# Module 24 – Operational Deployment & CI/CD Capsule

Version: 4.1 | Classification: Capsule Execution Lifecycle Control  
Scope: Deployment automation, capsule verification, runtime promotion and operational control.

**Capsule Initialization**

This capsule is instantiated whenever a MaxOneOpen runtime enters deployment, upgrade, or staged promotion. It enforces audit-bound capsule build verification, CI integrity checks, and runtime policy revalidation before production admission. No capsule enters a live environment without successful lifecycle validation.

**0. Purpose & Deployment Enforcement Logic**

This capsule defines operational control flows for capsule build, deployment, runtime promotion, and rollback assurance. It introduces standardized integration points for build pipelines, deployment staging, audit hooks, and CI/CD governance to ensure that runtime capsules are fully validated and traceable before activation.

**1. Build & Verification Strategy**

- All capsule builds are bound to cryptographic integrity anchors (hash-verified, signed).

- Build metadata includes policy references, simulation results, and audit flags.

- CI jobs must validate structural capsule compliance before containerization or packaging.

**2. CI/CD Integration Interfaces**

- Pipeline triggers must be audit-exposed (e.g. Git action logs, build traces).

- Capsule simulation results must block promotion if runtime policy validation fails.

- Federation admission logic is embedded as a CI-enforced step, not runtime dynamic.

**3. Deployment Environments & Federation Layers**

- Runtime capsules may be deployed in isolated, federated, or mirrored configurations.

- Deployment tiers include dev, simulation, verified, and certified.

- Sovereign capsules require geo-bound deployment matching policy signature.

**4. Runtime Admission & Promotion Gates**

- No runtime enters production without CI/CD capsule audit confirmation.

- Promotion gates must confirm simulation, license compliance, and fork integrity.

- All promotions are snapshot-bound for historical rollback and forensic trace.

**5. Operational Monitoring Hooks**

- Deployment capsules emit state hashes, log anchors, and live capsule sync traces.

- Monitoring endpoints may include Prometheus-compatible runtime probes.

- All failures trigger rollback capsule invocation with traceable state match.

**6. Federation Compliance & Deployment Escalation**

- Federation deployments follow a quorum-based validation logic.

- In case of deployment failure or policy mismatch, capsules revert to prior certified state.

- Federation observers may validate deployment snapshot hashes for reentry certification.

**7. Roadmap Commitments (v4.2 aligned)**

- Full CI/CD pipeline reference sets (e.g. GitLab CI, GitHub Actions, ArgoCD).

- Automated capsule registry integration with sovereign approval layers.

- Capsule-centric rollback recovery analytics for deployment forensics.

**Final CTO Statement**

This capsule governs every operational deployment flow within MaxOneOpen. It mandates CI-based verification, enforces runtime promotion gates, and integrates audit-bound deployment staging. It is mandatory for any federation-ready or sovereignty-linked capsule entering execution under v4.1.