The AI PM Builder's Manual

Building Governed AI Systems: A Practical Guide to Multi-Agent Development

- Jerome Davis

Purpose of This Manual

This User Manual is the definitive operational guide for the AI PM Builder's Template, an enterprise-grade, governance-first framework for building AI applications using multi-agent agentic coding platforms. It provides the procedural knowledge needed to deploy, operate, and sustain the framework across projects of any scale.

The manual is designed for three audiences: the Human Director who orchestrates strategy and approvals, the agent fleet operators who manage individual agent sessions, and compliance stakeholders who need to understand the governance evidence chain.

What the Framework Is

The AI PM Builder's Template is a complete consulting-grade toolkit for AI project governance and execution. It provides a 14-agent specialized workforce, 22 audit-ready governance templates, 5 mandatory operational directives, a self-correcting quality system, and a full brand identity system. Every component is engineered for traceability, repeatability, and standards compliance.

The framework transforms the Human Director from a hands-on doer into a strategic orchestrator. You set direction, make key decisions, and remove blockers. Your agent team handles tactical execution through structured handoffs, parallel workflows, and self-correcting quality protocols.

Standards Compliance

The framework harmonizes eight major AI governance and security standards into a unified operational model:

Standard	Coverage
CPMAI v7	Full 6-phase iterative lifecycle with hard phase gates
ISO/IEC 42001	AIMS Clauses 4–10 and Annex A controls (A.2–A.18)
NIST AI RMF 1.0	All four functions: Govern, Map, Measure, Manage
NIST SP 800-53 Rev 5	Security and privacy controls mapped per CPMAI phase
DoD CSRMC	All 8 modernization elements (MRP, CCV, AEP, CRPR, ACVR, Telemetry, Reciprocity, Resilience)
NIST SP 1270	Bias identification and mitigation protocols
NIST AI 100-1	Generative AI security considerations
OMB M-24-10	Federal AI governance expectations

These standards are not applied in isolation. The framework's crosswalk matrices harmonize them so that synthesized artifacts satisfy multiple standards simultaneously.

Key Capabilities Summary

- **14 Specialized AI Agents:** From Requirements BA through Program Analyst, each with defined roles, self-correction protocols, and documentation responsibilities
- **22 Governance Templates:** Audit-ready artifacts covering CPMAI v7 lifecycle, CSRMC modernization, ISO 42001, and NIST AI RMF
- **5 Mandatory Directives:** AI governance framework, self-annealing protocol, human reporting protocol, director interview protocol, and branding guide
- **Self-Annealing Quality System:** 4-phase error detection and correction loop (Validate → Execute → Verify → Correct)
- **Dual-Layer Architecture:** Reusable template layer at repo root with isolated project instances under projects/
- **Multi-Platform Support:** Claude Code (CLAUDE.md) and Google Anti-Gravity (GEMINI.md) with shared agent fleet
- **Algorithmic Authority Brand System:** Full Cyber-Secure Futurism design guide with hex values, typography, component specs, and AI prompt templates

System Architecture

Dual-Layer Architecture

The system operates as a two-tier architecture separating shared, reusable resources from project-specific artifacts.

Template Layer (Repository Root)

The template layer lives at the repository root and contains resources shared across all projects. This includes:

- **CLAUDE.md / GEMINI.md:** Universal context files every agent reads at session start
- **.agent/souls/:** 14 specialized agent SOUL files defining identity, values, working style, and expertise
- **directives/:** 22 governance templates and 5 mandatory directives
- **PORTFOLIO.md:** Cross-project portfolio tracker and showcase
- **new-project.sh:** Scaffolding script for creating isolated project instances
- **orchestration/:** Sprint planning templates and cross-project coordination
- **memory/:** Cross-project patterns, standards, and long-term learnings

Project Layer (projects/[project-name]/)

Each project is completely self-contained under its own directory with the following structure:

- **PROJECT.md:** Project identity card with business objectives, tech stack, scope, and demo links
- **execution/:** Implementation artifacts organized by domain (code/, docs/, artifacts/)
- **governance/:** Project-specific compliance evidence, audit trails, and decisions
- **memory/:** Daily activity logs (YYYY-MM-DD.md) and long-term project learnings (MEMORY.md)
- **orchestration/:** Task board (tasks.md) and dependency graph (dependencies.md)

Three-Layer Governance Architecture

Within each project, work flows through three governance layers:

Layer	Purpose	Location
Directives	Strategic rules and constraints shared across all projects	directives/ (repo root)
Orchestration	Agent coordination, task management, and dependency tracking	projects/[name]/orchestration/
Execution	Tactical implementation by specialized agents	projects/[name]/execution/

The architectural principle is that directives are shared but instantiations are isolated. The template for a security policy lives in directives/templates/. When a project begins, you create a project-specific instance in projects/[name]/governance/ with details tailored to that project's regulatory and mission requirements.

Repository Structure

The complete repository layout:

ai-pm-builder-template/ (repo root)

- CLAUDE.md — Claude Code shared agent context
- GEMINI.md — Google Anti-Gravity shared agent context
- PORTFOLIO.md — All projects overview and showcase
- README.md — Repository overview
- QUICK-START-GUIDE.md — Comprehensive setup guide
- QUICK-REFERENCE.md — Daily workflow reference card
- new-project.sh — Project scaffolding script
- .agent/souls/ — 14 specialized agent SOUL files
- directives/ — 5 mandatory directives + templates/ (22 governance templates)
- orchestration/ — Sprint planning templates
- memory/ — Cross-project learnings
- projects/ — Project instances directory

Agent Fleet

Fleet Roster

The framework deploys 14 specialized AI agents organized into five functional teams. Each agent has a dedicated SOUL file in .agent/souls/ that defines their identity, values, working style, and domain expertise.

Agent	Team	Primary Function
Scrum Master	Project Management	Sprint planning, blocker resolution, Director interface, task coordination
Requirements BA	Business Analysis	Stakeholder requirements elicitation, governance considerations
User Story BA	Business Analysis	Requirements-to-user-story conversion, acceptance criteria
Architecture SE	Systems Engineering	System design, ADRs, threat modeling, technology selection
Documentation SE	Systems Engineering	Documentation quality, ISO 42001 Clause 7.5 compliance
Database Engineer	Data Engineering	Schema design, data lineage, data governance, query optimization
Backend Developer	Development	API development, secure coding, business logic
Frontend Developer	Development	UI implementation, brand compliance, accessibility
UI/UX Designer	Design	Interface design, human oversight, accessibility, wireframes
QA Engineer	Quality	Functional testing, acceptance validation, test planning
Automation Test Engineer	Quality	Test automation, CCV rulesets, CI/CD integration
Pipeline DevOps	Operations	CI/CD pipelines, deployment security, secrets management
Performance DevOps	Operations	Telemetry, performance monitoring, alerting, runbooks
Program Analyst	Governance	CPMAI enforcement, phase gates, evidence orchestration

SOUL File Structure

Each agent's SOUL file is a comprehensive identity document containing:

- **Identity and Values:** Who the agent is, what they care about, and their working philosophy
- **Responsibilities:** Specific deliverables and ownership areas within the project
- **Verification Checklists:** Domain-specific self-review criteria for the self-annealing protocol
- **Handoff Requirements:** What must be included when passing work to downstream agents
- **Director Interview Procedures:** How to engage the Human Director when unknowns arise
- **Anti-Patterns:** What the agent should never do

Agent Communication Pattern

Agents coordinate through a structured handoff pattern:

Agent A completes work and creates artifacts in their designated execution directory

Agent A runs the Verify phase (self-review against acceptance criteria)

Agent A writes a detailed handoff in the daily memory file including self-review summary

Agent A updates the task board in orchestration/tasks.md

Agent B starts a new session and reads the shared context (CLAUDE.md or GEMINI.md)

Agent B reads their SOUL file for identity and working style

Agent B reads the daily memory file for handoff from Agent A

Agent B runs the Validate phase (pre-flight check on inputs)

Agent B executes their specialized work

Agent B repeats the cycle for Agent C

Work Modes

Agents operate in three work modes:

Solo Mode: Working independently on a well-scoped task. The agent checks the daily memory file for assignment, executes, documents work, and writes results back to project memory.

Coordinated Mode: Part of a sequential workflow where upstream or downstream dependencies exist. The agent checks task dependencies before starting and ensures complete handoff documentation.

Parallel Mode: Working simultaneously with other agents on independent tasks. Agents work in designated directories to avoid file conflicts and coordinate through the memory layer.

CPMAI v7 Lifecycle

Overview

Every AI project follows the CPMAI six-phase lifecycle. Phase gates are hard gates — no advancement without Program Analyst sign-off. The lifecycle is iterative: later phases can trigger re-evaluation of earlier phases when new information emerges.

Phase I — Business Understanding

Define business objectives, mission alignment, stakeholders, governance scope, risk criteria, and success metrics.

Key Deliverables: Mission Risk Profile (MRP), initial Statement of Applicability (SoA), Governance Scope Statement, and initial Risk Register.

Primary Agents: Requirements BA, Scrum Master, Program Analyst.

Standards Addressed: CSRMC MRP, NIST AI RMF GOVERN-1, ISO 42001 Clause 6.1.

Phase II — Data Understanding

Assess data sources, quality, provenance, lineage, bias risk, privacy constraints, and representativeness.

Key Deliverables: Telemetry Configuration, Reciprocity and Inheritance Register, Data Governance Documentation.

Primary Agents: Database Engineer, Architecture SE, Program Analyst.

Standards Addressed: CSRMC Telemetry/Reciprocity, NIST AI RMF MAP-2, ISO 42001 A.4.

Phase III — Data Preparation

Clean, transform, label, augment, and version datasets. Establish automated evidence packaging for data lineage. Validate telemetry instrumentation in data pipelines.

Key Deliverables: Automated Evidence Package (AEP), Data Lineage Record.

Primary Agents: Database Engineer, Backend Developer, Program Analyst.

Standards Addressed: CSRMC AEP, ISO 42001 Clause 7.5, NIST AI RMF cross-cutting.

Phase IV — Model Development

Select modeling approaches, train, test, evaluate early-stage performance. Incorporate threat modeling, adversarial testing, and explainability.

Key Deliverables: Cyber Resilience Posture Report (CRPR), Automated Control Validation Ruleset (ACVR), Threat Model, Bias Assessment, Architecture Decision Records (ADRs).

Primary Agents: Backend Developer, Architecture SE, QA Engineer, Program Analyst.

Standards Addressed: CSRMC CRPR/ACVR, NIST AI RMF MAP-5/MEASURE-2.6, NIST SP 1270.

Phase V — Model Evaluation

Independent evaluation of fairness, robustness, transparency, privacy, and mission alignment. Complete pre-deployment Continuous Compliance Validation cycle. Consolidate evidence into the Automated Evidence Package.

Key Deliverables: CCV Report, Go/No-Go Recommendation, Model Evaluation Report, consolidated AEP.

Primary Agents: QA Engineer, Automation Test Engineer, Program Analyst.

Standards Addressed: CSRMC CCV, CPMAI Phase V Gate, NIST AI RMF MEASURE-1 through MEASURE-4.

Phase VI — Operationalization

Deploy to production, activate monitoring, telemetry, CCV cycles, and incident response. Validate resilience. Transition to continuous governance operations.

Key Deliverables: Operational Monitoring Plan, Incident Response Plan, production Telemetry Configuration.

Primary Agents: Pipeline DevOps, Performance DevOps, Program Analyst.

Standards Addressed: CSRMC Resilience/Telemetry, NIST AI RMF MANAGE-4, ISO 42001 A.7/A.16.

Phase Gate Review Process

Each gate review uses a standardized template with eight sections: Project and Phase Information, Purpose of the Gate, Required Deliverables and Evidence Checklist, Acceptance Criteria, Findings and Required Corrective Actions, Residual Risks and Deviations Accepted, Decision and Leadership Sign-Off, and Archival Instructions.

Gate decisions are: Approved, Conditionally Approved (with corrective action timeline), or Not Approved (with required remediation). The Program Analyst enforces all phase gate reviews.

Phase Gate Deliverable Mapping

Phase Gate	Required Templates
Gate 1 — Business Understanding	Phase Gate Review, Mission Risk Profile, Governance Scope Statement, Statement of Applicability, Risk Register
Gate 2 — Data Understanding	Phase Gate Review, Telemetry Configuration, Reciprocity and Inheritance Register, Data Governance Documentation, Risk Register
Gate 3 — Data Preparation	Phase Gate Review, Automated Evidence Package, Data Lineage Record, Risk Register
Gate 4 — Model Development	Phase Gate Review, CRPR, ACVR, Threat Model, Bias Assessment, ADR, Risk Register
Gate 5 — Model Evaluation	Phase Gate Review, CCV Report, Go/No-Go Recommendation, Model Evaluation Report, AEP, Risk Register
Gate 6 — Operationalization	Phase Gate Review, Operational Monitoring Plan, Incident Response Plan, Risk Register

Governance Templates

Template Inventory

The framework provides 22 governance and evidence generation templates organized by CPMAI lifecycle phase. All templates are located in directives/templates/ and designed for AI agent population during project execution.

Each template contains metadata headers with Document ID, Version, Status, Phase, and Applicable Standards; instructional placeholders in brackets for agent or human population; standards cross-references at each section level; and revision history and approvals sections for audit traceability.

Phase I Templates

Template	File	CSRMC Element	Primary Standards
Mission Risk Profile (MRP)	mission-risk-profile.md	MRP	CSRMC, NIST AI RMF GOVERN-1, ISO 42001 Clause 6.1
Governance Scope Statement	governance-scope-statement.md	—	ISO 42001 Clause 4.3/5.1, NIST AI RMF GOVERN-2
Statement of Applicability	statement-of-applicability.md	—	ISO 42001 Clause 6.1.3, NIST SP 800-53 CA-2

Phase II Templates

Template	File	CSRMC Element	Primary Standards
Telemetry Configuration	telemetry-configuration.md	Telemetry	CSRMC, NIST AI RMF MEASURE-2, ISO 42001 A.7
Reciprocity & Inheritance Register	reciprocity-inheritance-register.md	Reciprocity	CSRMC, NIST SP 800-53 CA-2, ISO 42001 Clause 8.1
Data Governance Documentation	data-governance-documentation.md	—	ISO 42001 A.4, NIST AI RMF MAP-2, NIST SP 1270

Phase III Templates

Template	File	CSRMC Element	Primary Standards
Automated Evidence Package (AEP)	automated-evidence-package.md	AEP	CSRMC, ISO 42001 Clause 7.5, NIST AI RMF cross-cutting
Data Lineage Record	data-lineage-record.md	—	ISO 42001 A.4, NIST AI RMF MAP-2.3

Phase IV Templates

Template	File	CSRMC Element	Primary Standards
Cyber Resilience Posture Report	cyber-resilience-posture-report.md	CRPR	CSRMC, NIST AI RMF MANAGE-2, NIST SP 800-53 CA-7
Automated Control Validation Ruleset	automated-control-validation-ruleset.md	ACVR	CSRMC, NIST SP 800-53 CA-2/CA-7
Threat Model	threat-model.md	—	NIST AI RMF MAP-5, NIST AI 100-1, ISO 42001 A.6
Bias Assessment	bias-assessment.md	—	NIST SP 1270, NIST AI RMF MEASURE-2.6, ISO 42001 A.10
Architecture Decision Record	architecture-decision-record.md	—	ISO 42001 Clause 7.5, NIST AI RMF GOVERN-1.2

Phase V Templates

Template	File	CSRMC Element	Primary Standards
CCV Report	ccv-report.md	CCV	CSRMC, NIST SP 800-53 CA-7, ISO 42001 Clause 9.1
Go/No-Go Recommendation	go-no-go-recommendation.md	—	CPMAI Phase V Gate, NIST AI RMF MANAGE-1
Model Evaluation Report	model-evaluation-report.md	—	NIST AI RMF MEASURE-1 through MEASURE-4

Phase VI Templates

Template	File	CSRMC Element	Primary Standards
Operational Monitoring Plan	operational-monitoring-plan.md	Telemetry	CSRMC, NIST AI RMF MANAGE-4, ISO 42001 A.7
Incident Response Plan	incident-response-plan.md	Resilience	NIST SP 800-53 IR-1 through IR-10, ISO 42001 A.16

Cross-Cutting Templates

Template	File	Primary Standards
Phase Gate Review	phase-gate-review.md	CPMAI Phase Gates, ISO 42001 Clause 9.3
Risk Register	risk-register.md	ISO 42001 Clause 6.1, NIST AI RMF GOVERN-5/MANAGE-2
Evidence Index	evidence-index.md	ISO 42001 Clause 7.5, CSRMC AEP
Governance Review Template	governance-review-template.md	ISO 42001 Clause 9.3, NIST AI RMF GOVERN-1.5
Standards Crosswalk Matrix	standards-crosswalk-matrix.md	ISO 42001 Clause 6.1.3, NIST AI RMF cross-cutting
Corrective Action Register	corrective-action-register.md	ISO 42001 Clause 10.2, NIST SP 800-53 CA-5

Template Usage Procedure

Check the Template Index for the correct template for your deliverable

Copy the template to the appropriate evidence repository folder

Populate all instructional placeholders with project-specific content

Update the metadata header (Document ID, Version, Date, Author/Agent, Status)

Submit for review per the Human Reporting Protocol

Update the Evidence Index with the new artifact

Mandatory Directives

The framework enforces five mandatory directives that apply to all agents across all projects. These directives are located in the directives/ directory at the repository root.

Self-Annealing Protocol

The self-annealing protocol establishes a mandatory self-correction capability across the entire agent fleet. Every piece of work follows a four-phase loop:

Validate (Pre-Flight Check): Verify upstream completeness, upstream quality, context currency, and scope clarity before starting any work. If pre-flight fails, do not proceed — document the blocker and notify.

Execute (Do the Work): Perform specialized work as defined in the agent's SOUL file.

Verify (Self-Review Gate): Check acceptance criteria met, consistency with upstream inputs, completeness for downstream agents, edge cases, and regression impact. Write an explicit self-review summary in the handoff.

Correct (Fix and Learn): When verification reveals a problem, classify the error (Defect, Omission, Inconsistency, or Degradation), determine correction scope, apply the fix at the root cause, and document the learning as an Annealing Record.

Retry Protocol

When an operation fails: maximum 3 attempts, change something between each attempt, document each attempt, and escalate after 3 failures with full context.

Circuit Breaker Protocol

The circuit breaker triggers when the same error class occurs 3 or more times across different tasks, a correction creates a new error, two or more agents are blocked by the same root cause, or corrections take longer than the original work. When triggered: stop all affected work immediately, notify the Scrum Master, diagnose the systemic root cause, resolve before resuming, and document in MEMORY.md.

Self-Annealing Metrics

Metric	What It Measures	Target
Self-catch rate	Errors caught by producing agent vs. found downstream	> 80%
Correction cycle time	Time from error detection to verified fix	< 2 hours
Recurrence rate	Same error class after prevention was documented	< 10%
Circuit breaker frequency	How often systemic issues trigger circuit breaker	Decreasing over time
Rollback frequency	How often forward-fix fails and rollback needed	< 5% of corrections

Human Reporting Protocol

The Scrum Master serves as the single point of contact between the agent fleet and the Human Director. All agents report through the Scrum Master, who delivers structured briefings at five mandatory touchpoints:

Task Completion Briefing: Every time a task moves to Done. Includes task ID, agent, deliverables, self-review status, corrections, impact, and whether Director attention is needed.

Sprint Start Briefing: Beginning of each sprint. Includes sprint goal, committed work, capacity, key dependencies, risks, and decisions needed from the Director.

Blocker Escalation: Immediately when a blocker cannot be resolved within 2 hours. Includes severity, affected agents, what has been tried, specific Director input needed, and impact if unresolved.

Sprint Completion Briefing: End of each sprint. Includes goal achievement, completed work, carried-over work, sprint metrics (velocity, cycle time, blocker rate, self-annealing metrics), and recommended next-sprint priorities.

Circuit Breaker Notification: Immediately when the self-annealing circuit breaker fires. Includes affected agents, pattern detected, root cause, resolution plan, and estimated timeline impact.

Approval Gates

The following decisions require explicit Human Director approval: sprint scope and priorities, architecture direction changes, scope additions mid-sprint, rollback decisions, circuit breaker resolution approach, external dependency decisions, and go/no-go for deployment. No agent proceeds past an approval gate without documented Director approval.

Director Interview Protocol

This protocol governs how agents engage the Human Director when they encounter unknowns, ambiguities, or gaps. An agent must initiate a Director Interview when any of these conditions exist: missing business context, ambiguous requirements, template population gaps, risk tolerance decisions, policy or standard interpretation questions, cross-agent dependency unknowns, or scope and priority conflicts.

The interview follows five steps: State Context (identify yourself and the triggering artifact), Present What You Know (demonstrate due diligence), Present Specific Questions (numbered, with rationale and consequence if unanswered), Propose Defaults (when possible, for Director confirmation), and Document the Response (in the appropriate artifact, memory file, or task board).

Assumption Documentation

When agents proceed with assumptions on low-risk items, they must document them inline using the format: [ASSUMPTION: description | Risk: Low/Medium | Awaiting Director confirmation at next review | Agent: role | Date: YYYY-MM-DD]. All Medium-risk assumptions must be resolved before the artifact passes its phase gate review.

AI Governance Framework Directive

This directive defines the governance structure, lifecycle methodology, compliance requirements, and artifact expectations that all agents must follow. It establishes the seven-domain AI risk taxonomy (Technical, Ethical, Operational, Cybersecurity, Privacy, Regulatory, Mission-Driven), six evidence categories, the governance cadence (Operational Weekly, Governance Bi-Weekly/Monthly, Executive Quarterly, Audit Annual/Triggered), and the escalation model.

Branding Guide

The Algorithmic Authority branding guide defines the visual and verbal identity for all user-facing outputs. The aesthetic is Cyber-Secure Futurism. The guide specifies the complete color palette (Brushed Titanium, Electric Cyan, Deep Sky Blue, Charcoal, Clean White, Neutral Grey), typography hierarchy (Eurostile Bold Extended through JetBrains Mono), UI component standards (buttons, cards, navigation, tables, charts, forms), and brand voice rules (surgical, concise, authoritative, structured, high-level).

Operational Procedures

Creating a New Project

To scaffold a new project from the repository root:

Run ./new-project.sh my-project-name from the repo root. This creates the complete directory structure under projects/my-project-name/ with template files ready for customization.

Open projects/my-project-name/PROJECT.md and fill in the project-specific context: business problem, technologies chosen, stakeholders, success criteria.

Review which governance templates from directives/ apply to the project. Copy relevant templates into projects/my-project-name/governance/ and customize for specific constraints.

Create the initial task board in projects/my-project-name/orchestration/tasks.md with major epics and stories.

Begin the first agent session.

Starting an Agent Session

Every agent session follows the same initialization sequence:

Navigate to the repository root directory.

Start the agentic coding platform (Claude Code or Google Anti-Gravity).

Read the shared context file (CLAUDE.md or GEMINI.md).

Read the agent's SOUL file from .agent/souls/[agent-name].md.

Read the project context from projects/[project-name]/PROJECT.md.

Check today's memory file at projects/[project-name]/memory/YYYY-MM-DD.md for assignments and handoffs.

Review the task board at projects/[project-name]/orchestration/tasks.md.

Run the Validate (pre-flight) phase before beginning execution.

Ending an Agent Session

Before closing any agent session:

Run the Verify phase — self-review output against acceptance criteria.

Write a complete handoff summary in projects/[project-name]/memory/YYYY-MM-DD.md including self-review summary.

List all files created with their paths.

Note any corrections made (annealing records) or blockers discovered.

Update projects/[project-name]/orchestration/tasks.md if needed.

Commit to git with message format: Session: [PROJECT] [AGENT] — [what was accomplished].

Sequential Handoff Workflow

For sequential workflows where one agent's output feeds the next:

Session 1 (e.g., Requirements BA): Initialize the agent, execute their specialty work, write the handoff summary to the daily memory file explaining what was captured, where documents are located, and what the next agent needs to know. Close the session.

Session 2 (e.g., User Story BA): Initialize the new agent, have them read the daily memory file for the handoff from the previous agent. The agent picks up where the previous agent left off and continues the workflow.

Parallel Execution Workflow

For parallel workflows where multiple agents work simultaneously:

Identify independent tasks from the task board that have no mutual dependencies.

Open separate terminal sessions or browser instances, one per agent.

Each agent works in their designated execution subdirectory to avoid file conflicts.

Agents coordinate through the shared daily memory file.

The Scrum Master monitors the memory file and task board to track when parallel work completes.

Managing Multiple Projects

The dual-layer architecture supports simultaneous projects. Each project under projects/ is completely isolated. You might have a banking project in Phase IV with Backend Developer and Database Engineer working in parallel, a SaaS project in Phase I going through requirements, and an internal tool project deploying to production — all running concurrently without interference.

All projects share the same agent SOULs and governance templates from the repo root, but each instantiates them for its own specific constraints.

Completing a Project

When a project reaches production:

Write a project closure summary in projects/[name]/CLOSURE.md documenting start/end dates, final vs. initial scope, key deliverables, major blockers and resolutions, technology lessons, workflow lessons, metrics, and recommendations.

Update PORTFOLIO.md at the repository root with a project summary entry.

Archive the project directory (it remains in the repository for reference).

CSRMC Modernization Elements

The DoD Cybersecurity Risk Management Certification (CSRMC) framework defines eight modernization elements. The Program Analyst enforces these across the CPMAI lifecycle.

Element	Purpose	Lifecycle Touchpoints
Mission Risk Profile (MRP)	Mission-aligned risk evaluation	Phase I, updates II–VI
Critical Controls Identification	Prioritized security/resilience controls	Phase I–II
Telemetry Configuration	Real-time monitoring strategy	Phase II–III, VI
Reciprocity & Inheritance Register	Reuse of validated controls across systems	Phase II, updates III–VI
Cyber Resilience Posture Report (CRPR)	Resilience documentation and assessment	Phase IV–V, updates VI
Automated Evidence Package (AEP)	Audit-ready evidence bundle	Phase III–VI
Continuous Compliance Validation (CCV)	Automated compliance checks	Phase V–VI
Automated Control Validation Ruleset (ACVR)	Machine-readable CCV rules	Phase IV–V

Evidence Repository Structure

Each project maintains a structured governance evidence repository under projects/[name]/governance/:

Phase_Gates/ — Contains explicit go/no-go decision points for each CPMAI phase (Gate1 through Gate6).

Cross_Cutting/ — Contains standards that apply across the entire project: Risk_Register/, SoA/, CSRMC/, Evidence_Index/, and Governance_Cadence/.

Phase_I/ through Phase_VI/ — Phase-specific evidence artifacts produced by agents during execution.

Management_Reviews/ — Quarterly executive review documentation.

Incident_Response/ — Incident records and post-incident analysis.

Appendices/ — Supporting materials and reference documents.

Evidence Categories

Category	Description
Governance & Policy Evidence	Policies, governance structure, ethical frameworks, leadership oversight documentation
Risk & Security Evidence	Risk registers, threat assessments, resilience documentation, security controls, MRP
Data Governance Evidence	Data quality, lineage, privacy compliance, profiling results, pipeline integrity
Model Development Evidence	Experiments, performance metrics, explainability, bias assessments, robustness evaluations
Operational & Monitoring Evidence	Telemetry, CCV reports, incident response records, operational performance
Gate Approvals & Decision Records	Governance decisions, risk acceptance, deviations, executive approvals

Governance Cadence

Level	Frequency	Focus
Operational Review	Weekly	Telemetry, drift, anomalies, issue escalation
Governance Review	Bi-Weekly / Monthly	Risk register, SoA updates, documentation, CCV prep
Executive Review	Quarterly	Risk acceptance, strategic alignment, performance
Audit & Compliance	Annual / Triggered	Internal audits, external assessments, certification prep

AI Risk Taxonomy

All risks within the framework are classified using seven domains:

Domain	Description	Examples
Technical	Model behavior, performance, reliability	Model drift, data drift, hallucinations, overfitting
Ethical	Fairness, transparency, responsible outcomes	Bias, unfair impact, lack of explainability
Operational	Business continuity, operational performance	System downtime, dependency failures
Cybersecurity	Adversarial threats, security vulnerabilities	Adversarial attacks, poisoning, model extraction
Privacy	Data protection, privacy violations	Leakage, reidentification, improper handling
Regulatory	Compliance with laws, standards, policies	Audit failures, contractual violations
Mission-Driven	Mission success, critical organizational functions	Mission degradation, performance shortfall

Escalation Model

Routine project decisions → Program Manager / Scrum Master

Governance decisions, deviations, exceptions, material changes → Program Analyst → AI Governance Lead

Elevated or mission-impacting risk acceptance → Director (Human Director / Executive Sponsor)

Quality Standards and Memory System

Code Quality

- All code must include inline comments explaining the why, not just the what
- Follow existing patterns in the codebase
- Write tests for new functionality
- Update documentation when changing behavior

Documentation Quality

- Assume the reader has context from other agents' work
- Link to related artifacts (requirements, architecture docs, etc.)
- Use diagrams when explaining complex interactions
- Keep language clear and jargon-free unless technically necessary

Handoff Quality

- Output must be immediately usable by the next agent
- Never assume knowledge that only exists in your session
- If you make a decision, document why
- If you encounter ambiguity, flag it explicitly

Memory System

The memory system operates at two levels:

Project-Specific Memory

Located at projects/[name]/memory/. Contains daily activity logs (YYYY-MM-DD.md) where agents log work, decisions, and handoffs. Also contains MEMORY.md for long-term project learnings and decisions that should persist and influence future work.

Cross-Project Memory

Located at memory/ in the repository root. Contains MEMORY.md for patterns, standards, and decisions that apply across all projects. Captures organizational knowledge that transcends individual projects.

Continuous Learning

When agents encounter something that should be remembered long-term:

- **Project-specific learning:** Add to projects/[name]/memory/MEMORY.md
- **Cross-project patterns:** Also update memory/MEMORY.md at the repo root
- **Coordination changes:** Update CLAUDE.md or GEMINI.md
- **Role-specific learning:** Update the agent's SOUL file in .agent/souls/

Anti-Patterns to Avoid

The framework defines clear anti-patterns that all operators and agents must avoid:

Agent Work Anti-Patterns

- Do not work outside your specialization without coordinating with the team
- Do not make architectural decisions if you are not the Architecture SE
- Do not modify the database schema if you are not the Database Engineer
- Do not start work before checking if upstream dependencies are complete
- Do not complete work without documenting it in the shared memory
- Do not hand off work without completing the self-annealing Verify phase
- Do not pass flawed work forward with a note — either fix it or block the handoff
- Do not retry a failed operation without changing something between attempts

Self-Annealing Anti-Patterns

- Never skip the self-review because you are confident. Confidence is not verification.
- Never document a learning and then ignore it. If you wrote a prevention, follow it.
- Never correct errors silently. Other agents may have built on top of flawed output.
- Never let the circuit breaker become a way to defer hard problems.

Reporting Anti-Patterns

- Do not let agents bypass the Scrum Master for direct status updates to the Director
- Do not let reporting become a bottleneck — the Scrum Master reports asynchronously
- Do not skip sprint start or sprint completion briefings
- Do not approve by silence — explicit response is required at approval gates

Operational Anti-Patterns

- Do not start work without clear requirements from the Requirements BA
- Do not let work in progress pile up without completing tasks
- Do not skip handoff documentation — sloppy handoffs create triple rework downstream
- Do not ignore blockers — they compound over time
- Do not confuse shared templates (directives/) with project instances (projects/[name]/governance/)

Measuring Success

The framework's effectiveness is measured through operational indicators:

Key Performance Indicators

- **Cycle Time:** Work flows from requirements to production faster as coordination matures
- **Handoff Rate:** Agents rarely need to send work back for clarification
- **Blocker Rate:** Coordination patterns reduce blockages over time
- **Quality (Self-Catch Rate):** Producing agents catch >80% of errors before downstream discovery
- **Director Leverage:** The Human Director spends time on strategy, not tactical execution

Maturity Indicators

The system is operating at target maturity when the Human Director functions as a strategic orchestrator rather than a tactical doer, agent handoffs are consistently complete and require no clarification, the circuit breaker fires less frequently over time, annealing records show decreasing error recurrence, and the PORTFOLIO.md demonstrates a growing portfolio of governance-compliant delivered projects.

Appendix A: Daily Quick Reference

Essential File Locations

What	Where
Shared agent context	CLAUDE.md (or GEMINI.md)
Agent identity files	.agent/souls/[agent-name].md
Self-annealing protocol	directives/self-annealing-protocol.md
Human reporting protocol	directives/human-reporting-protocol.md
Director interview protocol	directives/director-interview-protocol.md
AI governance framework	directives/ai-governance-framework.md
Branding guide	directives/branding-guide.md
Template index	directives/templates/TEMPLATE-INDEX.md
Project creation script	./new-project.sh
Portfolio of all projects	PORTFOLIO.md
Project overview	projects/[name]/PROJECT.md
Today’s coordination log	projects/[name]/memory/YYYY-MM-DD.md
Long-term project knowledge	projects/[name]/memory/MEMORY.md
Task board	projects/[name]/orchestration/tasks.md
Governance docs	projects/[name]/governance/

Git Workflow

After each agent session, commit with the format:

Session: [PROJECT] [AGENT] — [what was accomplished]

Examples: Session: dashboard-app Requirements BA — Completed stakeholder interviews. Session: analytics-service Database Engineer — Created schema v1.

Troubleshooting

Agent forgets its role mid-conversation: Reminder: You are the [AGENT NAME]. Check your SOUL file at .agent/souls/[agent].md

Cannot find handoff from previous agent: Check the daily memory file for handoff entries. Search for the word Handoff.

Files in wrong location: Move to the correct execution/ subdirectory and update references in the memory file.

Appendix B: CSRMC Element Coverage

CSRMC Element	Template(s)	Phase(s)
Mission Risk Profile (MRP)	mission-risk-profile.md	I, updates II–VI
Critical Controls Identification	statement-of-applicability.md, standards-crosswalk-matrix.md	I–II
Telemetry Configuration	telemetry-configuration.md, operational-monitoring-plan.md	II–III, VI
Reciprocity & Inheritance Register	reciprocity-inheritance-register.md	II, updates III–VI
Cyber Resilience Posture Report	cyber-resilience-posture-report.md	IV–V, updates VI
Automated Evidence Package (AEP)	automated-evidence-package.md, evidence-index.md	III–VI
Continuous Compliance Validation	ccv-report.md	V–VI
Automated Control Validation Ruleset	automated-control-validation-ruleset.md	IV–V

Appendix C: Standards Coverage Summary

Standard	Templates Addressing
CPMAI v7	All 22 (lifecycle structure)
ISO/IEC 42001	20 of 22 templates
NIST AI RMF 1.0	18 of 22 templates
NIST SP 800-53 Rev 5	14 of 22 templates
DoD CSRMC	10 of 22 templates (all 8 elements covered)
NIST SP 1270	3 templates (bias/fairness focused)
NIST AI 100-1	2 templates (security focused)
OMB M-24-10	2 templates (governance focused)

— End of Document —