DAFITC 2026

UNCLASSIFIED

Mission-Ready Software: Accelerating Cloud Migration with Multi-Agent AI

Modernizing the defense ecosystem through agentic AI software teams

STRATIAS · BRAVO17

SESSION OVERVIEW

Who is presenting, and what we will cover



Mac Sims

Chief Executive Officer, Stratias LLC

Former Transformation Branch Chief, HAF/A4



Adam Flater

Chief Technology Officer, Bravo17

01 The operational risk carried in legacy defense infrastructure

02 The digital flightline: a multi-agent engineering team

03 Human control gates and the authorization path

04 Live demo: COBOL ingest through implemented tickets

PROBLEM STATEMENT

The legacy inventory

Mission-critical footprint

Decades of COBOL and other legacy code still run finance, logistics, and personnel workloads across the enterprise.

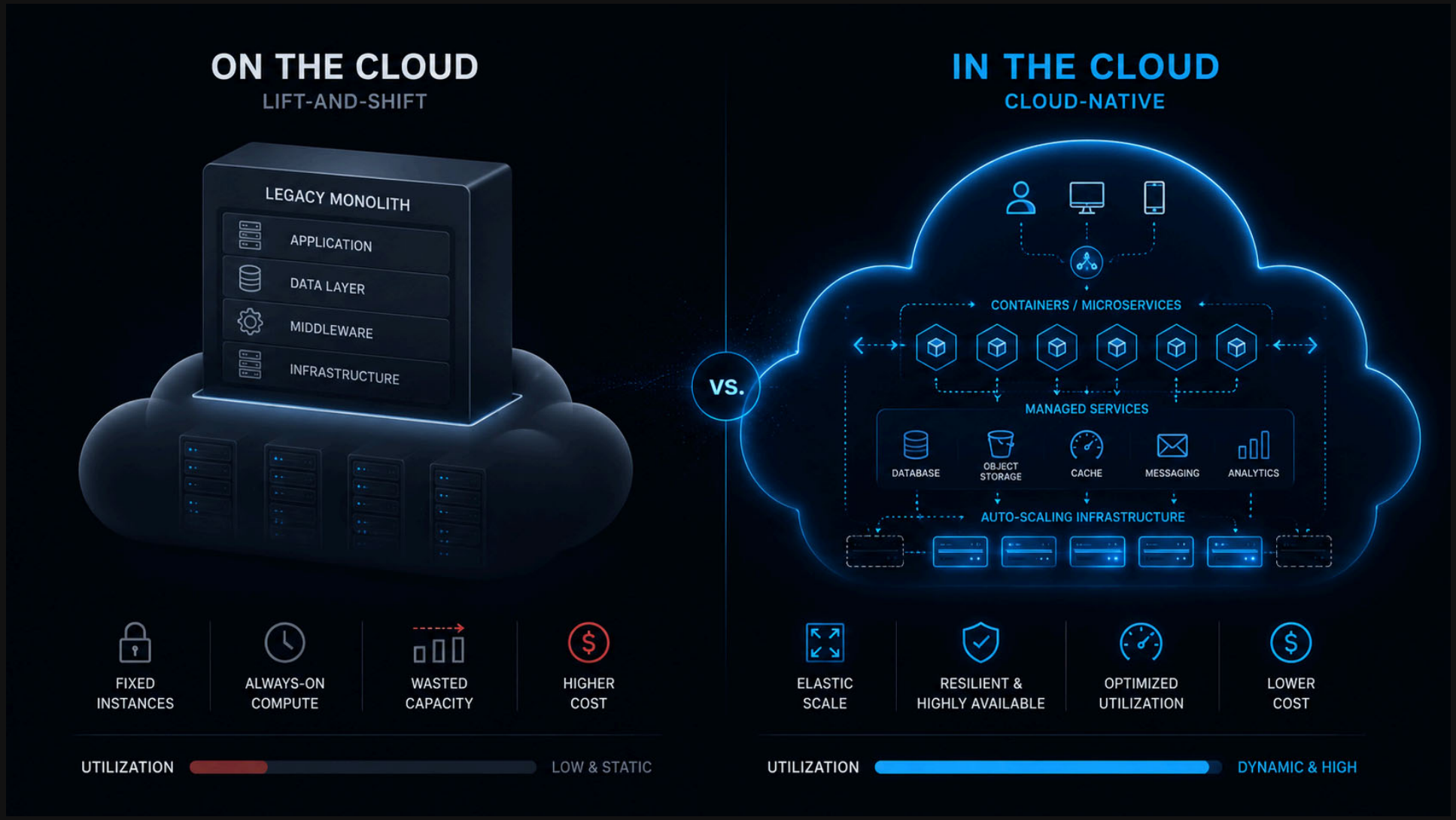
Thinning expertise

Documentation is sparse, inconsistent, or wrong. The architecture is tightly coupled and the people who could explain it have retired.

The velocity gap

Manual refactoring is too slow and too resource-intensive to keep pace with operational demand at the digital edge.

In the A4's Log IT Portfolio, hundreds of systems were queued for refactor. At manual throughput, that queue outlives the knowledge needed to work it.


ON THE CLOUD
LIFT-AND-SHIFT
LEGACY MONOLITH
APPLICATION
DATA LAYER
MIDDLEWARE
INFRASTRUCTURE
FIXED INSTANCES
ALWAYS-ON COMPUTE
WASTED CAPACITY
HIGHER COST
UTILIZATION
LOW & STATIC
VS.
IN THE CLOUD
CLOUD-NATIVE
CONTAINERS / MICROSERVICES
MANAGED SERVICES
DATABASE
OBJECT STORAGE
CACHE
MESSAGING
ANALYTICS
AUTO-SCALING INFRASTRUCTURE
ELASTIC SCALE
RESILIENT & HIGHLY AVAILABLE
OPTIMIZED UTILIZATION
LOWER COST
UTILIZATION
DYNAMIC & HIGH

SOLUTION

The digital flightline: an agentic scrum team

SCRUM MASTER

Sequences the work, holds the ceremonies, and keeps the other agents unblocked and in sync.

PRODUCT OWNER

Owns the requirements document, decomposes it into tickets, and defends intent against drift.

SOFTWARE ENGINEER

Implements tickets as merge requests on GitLab, each one traceable to a requirement.

SECURITY ENGINEER

Turns cyber readiness requirements into hardening tickets and validates the fixes.

UX ANALYST

Redesigns the interface against human-centered standards and instruments it for telemetry.

THE DIFFERENCE

A single-prompt coding tool translates files. A crew runs a lifecycle: requirements, planning, implementation, security, and user experience, coordinated as one agentic DevSecOps pipeline.

SOLUTION · STARGATE

Requirements to release

1

Ingest and understand

The platform captures the legacy source into a controlled repository and analyzes control flow, data structures, and business logic to determine what the system actually does.

2

Reverse-engineer requirements

Business rules buried in decades of patches are surfaced in plain language as an editable, version-controlled requirements document: the source of truth for the effort.

3

Plan and decompose

Approved requirements become a project plan and a ticket backlog, with the target language, cloud, and database set deliberately rather than inherited.

4

Implement with traceability

Implementation agents produce code through a standard merge-request workflow. Every line traces to a requirement; every requirement traces to observed legacy behavior.

The delivered artifact is not just an application. It is the requirements that defined it, the tickets that built it, and the version history connecting all three.

GOVERNANCE

Three human control gates

GATE 1	Architectural alignment	The architect confirms target language, cloud, and database against the team's expertise and operating environment.
GATE 2	Requirements validation	Subject matter experts read and correct the generated business rules before any code is written. This is where operator knowledge re-enters the system.
GATE 3	Implementation oversight	Technical leadership sets migration pace by reviewing units of work in a standard merge-request workflow, holding both functional and UX standards.

Between the gates, the agents do the work. At each gate, a human signs off.

PIPELINE

Hardening, telemetry, and authorization

Behavioral equivalence

Tests generated from the original code define what the system does, then validate the modern stack against it. The suite ships as the maintaining team's regression baseline.

Cyber hardening cycle

Cyber readiness requirements enter the same loop as everything else: the security agent generates hardening tickets, implementation agents apply them, and each change is documented.

Operationalized UX

Telemetry hooks are embedded during code generation, producing a UXR score across task success, latency, and interaction density. High-friction workflows surface as evidence, not anecdote.

Accredited deployment

Infrastructure-as-code is generated from the first gate. In the proving deployment, a sidecar brings AWS Bedrock models inside the GovCloud boundary, keeping inference in the accredited environment.

RMF Legacy code to requirements to tickets to modern code: the traceability chain is the evidence an authorization package asks for, assembled as the work happens.

LIVE DEMO

From COBOL ingest to implemented tickets

01	Ingest the code base	Connect a legacy COBOL repository and capture an auditable behavioral baseline.
02	Code to requirements	Read the generated design specification in plain language and correct a business rule.
03	Requirements to tickets	Watch the approved requirements decompose into a planned ticket backlog.
04	Tickets to implementation	Implementation agents pick up tickets and open merge requests against them.

STRATIAS · BRAVO17 DAFITC 2026