DRAFT — UNOFFICIAL — NOT FOR OPERATIONAL USE

ARCHITECTURE REFERENCE

ODT-CDA

USAREUR-AF

Architectural Constraints and Directives

Architecture Reference

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

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

20 MARCH 2026

DRAFT — UNOFFICIAL — NOT FOR OPERATIONAL USE

sidebar_position: 2 title: "Constraints & Directives"

ARCHITECTURAL CONSTRAINTS AND DIRECTIVES

Governing principles for the Cross-Domain Architecture derived from operational leadership, theater-level experience, and observed combat outcomes. These directives are not aspirational — they are constraints. Systems, data models, and processes that violate them are non-compliant.

Source: USAREUR-AF G37/ODT organizational guidance, derived from theater-level operational experience and observed outcomes in Ukraine, the Middle East, and LSCO preparation.

DOCTRINAL FOUNDATION

Publication	Title	Relevance
ADP 3-0	Operations	Operational framework within which CDA architecture operates
ADP 5-0	The Operations Process	Plan-prepare-execute-assess cycle that CDA supports
ADP 6-0	Mission Command	C2 warfighting function and shared understanding
AR 25-1	Army Information Technology	Statutory authority for data governance and IT architecture

Strategic Guidance:

The following are strategic guidance documents — not doctrine — that inform CDA architecture and operational context.

Document	Authority	Relevance
NATO Data Strategy for the Alliance (Feb 2025)	NATO	Alliance-wide data governance mandate — coalition alignment requirement for EUCOM AOR
NATO Data Centric Reference Architecture v2 (2025)	NATO	Reference architecture for digital transformation — target architecture for coalition data interoperability

NOTE

CDA architectural constraints are derived from these doctrinal principles as applied through CG operational guidance. They are not doctrine themselves but implement doctrine within the MSS operational context.

Theater-Level Implementation Examples:

Initiative	Date	Implementation Context
EUCOM BRAVO Hackathon Series (BRAVO 100, BRAVO 101)	Mar–Summer 2024	EUCOM Chief Data and AI Office innovation events where small teams build AI prototypes for theater warfighting challenges in 72–96 hours. 26 prototype solutions produced. Demonstrates the Understand-Adapt-Integrate cycle (Constraint III) at operational tempo — rapid prototyping under architectural constraints, with operational relevance as the acceptance criterion.
EUCOM Thunderforge AI Planning Ecosystem (DIU/Scale AI contract)	2024–25	AI agents simulate wargaming and planning scenarios for EUCOM, enabling AI-augmented MDMP at theater level. Implements Constraints VI (decision speed as the critical metric), VII (observability of AI agent outputs), and IV (theater-level architecture enabling subordinate planning).

I. THE CHANGE FRAMEWORK

Every architectural initiative must answer four questions in sequence. Skipping any step produces temporary change that does not survive leadership transitions or operational stress.

Step	Requirement	Architectural Implication
1. Why	Clearly articulate why the current state is unacceptable	Every architecture decision record (ADR) must open with the operational problem, not the technical solution
2. Vision	Define the end state in terms the formation can internalize	Capability maps and target architectures must be communicable to non-technical leaders at echelon
3. Plan	Turn vision into sequenced, resourced actions	Roadmaps must have dependencies, milestones, and measurable checkpoints — not just timelines
4. Culture + Process	Embed change into how the organization actually operates	If a change isn't encoded in a process (SOP, pipeline, governance gate), it doesn't exist yet

Culture without process does not survive. Change must be encoded in SOPs, pipelines, and governance gates — otherwise it does not persist beyond the leadership that drove it.

II. MEASURABILITY AS A HARD CONSTRAINT

Any capability, system, or data product that cannot demonstrate measurable change is not a deliverable — it is an experiment.

- Every data product must define its measures of performance (MOP) and measures of effectiveness (MOE) before development begins
- Baselines must be captured on arrival; outcomes must be captured on departure — the lethality study model applies to data architecture maturity the same way it applies to gunnery tables
- Dashboards and reports that do not connect to a decision or an action are waste

The ability to measure change is not optional — it is a prerequisite for knowing whether change is actually occurring. Any capability that cannot demonstrate measurable progress is not a deliverable.

III. UNDERSTAND, ADAPT, INTEGRATE

This is the operational cycle observed in units that succeed in combat and the one that fails in units that don't. It applies directly to how this architecture must function.

Phase	In Combat	In Architecture
Understand	Read the environment, know what the adversary is doing	Maintain current-state inventory; know what data, systems, and capabilities exist and how they perform
Adapt	Change TTPs based on what you're seeing	Update ontologies, pipelines, and models when doctrine changes, when new capabilities arrive, or when assessments reveal gaps
Integrate	Bring new capabilities into the fight at speed	New data sources, tools, and platforms must integrate through governed patterns, not ad hoc connections

Units that can execute the understand-adapt-integrate cycle gain decisive advantage over those that cannot. The gap is not technological — it is organizational and procedural.

Constraint: Architecture artifacts that are not updated within 30 days of a significant doctrinal, organizational, or capability change are stale and must be flagged.

IV. ECHELON-APPROPRIATE INFLUENCE

Technology now enables every echelon to influence the fight in ways it never could before. The architecture must reflect this without undermining subordinate initiative.

- Theater-level architecture enables and constrains — it does not prescribe tactical implementation
- Corps and division architectures must have freedom of action within theater standards
- Company-level systems must remain usable by soldiers without architectural expertise

Theater-level architecture enables subordinate formations — it does not replace their judgment or constrain their freedom of action. No matter what technology is layered above, company troops and batteries are still the ones who close with the enemy.

Directive: Data models and platform governance must be layered by echelon. Theater defines authoritative sources, semantic standards, and cross-domain flow. Subordinate echelons define how they consume and operationalize within those boundaries.

V. NEVER FIGHT THE LAST WAR

Whatever we observe in current conflicts informs the future but does not define it.

- Architecture must not be optimized exclusively for the current threat or current platform
- Vendor-specific, platform-locked designs violate this principle — the architecture must be portable
- Doctrine-grounded ontologies survive platform transitions; ad hoc schemas do not

Current conflicts inform the future but do not define it. Architecture optimized exclusively for today's observed threat will be obsolete before it is fielded.

Constraint: No system architecture may have a single-vendor dependency without a documented migration path. Ontology designs must be grounded in published doctrine (MIM, ADP, FM), not in the schema of the platform they happen to be deployed on.

VI. NEW FORMS OF MASS AND THE OFFENSIVE PROBLEM

Stationary defense is a solvable problem with current technology. Offensive operations under persistent surveillance and cheap precision strike are the hard architectural challenge.

- Architecture must prioritize capabilities that enable maneuver, not just capabilities that enable defense

- Decision speed — the time from sensor to shooter to assessment — is the critical architectural metric
- Cheaper, more distributed capabilities that free high-end assets for deeper employment are preferred over expensive, centralized solutions

Persistent surveillance and cheap precision strike have made stationary defense a solvable problem — but the harder and still-unsolved challenge is enabling offensive operations under those same conditions.

Directive: Data architectures must support real-time or near-real-time decision cycles. Batch-only pipelines that cannot support time-sensitive targeting or maneuver decisions are incomplete.

VII. OBSERVABILITY AND REMEDIATION

Algorithms, unmanned systems, and automated decision aids will drift. The architecture must include the ability to detect drift and correct it under operational conditions.

- Every algorithm and model deployed in an operational context must have observability built in — not bolted on
- Operators must be able to understand when a system is performing outside acceptable parameters without needing data science expertise
- Remediation procedures must be documented and exercisable before deployment, not developed under fire

Observability is the operational requirement: knowing in real time how unmanned systems and algorithms are performing, and having the means to adjust them when they drift.

Constraint: No AI/ML model or algorithmic decision aid may be fielded without a defined observability framework that includes: performance baselines, drift detection thresholds, operator-accessible alerts, and documented remediation procedures.

VIII. CONVERGENCE OVER DUPLICATION

Plans and headquarters have converged. Architectures must converge with them.

- USAREUR-AF operates as theater army for EUCOM, NATO, and AFRICOM simultaneously — the architecture cannot treat these as separate concerns
- Shared data, shared ontology, shared governance — with classification and releasability controls that enable appropriate access without duplicating the entire architecture per mission set
- Regional plans have converged; the data architecture that supports them must converge too

Headquarters have converged because the plans that drive them have converged. The data architecture must reflect that operational reality — not preserve legacy stovepipes that no longer match how the theater operates.

Directive: One ontology governs all three regional plan sets. Classification, releasability, and coalition sharing are handled through metadata and access controls on the shared model — not through parallel, disconnected architectures.

IX. SELF-DEVELOPMENT IS NOT OPTIONAL

Leaders at every level must understand the technology they are responsible for employing. The architecture must support this.

- Training materials, reference documents, and decision aids must be accessible to non-specialists
- The course portal and training pipeline exist to close the gap between the people building the architecture and the people who depend on it
- If leadership cannot explain the architecture to a subordinate in plain language, the architecture is too complex or the training materials are insufficient

Self-development is an organizational responsibility, not a personal preference. Leaders at every level are expected to understand the technology they employ — and the training materials, reference documents, and decision aids to support that must be accessible and current.

Directive: Every major architectural capability must have an associated training product in the course portal. If it doesn't have a deck, it doesn't have adoption.

X. NO GROUPTHINK

Diverse perspectives produce better architectures. Homogeneous teams produce blind spots.

- Architecture review boards must include operators, not just engineers
- Red-team reviews of data models and system designs are required, not optional
- Rotate review responsibilities — the same team reviewing the same artifacts will stop seeing the problems

Homogeneous teams with long-tenure reviewers stop seeing the problems in their own work. Rotation is not merely personnel management — it is a mechanism for maintaining architectural rigor.

Constraint: No architecture artifact may be approved without review by at least one person from outside the authoring team. ARB membership must rotate annually.

XI. TRUST AND ACCOUNTABILITY

Trust subordinates from day one. Hold each other accountable publicly. Own mistakes immediately.

- Architecture governance must trust subordinate echelons to implement within standards — not micromanage their technical choices
- When an architecture decision is wrong, document it openly, update the ADR, and move on
- Governance exists to maintain standards, not to create bureaucracy that prevents action

When an architecture decision is wrong, document it openly, update the record, and move on. Concealing failures does not protect the architecture — it compounds the problem.

XII. LEADERSHIP IS ALWAYS NUMBER ONE

Leadership is the decisive factor — above technology, data, algorithms, and platforms. No system, pipeline, or analytical tool compensates for a formation that lacks leaders who can employ it, develop people around it, and drive change through it.

This architecture exists to make leaders more effective. If it makes their jobs harder, it has failed.

APPLICABILITY

These constraints and directives apply to: - All ontology and data model design decisions - All system architecture and integration patterns - All governance processes (ARB, data governance, acquisition review) - All training and adoption strategies - All roadmap and investment prioritization

Non-compliance must be documented as an accepted risk with a remediation timeline, not ignored.