sprint. Calculated after three or more sprints. The PM uses velocity to forecast delivery timelines and plan release milestones.

WIP Limit — Work-in-progress limit. A Kanban discipline that caps the number of stories a team member can have In Progress simultaneously. Enforces completion before starting new work. Recommended: 2 per developer.

Workshop — The Foundry application builder tool. SL 4J PMs build sprint boards, stakeholder-facing dashboards, and commander status products in Workshop. The primary PM-facing tool for project tracking on MSS.

SL 4J — Technical Manual: Program Manager (Technical), Maven Smart System Headquarters, United States Army Europe and Africa, Wiesbaden, Germany Army CIO Memo (April 2024) — Unified Data Reference Architecture v1.1 (February 2025)

DoD and Army Strategic References:

- **DoDI 5000.87, Software Acquisition Pathway (October 2020)** — Establishes the software acquisition pathway for rapid, iterative software delivery
- **Army Directive 2024-02, Agile Software Development (December 2024)** — Army policy for agile software development practices and delivery