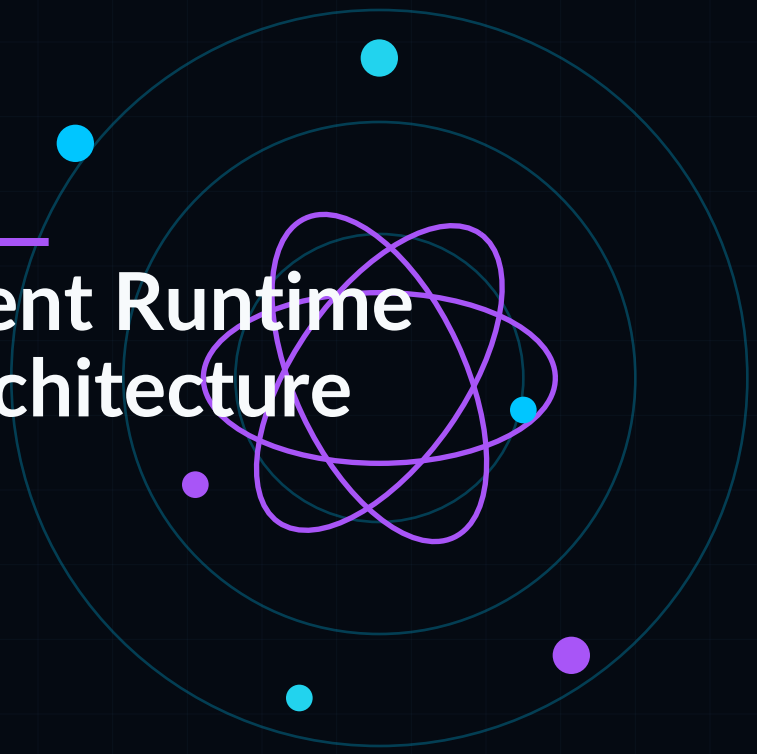


REFERENCE ARCHITECTURE / ENGINEERING STANDARDS LIBRARY

# Governed Multi-Agent Runtime and PANTHEON Architecture

A governed, durable multi-agent architecture that separates reasoning, authorization, deterministic execution, verification, memory, observability, and evidence.

**PERMANENT DOCUMENT ID**
RA-001

**PUBLICATION**
Candidate Reference Architecture

**ARCHITECTURE SET**
Canonical Set

# Architecture identity and operating position

| ARCHITECTURE SET | DOMAIN | LAYERS | COMPONENTS | ACCEPTANCE | DEPENDENCIES |
|---|---|---|---|---|---|
| Canonical Set | Artificial Intelligence | 6 | 6 | 18 | 0 |

## SYSTEM CONTEXT


## EXECUTIVE OUTCOMES

1. Convert ambiguous human intent into explicit missions and typed plans.
2. Permit models and agents to propose actions without silently acquiring execution authority.
3. Execute consequential work through deterministic, policy-bound mechanisms.
4. Preserve evidence, checkpoints, approvals, memory boundaries, and verified outcomes.

**ARCHITECTURE DOCTRINE**

Interpret the architecture as a bounded operating model: explicit ownership, least authority, separable desired/runtime/observed/evidence state, safe degradation, reversible change, measurable outcomes, and evidence-backed acceptance.

# Layered architecture and component model

LAYER 01
**Experience and Intent**

LAYER 02
**Mission and Plan Compiler**

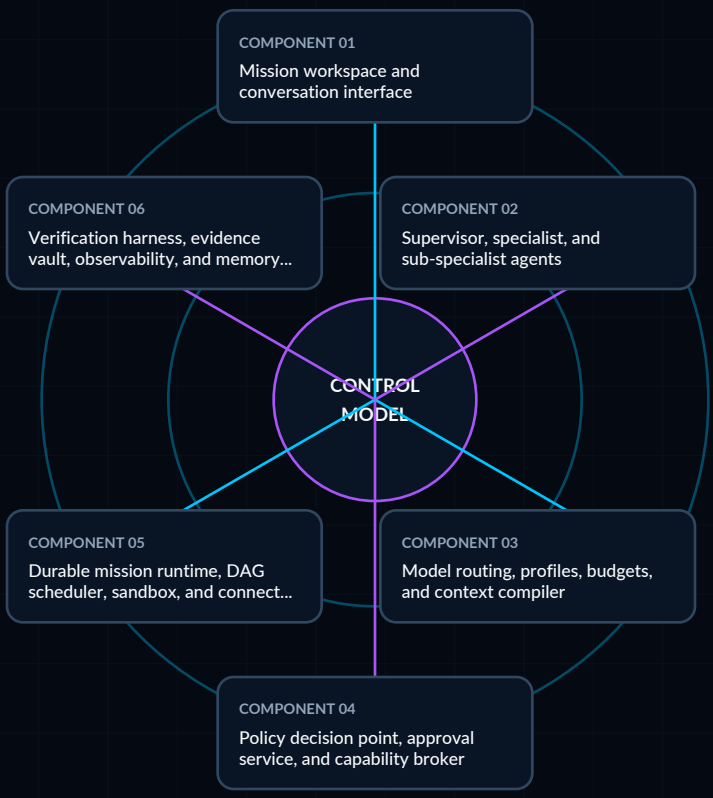
LAYER 03
**Agent and Model Fabric**

LAYER 04
**Governance and Authorization**

LAYER 05
**Deterministic Execution**

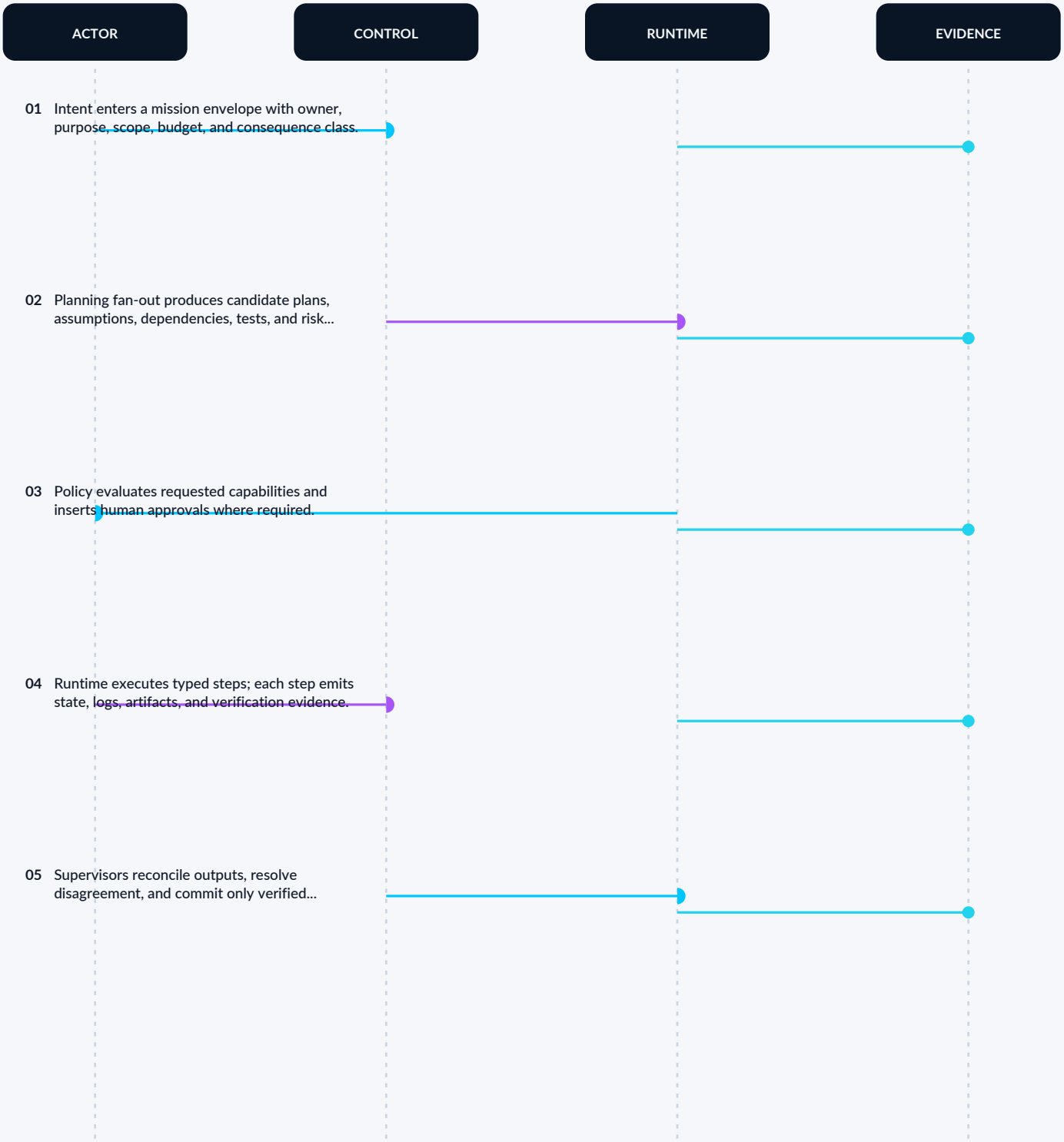
LAYER 06
**Verification, Evidence, and Memory**

## CORE COMPONENTS




Each layer and component owns an explicit interface, state model, authority boundary, failure behavior, telemetry contract, recovery path, and evidence obligation.

# Primary sequence and control flow




**EVIDENCE THREAD** Identity + authorization + version + state transition + output + validation + error + timing + accountable disposition.

# State, data, authority, and trust flow



## TRUST BOUNDARY REGISTER

**TB-01**
Human intent versus model interpretation

**TB-02**
Model and agent proposal versus authorization

**TB-03**
Authorization versus deterministic execution

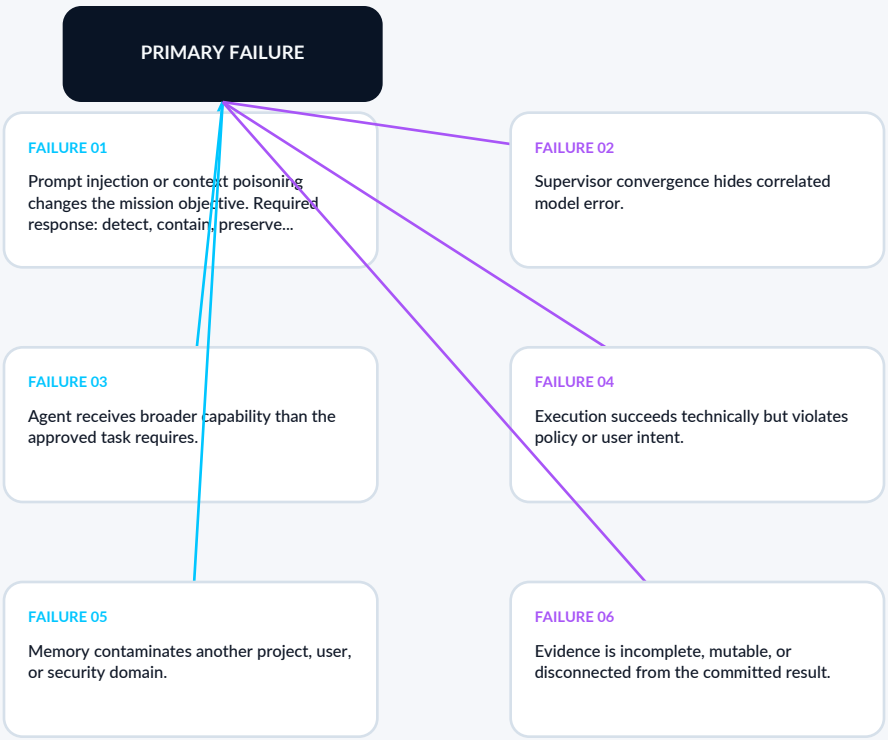
**TB-04**
Private project context versus shared organizational memory

**TB-05**
Tool connector versus external infrastructure

**TB-06**
Runtime evidence versus mutable conversational narrative

# Failure propagation and observability

## FAILURE PATHS




## OBSERVABILITY STACK

**OUTCOME**
User / mission result

**SERVICE**
SLOs / availability

**CONTROL**
Policy / authorization

**EXECUTION**
State / errors / retries

**DEPENDENCY**
External health / latency

**EVIDENCE**
Trace / artifact / receipt

**RECOVERY RULE** Closure requires containment, causal explanation, restored service, reconciled intended vs observed state, preserved evidence, and an explicit residual-risk decision.

# Deployment profiles and implementation roadmap

## DEPLOYMENT PROFILES

**PROFILE 01** — **SINGLE-USER LOCAL WORKSTATION**

Bounded proof

**PROFILE 02** — **GOVERNED ENGINEERING TEAM WORKSPACE**

Team-scale governance

**PROFILE 03** — **AIR-GAPPED OR SOVEREIGN ENVIRONMENT**

Sovereign / high-assurance

**PROFILE 04** — **ENTERPRISE CONTROL PLANE WITH DISTRIBUTED RUNNERS**

Distributed enterprise

## IMPLEMENTATION ROADMAP

| 0-30 DAYS | 31-90 DAYS | 3-6 MONTHS | 6-12 MONTHS | ONGOING |
|---|---|---|---|---|
| Scope / baseline | Pilot / contracts | Scale / integrate | Assure / automate | Measure / revalidate |

Progression is evidence-gated. Architecture adoption does not advance because a component is installed or demonstrated; each stage requires measurable service, safety, security, recovery, and evidence outcomes.

# Verification plan, dependencies, and reading map

## ACCEPTANCE CRITERIA - REPRESENTATIVE SET

| | |
|---|---|
| AC-01 | The architecture has a named service owner, technical owner, security owner, data owner, and recovery authority. |
| AC-02 | The system boundary, external dependencies, retained responsibilities, and trust transitions are documented and |
| AC-03 | Identity, authorization, secrets, policy, and privileged operations are explicit and independently auditable. |
| AC-04 | Desired state, runtime state, observed state, and evidence state are distinguishable and reconciled. |
| AC-05 | Consequential changes require preconditions, approval policy, bounded execution, verification, and rollback. |
| AC-06 | Telemetry supports service health, security detection, forensic reconstruction, and conformance evidence. |
| AC-07 | Failure modes include safe degradation, dependency loss, corrupted state, operator error, and malicious action. |
| AC-08 | Backup, restoration, reconstitution, and exit procedures are exercised against defined recovery objectives. |

## RELATED REFERENCE ARCHITECTURES

No mandatory RA dependencies declared

## MAPPED MYTHOS STANDARDS

MYTHOS-AI-0001, MYTHOS-AI-0002, MYTHOS-AI-0004, MYTHOS-AI-0007, MYTHOS-KNW-0001, MYTHOS-DAT-0005, MYTHOS-ID-0002

## DETAILED SPECIFICATION READING MAP



The following pages preserve the detailed candidate architecture specification while this visual dossier adds explicit architecture diagrams, sequence views, state and evidence flows, observability, failure propagation, and implementation gates.

# PART I - EXECUTIVE DIRECTION

## OUTCOMES, RISKS, AND DECISIONS

**ARCHITECTURE MANDATE**

A governed, durable multi-agent architecture that separates reasoning, authorization, deterministic execution, verification, memory, observability, and evidence.

### EXECUTIVE OUTCOMES

- Convert ambiguous human intent into explicit missions and typed plans.
- Permit models and agents to propose actions without silently acquiring execution authority.
- Execute consequential work through deterministic, policy-bound mechanisms.
- Preserve evidence, checkpoints, approvals, memory boundaries, and verified outcomes.

### DECISIONS REQUIRED

- Accountable service and risk ownership.
- Accepted architecture profile and consequence class.
- Authoritative identity, data, policy, configuration, and evidence systems.
- Mandatory approval, recovery, privacy, safety, and supplier-exit gates.
- Funding for operations, observability, validation, and lifecycle—not only implementation.

# PART II - SCOPE, PRINCIPLES, AND SYSTEM CONTEXT

## ARTIFICIAL INTELLIGENCE

### IN SCOPE

A governed, durable multi-agent architecture that separates reasoning, authorization, deterministic execution, verification, memory, observability, and evidence.

### OUT OF SCOPE

Vendor selection, exact sizing, contractual obligations, detailed low-level design, product certification, regulatory approval, and proof that any named platform is implemented.

### ARCHITECTURE INVARIANTS

Explicit ownership; least authority; separable desired, runtime, observed, and evidence state; bounded automation; tested recovery; product-neutral interfaces; source and claim currency.

### ASSUMPTIONS

Organizations tailor the pattern through documented risk, environment, criticality, data, safety, regulatory, workforce, and supplier analysis.

# SYSTEM CONTEXT AND RESPONSIBILITY MODEL

## ACTORS, ENVIRONMENTS, AND EXTERNAL DEPENDENCIES

### ACTORS AND RESPONSIBILITY

- Service consumers and affected users
- Product, platform, network, security, data, reliability, and support teams
- Human administrators, operators, reviewers, incident commanders, and risk owners
- Machine identities, agents, workloads, devices, controllers, and external services
- Suppliers, providers, carriers, integrators, assessors, and regulators where applicable

### RESPONSIBILITY BOUNDARY

- The adopting organization retains accountability for purpose, risk, configuration, data, identity, operation, recovery, and evidence even where a provider performs underlying functions.
- Inherited controls are recorded with provider, scope, evidence, limitations, shared responsibility, incident path, and exit conditions.
- No component is considered trusted solely because it is internal, managed, signed, cloud-hosted, model-generated, or supplied by a recognized vendor.

**ARCHITECTURE POSITION**

This candidate architecture describes the specified PANTHEON direction. It is not evidence that the platform is released, implemented in full, or production-certified.

# PART III - LOGICAL ARCHITECTURE

## LAYERS AND RESPONSIBILITIES

**01**

### Experience and Intent

Experience and Intent owns a bounded set of responsibilities, interfaces, state, controls, failure behavior, telemetry, recovery procedures, and evidence. It does not silently absorb authority assigned to another layer.

**02**

### Mission and Plan Compiler

Mission and Plan Compiler owns a bounded set of responsibilities, interfaces, state, controls, failure behavior, telemetry, recovery procedures, and evidence. It does not silently absorb authority assigned to another layer.

**03**

### Agent and Model Fabric

Agent and Model Fabric owns a bounded set of responsibilities, interfaces, state, controls, failure behavior, telemetry, recovery procedures, and evidence. It does not silently absorb authority assigned to another layer.

**04**

### Governance and Authorization

Governance and Authorization owns a bounded set of responsibilities, interfaces, state, controls, failure behavior, telemetry, recovery procedures, and evidence. It does not silently absorb authority assigned to another layer.

**05**

### Deterministic Execution

Deterministic Execution owns a bounded set of responsibilities, interfaces, state, controls, failure behavior, telemetry, recovery procedures, and evidence. It does not silently absorb authority assigned to another layer.

**06**

### Verification, Evidence, and Memory

Verification, Evidence, and Memory owns a bounded set of responsibilities, interfaces, state, controls, failure behavior, telemetry, recovery procedures, and evidence. It does not silently absorb authority assigned to another layer.

# CORE COMPONENTS - A

## COMPONENTS 01-03

### COMPONENT 01

Mission workspace and conversation interface

Required contract: owner, purpose, inputs, outputs, authority, data, dependencies, health, failure behavior, recovery, lifecycle, and evidence.

### COMPONENT 02

Supervisor, specialist, and sub-specialist agents

Required contract: owner, purpose, inputs, outputs, authority, data, dependencies, health, failure behavior, recovery, lifecycle, and evidence.

### COMPONENT 03

Model routing, profiles, budgets, and context compiler

Required contract: owner, purpose, inputs, outputs, authority, data, dependencies, health, failure behavior, recovery, lifecycle, and evidence.

# CORE COMPONENTS - B

## COMPONENTS 04-06

### COMPONENT 04

Policy decision point, approval service, and capability broker

Required contract: owner, purpose, inputs, outputs, authority, data, dependencies, health, failure behavior, recovery, lifecycle, and evidence.

### COMPONENT 05

Durable mission runtime, DAG scheduler, sandbox, and connector plane

Required contract: owner, purpose, inputs, outputs, authority, data, dependencies, health, failure behavior, recovery, lifecycle, and evidence.

### COMPONENT 06

Verification harness, evidence vault, observability, and memory services

Required contract: owner, purpose, inputs, outputs, authority, data, dependencies, health, failure behavior, recovery, lifecycle, and evidence.

# PART IV - DATA, CONTROL, AUTHORITY, AND EVIDENCE FLOW

## FLOW CONTRACTS

1. Intent enters a mission envelope with owner, purpose, scope, budget, and consequence class.

   Evidence: request and actor identity, policy decision, versions, state transitions, outputs, tests, errors, timing, and accountable disposition.

2. Planning fan-out produces candidate plans, assumptions, dependencies, tests, and risk statements.

   Evidence: request and actor identity, policy decision, versions, state transitions, outputs, tests, errors, timing, and accountable disposition.

3. Policy evaluates requested capabilities and inserts human approvals where required.

   Evidence: request and actor identity, policy decision, versions, state transitions, outputs, tests, errors, timing, and accountable disposition.

4. Runtime executes typed steps; each step emits state, logs, artifacts, and verification evidence.

   Evidence: request and actor identity, policy decision, versions, state transitions, outputs, tests, errors, timing, and accountable disposition.

5. Supervisors reconcile outputs, resolve disagreement, and commit only verified outcomes.

   Evidence: request and actor identity, policy decision, versions, state transitions, outputs, tests, errors, timing, and accountable disposition.

# TRUST BOUNDARIES AND ENFORCEMENT

## EVERY CROSSING REQUIRES AN EXPLICIT CONTRACT

**TRUST BOUNDARY 01**

### Human intent versus model interpretation

Minimum contract: authenticated parties, authorized actions, data classification, protocol, encryption, validation, rate and resource limits, observability, failure behavior, revocation, and evidence.

**TRUST BOUNDARY 02**

### Model and agent proposal versus authorization

Minimum contract: authenticated parties, authorized actions, data classification, protocol, encryption, validation, rate and resource limits, observability, failure behavior, revocation, and evidence.

**TRUST BOUNDARY 03**

### Authorization versus deterministic execution

Minimum contract: authenticated parties, authorized actions, data classification, protocol, encryption, validation, rate and resource limits, observability, failure behavior, revocation, and evidence.

**TRUST BOUNDARY 04**

### Private project context versus shared organizational memory

Minimum contract: authenticated parties, authorized actions, data classification, protocol, encryption, validation, rate and resource limits, observability, failure behavior, revocation, and evidence.

**TRUST BOUNDARY 05**

### Tool connector versus external infrastructure

Minimum contract: authenticated parties, authorized actions, data classification, protocol, encryption, validation, rate and resource limits, observability, failure behavior, revocation, and evidence.

**TRUST BOUNDARY 06**

### Runtime evidence versus mutable conversational narrative

Minimum contract: authenticated parties, authorized actions, data classification, protocol, encryption, validation, rate and resource limits, observability, failure behavior, revocation, and evidence.

# PART V - DEPLOYMENT PROFILES

## PROFILE CHANGES SCALE, NOT ASSURANCE OUTCOMES

### PROFILE 01 / SINGLE-USER LOCAL WORKSTATION

Use this profile to establish a bounded implementation with explicit scale, criticality, users, data, autonomy, external dependencies, recovery objectives, staffing, and validation depth. The profile may simplify technology or topology but must preserve identity, least authority, evidence, safe failure, recovery, lifecycle, and accountable risk acceptance.

### PROFILE 02 / GOVERNED ENGINEERING TEAM WORKSPACE

Use this profile to establish a bounded implementation with explicit scale, criticality, users, data, autonomy, external dependencies, recovery objectives, staffing, and validation depth. The profile may simplify technology or topology but must preserve identity, least authority, evidence, safe failure, recovery, lifecycle, and accountable risk acceptance.

### PROFILE 03 / AIR-GAPPED OR SOVEREIGN ENVIRONMENT

Use this profile to establish a bounded implementation with explicit scale, criticality, users, data, autonomy, external dependencies, recovery objectives, staffing, and validation depth. The profile may simplify technology or topology but must preserve identity, least authority, evidence, safe failure, recovery, lifecycle, and accountable risk acceptance.

### PROFILE 04 / ENTERPRISE CONTROL PLANE WITH DISTRIBUTED RUNNERS

Use this profile to establish a bounded implementation with explicit scale, criticality, users, data, autonomy, external dependencies, recovery objectives, staffing, and validation depth. The profile may simplify technology or topology but must preserve identity, least authority, evidence, safe failure, recovery, lifecycle, and accountable risk acceptance.

# PART VI - OPERATING LIFECYCLE

## FROM DISCOVERY THROUGH RETIREMENT

### 01 / Register mission and consequence

Entry and exit criteria, owner, evidence, exceptions, dependencies, rollback, and next decision are recorded. No phase is complete solely because components were purchased, configured, or demonstrated.

### 02 / Compile plan and acceptance tests

Entry and exit criteria, owner, evidence, exceptions, dependencies, rollback, and next decision are recorded. No phase is complete solely because components were purchased, configured, or demonstrated.

### 03 / Authorize capabilities and budgets

Entry and exit criteria, owner, evidence, exceptions, dependencies, rollback, and next decision are recorded. No phase is complete solely because components were purchased, configured, or demonstrated.

### 04 / Execute checkpointed work

Entry and exit criteria, owner, evidence, exceptions, dependencies, rollback, and next decision are recorded. No phase is complete solely because components were purchased, configured, or demonstrated.

### 05 / Verify outcome and collect evidence

Entry and exit criteria, owner, evidence, exceptions, dependencies, rollback, and next decision are recorded. No phase is complete solely because components were purchased, configured, or demonstrated.

### 06 / Integrate or roll back

Entry and exit criteria, owner, evidence, exceptions, dependencies, rollback, and next decision are recorded. No phase is complete solely because components were purchased, configured, or demonstrated.

### 07 / Retain, summarize, and reassess memory

Entry and exit criteria, owner, evidence, exceptions, dependencies, rollback, and next decision are recorded. No phase is complete solely because components were purchased, configured, or demonstrated.

# SECURITY, PRIVACY, SAFETY, AND RIGHTS

## CROSS-CUTTING ASSURANCE

### SECURITY

Threat model identities, trust boundaries, supply chain, control planes, data, protocols, privilege, abuse, detection, response, recovery, and residual risk.

### PRIVACY

Purpose limitation, minimization, transparency, consent where required, access, correction, retention, deletion, sovereignty, metadata, and third-party processing.

### SAFETY

Hazards, unsafe states, human intervention, physical consequence, dependency failure, automation limits, emergency stop, recovery sequence, and validation.

### RIGHTS AND USABILITY

Accessibility, contestability, explanation, operator workload, affected-user communication, grievance, misuse prevention, and equitable performance.

### SUPPLY CHAIN

Provenance, version, SBOM or component inventory, signature, vulnerability, support, update, provider, subcontractor, and exit.

### EVIDENCE

Design decisions, tests, telemetry, incidents, approvals, exceptions, recovery exercises, source currency, and review findings remain linked to the architecture version.

# FAILURE MODES AND RESILIENCE

## DESIGN FOR SAFE DEGRADATION AND TRUSTWORTHY RECOVERY

### FAILURE SCENARIO 01

Prompt injection or context poisoning changes the mission objective.

Required response: detect, contain, preserve evidence, maintain or enter safe service, communicate, recover, verify, learn, and prevent recurrence.

### FAILURE SCENARIO 02

Supervisor convergence hides correlated model error.

Required response: detect, contain, preserve evidence, maintain or enter safe service, communicate, recover, verify, learn, and prevent recurrence.

### FAILURE SCENARIO 03

Agent receives broader capability than the approved task requires.

Required response: detect, contain, preserve evidence, maintain or enter safe service, communicate, recover, verify, learn, and prevent recurrence.

### FAILURE SCENARIO 04

Execution succeeds technically but violates policy or user intent.

Required response: detect, contain, preserve evidence, maintain or enter safe service, communicate, recover, verify, learn, and prevent recurrence.

### FAILURE SCENARIO 05

Memory contaminates another project, user, or security domain.

Required response: detect, contain, preserve evidence, maintain or enter safe service, communicate, recover, verify, learn, and prevent recurrence.

### FAILURE SCENARIO 06

Evidence is incomplete, mutable, or disconnected from the committed result.

Required response: detect, contain, preserve evidence, maintain or enter safe service, communicate, recover, verify, learn, and prevent recurrence.

# PART VII - INTEGRATION AND DEPENDENCY MODEL

## CONNECTED REFERENCE ARCHITECTURES AND STANDARDS

### REFERENCE ARCHITECTURE DEPENDENCIES

- RA-006 - Local-First AI Inference Cluster and Edge Node
- RA-007 - AI-Assisted Software Engineering Pipeline
- RA-009 - Zero-Trust Identity, Policy, Secrets, and Access Fabric
- RA-010 - Local-First Knowledge, Memory, and Evidence Platform
- RA-011 - Digital Twin, Infrastructure Automation, and Change Assurance
- RA-012 - Enterprise Observability, SRE, and Incident Command Platform
- RA-016 - Realtime Voice, Media, and Multimodal Interaction Platform
- RA-019 - Sovereign Browser, Remote Access, and Web Automation
- RA-020 - Adaptive Learning, Cyber Range, and Workforce Assurance

### MAPPED MYTHOS STANDARDS

- MYTHOS-AI-0001
- MYTHOS-AI-0002
- MYTHOS-AI-0004
- MYTHOS-AI-0007
- MYTHOS-KNW-0001
- MYTHOS-DAT-0005
- MYTHOS-ID-0002

**DEPENDENCY DOCTRINE**

A dependency is not a blanket trust grant. Each connection requires an interface, identity, policy, data, availability, version, failure, recovery, and evidence contract. Correlated failure and circular authority must be analyzed across the full architecture graph.

# PART VIII - IMPLEMENTATION AND ADOPTION ROADMAP

## PHASED, EVIDENCE-DRIVEN TRANSITION

| | |
|---|---|
| **0-30 DAYS** | Inventory purpose, owners, users, systems, data, dependencies, risks, current diagrams, incidents, recovery, and evidence gaps. |
| **31-90 DAYS** | Establish authoritative identity, configuration, policy, data, source, and architecture records; define target profile and acceptance tests. |
| **3-6 MONTHS** | Implement bounded foundations, isolation, telemetry, change control, backup, recovery, and representative pilot workflows. |
| **6-12 MONTHS** | Expand through standard patterns, automation, service ownership, SLOs, independent assessment, failure injection, and supplier-exit exercises. |
| **ONGOING** | Reconcile drift, review sources, reassess risk, measure outcomes, exercise recovery, renew exceptions, migrate dependencies, and retire obsolete paths. |

# ARCHITECTURE ACCEPTANCE CRITERIA - A

## CRITERIA 01-09

| | |
|---|---|
| **AC-01** | The architecture has a named service owner, technical owner, security owner, data owner, and recovery authority. |
| **AC-02** | The system boundary, external dependencies, retained responsibilities, and trust transitions are documented and reviewed. |
| **AC-03** | Identity, authorization, secrets, policy, and privileged operations are explicit and independently auditable. |
| **AC-04** | Desired state, runtime state, observed state, and evidence state are distinguishable and reconciled. |
| **AC-05** | Consequential changes require preconditions, approval policy, bounded execution, verification, and rollback. |
| **AC-06** | Telemetry supports service health, security detection, forensic reconstruction, and conformance evidence. |
| **AC-07** | Failure modes include safe degradation, dependency loss, corrupted state, operator error, and malicious action. |
| **AC-08** | Backup, restoration, reconstitution, and exit procedures are exercised against defined recovery objectives. |
| **AC-09** | Data classification, retention, privacy, sovereignty, and disposition controls follow the actual information flow. |

# ARCHITECTURE ACCEPTANCE CRITERIA - B

## CRITERIA 10-18

| | |
|---|---|
| **AC-10** | Supplier, model, protocol, platform, and software dependencies have provenance, version, support, and replacement records. |
| **AC-11** | Representative performance and capacity tests include saturation, contention, latency, and degraded conditions. |
| **AC-12** | Exceptions are time-bounded, visible, approved by accountable risk owners, and linked to compensating controls. |
| **AC-13** | Every autonomous or semi-autonomous action resolves to a mission, actor, policy decision, capability grant, execution trace, verification result, and accountable owner. |
| **AC-14** | Model output cannot directly bypass deterministic execution controls for consequential actions. |
| **AC-15** | Parallel agent and multi-model disagreement is retained and resolved through explicit arbitration criteria. |
| **AC-16** | Memory promotion, sharing, correction, expiration, and deletion are governed independently from chat history. |
| **AC-17** | PANTHEON-specific labels are treated as architecture assignments rather than proof of current implementation. |
| **AC-18** | A kill switch, pause, checkpoint recovery, budget stop, and safe rollback are demonstrated for representative missions. |

# METRICS, EVIDENCE, AND DECISION GATES

## MEASURE SERVICE OUTCOMES AND ASSURANCE

### COVERAGE

Percent of components, interfaces, identities, data flows, dependencies, risks, and controls represented in current architecture evidence.

### CHANGE QUALITY

Plan success, rollback, verification pass, escaped defect, drift, and emergency-change rates.

### RELIABILITY

Availability, latency, saturation, dependency health, recovery time, recovery point, and safe-degradation success.

### SECURITY

Privilege exposure, policy denial, identity compromise, vulnerable dependency, detection coverage, response, and evidence completeness.

### HUMAN OUTCOME

Task success, operator workload, accessibility, override, contestability, training, and user-impact measures.

### LIFECYCLE

Version currency, unsupported component, exception age, migration readiness, restore exercise, and retirement completeness.

Release or expansion gate: mandatory acceptance criteria pass; failed safety, privacy, identity, recovery, evidence, or legal gates block promotion regardless of aggregate score.

# SOURCE AUTHORITY REGISTER

## CURRENT PRIMARY AND OFFICIAL REFERENCES

**SOURCE 01** **NIST AI Risk Management Framework 1.0 (revision in progress as of publication date)**

https://www.nist.gov/itl/ai-risk-management-framework

**SOURCE 02** **NIST AI 600-1, Generative AI Profile**

https://doi.org/10.6028/NIST.AI.600-1

**SOURCE 03** **NIST SP 800-53 Rev. 5**

https://doi.org/10.6028/NIST.SP.800-53r5

**SOURCE 04** **NIST SP 800-207, Zero Trust Architecture**

https://csrc.nist.gov/pubs/sp/800/207/final

**SOURCE 05** **NIST SP 800-218, Secure Software Development Framework**

https://csrc.nist.gov/pubs/sp/800/218/final

**SOURCE 06** **OpenTelemetry Specification**

https://opentelemetry.io/docs/specs/

Source currency note. Official frameworks, specifications, and implementation guidance evolve. Assessors must verify the current version, status, errata, applicability, and effective date at adoption and scheduled review. NIST AI RMF 1.0 was under revision at the publication date.

# PUBLICATION CONTROL AND REVISION

## PERMANENT IDENTITY / CONTROLLED EVOLUTION

| | |
|---|---|
| PERMANENT DOCUMENT IDENTITY | RA-001 |
| VERSION | 1.0.0-candidate |
| PUBLICATION DATE | 2026-07-24 |
| SCHEDULED REVIEW | 2027-01-24 |
| STATUS | Candidate Reference Architecture |
| ARCHITECTURE SET | Canonical Set |
| CLASSIFICATION | Public |
| PUBLISHER | Mythos Systems |

### REVISION RECORD

1.0.0-candidate (2026-07-24) - Permanent-publication candidate edition regenerated under the Mythos Systems architecture publication system. Production sequence metadata is excluded from public identity.