COURSE SYLLABUS

SL 5L

USAREUR-AF

COURSE SYLLABUS — SL 5L: ADVANCED SOFTWARE ENGINEERING

Maven Smart System (MSS) — USAREUR-AF

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

26 MARCH 2026

DRAFT — UNOFFICIAL — NOT FOR OPERATIONAL USE

COURSE SYLLABUS — SL 5L: ADVANCED SOFTWARE ENGINEERING

MAVEN SMART SYSTEM (MSS) — USAREUR-AF

Field	Detail
Level	SL 5L — Advanced Software Engineer Specialist Track
Duration	5 days (40 hours)
Prerequisites	SL 4L complete (Go evaluation on file); 18+ months active software engineering experience on MSS or equivalent Foundry/OSDK platform; demonstrated proficiency with TypeScript and Python in production; experience with CI/CD pipelines and code review processes
Audience	Senior SWEs, platform engineers, SWE leads building enterprise-scale MSS applications and platform infrastructure
Format	Advanced lab + architecture design review + evaluated system build
Location	MSS Training Environment (OSDK TypeScript/Python, CI/CD pipeline access, advanced Ontology permissions required)

PREREQUISITE WARNING: SL 5L is not required for most SWE billets. It is intended for engineers designing MSS platform architecture, establishing development standards for a team, implementing enterprise CI/CD, or building OSDK-backed systems consumed by other applications.

BLUF: SL 5L addresses the platform engineering challenges that arise when MSS-backed applications move from individual development to team-scale production — OSDK-first architecture patterns, type-safe Action design, enterprise CI/CD pipelines, security review standards, and the technical governance that prevents a collection of individually correct code from becoming an unmaintainable system. SL 5L practitioners design the platform that other developers build on.

LEARNING OBJECTIVES

#	Objective
1	Design OSDK-first application architectures: Object Type modeling for application consumption, OSDK query pattern optimization, type-safe Action interface design
2	Implement enterprise CI/CD pipelines for MSS applications: automated testing (unit, integration, contract), branch protection, promotion gates, and rollback procedures
3	Apply TypeScript advanced patterns for Foundry Functions: memoization, bulk query patterns, Object Set operations at scale, type narrowing for Action inputs
4	Design and implement a security review process for MSS applications: input validation at all boundaries, authentication/authorization pattern review, OSDK credential handling
5	Build a developer platform toolchain: shared Transform libraries, common OSDK query utilities, code generation from Object Type schemas
6	Conduct a technical architecture review of an existing MSS application: identify scalability constraints, security gaps, technical debt, and recommend a refactoring roadmap
7	Document a platform architecture for a technical audience: system diagram, OSDK interface contracts, API versioning, and deprecation policy

PRE-COURSE CHECKLIST

Complete **7+ duty days before Day 1:**

- ☐ Confirm OSDK TypeScript SDK access (advanced — branch manipulation, Function authoring)
- ☐ Confirm CI/CD pipeline access in the training environment (C2DAO request)
- ☐ Read TM-50L, Chapter 1 (Introduction and Scope) and Chapter 6 (DevSecOps for Foundry Environments) before Day 1
- ☐ Prepare a brief (1-page) architecture overview of an MSS application you built or maintain — you will present it for peer review on Day 4

DAILY SCHEDULE

Day 1 — OSDK-First Architecture and Object Type Design

Time	Block	Method	Content
0800–0900	1	Seminar	OSDK-first architecture: designing for consumption, not just for data; Object Type interface contracts; API design principles for Ontology
0900–1100	2	Lab	Object Type modeling for application consumption: primary key design, property typing, link traversal optimization, Object Set design
1100–1115	—	Break	
1115–1200	3	Lab	OSDK query optimization: filter push-down, pagination patterns, bulk query patterns for large Object Sets
1200–1300	—	Lunch	
1300–1500	4	Lab	Type-safe Action interface design: input validation in TypeScript, error type design, idempotency patterns
1500–1515	—	Break	
1515–1700	5	Lab	OSDK interface contract documentation: OpenAPI-style docs for Foundry Actions and query patterns

Evening reading: TM-50L, Chapter 3 (High-Performance Foundry Development) — memoization and bulk query sections.

Day 2 — TypeScript Advanced Patterns and Function Architecture

Time	Block	Method	Content
0800–0830	—	Review	Object Type design review; common OSDK query anti-patterns
0830–1030	6	Lab	TypeScript advanced patterns: memoization for computed properties, bulk query batching, Object Set operators for complex filters
1030–1045	—	Break	

Time	Block	Method	Content
1045–1200	7	Lab	Type narrowing and discriminated unions for Action inputs; exhaustive pattern matching for multi-type Object operations
1200–1300	—	Lunch	
1300–1500	8	Lab	Function composition patterns: shared Function libraries, code generation from Object Type schema, reducing duplication across applications
1500–1515	—	Break	
1515–1700	9	Lab	Function testing: unit testing TypeScript Functions with mock OSDK, contract testing between Functions and downstream consumers

Evening reading: TM-50L, Chapter 6 (DevSecOps for Foundry Environments).

Day 3 — Enterprise CI/CD and Security Review

Time	Block	Method	Content
0800–0830	—	Review	Function architecture questions
0830–1030	10	Lab	CI/CD pipeline design: automated unit + integration test execution, branch protection rules, promotion gate configuration
1030–1045	—	Break	
1045–1200	11	Lab	Contract testing between MSS applications: consumer-driven contract tests, OSDK interface version compatibility checks
1200–1300	—	Lunch	
1300–1500	12	Lab	Security review process: input validation audit checklist, authentication/authorization pattern review, OSDK credential handling review
1500–1515	—	Break	
1515–1700	13	Lab	Security testing: injection attack surface review for Action inputs, OPSEC review for data exposed through OSDK queries

Evening reading: TM-50L, Chapter 7 (Security Assessment and Secure Code Review) — the peer review checklist.

Day 4 — Architecture Review and Platform Toolchain

Time	Block	Method	Content
0800–0900	14	Seminar	Technical architecture review methodology: scalability, security, maintainability, and operational observability dimensions
0900–1100	15	Workshop	Participant architecture review: present your prepared MSS application; peer and instructor critique using the SL 5L review checklist
1100–1115	—	Break	
1115–1200	16	Workshop	Refactoring roadmap: given a reviewed application, produce a prioritized technical debt remediation plan
1200–1300	—	Lunch	
1300–1500	17	Lab	Developer platform toolchain: shared Transform library design, common OSDK utilities, code scaffolding templates
1500–1515	—	Break	
1515–1700	18	Prep	Evaluation scenario brief; platform architecture document template review

Day 5 — Evaluated Platform Architecture Build

Time	Block	Method	Content
0800–1700	19	**Eval**	**Evaluated exercise:** Design and build an OSDK-backed application skeleton with CI/CD pipeline, security review checklist applied, TypeScript Function library, and architecture documentation; defend to evaluator

Go standard: OSDK application skeleton functional with at least 2 Object Types and 1 Action; CI/CD pipeline configured; security checklist applied and documented; architecture document includes OSDK interface contracts and deprecation policy.

PEER ADVANCED TRACKS

Track	Relevance to SL 5L
SL 5H (Advanced AI Eng)	OSDK-backed AI system integrations; security patterns for AI applications on Foundry
SL 5M (Advanced ML Eng)	ML model serving infrastructure built on OSDK; platform toolchain for data science workflows
SL 5J (Advanced PM)	Delivery governance for SWE-led programs; technical debt and architecture debt in portfolio risk reporting

USAREUR-AF Operational Data Team Syllabus SL 5L | Version 1.0 | March 2026