DRAFT — UNOFFICIAL — NOT FOR OPERATIONAL USE

CONCEPTS GUIDE

SL 5M

USAREUR-AF

CONCEPTS GUIDE — SL 5M COMPANION — ADVANCED MACHINE LEARNING ENGINEER — MAVEN SMART SYSTEM (MSS)

Specialist Course Manual

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

DRAFT — NOT FOR OFFICIAL USE. FOR TRAINING PLANNING PURPOSES ONLY.

26 MARCH 2026

DRAFT — UNOFFICIAL — NOT FOR OPERATIONAL USE

CONCEPTS GUIDE — SL 5M COMPANION — ADVANCED MACHINE LEARNING ENGINEER — MAVEN SMART SYSTEM (MSS)

Forward: SL 5M marks a professional transition. You are no longer primarily a model builder. You are an ML systems engineer — responsible for the infrastructure, processes, and standards that make reliable ML possible at scale. **Purpose:** Extends the mental models of the TM-40M Concepts Guide to advanced ML engineering on MSS. Prerequisite: TM-40M Concepts Guide and SL 4M qualification. *HQ USAREUR-AF · v1.0 · 2026 · DISTRIB: USG only*

SECTION 1 — FROM MODEL BUILDER TO ML SYSTEMS ENGINEER

BLUF: SL 5M marks a professional transition. You are no longer primarily a model builder. You are an ML systems engineer — responsible for the infrastructure, processes, and standards that make reliable ML possible at scale.

1-1. The Distinction That Matters

A model that works is not the same as an ML system that reliably serves operational decisions over time.

A SL 4M MLE produces a model that achieves acceptable performance, deploys it to the Ontology, and it provides value. But that model will encounter data it was not trained on. It will be maintained by someone who did not build it. Its training data will become stale. Over 6-18 months, a carefully built SL 4M model will degrade in ways that are hard to detect — potentially invisible to operational users who have come to depend on it.

An ML system is a model plus everything required to keep it reliable over time: the feature pipeline that feeds it, monitoring that detects drift, governance records enabling independent verification, retraining infrastructure that can update it without manual heroics, and human-in-the-loop gates ensuring someone accountable reviews its behavior before it continues influencing decisions.

SL 4M Definition of "Done"	SL 5M Definition of "Done"
Model is in production and performing within	Model is in production, performance is monitored, monitoring is connected to an alerting and response protocol, training pipeline is documented and reproducible,

SL 4M Definition of "Done"	SL 5M Definition of "Done"
acceptable bounds	governance record is complete, and a plan exists for retraining or retirement

1-2. The Systems Engineer's Mental Shifts

Model Builder Instinct	ML Systems Engineer Extension
Build features optimized for this model	Ask whether this feature could be useful to another model and should be computed once and shared
Get good results and move to deployment	Record the experiment in enough detail that someone else could reproduce it 18 months later and understand why this approach was chosen
Design a model for current operational requirements	Design with an explicit retirement plan: what conditions force a retrain, what force a retire, and who makes that call

1-3. Vignette: Theater-Level Readiness Prediction

At SL 4M, a capable MLE could build a readiness prediction model for one BCT. At SL 5M, the challenge is different: twenty-plus subordinate formations each with slightly different data collection practices, feeding a corps-level prediction that must be coherent and comparable across formations.

Features must be defined consistently. The model must retrain when readiness reporting practices change. When one formation's sensor data goes offline, the prediction for that formation must degrade gracefully, not silently. Someone must own the monitoring for all twenty-plus pipelines simultaneously.

That is the problem SL 5M is designed to equip you to solve.

SECTION 2 — FEATURE STORE ARCHITECTURE

BLUF: A feature store is shared infrastructure for computed features — eliminating duplication, enforcing consistency, and creating a governance layer around the data that feeds your models.

2-1. Why Feature Duplication Is a Production Risk

In a team with multiple MLEs building models independently, feature computation happens in each model's training pipeline. The "last 90 days of maintenance events per vehicle class" feature gets computed three different ways by three engineers. Each version handles edge cases slightly differently.

Models trained on these three feature variants cannot be meaningfully compared. When predictions diverge on the same vehicle fleet, it is impossible to determine whether the difference comes from model architecture or feature computation.

A feature store solves this by making feature computation a shared service. The feature is computed once, stored with a version identifier, and consumed by any model that needs it.

2-2. Feature Governance Requirements

Shared features create a governance challenge: if a feature is owned by one team and consumed by another, what happens when the owning team wants to change the feature definition?

Governance Element	Purpose
Feature owner assignment	Single accountable party for definition changes and quality
Semantic versioning	Consuming models can pin to a version while evaluating upgrades
Change notification protocol	Owners notify consuming models before breaking changes go live
Deprecation policy	Clear timeline for how long old feature versions remain supported
Quality monitoring	Feature values monitored for distribution drift, nullity, and staleness

2-3. Designing for Reuse

Reusable features are designed differently from single-model features:

Temporally parameterized. Rather than hardcoding a 90-day lookback window, a reusable feature accepts a window parameter.

Entity-aligned. Features computed at a consistent entity grain (per vehicle, per unit, per personnel record). Inconsistent grains require implicit join assumptions that are easy to get wrong.

Self-documenting. Feature definition includes: what it measures, source data, missing data handling, expected distribution, and operational context. This lives with the feature in the registry, not in the original notebook.

Computationally bounded. A feature requiring four hours to compute cannot serve a real-time inference requirement. Design accounts for latency requirements of consuming models.

2-4. Breaking vs. Non-Breaking Feature Changes

Change Type	Examples	Action Required
Non-breaking	Performance optimizations producing identical outputs; adding documentation; adding optional parameter with backward-compatible default	Version increment not required; consuming models unaffected
Breaking	Changing missing data edge case handling; changing output units (hours to days); changing entity grain; removing a parameter	Version increment required; consuming models must re-evaluate before adopting

SECTION 3 — ADVANCED MODEL EVALUATION: BEYOND ACCURACY

BLUF: At SL 5M level, model evaluation is multi-dimensional. Statistical performance is necessary but not sufficient. A model with 92% accuracy may still be unsuitable for operational deployment if it is poorly calibrated, not robust to input perturbations, or accurate on average but unreliable in the specific cases that matter operationally.

3-1. The Evaluation Dimensions

Dimension	What It Measures	Why It Matters Operationally
Statistical performance	Accuracy, precision, recall, F1, AUC	Baseline fitness for purpose
Calibration	Agreement between predicted probabilities and observed frequencies	Confidence scores must be usable, not just directionally correct
Robustness	Performance under input perturbation and edge cases	Production data is messier than training data
Operational utility	Performance on the specific cases that drive decisions	Average accuracy hides performance on rare but critical cases
Fairness	Performance consistency across subgroups	Uneven performance creates systematic blind spots

DRAFT

3-2. Calibration: The Underemphasized Dimension

A calibrated model that outputs 80% confidence is right 80% of the time. A model that is accurate but poorly calibrated may output 80% confidence and be right 60% or 95% of the time. Accuracy cannot tell you which.

For operational users, calibration matters. A commander using a readiness risk dashboard does not just need to know which units are flagged — they need to know how much confidence to assign to the flags. If confidence scores are systematically miscalibrated, only the binary flag can be used, discarding the continuous signal the model produced.

Calibration is measured with reliability diagrams and metrics such as Expected Calibration Error (ECE) and Maximum Calibration Error (MCE). When poorly calibrated, correction is often possible post-training with Platt scaling or isotonic regression — but must be validated on a separate calibration set.

3-3. Operational Utility: Average vs. Operationally Critical Cases

Accuracy on a balanced holdout measures average performance. Operational utility measures performance on the cases that drive decisions. These are not the same.

A readiness prediction model where 95% of unit-days show no significant risk can achieve 95% accuracy by predicting "no risk" for every case — with zero utility. More subtly: a model achieving 90% overall accuracy but performing at 65% accuracy on units with partial data (the units most likely to have readiness problems) may be producing its worst performance exactly where it is most needed.

Operationally critical case analysis requires defining which cases matter most *before* evaluation begins, then building the holdout set to contain enough of these cases to evaluate specifically. This requires working with operational SMEs during evaluation design — not just during requirements definition.

SECTION 4 — DRIFT TAXONOMY AND RESPONSE PROTOCOLS

BLUF: Drift is not a single phenomenon. Data drift, concept drift, and prediction drift are distinct failure modes requiring different monitoring approaches and responses. Treating them as interchangeable leads to both missed detections and inappropriate responses.

DRAFT

4-1. The Three Drift Types

Drift Type	Definition	USAREUR-AF Example
Data drift	Statistical distribution of input features changes; relationship between inputs and outputs is unchanged	Logistics demand model trained on pre-exercise data; exercise begins and supply request volume and composition changes
Concept drift	Relationship between inputs and outputs changes; model's internal logic no longer reflects how the outcome is actually determined	Personnel readiness model trained before a major policy change; old predictors no longer drive readiness
Prediction drift	Model's output distribution changes without obvious change in inputs	Symptom of data drift, concept drift, pipeline change, or feedback loop dynamics

4-2. Monitoring Architecture for Drift Discrimination

Monitor	Signal	Drift Type Indicated
Feature distribution monitoring	Statistical distance (PSI, KL divergence) per feature	Data drift
Label distribution monitoring	Frequency of each output class over time	Concept drift candidate
Model performance monitoring	Accuracy/calibration against available ground truth	Any type
Prediction distribution monitoring	Output probability distribution over time	Prediction drift
Feature pipeline health monitoring	Null rates, value ranges, freshness per feature	Data quality / pipeline failure

The pattern of which monitors trigger simultaneously narrows the diagnosis. Data drift with stable prediction distribution suggests the drift is in features the model does not weight heavily. Prediction drift without feature distribution changes suggests upstream pipeline issues or feedback loop dynamics.

4-3. Operational Response Framework

Drift Signal	Severity	Response Options
Minor data drift, stable performance	Low	Log; schedule investigation at next review cycle
Major data drift, stable performance	Medium	Investigate feature pipeline; assess retraining timeline

Drift Signal	Severity	Response Options
Minor data drift, degrading performance	Medium-High	Expedite retraining; consider fallback model
Concept drift (confirmed)	High	Immediate model review; possible rollback; mandatory retraining
Prediction drift with unknown cause	High	Investigate pipeline and feedback dynamics before any retraining
Model performance failure (severe)	Critical	Rollback to last validated version; notification to mission owner

The decision to roll back versus retrain versus investigate is an operational decision that must involve the mission owner. The MLE provides the drift assessment and options; the mission owner decides on operational continuity tradeoffs.

4-4. The Feedback Loop Problem

Prediction drift deserves special attention when the model's outputs influence future training data. In a readiness prediction system: if predicted high-risk units receive additional resources and recover, the ground truth label for those units will be "no readiness problem" — because the model's prediction triggered an intervention that resolved the problem. Future training data systematically underrepresents "high risk, no intervention" scenarios.

This is not an edge case. It is a structural property of any ML system that drives consequential interventions. The monitoring and governance design must explicitly account for it, including how ground truth is labeled for cases where the model's prediction led to an intervention that changed the outcome.

SECTION 5 — MULTI-MODEL SYSTEMS: ENSEMBLES AND STACKING IN OPERATIONAL CONTEXT

BLUF: Ensemble models are often more accurate than single models, but accuracy is not the only operational requirement. Before designing an ensemble, explicitly evaluate the interpretability tradeoff. A commander who cannot understand why a model flagged a readiness risk may not be able to act on it — and may stop using the system.

5-1. When Ensembles Add Value

Ensembles improve prediction reliability by combining models whose errors are not perfectly correlated. In operational contexts with complex, high-dimensional input spaces — logistics demand forecasting across diverse supply chain variables, readiness prediction across heterogeneous equipment fleets — ensemble methods often outperform single models enough to justify their cost.

The cost has two components: - **Computational cost:** Training and serving multiple models instead of one - **Interpretability cost:** Ensemble output is a combination of multiple models' outputs, harder to explain to operational users

5-2. The Interpretability Requirement

In USAREUR-AF operational settings, model interpretability is an operational requirement with governance implications. If the G3 cannot understand what drove a high-risk flag for a specific BCT, they cannot: - Validate the prediction against their own situational awareness - Communicate the risk basis to the commanding general - Identify whether the model flagged a real operational risk or a data quality artifact

Common interpretability strategies for ensembles:

Approach	Description	When to Use
SHAP values at prediction time	Decomposes each prediction into feature contributions; provides per-prediction explanations	When per-case explanation is required by operational users
Ensemble distillation	Train a single, more interpretable model to mimic the ensemble's predictions	When overall interpretability is required and slight accuracy loss is acceptable
Tiered explanation architecture	High-level summary for operational users; detailed breakdown for analysts	When multiple audience types with different explanation needs must be served

5-3. Vignette: Multi-Model Logistics Demand Forecasting for 21st TSC

A stacking approach trains specialized models for each commodity class (Class III, V, IX) and a meta-model that combines outputs, accounting for cross-commodity correlations. The stacked system outperforms any single model.

But the G4 needs to explain demand forecasts to subordinate units and to USAREUR-AF for resourcing decisions. The stacking architecture requires an interpretability layer that can answer: "Why is the Class IX forecast spiking for this brigade?" The answer should come from the Class IX specialist model — the

meta-model adjusts for operational tempo correlations, but the Class IX-specific drivers are in the specialist model.

Designing the explanation interface requires understanding which layer holds the operationally relevant reasoning for each type of question. This is architecture work that must happen before deployment.

A parallel demand forecasting requirement from SETAF-AF introduces an additional complication: Africa AOR logistics data is sparser, supply chains are longer, and consumption patterns differ fundamentally from European theater operations. The stacking architecture must either train separate specialist models for SETAF-AF data distributions or introduce a domain adaptation layer. This decision affects model governance — a single model trained on combined data may underperform for both theaters, while separate models double the monitoring and retraining burden.

SECTION 6 — EXPERIMENT TRACKING AS OPERATIONAL DISCIPLINE

BLUF: An ML experiment record is an operational record. At SL 5M level, you are managing a portfolio of experiments across months and team members. The tracking system exists to make any result reproducible, enable cross-experiment comparison, and enable onboarding without institutional knowledge loss.

6-1. The Minimum Experiment Record

Field	Content	Why Required
Data version	Snapshot identifier or date range for training and validation data	Reproducibility — without this, the experiment cannot be recreated
Feature set	Explicit list of features used, including versions if drawn from the feature store	Reproducibility requires knowing exactly what inputs were used
Hyperparameters	Complete parameter configuration, not just final values	Debugging and re-running requires the full configuration
Evaluation results	All metrics across all relevant subgroups, not just overall accuracy	Governance requires subgroup performance documentation
Environment	Python version, library versions, compute environment	Reproducibility depends on consistent execution environments
Deployment decision	Was this promoted to production? Why or why not?	The most frequently omitted element — and the most critical for institutional memory

6-2. Portfolio-Scale Failure Modes

Failure Mode	Description	Prevention
Experiment orphaning	Experiment started, produces inconclusive results, abandoned without record	Require closure records for all experiments, including abandoned ones
Configuration drift	Configuration changes incrementally across runs; final run does not match initial record	Record configuration at start of each run, not just at conclusion
Cross-experiment comparison without data version control	Two experiments produce different results on "the same dataset" that was updated between runs	Enforce data version pinning; flag comparisons across different data versions
Evaluation metric inflation	Successive experiments evaluated on growing holdout sets; early experiments appear worse than they are	Pin evaluation set across comparison experiments

6-3. Onboarding as the Test of Institutional Memory

The practical test: can a new MLE arriving six months after a production model was deployed understand what was tried, why the deployed approach was chosen, what the known limitations are, and what follow-on experiments would be highest value?

If the answer is no, the tracking system is not serving its purpose. In USAREUR-AF, rotation cycles mean the MLEs who built a system are often not the MLEs who will maintain it. An ML portfolio that depends on the original builders' knowledge degrades with every PCS cycle.

SECTION 7 — MODEL GOVERNANCE AT PORTFOLIO SCALE

BLUF: A model registry is governance infrastructure, not just a version control system. At portfolio scale, the registry is how C2DAO maintains accountability for every ML capability on MSS.

7-1. The Registry as Source of Truth

NOTE

SL 4M now maps DDOF Phase 6 (Govern) to model versioning, A/B testing protocols, retraining trigger definitions, and federated ML considerations for coalition environments. The portfolio-scale governance below assumes familiarity with those single-model governance foundations; review SL 4M Sections 1-1 through 1-3 before applying registry governance at enterprise scale.

The model registry must answer these questions for every production model:

Question	Registry Field
What is this model and what does it do?	Model name, purpose, operational context
Who is accountable for it?	Mission owner (operational), ML owner (technical)
What data was it trained on?	Training dataset version, date range, source systems
What was its evaluated performance at deployment?	Evaluation metrics at deployment time, subgroup breakdown
What is its current performance?	Current performance metrics from production monitoring
Has it been reviewed since deployment?	Date of last governance review, reviewer
What are its known limitations?	Documented edge cases, known failure modes, subgroup performance gaps
When does it require revalidation?	Scheduled review date, revalidation trigger conditions

Without these fields, the registry is a deployment log, not governance infrastructure.

7-2. Mandatory Revalidation Triggers

Some events require immediate revalidation outside the normal review cycle:

Trigger	Reason
Significant data distribution change	Performance estimates from deployment time may no longer be valid
Major platform upgrade	Platform changes can introduce subtle numerical differences affecting calibration
Operational requirement change	New task organization, concept, or decision threshold changes evaluation criteria

Trigger	Reason
Governance policy change	New DoD/Army/USAREUR-AF guidance may require revalidation for compliance

Revalidation is not retraining. Revalidation means re-evaluating the existing model against current data and standards. If the model remains performant and compliant, it continues in production. If gaps are found, retraining or retirement is required.

SECTION 8 — RESPONSIBLE ML: ADVANCED CONSIDERATIONS

BLUF: Basic fairness checks at SL 4M level are necessary but not sufficient at SL 5M. Advanced responsible ML requires intersectional fairness analysis, temporal fairness monitoring, and feedback loop detection — with documentation that these analyses were performed for the governance record.

8-1. Intersectional Fairness

Single-axis fairness analysis (performance by gender, by rank, by MOS independently) can miss systematic performance gaps that only appear at the intersection of multiple characteristics. A model may perform equivalently for men and women overall, and equivalently for enlisted and officer personnel overall, but perform significantly worse for junior enlisted women.

When intersectional subgroups are too small for reliable statistical evaluation, the governance documentation must acknowledge this explicitly: "The evaluation set contains insufficient representation of [subgroup] to characterize model performance with confidence." This is honest documentation of a known gap. Running single-axis analysis only and documenting it as a complete fairness evaluation is not acceptable.

8-2. Temporal Fairness

Model performance can degrade differently across subgroups over time — temporal fairness drift. A model equitable at deployment may become inequitable in production if data drift affects some subgroups' input features more than others.

Temporal fairness monitoring tracks performance metrics across fairness-relevant subgroups over time. A readiness prediction model trained primarily on high-data-quality formations may degrade faster for lower-data-quality formations — but this degradation is invisible in overall performance metrics because high-data-quality formations dominate the average.

8-3. Feedback Loop Detection

Feedback loop dynamics can produce these failure modes:

Failure Mode	Mechanism
Performance inflation	Model predicts risk; cases receive resources that resolve the risk; future training shows fewer adverse outcomes, making model appear more accurate than it is
Underrepresentation of intervention scenarios	Because intervention resolves risk, future training has fewer "high risk, no intervention" examples; model becomes less able to recognize early warning signals
Runaway confidence	Model becomes progressively more confident in classifications that drive interventions, because interventions confirm predictions

Detection requires tracking intervention rates alongside model predictions — specifically, whether cases the model flagged as high-risk received interventions that would have changed their ground truth outcome. This linkage must be established in the data architecture before deployment.

8-4. Documenting Responsible ML Analysis for the Governance Record

Analysis	Documentation Requirement
Single-axis fairness	Performance metrics by each protected/relevant characteristic, with sample sizes
Intersectional fairness	Performance metrics for available intersectional subgroups; explicit acknowledgment of subgroups too small to evaluate
Temporal fairness plan	Monitoring specification for per-subgroup performance tracking post-deployment
Feedback loop assessment	Analysis of whether deployment will influence future training data; if yes, documentation of how this is addressed
Calibration analysis	Reliability diagram and ECE/MCE metrics, with calibration correction procedure if applied

SECTION 9 — ADVANCED FAILURE MODES: WHAT SL 5M ENGINEERS GET WRONG

BLUF: SL 5M failure modes are harder to detect than SL 4M failures because they are systemic, not local. They often manifest not as obvious errors but as degraded reliability over time, governance gaps discovered during audits, or operational users who quietly stop trusting the system.

Failure Mode	Description	Prevention
Feature pipeline training-production skew	Features computed during training are not identical to features computed at inference; production diverges from training silently	Use the same feature computation code for training and inference; monitor feature distributions immediately after deployment
Deploying ensembles without interpretability	Operational users encounter predictions they cannot validate; they stop trusting the system	Design the interpretability interface before deployment; validate with operational users during testing; monitor system utilization
Treating governance as one-time	Model deployed with accurate governance record in 2025 has materially inaccurate record by 2026 as training data ages, use cases evolve, and new policy is issued	Build governance maintenance into the model lifecycle from design; update at every retrain and on scheduled review cycle
Failing to plan model retirement	Model becomes difficult to retire because it is embedded in workflows with no documented replacement procedure	Build the retirement plan before deployment; design operational interfaces to be replaceable; review plan at each governance cycle

9-1. Training-Production Skew in Detail

Sources of skew include: training uses batch computation over a historical dataset while inference computes in near-real-time from a live source; training handles missing values differently from inference; training includes data not available at inference time.

Detecting skew requires comparing feature distributions during training against distributions during inference. If feature monitoring shows drift immediately after deployment — before any real operational data shift — the drift is likely skew, not environmental change.

SUMMARY — THE SL 5M MENTAL MODEL

SL 4M Mental Model	SL 5M Extension
Build a model	Design an ML system
Train on current data	Design for data that will change
Evaluate accuracy	Evaluate accuracy, calibration, fairness, and robustness
Monitor for drift	Monitor for drift types, distinguish causes, define response protocols

SL 4M Mental Model	SL 5M Extension
Deploy one model	Manage a portfolio of models
Record the model	Record the experiment, the decision rationale, and the governance lifecycle
Governance at deployment	Governance as a continuous process

The ML systems built at SL 5M level inform readiness decisions for III Corps, V Corps, 10th AAMDC, 56th MDC-E, and SETAF-AF formations, logistics decisions for 21st TSC, and intelligence analysis for USAREUR-AF G2. The difference between a technically deployed ML model and an ML system that reliably supports those decisions over time is the difference between SL 4M and SL 5M work.

APPENDIX — PEER SL 5 CROSS-REFERENCES AND WFF INTEGRATION

Peer SL 5 Publications. Senior MLEs should coordinate with practitioners in these companion advanced-track publications.

Publication	Track	Coordination Point
SL 5G	Advanced ORSA	Statistical validation; uncertainty quantification for ML outputs
SL 5H	Advanced AI Engineer	AIP Logic integration; fine-tuning infrastructure; adversarial robustness
SL 5J	Advanced Program Manager	ML program lifecycle; governance documentation
SL 5K	Advanced Knowledge Manager	Feature data governance; training corpus design
SL 5L	Advanced Software Engineer	Platform SDK infrastructure; OSDK model-serving integrations
SL 5N	Advanced UI/UX Designer	Feature engineering UI; model monitoring dashboards
SL 5O	Advanced Platform Engineer	ML platform infrastructure; training pipeline orchestration

WFF Operational Consumer Note. ML models built and maintained by SL 5M engineers serve the six Warfighting Function (WFF) tracks: Intelligence (SL 4A), Fires (SL 4B), Movement and Maneuver (SL 4C), Sustainment (SL 4D), Protection (SL 4E), and Mission Command (SL 4F). Readiness prediction models serve G3 (Movement and Maneuver). Logistics demand forecasting serves G4 (Sustainment). Personnel

risk models may serve G1 and G6. The mental model shift described in this guide — from model builder to ML systems engineer — must account for the diversity of WFF consumers: different latency requirements, different interpretability needs, and different consequences of model failure.

CONCEPTS GUIDE — SL 5M COMPANION // ADVANCED MACHINE LEARNING ENGINEER
HEADQUARTERS, UNITED STATES ARMY EUROPE AND AFRICA // WIESBADEN, GERMANY // 2026
DISTRIBUTION RESTRICTION: Distribution authorized to U.S. Government agencies and their contractors only. Other requests must be referred to Headquarters, C2DAO, Wiesbaden, Germany.

DRAFT