

The IRGF Implementation Handbook

A Practitioner's Guide to the Intent-to-Runtime Governance
Framework, Version 1.0 — for Enterprise Architecture and
Governance Teams

Dr. Rajan Kumar Upadhyay
August 2026

How to Use This Handbook

This handbook serves as the operational companion to "An Enterprise Architecture Framework for the Governance of AI Build and Digital Transformation," the research report that designed and stress-tested IRGF. The report addresses whether the framework is defensible. This document answers a different question: what actions should you take on Monday morning? The two documents divide the work. While the report justifies a mechanism, the handbook explains how to implement it, what evidence to collect, who signs off, what typically goes wrong, and how to determine if it is working. When the report notes a limitation, the handbook advises on how to work around it. Nothing here overrides the specification in Chapter 14 of the report; if the two seem to disagree, the specification prevails, and the discrepancy is a defect in this handbook.

Who this is written for

You are an enterprise architect, AI architect, risk and compliance lead, or governance practitioner in a large organization. You have a standing architecture function, some form of architecture review board, and identifiable risk and security partners. Enterprise architecture vocabulary is assumed throughout: domains, capabilities, reference patterns, decision records, and portfolio governance are used without definition.

The handbook does not explain what enterprise architecture is, why AI governance matters, or how large language models work. It assumes you have already decided to adopt IRGF, or are seriously evaluating it, and need to know what adoption involves.

What is in each part

Part A — Orientation establishes what IRGF governs, what it deliberately does not, and how its eight-entity Core Model maps onto records you probably already keep. It closes with a readiness assessment you can run in a week.

Part B — The AI Build Lifecycle is the operational core. Eleven stages and five gates, one chapter per cluster, each with entry conditions, evidence requirements, decision authority, worked examples, and the failure modes that recur. Risk classification receives the longest treatment because every other tier-scaled decision depends on it.

Part C — Agents and Autonomy covers the governance chain for systems that act, not advise, and the specific problem of detecting autonomy increases that were never authorized.

Part D — Continuous Assurance covers standing up the Control Plane: which telemetry to connect first, how to triage each drift category, and where policy-as-code helps versus where it creates noise.

Part E — The Operating Model covers the two governance bodies, decision rights, escalation, and exception handling.

Part F — The Transformation Interface covers what architecture owes portfolio governance and what it must not try to own.

Part G — Artifacts provides the full record set with filled examples rather than blank templates.

Part H — Adoption covers maturity assessment, the phased roadmap, the Minimum Viable Framework for constrained starts, recurring anti-patterns, and how to measure whether governance is working.

Part I — Reference holds the pattern library, gate checklists, scoring worksheets, and the glossary.

Reading paths

Few practitioners will read this front to back. Three shorter paths cover most needs.

Table 1. Reading paths

If you are	Read	Then
Deciding whether to adopt IRGF	Part A, then Chapter 3 (readiness) and Chapter 28 (failure modes)	Chapter 27 (Minimum Viable Framework) to see the smallest credible start
Standing up governance from scratch	Part A, Chapter 25 (maturity), Chapter 26 (roadmap), Part E	Part B in sequence as each gate goes live
Running an AI system through governance today	Part B, in stage order	Part G for the records you must produce at each gate
Building the Control Plane	Part D	Chapter 18 for policy-as-code rules worth writing first

A status warning you should not skip

IRGF has not been deployed in a real organization. Its mechanisms were designed against a structured requirements set and then stress-tested through simulated adversarial review, not through field use. No organization has been observed running these gates, scoring these dimensions, or progressing through the maturity levels described in Chapter 26.

This matters for how you should read the specific numbers in this handbook. Thresholds, cadences, timeboxes, and scoring anchors are reasoned defaults, not calibrated ones. The 90-day exception ceiling, the five-business-day Tier 3 alert window, the four-tier granularity, and the eight-entity Core Model are all defensible design choices with no empirical calibration behind them. Treat each as a starting position to be adjusted against your own incident data, and expect to adjust several of them within the first year.

Three known weaknesses deserve particular attention before you commit, and each is addressed practically in the chapter noted.

Table 2. A status warning you should not skip

Known weakness	Why it matters operationally	Where addressed
Risk-tier self-scoring is gameable in the mid-range	A team under delivery pressure can score D1–D5 to land just below a gate threshold. The safety-override floor catches extremes, not judgment calls	Chapter 7, §7.6–7.8
Gate consolidation reduced labels more than burden	Five gates replaced eleven touchpoints, but the evidence required did not shrink proportionally. Do not promise a lighter process	Chapter 5, §5.4
The two-body model does not scale on volume	ARB and the AI Governance Body become bottlenecks as Tier 3–4 volume grows within one jurisdiction	Chapter 20, §20.6

Presenting an unvalidated framework in a confident operational register creates a specific risk: readers mistake specificity for evidence. Where this handbook gives a precise number, that precision reflects the need to give you something usable, not the existence of data behind it.

Conventions used throughout

Identifiers follow the research report. **S0–S10** are lifecycle stages, **G1–G5** are governance gates, **T0–T7** are transformation lifecycle stages, **D1–D5** are risk dimensions, **FR-*nn*** are the formal design requirements from Chapter 3 of the report, and **D-*nn*** are design decisions recorded in the report's consolidated design log. Where this handbook recommends something the specification does not require, it is marked **[Practice recommendation]** so you can tell the framework's requirements from this handbook's advice.

Worked examples use a deliberately generic organization. No industry framing is used, because sector-specific detail tends to obscure which parts of a mechanism are general and which are contingent. Where a sector materially changes how a mechanism should run, that is called out as a variation rather than built into the example.

Contents

How to Use This Handbook.....	2
List of Figures.....	7
List of Tables	8
Chapter 1. What IRGF Governs.....	11
Chapter 2. The Core Model in Practice.....	14
Chapter 3. Readiness.....	17
Chapter 4. Running the AI Build Lifecycle	19
Chapter 5. Opportunity and Qualification: S0, S1, and Gate G1	26
Chapter 6. Risk Classification: The Five Dimensions	30
Chapter 7. Risk Classification: Tiering, Calibration, and Gaming	35
Chapter 8. Architecture Decision: S3	41
Chapter 9. Data Readiness and Gate G2: S4.....	44
Chapter 10. Build, Configure, Acquire: S5.....	47
Chapter 11. AI Assurance: S6.....	49
Chapter 12. Deployment Authorization: S7 and Gate G3	51
Chapter 13. Runtime Operation and Monitoring: S8	53
Chapter 14. Change: S9 and Gate G4	55
Chapter 15. Retirement: S10 and Gate G5.....	58
Chapter 16. Governing Agents.....	60
Chapter 17. Autonomy Change and Its Detection	64
Chapter 18. Standing Up the Control Plane.....	66
Chapter 19. Drift Triage.....	71
Chapter 20. The Governance Bodies	74
Chapter 21. Roles, Decision Rights, and RACI.....	77
Chapter 22. Escalation and Exceptions.....	80
Chapter 23. The Transformation Interface.....	82
Chapter 24. The Artifact Set	85
Chapter 25. Maturity Assessment	89
Chapter 26. The Adoption Roadmap	91
Chapter 27. The Minimum Viable Framework.....	93
Chapter 28. Failure Modes and Anti-Patterns.....	95
Chapter 29. Measuring Whether It Works	97
Chapter 30. Operating Rhythm	99
Chapter 31. Introducing IRGF to Delivery Teams	101
Chapter 32. End-to-End Walkthrough	103
Chapter 33. Module Reference: Foundations, Classification, Lifecycles	110
Chapter 34. Module Reference: Autonomy, Assurance, Operating Model, Evidence	118
Chapter 35. The Reference Pattern Library	126

Chapter 36. Checklists and Job Aids	129
Chapter 37. Glossary.....	132
Appendix A. Known Limitations Register	134
Appendix B. Departures From the Specification	135

DO NOT COPY

List of Figures

Figure 1. The IRGF lifecycle wheel: eleven stages, five gates, and Risk and Governance at the hub	20
Figure 2. The build phase: S0 to G3, with decision authority by tier	21
Figure 3. The operate phase: S7 to G5, with the loop-backs that make the lifecycle non-linear. 22	
Figure 4. Risk classification: from five dimension scores to a countersigned tier.....	36
Figure 5. The agent governance chain: eight links, each a control point.....	61
Figure 6. The Control Plane loop, with the automation boundary that constrains Enforce	67
Figure 7. The IRGF module map: twenty modules in seven groups, and what each depends on	111

List of Tables

Table 1. Reading paths	3
Table 2. A status warning you should not skip	3
Table 3. What IRGF is not	11
Table 4. The boundary that matters most: architecture versus engineering.....	12
Table 5. The seventeen questions.....	13
Table 6. The eight entities	14
Table 7. Mapping to what you already have	15
Table 8. The one-week readiness assessment	17
Table 9. The shape of the lifecycle.....	19
Table 10. Where the gates sit and why there are five.....	23
Table 11. Intensity scales, procedure does not.....	23
Table 12. Governance weight by build type.....	24
Table 13. The six-way sourcing question.....	26
Table 14. Gate G1 — Qualification and Provisional Risk	28
Table 15. Common G1 failures	28
Table 16. Why five dimensions.....	30
Table 17. D1 — Decision Consequence Severity	30
Table 18. D2 — Autonomy.....	31
Table 19. D3 — Reversibility Deficit	32
Table 20. D4 — Exposure and Scale	32
Table 21. D5 — Data Sensitivity and Model Uncertainty	33
Table 22. Scoring in practice: who, when, how long	33
Table 23. Two axes, not a sum.....	35
Table 24. Re-scoring: when the tier must be revisited	38
Table 25. Tier semantics: what each tier actually buys.....	38
Table 26. Recognizing and countering under-tiering.....	40
Table 27. What S3 decides	41
Table 28. The sourcing decision framework	41
Table 29. Writing an ADR that survives contact with the Control Plane	42
Table 30. Gate G2 — Architecture and Data Readiness.....	44
Table 31. Common G2 failures	46
Table 32. Where drift begins.....	47
Table 33. Assurance is not one thing	49
Table 34. Assurance scope by build type	50
Table 35. What G3 authorizes.....	51
Table 36. What must be observable	53
Table 37. Monitoring intensity by tier.....	53
Table 38. The classification principle	55
Table 39. The change type table.....	55
Table 40. Gate G4 — Material Change.....	56
Table 41. What S10 must produce	58

Table 42. Gate G5 — Retirement.....	59
Table 43. The governance chain.....	61
Table 44. The Agent Card	62
Table 45. What counts as boundary-widening	64
Table 46. What the Control Plane is and is not	66
Table 47. Telemetry sources and connection order.....	68
Table 48. Suggested connection order	68
Table 49. The automation boundary.....	69
Table 50. Policy-as-code: the rules worth writing first	69
Table 51. Evidence-as-code	70
Table 52. The eight categories	71
Table 53. Triage: severity and routing	72
Table 54. From drift to decision.....	73
Table 55. Why two bodies and not six	74
Table 56. The role catalogue	77
Table 57. Decision rights by tier	77
Table 58. The AI Build Lifecycle RACI.....	78
Table 59. The escalation model.....	80
Table 60. Shallow by design	82
Table 61. The transformation lifecycle	82
Table 62. The Architecture Input Package.....	83
Table 63. Benefits and the value scorecard.....	84
Table 64. The primary records	85
Table 65. Risk Classification Record.....	86
Table 66. Architecture Decision Record, AI-extended	87
Table 67. Architecture Exception.....	87
Table 68. AI System Card, derived view	88
Table 69. Tier-scaled documentation depth	88
Table 70. The levels	89
Table 71. A self-assessment instrument.....	89
Table 72. The phases.....	91
Table 73. Months 0–3 in detail.....	91
Table 74. What tends to go wrong, by phase	92
Table 75. What the MVF gives up, honestly.....	93
Table 76. Instrument from day one	97
Table 77. A working calendar	99
Table 78. Architecture Review Board, standing agenda	100
Table 79. AI Governance Body, standing agenda.....	100
Table 80. Signs adoption is going well	102
Table 81. S1 — Use-Case Qualification	103
Table 82. S2 — Risk Classification and countersignature	104
Table 83. S4 — Data Readiness.....	105
Table 84. S6 — Assurance	106

Table 85. S9 and Gate G4 — the autonomy change	108
Table 86. S10 and Gate G5 — retirement, two years on.....	108
Table 87. How to read a module entry	110
Table 88. Module map at a glance	111
Table 89. M1 — Core Model	112
Table 90. M2 — Architecture Domains.....	113
Table 91. M3 — Three Views.....	113
Table 92. M4 — Risk Dimensions.....	114
Table 93. M5 — Tiering Method	114
Table 94. M6 — Independent Verification	114
Table 95. M7 — AI Build Lifecycle	115
Table 96. M8 — Transformation Lifecycle	116
Table 97. M9 — Architecture Lifecycle	116
Table 98. M10 — Agent Governance Chain.....	118
Table 99. M11 — Autonomy Change Control.....	118
Table 100. M12 — Control Plane	119
Table 101. M13 — Drift Taxonomy	120
Table 102. M14 — Policy- and Evidence-as-Code.....	120
Table 103. M15 — Two Standing Bodies.....	121
Table 104. M16 — Decision Rights and RACI	122
Table 105. M17 — Escalation and Exceptions	122
Table 106. M18 — Primary Records	123
Table 107. M19 — Derived Views	123
Table 108. M20 — Reference Pattern Library.....	124
Table 109. Module dependency summary	124
Table 110. Risk scoring worksheet	130
Table 111. Drift triage card	131
Table 112. Change classification quick reference.....	131
Table 113. Known Limitations Register	134
Table 114. Departures From the Specification.....	135

Chapter 1. What IRGF Governs

Scope, boundaries, and the specific claim IRGF makes. Read this before anything else, because most adoption failures trace back to a team applying the framework to work it was never designed to govern.

1.1 The one-sentence version

IRGF makes a single AI system traceable from the enterprise intent that justified it, through the architecture that realizes it, the risk tier that governs it, and the authorization that released it, to the runtime evidence that it still behaves as approved. That chain, end to end, without switching frameworks partway, is the whole proposition.

Everything else in this handbook exists to make one link of that chain operable.

1.2 What IRGF is not

IRGF is a complementary governance framework. It is not a replacement for TOGAF, not a next-generation enterprise architecture method, and not a new paradigm. It introduces no AI risk taxonomy, no AI management system standard, no general IT governance model, and no security control catalog. Each of those already exists in mature form, and IRGF defers to them.

This deferral is operationally significant, not a disclaimer. If your organization already runs ISO/IEC 42001, IRGF does not ask you to restate its management-system requirements. If you run COBIT, IRGF operates inside that structure, not beside it. If you have a DAMA-aligned data governance function, IRGF extends it at exactly one seam — the lineage between a governed data source and an AI system it grounds — and leaves the rest alone.

Table 3. What IRGF is not

Domain	What IRGF does	What it defers to
AI risk taxonomy	Provides D1–D5 as an operational scoring schema that keys architecture gates	NIST AI RMF, ISO/IEC 23894 for the underlying risk methodology
AI management system	References conformance evidence	ISO/IEC 42001
General IT governance	Operates as the architecture and AI arm of existing governance	COBIT, ISO/IEC 38500
Security controls	Names where AI-specific security concerns attach	ISO/IEC 27001, NIST CSF
Data governance	Extends lineage to AI grounding relationships only	DAMA-DMBOK2
Transformation and change management	Supplies structured input to portfolio decisions	Your PMO and change function
Model engineering	Governs a model's existence, sourcing, and authority boundary	ML engineering owns training, tuning, and evaluation method

The last two rows cause the most trouble in practice. Teams adopting IRGF frequently try to pull transformation delivery and model engineering into its scope, because both feel adjacent and both have visible governance gaps. Resisting that pull is a condition of the framework working. A governance framework that absorbs everything nearby stops being architecture governance and becomes an unbounded review function, which is the failure mode the design was built to avoid.

1.3 The boundary that matters most: architecture versus engineering

The single most useful line to internalize is where AI Architecture stops.

AI Architecture governs the existence, connection, and authority boundary of a component. It does not govern the component's internal construction.

This is the same line TOGAF already draws between enterprise and solution architecture. AI is not exempt from it because the technology is newer.

Table 4. The boundary that matters most: architecture versus engineering

Component	Governed by IRGF	Left to engineering
Foundation models	Which model families the enterprise authorizes; concentration risk across vendors	Model weights, training methodology
Hosting	Deployment pattern (vendor API, private hosting, on-premises) and its sovereignty and continuity implications	Infrastructure sizing, autoscaling
Model gateway	That a gateway exists, which models it may route to, what it logs	Gateway implementation, routing algorithm
Retrieval and grounding	That a grounding relationship exists between a system and a governed source	Chunking strategy, embedding model, retrieval tuning
Vector stores	That one exists, what governed data enters it, its retention and classification policy	Product choice, index type, dimensionality
Agents	That an agent exists, its authorized tools, action boundaries, escalation thresholds	Internal planning and reasoning implementation
Multi-agent composition	Which agents may invoke which others, and the authority each delegation carries	Orchestration framework choice
Guardrails	Which guardrail categories a risk tier requires	The classifier or filter implementing one
Evaluation	That evaluation gates exist and bind to deployment	Benchmark suite composition, evaluation datasets
AI identity	That every agent holds a registered machine identity	IAM platform configuration
Human oversight	That an oversight point exists and where it sits in the flow	The interface implementing it

A practical test when you are unsure: **could a reasonably informed architecture review board member evaluate this without becoming a machine learning engineer?** If yes, it is probably in scope. If evaluating it requires reading training code, it is not.

1.4 What “governance” means here

IRGF’s governance is not primarily a review activity. Reviews exist, but they are the smallest part of the design. Three mechanisms carry most of the weight.

Classification. A risk tier is assigned early and re-assigned whenever a change plausibly moves one of the five risk dimensions. Almost every other decision in the framework keys off that tier. Getting classification right matters far more than getting any individual review right.

Federation with verification. Tier 1 and Tier 2 work is self-service by default. Product and solution teams make their own architecture and deployment decisions inside pre-approved patterns and policy guardrails. This is only safe because the Control Plane checks their self-certifications continuously against the same rules a reviewer would apply. Federation without that verification is delegation without accountability, and the framework does not support it.

Continuous comparison. The framework’s distinguishing claim is that deployed state is compared against approved state on an ongoing basis, not at review checkpoints. The comparison target is specific: not a generic security baseline but a named Decision record made by an accountable owner under a stated risk tier.

If you adopt only the review gates and skip classification discipline and continuous comparison, you will have added ceremony without adding assurance. That is the most common way to implement IRGF badly.

1.5 The seventeen questions

IRGF exists to let an enterprise answer seventeen questions about any AI system, continuously, from one connected structure. The list is worth keeping visible, because it is the acceptance test for whether your implementation is working. If you cannot answer one of these for a production Tier 3 system, you have found a gap.

Table 5. The seventeen questions

#	Question	Where the answer lives
1	Why are we using this AI?	Use-Case Canvas, linked Capability Record
2	Which capability does it serve?	Capability Record
3	Who owns it?	AI System Owner field, Deployment Authorization Record
4	What data does it use?	Data Lineage and Sensitivity Record
5	Which model or service does it depend on?	Architecture Decision Record
6	What decisions can it make?	Risk Classification (D1), Agent Card
7	What actions can it perform?	Agent Card authorized tool and action list
8	How autonomous is it?	Risk Classification (D2), Agent Card boundary
9	What risk does it carry?	Risk Classification Record, current tier
10	Which policies and regulations apply?	Control Matrix, Regulatory Overlay Reference
11	What controls mitigate that risk?	Control Matrix
12	Who approved it?	Deployment Authorization Record
13	Is deployed architecture still compliant with approved?	Control Plane drift signals
14	Is it behaving as expected?	AI Assurance Summary, behavioral drift signals
15	Has its risk changed?	Risk Classification history, change records
16	What value is it generating?	Benefit Record, Business Outcome Assurance
17	Should it change, be restricted, or retire?	Change Record, retirement trigger review

Notice that no single artifact answers more than two questions, and that the AI System Card — the one-page summary practitioners most want — is a compiled view over the others instead of a record anyone maintains by hand. That is deliberate, and Chapter 24 explains why duplicating these fields into a hand-maintained summary reliably produces stale governance data.

Chapter 2. The Core Model in Practice

Eight entities, what each means when you have to record it, and how they map onto artifacts you may already keep.

2.1 The eight entities

The Core Model holds the entities without which IRGF’s traceability claims stop working. Everything else in the metamodel — roughly twenty-three further entities including Strategy, Objective, Value Stream, Dataset, Model, Agent, Policy, and Metric — is supporting structure.

Table 6. The eight entities

Entity	Operational meaning	Typically recorded in	Owner
Intent	The enterprise purpose or objective a system serves	Capability Record, Use-Case Canvas	Business Sponsor
Capability	The business capability the system enables or improves	Capability Record	Business Sponsor with Architect
AI System	The governed unit: a deployed system with a boundary, an owner, and a tier	AI System Card (compiled)	AI System Owner
Decision	A recorded, authorized choice with a named accountable human	ADR, Deployment Authorization Record	Architect, governance body
Risk	The current classification attached to the system	Risk Classification Record	Risk and Compliance Lead
Control	A mitigation mapped to a specific risk	Control Matrix	Risk and Compliance Lead
Outcome	The measurable result the system was authorized to produce	Benefit Record	Business Sponsor
Assurance	Evidence that the system still performs and conforms	AI Assurance Summary, drift records	Varies by assurance type

A status caveat carried from the research: this set is asserted rather than derived. No minimality test was performed, and no expert panel has validated that eight is the right number or that these are the right eight. It is the organizing structure because no better tested alternative exists, not because its sufficiency is established.

2.2 Why Decision is the load-bearing entity

Practitioners consistently underestimate Decision. It is worth understanding why the framework treats it as central.

Every tier-scaled control in IRGF ultimately compares something against a Decision. A drift alert is meaningful because it says “actual state diverges from what was decided by a named person under a stated tier.” Without a Decision record, a drift alert degrades into “configuration changed,” which is a technical observation with no governance content and no obvious owner.

This has a direct operational consequence. **If your Decision records are thin, late, or unowned, the Control Plane cannot produce governance signal regardless of how good your telemetry is.** Organizations that invest heavily in telemetry before their decision records are trustworthy generate a large volume of alerts that nobody can act on, and then conclude the Control Plane does not work.

Get Decision records right first. Telemetry second.

2.3 Mapping to what you already have

Most large organizations already hold some version of most of these entities. Adoption is usually less about creating records than about connecting existing ones and closing specific gaps.

Table 7. Mapping to what you already have

Core Model entity	Common existing equivalent	Typical gap to close
Intent	Strategy documents, OKRs	Rarely linked to individual systems
Capability	Capability model, business architecture	Often stale; AI-enablement flag absent
AI System	Application portfolio or CMDB entry	AI systems frequently absent or recorded as generic applications
Decision	Architecture decision records	Exist for major programs, rarely for AI-specific choices; often no risk tier recorded
Risk	Enterprise risk register, model risk inventory	Not connected to architecture records; no tier driving gate behavior
Control	Control library, compliance mappings	Mapped to regulations, seldom to a specific AI system's risk
Outcome	Benefits tracking	Usually stops at program level, not per system
Assurance	Test results, audit evidence	Point-in-time; not continuously refreshed

The pattern is consistent. The entities exist, but they exist in separate systems owned by separate functions, and nothing joins them at the level of an individual AI system. That join is what IRGF asks you to build, and it is why the framework's contribution is described as connective, not novel.

[Practice recommendation] Before designing anything, run a one-week exercise: pick three production AI systems and try to answer the seventeen questions in §1.5 using only records that exist today. The specific questions you cannot answer are your adoption backlog, and they are usually more informative than a maturity assessment.

2.4 The three views

The same metamodel is read three ways. These are views over one model, not three separate models, and the distinction matters because it determines where a fact is authored.

The Structural View answers what exists and how it connects: domains, capabilities, systems, data products, models, agents, and the relationships between them. Architects work here most of the time.

The Lifecycle View answers where something is in its governed progression: which stage, which gate passed, what evidence exists, what happens next. Delivery teams and governance bodies work here.

The Control View answers whether reality matches the record: drift signals, policy evaluation results, assurance status. Operations, risk, and the Control Plane work here.

A fact is authored in exactly one view and referenced from the others. An AI system's risk tier is authored in the Lifecycle View at S2, read by the Structural View as an attribute, and used by the

Control View to set alert severity. Re-entering it anywhere else creates the duplicate-maintenance problem the artifact model exists to prevent.

DO NOT COPY

Chapter 3. Readiness

What must be true before IRGF can work, and how to find out in a week.

3.1 Five preconditions

IRGF assumes several things that are not universally present. Where a precondition is missing, adoption does not fail immediately; it fails about six months in, when a mechanism that depends on it produces nothing usable.

A functioning architecture review body. IRGF expands an existing ARB instead of creating one. If no body currently makes binding architecture decisions, you are standing up architecture governance and AI governance simultaneously, which is a materially larger program than adopting IRGF.

An accountable-owner convention. Every AI system needs exactly one named human owner at all times. Organizations that cannot name an owner for an existing production system will not be able to name one for a new one, and the “no orphaned systems” principle will be violated from the first week.

Some inventory of AI in production. You cannot risk-classify what you cannot enumerate. Most organizations discover during adoption that their AI footprint is substantially larger than recorded, largely because of AI features embedded in purchased software.

A risk function willing to countersign. Independent countersignature of risk scores is the single highest-leverage control in the framework. It requires a risk or compliance partner with capacity to review scores and authority to change them. Without that, self-scoring stands unverified and the framework’s most confirmed weakness goes unaddressed.

Executive tolerance for saying no. Gates that never reject anything are not gates. If the organization’s culture makes rejection or conditional approval unavailable in practice, IRGF becomes documentation overhead.

3.2 The one-week readiness assessment

Run this before committing to a roadmap. It is deliberately small, and the output is a decision about where to start rather than a maturity score.

Table 8. The one-week readiness assessment

Day	Activity	Output
1	Enumerate AI systems in production, including vendor-embedded features. Ask procurement, not just engineering	Draft inventory, deliberately incomplete
2	For three systems spanning low and high consequence, attempt the seventeen questions (§1.5)	Gap list
3	Interview the ARB chair and a risk partner on capacity and appetite	Governance capacity note
4	Sample five recent architecture decisions. Check whether each records a rationale, an owner, and any risk	Decision-record quality baseline

Day	Activity	Output
	view	
5	Identify which telemetry sources already exist and who owns them	Control Plane feasibility note

Two findings from this week usually determine the shape of adoption. The first is how large the unrecorded AI footprint is; a large gap means retroactive classification becomes a program in its own right. The second is decision-record quality; poor records mean Part B's gates will produce evidence nobody trusts, and remediating that comes before any Control Plane work.

3.3 Sizing the effort honestly

The research report is explicit that no cost, staffing, or service-level model exists for IRGF, and this handbook will not invent one. What can be said is structural rather than numeric.

The dominant recurring cost is **risk classification and its countersignature**, because it applies to every new initiative and re-triggers on every material change. The dominant one-off cost is **retroactive classification of existing systems**, which scales with the size of the unrecorded footprint found in §3.2. Control Plane work is usually the largest technical investment but can be deferred and staged; governance body time is the constraint most likely to bind first, and it binds on Tier 3–4 volume specifically.

Any figure you are given for FTE or turnaround time, including by a vendor, is an assertion until your own first two quarters produce data. Instrument the process from day one so that by month six you can replace assumptions with measurements. Chapter 29 sets out which measurements are worth taking.

Chapter 4. Running the AI Build Lifecycle

How the eleven stages and five gates fit together, how intensity scales with tier, and the sequencing rules that are commonly misread.

4.1 The shape of the lifecycle

Eleven stages, numbered S0 to S10. Five gates, G1 to G5, placed at the points where an accountable decision must be recorded, not at every stage boundary.

Table 9. The shape of the lifecycle

Stage	Name	Core question	Gate that follows
S0	Opportunity	Should this problem use AI at all?	—
S1	Use-Case Qualification	Is there value, feasibility, and a data path?	G1
S2	Risk Classification	What tier does this carry?	G1
S3	Architecture Decision	What architecture delivers it, and does it conform to a pattern?	G2
S4	Data Readiness	Is the grounding data governed, traced, and classified?	G2
S5	Build / Configure / Acquire	Construct, integrate, or procure	—
S6	AI Assurance	Does it perform as intended and as safely as its tier requires?	G3
S7	Deployment Authorization	Is it authorized, by whom, on what evidence?	G3
S8	Runtime Operation and Continuous Monitoring	Is it still behaving, compliant, and valuable?	—
S9	Change	Has something changed enough to re-enter an earlier stage?	G4
S10	Retirement	Has it been decommissioned with evidence preserved?	G5

IRGF — the Intent-to-Run Governance Lifecycle

Eleven governed stages, five decision gates, and an outer band in which the Control Plane compares deployed state against approved state without pause

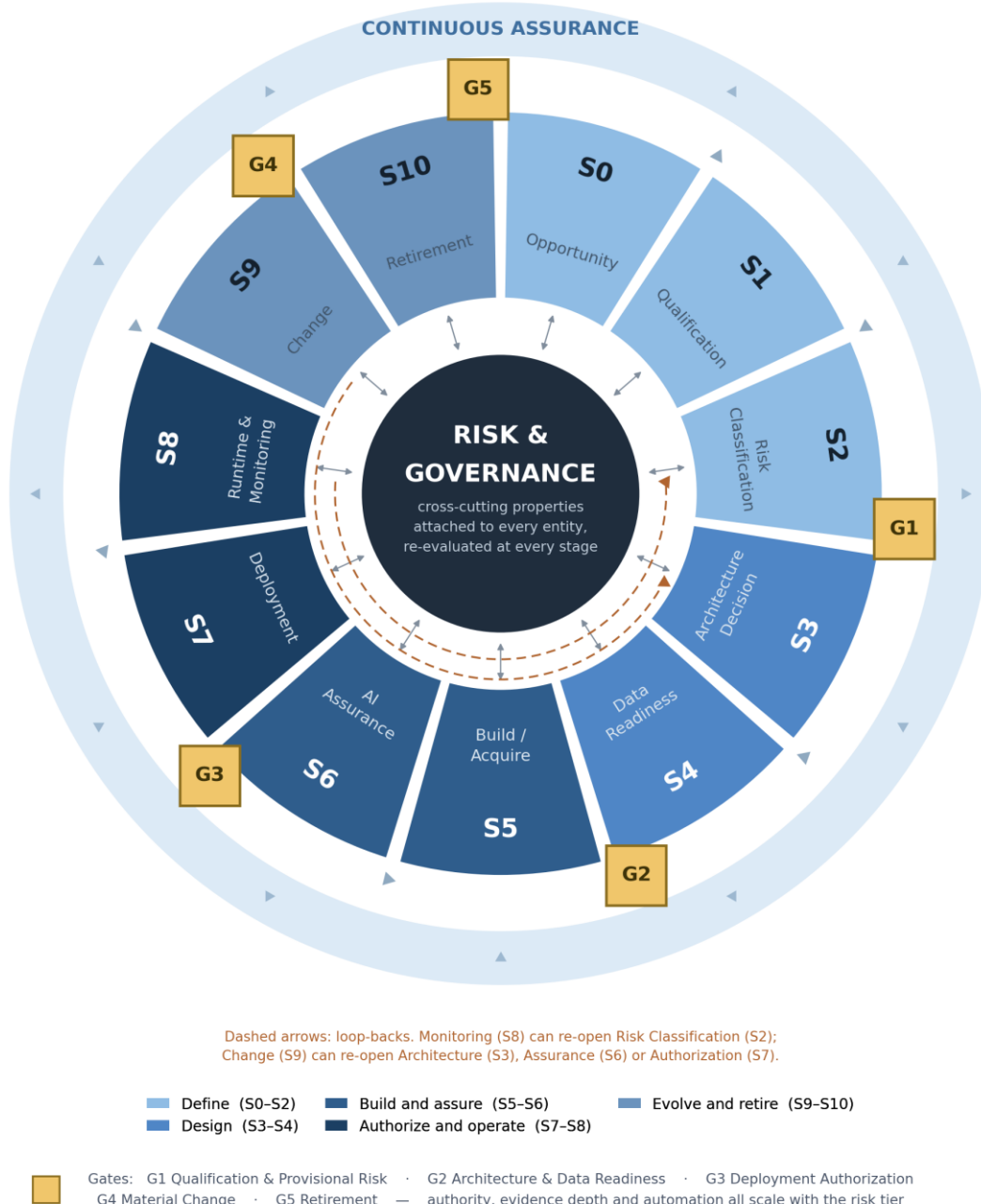


Figure 1. The IRGF lifecycle wheel: eleven stages, five gates, and Risk and Governance at the hub

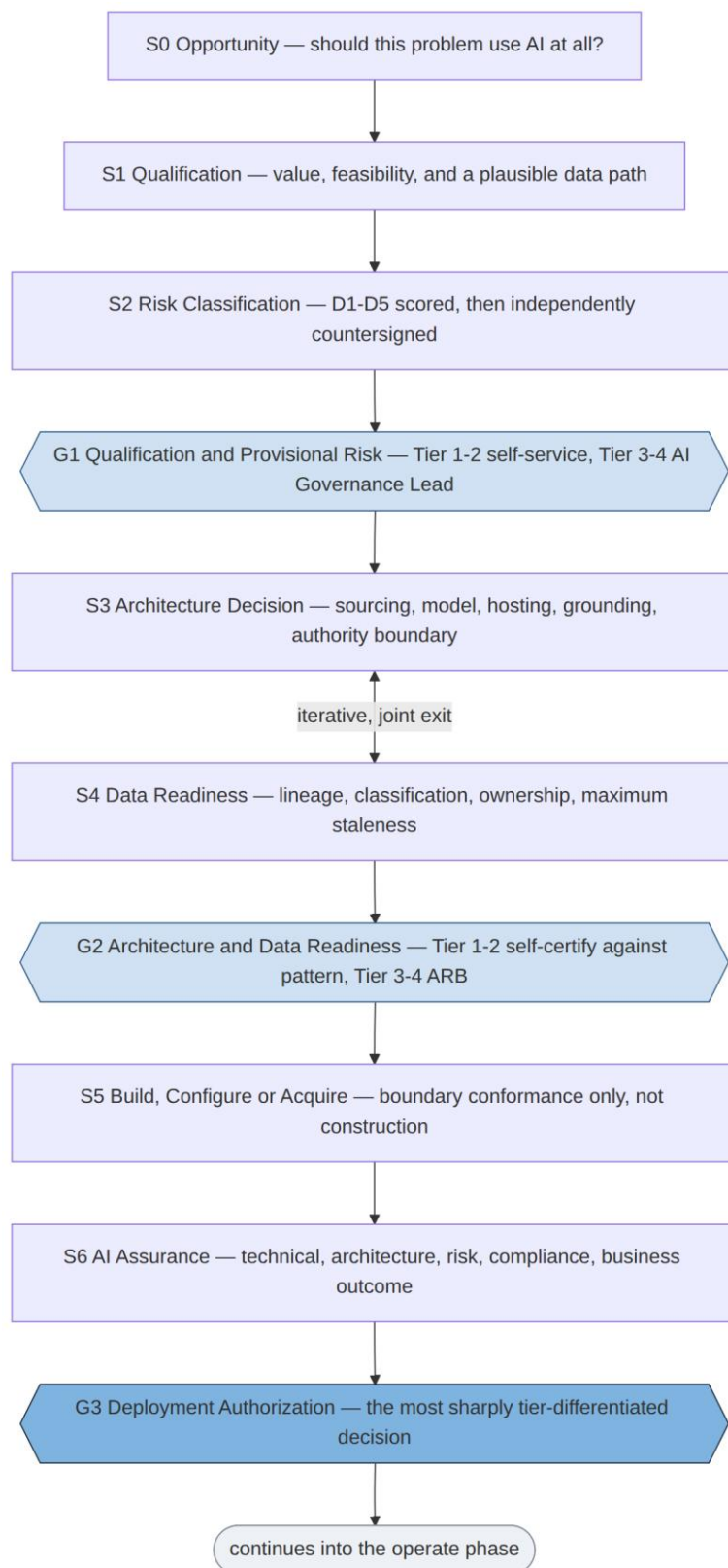


Figure 2. The build phase: S0 to G3, with decision authority by tier

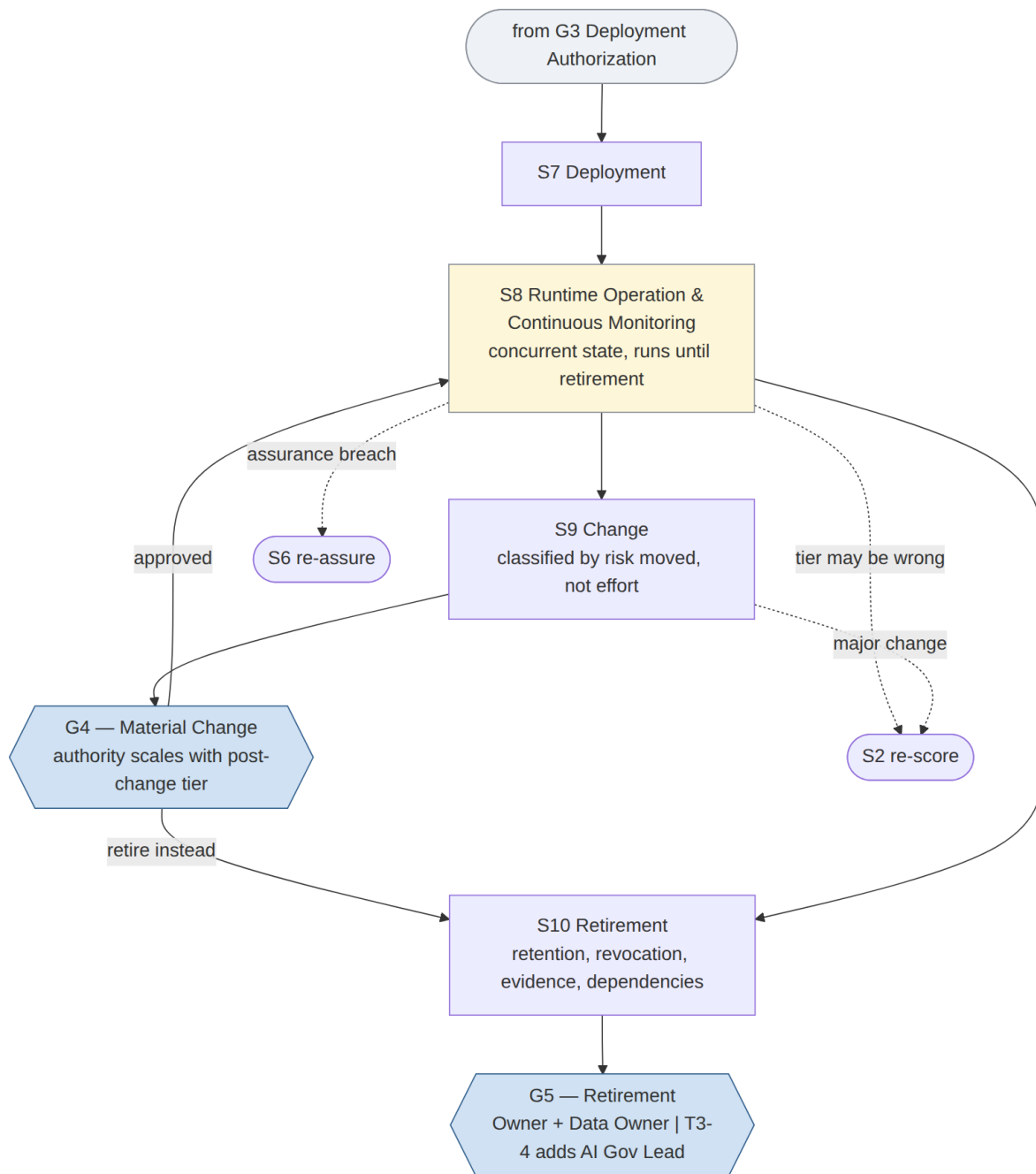


Figure 3. The operate phase: S7 to G5, with the loop-backs that make the lifecycle non-linear

Three features of this structure are easy to miss and cause most sequencing disputes.

S8 is a concurrent state, not a stage you pass through. Runtime operation and monitoring begin at deployment and continue until retirement. They are deliberately not modeled as two sequential stages, because placing monitoring after operation reintroduces the periodic assumption the framework exists to remove.

S2 is not visited once. Risk classification has one identifiable first occurrence, but it re-triggers whenever a change plausibly moves a risk dimension, and whenever runtime evidence suggests the original tier was wrong. Risk travels with the system.

Loop-backs are structural, not exceptional. S8 can re-open S2, S6, or S9. S9 routes to S2, S3, S6, or S7 depending on change classification. A lifecycle diagram showing only forward arrows would misrepresent how the framework actually operates.

4.2 Where the gates sit and why there are five

The framework consolidated an illustrative eleven checkpoints into five gates. The consolidation criterion was specific: a gate exists where a distinct accountable decision must be recorded, not wherever a stage ends.

Table 10. Where the gates sit and why there are five

Gate	Sits after	The decision being recorded
G1	S1–S2	Is this use case justified, and what tier does it provisionally carry?
G2	S3–S4	Is the architecture and its data foundation acceptable?
G3	S6	Is this authorized to run in production?
G4	S9	Does this change require re-authorization?
G5	S10	Is decommissioning complete and evidenced?

Gates that were proposed and rejected are as informative as those retained. A separate “AI necessity” gate distinct from strategic alignment was rejected because it does not correspond to a decision anyone records separately; it is an entry condition on G1. A separate data-readiness gate was folded into G2 because architecture and data readiness are assessed against each other, and separating them produces two reviews that each lack half the context.

4.3 Intensity scales, procedure does not

The same five gates apply at every tier. What changes is who decides, how much evidence is required, and how much can be automated.

Table 11. Intensity scales, procedure does not

Gate	Tier 1	Tier 2	Tier 3	Tier 4
G1	Self-service, rule-based auto-approval possible	Self-service with notification	AI Governance Lead review	AI Governance Lead review
G2	Architect self-certifies against unmodified pattern	Architect self-certifies, ARB notified	ARB decision	ARB decision, Enterprise Sponsor briefed
G3	System Owner	System Owner, AI Governance Lead notified	AI Governance Body	AI Governance Body plus Enterprise Sponsor sign-off
G4	Often bypassed (minor changes)	Scaled to post-change tier	AI Governance Body	AI Governance Body plus Enterprise Sponsor
G5	Owner plus Data Owner	Owner plus Data Owner	Adds AI Governance Lead confirmation	Adds AI Governance Body

G3 is the most sharply differentiated gate in the framework. At Tier 1 it is a single owner clicking approve against automated evidence checks. At Tier 4 it is an executive decision with board-level visibility. If your implementation makes G3 feel similar across tiers, the tiering is not doing its job.

4.4 What the five gates actually cost

This section exists because the framework’s own stress-testing produced an uncomfortable finding, and adopters who are not told about it form the wrong expectation.

Reducing eleven illustrative checkpoints to five gates **reduced the number of labeled decision points more than it reduced the evidence teams must produce**. Independent review characterized the consolidation as largely bundling. The evidence required at G2, for instance, is the union of what an architecture gate and a separate data gate would each have demanded; merging the reviews saved a meeting, not a document.

Two consequences follow for how you introduce IRGF.

Do not market it as lighter governance. Teams told to expect a lighter process and then asked for an ADR, a data lineage record, a pattern conformance result, and a risk score at a single gate will conclude the framework was oversold, and that impression is difficult to reverse.

Look for burden reduction where it is real. Genuine reduction comes from three places, none of which is gate consolidation: pattern-based self-certification at Tier 1–2, evidence-as-code capture of roughly two-thirds of the minimum evidence set, and the tier system keeping most work out of standing bodies entirely. Those are worth promising. Fewer gates is not.

[Practice recommendation] Track evidence-production time per gate from the first month, separated by tier. It is the only way to know whether your implementation is achieving proportionality or simply relabeling a uniform process.

4.5 Governance weight by build type

The eleven stages apply to every AI build type, but the weight of each stage differs sharply. Applying identical rigor to a vendor-embedded feature and a multi-agent system would breach risk proportionality as surely as under-governing the latter.

Table 12. Governance weight by build type

Build type	S3 weight	S4 weight	Distinguishing concern
Traditional ML / predictive	Moderate	High	Training data governance; well covered by existing model-risk practice
External foundation model API	High	Moderate	Vendor concentration; what leaves the enterprise boundary
General-purpose generative AI	High	Moderate	Non-determinism; guardrail configuration
Retrieval-augmented generation	High	High	Grounding lineage is the dominant concern
Fine-tuned models	High	High	Training-data governance; treated as a major change
Copilots (human-paired)	Moderate	Moderate	Placement and usability of the oversight point
Single tool-using agent	High	Moderate–High	Authority and permission boundary
Multi-agent systems	Highest	High	Inter-agent delegation; weakest standards coverage
Vendor-embedded AI features	Low	Low–Moderate	Risk of being governed as a feature toggle

The last row deserves specific attention. Because S3 to S5 are light for embedded vendor AI, there is a standing temptation to skip S2 entirely on the reasoning that it is just a purchased feature. **S2 applies to every build type without exception.** A vendor feature that influences credit decisions carries the risk of a credit decision system regardless of how it was acquired. Chapter 28 treats this as a named anti-pattern because it recurs reliably.

Chapter 5. Opportunity and Qualification: S0, S1, and Gate G1

The cheapest place to stop bad work, and the stage most often skipped.

5.1 S0 — Opportunity

Question: should this problem use AI at all, and if so, what kind?

S0 exists because the most expensive governance failures are systems that should never have been built. It is a short stage, often a single conversation, and it produces one decision: proceed to qualification, or stop.

The stage has a specific discipline attached. AI is never self-justifying. A proposal must name a capability or stakeholder need that the system addresses. “We should use AI for this” is not an opportunity; “claims triage takes eleven days and the capability target is four” is.

The six-way sourcing question

S0 also frames the sourcing decision that S3 will settle. Raising it this early prevents teams arriving at architecture having already committed to an approach for reasons nobody recorded.

Table 13. The six-way sourcing question

Option	Fits when	Governance weight
No AI	A deterministic rule, better process design, or a report solves it	Stops here
Vendor AI SaaS	The capability is undifferentiated and a mature product exists	Procurement-led; due-diligence evidence required at G2
Foundation model API	General capability, no proprietary advantage in the model itself	Vendor and data-flow decisions dominate
Retrieval augmentation	The gap is enterprise knowledge, not model capability	Grounding lineage dominates
Fine-tuning	Domain language or format is genuinely distinctive and data exists	Treated as major change on every subsequent revision
Build a model	The model is a durable competitive differentiator	Rare; highest sustained governance cost

[Practice recommendation] Require the “no AI” option to be explicitly considered and its rejection recorded in one line. Teams that must write down why a rule engine was insufficient produce noticeably better use cases, and a small number abandon the initiative at this point, which is the cheapest possible outcome.

S0 checklist

- A named capability or stakeholder need is stated, with a current and target measure
- A named business sponsor exists
- The “no AI” alternative has been considered and its rejection recorded
- A provisional sourcing direction is identified, understood as non-binding
- No architecture work has begun

5.2 S1 — Use-Case Qualification

Question: is there genuine value, feasibility, and a plausible data path?

S1 converts an opportunity into a qualified candidate. Its output is the AI Use-Case Canvas, which becomes the evidence base for G1.

Four tests apply, and a failure on any one should stop or reshape the initiative.

Value. An expected outcome stated as a measure with a baseline. “Improved customer experience” fails this test. “Reduce first-response time from 6 hours to under 1 hour for tier-2 queries” passes, because it can be measured later against the same definition.

Feasibility. A plausible technical path, not a proven one. The question is whether anything in the proposal is known to be beyond current capability, not whether success is certain.

Data path. Whether the data the system needs exists, is accessible, and is governed. This is the test that most often fails late when skipped here. A retrieval system grounded on a knowledge base nobody maintains will pass every gate and then degrade quietly.

Provisional risk signal. Enough D1–D5 indication to know roughly what tier is coming. Precision is not required at S1; direction is.

Worked example

Use case. Reduce claims-processing cycle time using a generative triage assistant that summarizes submitted claims and proposes a routing category.

Capability. Claims Handling. Baseline maturity 2, target 4. Current mean cycle time 11 days against a 4-day target.

Value. 30% cycle-time reduction on the triage step, measured as mean hours from submission to routing decision, baselined over the preceding quarter.

Feasibility. Summarization and classification against an existing claims corpus. No novel capability required.

Data path. Claims documents exist in a governed repository, classified Restricted. Routing taxonomy is maintained by the claims operations team.

Provisional risk. D1 moderate (a misrouted claim delays but does not deny), D2 low (advisory only, adjuster decides), D3 low (routing is trivially reversible), D4 moderate (all claims, internal users), D5 moderate (restricted personal data).

Sponsor. Claims Operations Director.

This canvas is enough to pass G1 at the tier it implies. Note what it does not contain: no model choice, no vendor, no architecture. Those belong at S3, and including them here invites the gate to approve an architecture nobody has reviewed.

5.3 S2 in brief, and why it sits before the gate

Risk classification runs before G1, not after. The gate needs a provisional tier to know its own decision authority. A Tier 4 candidate must not be waved through by a product owner because the tier had not yet been assigned when the gate ran.

Chapters 6 and 7 cover classification in full. For G1 purposes, provisional scores are acceptable, and the record must be marked provisional so that the S3 re-score is expected, not treated as a correction.

5.4 Gate G1 — Qualification and Provisional Risk

Table 14.

Element	Content
Purpose	Confirm a business justification exists and assign a provisional tier before architecture work begins
Entry criteria	Use-case description, expected value with baseline, D1–D5 inputs at whatever precision is available
Required evidence	AI Use-Case Canvas; provisional Risk Classification Record; named business sponsor
Decision authority	Tier 1–2: capability or product owner, self-service. Tier 3–4: AI Governance Lead review
Outcomes	Approve / Approve with conditions / Remediate / Reject
Automation	High at Tier 1–2; rule-based provisional scoring can pre-clear routine cases

What “approve with conditions” should mean

Conditional approval is the most useful and most abused outcome at G1. Used well, it lets an initiative proceed while a specific, named deficiency is closed by a specific point. Used badly, it becomes a way to approve everything.

A conditional approval is only valid if it names the condition, the owner, and the gate by which it must be satisfied. “Approve with conditions: data path to be confirmed” is not a condition; it is a deferral with no end. “Approve with conditions: Data Owner to confirm claims corpus classification and retention before G2 entry, owner J. Okafor” is a condition.

[Practice recommendation] Cap open conditions per initiative at three. Beyond that, the honest outcome is Remediate, and treating it as conditional approval accumulates governance debt that surfaces at G3 when it is expensive.

Common G1 failures

Table 15.

Failure	What it looks like	Correction
Architecture smuggled in	Canvas names a specific vendor and model	Strip to capability and value; sourcing is provisional until S3
Unmeasurable value	Outcome stated qualitatively	Require a baseline and a unit; refuse to proceed without one
Tier assigned by aspiration	Team scores low to stay self-service	Independent countersignature at S2 (Chapter 7)
Gate skipped for small work	“It’s just a pilot”	Pilots reach production. Apply G1 at proportionate weight, not zero
Sponsor is the delivery team	No business accountability	The sponsor must be able to stop the work and own the outcome

The “just a pilot” failure deserves emphasis. Pilots become production systems more often than they are retired, and they do so without passing any gate they skipped. **[Practice**

recommendation] Treat pilot as a deployment state with a mandatory expiry date, not a governance exemption. At expiry the system either enters full governance or is switched off.

DO NOT COPY

Chapter 6. Risk Classification: The Five Dimensions

S2 in detail. This is the longest treatment in the handbook because nearly every tier-scaled decision in IRGF depends on getting it right, and because scoring is where implementations most often go quietly wrong.

6.1 Why five dimensions

Twelve candidate risk factors were considered during design: autonomy, decision impact, reversibility, data sensitivity, financial impact, safety, legal exposure, stakeholder impact, model uncertainty, external exposure, scale, and systemic impact. Scoring all twelve independently would force assessors to make distinctions that are rarely independent in practice. A decision severe enough to cause major financial harm is very often the same decision carrying major legal exposure.

They were consolidated into five composite dimensions, each scored 1 to 4.

Table 16. Why five dimensions

Dimension	Subsumes	What it asks
D1 Decision Consequence Severity	Financial impact, safety, legal exposure, stakeholder impact	How bad is it if the system is wrong?
D2 Autonomy	Autonomy	How much human judgment sits between output and effect?
D3 Reversibility Deficit	Reversibility	How hard is it to undo a wrong outcome?
D4 Exposure and Scale	External exposure, scale, systemic impact	How many, how far, how connected?
D5 Data Sensitivity and Model Uncertainty	Data sensitivity, model uncertainty, explainability need	How sensitive the inputs, how understood the behavior?

The consolidation is a considered trade of precision for usability. It has not been empirically tested for whether it loses discriminating power in edge cases, and that remains an open question in the research. Where you find a case the five dimensions cannot separate, record it; those cases are exactly what a future recalibration needs.

6.2 D1 — Decision Consequence Severity

The question: if this system produces a wrong output and that output is acted on, how severe is the consequence?

Two clarifications prevent most scoring errors. D1 asks about the consequence of being wrong, not the value of being right. A system that saves considerable money when correct but causes trivial harm when wrong scores low on D1. And D1 assumes the output is acted on; the probability that a human catches the error belongs to D2, not here.

Table 17. D1 — Decision Consequence Severity

Score	Anchor	Illustrative characteristics
1	Negligible	A wrong output wastes a few minutes. No external party affected. No money, safety, or rights implication
2	Moderate	Rework, minor cost, internal frustration. Recoverable

Score	Anchor	Illustrative characteristics
3	Serious	within normal operations. No individual materially harmed Material financial loss, a customer materially disadvantaged, a regulatory reporting concern, or a decision affecting someone's access to a service
4	Severe	Harm to people or safety, denial of a significant right or benefit, major financial or legal exposure, or damage that cannot be made good

The D1 = 4 threshold is consequential. A score of 4 triggers the safety-override floor and forces at least Tier 3 regardless of every other dimension. Assessors under delivery pressure feel that weight and reach for 3. The countersignature control (§7.6) exists largely for this boundary.

A useful test for the 3/4 boundary: **would a reasonable person affected by a wrong output describe the result as a serious harm, or as an inconvenience they could get fixed?** If someone loses access to a service, a benefit, or money they cannot readily recover, it is 4.

6.3 D2 — Autonomy

The question: how much human judgment sits between the system's output and its effect on the world?

Table 18. D2 — Autonomy

Score	Anchor	Characteristics
1	Advisory only	A human makes every decision. The system informs; it cannot act
2	Recommendation with review	The system proposes a specific action; a human approves each one before effect
3	Acts with oversight	The system acts, but a human monitors and can intervene before or shortly after effect. Exceptions route to a human
4	Acts without review	The system acts and the effect occurs without a human in the path

The formal-versus-effective autonomy problem

This is the most important caveat in the entire scoring model, and it is an unresolved weakness instead of a solved one.

A system scored D2 = 1 because “a human makes every decision” may carry far higher effective autonomy if the human approves essentially everything the system proposes. Automation bias is well established: when a system is usually right, review degrades into assent. The framework's stress-testing surfaced this specifically for advisory clinical and advisory financial systems, where the formal score understated real autonomy substantially.

IRGF does not solve this. What it asks you to do is detect it.

[Practice recommendation] For any system scored D2 = 1 or 2 where D1 is 3 or 4, instrument the override rate — the proportion of recommendations the human changes or rejects. Review it at the first post-deployment assurance cycle. An override rate approaching zero over a meaningful volume is evidence that effective autonomy exceeds the recorded score, and it should trigger a re-score rather than a note. Set the threshold in advance so that the conversation is about a pre-agreed number rather than an argument after the fact.

Two architectural options reduce the problem more reliably than policy does. Design the oversight point so the human must supply something the system did not (a reason code, a selection among genuinely different options), not merely clicking approve. And cap autonomy in architecture rather than in policy: a system that has no technical path to act cannot drift into acting.

6.4 D3 — Reversibility Deficit

The question: if a wrong outcome occurs, how hard is it to undo?

D3 is scored against **consequence reversibility, not mechanism reversibility**. This anchoring was tightened during the framework’s revision because the original wording invited a serious error.

The error looks like this. An agent sends an incorrect email to a customer. The mechanism is trivially reversible: delete the record, send a correction. The consequence is not reversible at all, because the customer has read it. Scoring D3 = 1 on the basis that the database row can be rolled back is wrong. **Ask whether the effect on the world can be undone, not whether the system’s state can be.**

Table 19. D3 — Reversibility Deficit

Score	Anchor	Characteristics
1	Trivially reversible	The effect can be undone immediately, at no cost, with no residue. Nothing has left the system boundary
2	Reversible with effort	Undoing requires a defined process, some cost, and some delay, but restores the prior position
3	Partially reversible	Some consequences can be remedied, others cannot. Compensation is possible; restoration is not
4	Effectively irreversible	The consequence stands. Money has moved irrecoverably, information has been disclosed, a decision has been communicated and acted on, or physical effect has occurred

Communication to a third party, disclosure of information, and execution of a financial transfer are the three most commonly mis-scored cases. All three are usually 3 or 4 regardless of how easy the system-side rollback is.

6.5 D4 — Exposure and Scale

The question: how many parties are affected, how far outside the enterprise does the effect reach, and can effects cascade?

Table 20. D4 — Exposure and Scale

Score	Anchor	Characteristics
1	Contained	Single user or small internal team. No external exposure. No downstream dependency
2	Departmental	A business function’s worth of internal users. Effects stay inside the enterprise
3	Enterprise-wide or limited external	Used across the enterprise, or externally by a defined and bounded population
4	Broad external or systemic	Large external population, or the system feeds others such that an error propagates beyond its own boundary

The cascade clause carries more weight than most assessors give it. A system with a small direct user population that writes into a system of record consumed by many others scores on the reach of the consequence, not the size of its own user base.

D4 is also the dimension most likely to change without anyone noticing. A pilot released to twelve internal users and later opened to customers has moved from 1 to 3 or 4. That change is classified Major and re-enters classification, which is why user-population change appears in the change model in Chapter 14.

6.6 D5 — Data Sensitivity and Model Uncertainty

The question: how sensitive are the inputs, and how well understood is the behavior?

D5 fuses two things that the source standards generally keep separate. The framework's own review found the fusion loses information, and the revision requires both sub-scores to be recorded alongside the composite even though the tiering formula uses only the composite.

Record all three. Score the composite as the higher of the two sub-scores, not their average, so that highly sensitive data in a well-understood model is not diluted into moderation.

Table 21. D5 — Data Sensitivity and Model Uncertainty

Score	Data Sensitivity anchor	Model Uncertainty anchor
1	Public or non-sensitive internal data	Deterministic or well-characterized; behavior is interpretable and stable
2	Internal confidential; no personal data	Understood, with documented evaluation and predictable failure modes
3	Personal data, commercially sensitive, or regulated	Generative or complex; failure modes known in kind but not in detail
4	Special-category personal data, safety-relevant, or subject to specific legal restriction	Poorly characterized behavior, opaque reasoning, or known to produce confident errors

Most generative systems will not score below 3 on model uncertainty. That is appropriate rather than a scoring inflation problem: confident plausible error is a real property of these systems, and the framework should reflect it.

6.7 Scoring in practice: who, when, how long

Scoring is a short, structured conversation, not a workshop. A well-run S2 for a typical system takes under an hour with the right people in the room.

Table 22. Scoring in practice: who, when, how long

Participant	Contributes
Risk and Compliance Lead	Accountable for the score. Runs the session, applies anchors, records justification
Business Sponsor	D1 and D4. Only the business can say what a wrong output actually costs
Architect	D2 and D3. Autonomy and reversibility are properties of the design
Data Owner	D5 data sensitivity sub-score
Security Architect	Consulted on D4 exposure and D5 where security-relevant

Every score carries a one-line justification. This is not bureaucracy: the justification is what a re-score six months later compares against, and it is what makes the countersignature meaningful. A score without a justification cannot be challenged, only overruled.

DO NOT COPY

Chapter 7. Risk Classification: Tiering, Calibration, and Gaming

Turning five scores into a tier, and defending that tier against the pressure it will come under.

7.1 Two axes, not a sum

Averaging five dimensions would let one catastrophic factor be diluted by four benign ones. A system with D1 = 4 and low scores elsewhere would average into apparent moderation, which is precisely the failure the tiering model must avoid.

The model uses two axes adapted from risk-matrix practice, with **Impact** and **Control Deficit** replacing probability and impact. For AI systems the decision-relevant question is not how likely failure is, but how much human control exists over whether failure becomes consequential.

Impact = average(D1, D4, D5), rounded up.

Control Deficit = average(D2, D3), rounded up.

D2 and D3 are paired deliberately. High autonomy combined with low reversibility is the combination that most sharply removes the enterprise's ability to intervene before harm lands.

Table 23. Two axes, not a sum

Impact ↓ / Control Deficit →	1	2	3	4
1	Tier 1	Tier 1	Tier 2	Tier 2
2	Tier 1	Tier 2	Tier 2	Tier 3
3	Tier 2	Tier 2	Tier 3	Tier 4
4	Tier 3	Tier 3	Tier 4	Tier 4

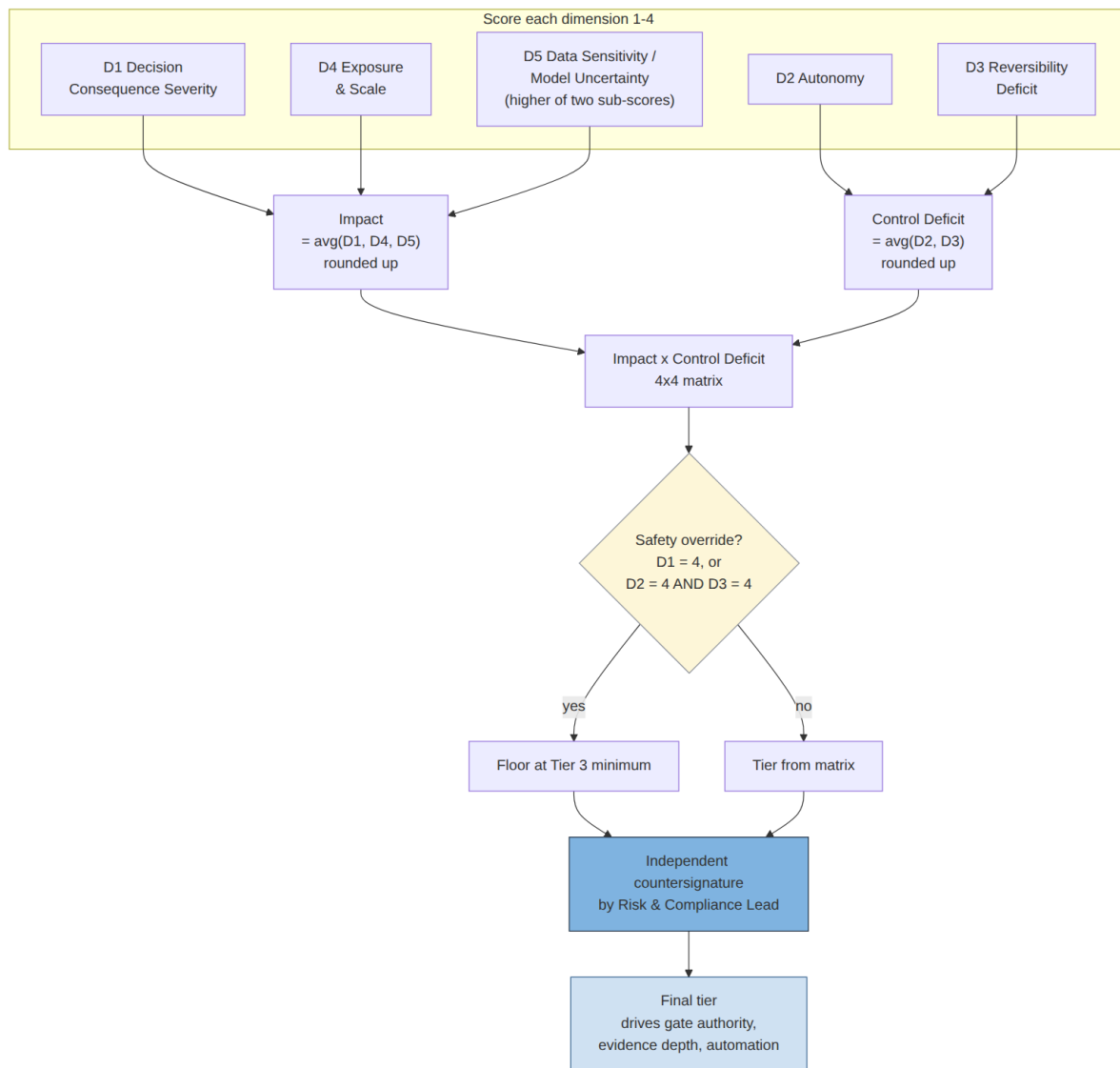


Figure 4. Risk classification: from five dimension scores to a countersigned tier

7.2 The safety-override floor

Regardless of what the matrix returns, a system is floored at **Tier 3 minimum** if either condition holds:

- **D1 = 4** — catastrophic consequence if wrong; or
- **D2 = 4 and D3 = 4** — fully autonomous and effectively irreversible.

The floor exists to stop averaging from masking a single severe factor. It is a blunt instrument by design, and its bluntness is the point: it does not require judgment to apply.

What the floor does not do. It catches extremes only. It does nothing about the mid-range, where a system scored $D1 = 3$ that arguably warrants 4 lands two tiers lower with no automatic backstop. Three independent reviewers identified this during the framework’s stress-testing as the live gaming vector, and it remains unresolved. Sections 7.6 to 7.8 are the practical response.

7.3 Worked examples

Example A — Internal document summarizer

An internal tool that summarizes long policy documents for staff. No decisions, no external exposure, internal-confidential inputs.

D1 = 1 (a poor summary wastes minutes), D2 = 1 (advisory), D3 = 1 (nothing leaves), D4 = 2 (departmental), D5 = 2 (internal confidential; generative but low stakes, uncertainty sub-score 3, sensitivity 2 → composite 3).

Impact = $\text{avg}(1, 2, 3) = 2.0 \rightarrow 2$. Control Deficit = $\text{avg}(1, 1) = 1$.

Matrix: Impact 2, Control Deficit 1 → **Tier 1**. No override triggered.

Note that D5 was scored 3 on the higher-of-two rule rather than averaged to 2.5. It did not change the tier here, but it would have if D1 or D4 had been one point higher, which is the kind of case the rule exists for.

Example B — Claims triage assistant

The worked example from Chapter 5, now scored fully.

D1 = 3 (misrouting delays a claim and may disadvantage a claimant), D2 = 2 (proposes a routing category, adjuster approves each), D3 = 2 (routing is reversible, though delay is not recoverable), D4 = 3 (all claims, internal users, feeds the claims system of record), D5 = 3 (restricted personal data; generative).

Impact = $\text{avg}(3, 3, 3) = 3$. Control Deficit = $\text{avg}(2, 2) = 2$.

Matrix: Impact 3, Control Deficit 2 → **Tier 2**. No override.

This result deserves scrutiny, and it illustrates why review matters. Tier 2 means architect self-certification at G2 and owner approval at G3. Is that right for a system touching every claim and handling restricted personal data? The scores are defensible individually. The outcome sits one point of D1 away from Tier 3. This is precisely the boundary case where an independent countersigner earns their place, and where the override-rate instrumentation from §6.3 should be mandatory: if adjusters accept 99% of routings, effective D2 is 3 and the tier moves.

Example C — Autonomous reconciliation agent

An agent that identifies and posts correcting journal entries in a financial ledger without human review below a value threshold.

D1 = 3 (misposting is a financial control issue, correctable but reportable), D2 = 4 (acts without review below threshold), D3 = 3 (a posted entry can be reversed, but the audit trail and any downstream reporting cannot be un-run), D4 = 3 (enterprise financial records, feeds reporting), D5 = 2 (internal financial data, deterministic-ish logic).

Impact = $\text{avg}(3, 3, 2) = 2.67 \rightarrow 3$. Control Deficit = $\text{avg}(4, 3) = 3.5 \rightarrow 4$.

Matrix: Impact 3, Control Deficit 4 → **Tier 4**. Override not triggered, but not needed.

The matrix does the work here. The combination of autonomy and partial irreversibility pushes Control Deficit to 4, and Tier 4 brings AI Governance Body plus Enterprise Sponsor sign-off, mandatory preventive controls, and a pre-approved kill-switch condition. That is a proportionate outcome for an agent posting to a ledger.

Example D — A scoring dispute

The same reconciliation agent, as the delivery team scored it.

D1 = 2 (“entries are reversible, so the consequence is minor”), D2 = 4, D3 = 1 (“a journal reversal takes one transaction”), D4 = 2 (“only the finance team uses it”), D5 = 2.

Impact = $\text{avg}(2, 2, 2) = 2$. Control Deficit = $\text{avg}(4, 1) = 2.5 \rightarrow 3$.

Matrix: Impact 2, Control Deficit 3 → **Tier 2**.

Two tiers lower, and every score is arguable in isolation. The errors are instructive. D3 was scored on mechanism reversibility, not consequence: the ledger entry reverses, the audit trail and any interim reporting do not. D4 counted direct users rather than the reach of the financial records the agent writes into. D1 followed from the D3 error.

None of this requires bad faith. It is what happens when anchors are applied loosely by people who also own the delivery date. §7.6 is the control.

7.4 Re-scoring: when the tier must be revisited

Classification is not a one-time event. Six triggers require a re-score.

Table 24. Re-scoring: when the tier must be revisited

Trigger	Source	Typical dimension moved
Any change classified Material or Major	Change model (Chapter 14)	Varies by change type
Autonomy increase of any kind	Change model	D2
User population change	Change model	D4
New or changed grounding source above current sensitivity	Change model	D5
Control Plane risk-relevant drift alert	Chapter 19	Varies
Scheduled review: annual minimum for Tier 3–4	This handbook	Any
Override-rate evidence contradicting recorded D2	§6.3	D2

Preserve score history instead of overwriting it. A Risk Classification Record that shows a system moved from Tier 3 to Tier 2 six weeks before an incident is a governance artifact of considerable value, and it is destroyed by editing in place.

7.5 Tier semantics: what each tier actually buys

Practitioners find tiers easier to apply when they know what each one costs.

Table 25. Tier semantics: what each tier actually buys

Tier	Governance posture	Practical consequences
Tier 1	Self-service within patterns	Architect self-certifies G2 against an unmodified pattern. Owner approves G3. Assurance is sampling-based. Audit-sampled periodically
Tier 2	Self-service with visibility	As Tier 1, with notification to ARB and AI Governance Lead. Fuller assurance evidence. Agent Card required if any tool access exists
Tier 3	Reviewed	ARB decides G2. AI Governance Body decides G3 and G4. Full assurance. Annual re-score. Preventive controls mandatory for standing-authority agents
Tier 4	Reviewed and escalated	As Tier 3, plus Enterprise Governance Sponsor sign-off, pre-approved kill-switch conditions, no automatic

Tier	Governance posture	Practical consequences
		exception renewal, and same-business-day alert response

A common and reasonable question is whether four tiers are the right number. The answer in the research is a reasoned judgment instead of a tested one: four is thought to be roughly the finest granularity assessors can apply consistently without significant inter-rater disagreement. That claim is untested. If your own experience shows assessors cannot reliably separate Tier 2 from Tier 3, the honest response is to sharpen the anchors, not to add a tier.

7.6 Independent countersignature

This is the highest-leverage control in IRGF, and the one most likely to be quietly dropped during adoption because it costs a person's time.

The rule. A D1–D5 score produced by a delivery team is not final until countersigned by the Risk and Compliance Lead, who is independent of delivery. The countersigner may change any score and must record a reason when they do.

Why it matters more than anything else. Unverified self-scoring was the single most confirmed weakness across the framework's entire adversarial review, converging from multiple independent reviewers. Every tier-scaled control downstream inherits the accuracy of this one number. A framework with excellent gates and wrong tiers governs the wrong things rigorously.

What countersignature is not. It is not a second full assessment. The countersigner reviews the justifications, tests the two or three scores nearest a threshold, and either accepts or adjusts. A thorough countersignature on a routine system takes fifteen minutes.

Where to concentrate it. Not uniformly. Concentrate on boundary cases: any score of 3 where 4 is arguable, any D3 justified on mechanism, not consequence, any D2 of 1 or 2 on a system with $D1 \geq 3$, and any case where the matrix result sits one point from a tier change.

7.7 Audit sampling for Tier 1–2

Countersignature covers the initial score. Sampling covers the systems that were scored low and then never looked at again.

[Practice recommendation] Sample 10% of Tier 1–2 self-certifications per quarter, selected with deliberate bias, not at random: weight selection toward systems whose D-scores sit one point below a tier boundary, systems whose owner has multiple low-scored systems, and systems whose function has changed since scoring. Random sampling of a population where most cases are correctly scored wastes the sample on easy cases.

Publish the outcome as a rate, not as individual findings. “12% of sampled Tier 1 systems were re-tiered upward” is a governance signal the organization can act on. Naming teams turns sampling into an enforcement activity and drives scoring underground.

7.8 Recognizing and countering under-tiering

Under-tiering is rarely dishonest. It is the predictable result of asking people with a delivery commitment to assess their own governance burden. Treat it as a systems problem.

Table 26. Recognizing and countering under-tiering

Signal	What it suggests	Response
A cluster of systems scoring just below a tier boundary	Anchoring to the threshold rather than to the system	Review the cluster together; recalibrate anchors with examples
D3 justified by rollback capability	Mechanism-reversibility error	Re-anchor to consequence; this is the most common single error
D2 = 1 with near-zero override rate	Effective autonomy exceeds formal	Re-score D2; consider architectural cap
Vendor AI consistently unscored	Treated as a feature, not a system	S2 applies to all build types; route through Procurement (Chapter 21)
Scores unchanged after significant change	Re-score triggers not firing	Check change classification discipline (Chapter 14)
Tier distribution heavily weighted to Tier 1	Possible systemic under-tiering	Compare against incident data; tier distribution should roughly track consequence distribution

The last row is the most useful portfolio-level diagnostic available. **[Practice recommendation]** Publish the tier distribution quarterly alongside the incident and near-miss record. If almost everything is Tier 1 and incidents keep arising in Tier 1 systems, the classification model is not describing your estate, and no amount of gate rigor will compensate.

Chapter 8. Architecture Decision: S3

Settling sourcing, model, hosting, grounding, and authority boundary — and recording the decision so that everything downstream has something to compare against.

8.1 What S3 decides

S3 produces the Architecture Decision Record that G2 assesses and that the Control Plane later compares deployed reality against. Its scope is the architecture boundary, not the engineering interior (Chapter 1, §1.3).

Six decisions belong here.

Table 27. What S3 decides

Decision	Recorded as	Downstream dependency
Sourcing	Build, fine-tune, retrieve, API, SaaS, or no AI	Determines governance weight (Chapter 4, §4.5)
Model or service	Named model family from the approved list, or an exception	Policy-as-code checks deployed model against this
Hosting pattern	Vendor API, private hosting, on-premises	Sovereignty and continuity implications
Grounding	Which governed data sources, if any, ground the system	Lineage record at S4; lineage drift detection later
Authority boundary	For agentic systems: tools, actions, limits	Agent Card; machine-readable boundary at G3
Pattern conformance	Which reference pattern, unmodified, modified, or none	Determines whether the fast path at G2 is available

8.2 The sourcing decision framework

S0 raised the sourcing question provisionally. S3 settles it. Twelve considerations bear on the choice; not all matter for every case, but each should be consciously dismissed, not overlooked.

Table 28. The sourcing decision framework

Consideration	Pushes toward build or fine-tune	Pushes toward API or SaaS
Differentiation	The capability is a durable competitive advantage	The capability is table stakes
Domain specificity	Domain language or format is genuinely distinctive	General language suffices
Data sensitivity	Data cannot leave the enterprise boundary	Data is low-sensitivity or can be redacted
Data sovereignty	Jurisdictional restrictions on processing location	No location constraint
Intellectual property	Outputs or training data create IP that must be retained	No IP consequence
Explainability need	High-consequence decisions requiring defensible reasoning	Explainability not material
Model control	Behavior must be stable and version-pinned	Vendor-managed updates acceptable
Latency	Strict latency requirements favor local inference	Network latency tolerable
Cost profile	High sustained volume amortizes fixed cost	Variable, low, or unpredictable volume
Vendor dependency	Concentration risk is unacceptable	Dependency acceptable and diversifiable
Compliance evidence	You must be able to produce evidence a vendor will not supply	Vendor attestations suffice
Skills	You have and can retain the capability	You cannot staff it durably

Two observations from the framework’s design worth carrying into the conversation. First, **the sourcing decision is the architecture decision** for externally consumed models; the vendor and data-flow choice is where nearly all the governable content sits. Second, fine-tuning commits you to treating every subsequent training-data change as a major change with full revalidation. That recurring cost is frequently omitted from the business case.

Sourcing and the “no AI” option

The no-AI option remains live at S3, not just S0. Architecture work sometimes reveals that the actual requirement is a deterministic rule with good data behind it. Recording that outcome as a valid S3 decision, not as a failed initiative, makes it available as a real choice.

8.3 Reference patterns and the fast path

Ten pre-approved patterns exist (Chapter 35). Their governance function is specific: **a Tier 1–2 system built from an unmodified pattern may use the largely automated G2 path**. That is the framework’s main genuine burden reduction, and it only works if pattern conformance is assessed honestly.

Three conformance outcomes are possible, and the distinction matters.

Conformant. The system uses the pattern as published, with only the configuration the pattern anticipates. Fast path available at Tier 1–2.

Modified. The system follows the pattern’s structure but changes something material — an additional data path, a different trust boundary, an extra integration. Fast path not available. The modification is documented in the ADR and reviewed on its merits.

Novel. No pattern applies. Full review regardless of tier, and a candidate for promotion into the pattern library if it proves reusable (Chapter 35, §35.3).

[Practice recommendation] Watch for pattern-conformance inflation, where teams describe a materially modified architecture as conformant to obtain the fast path. The counter is not suspicion but specificity: patterns should be published with an explicit list of what may vary and what may not. A pattern that says only “use a gateway” cannot be assessed for conformance and will be claimed for everything.

8.4 Writing an ADR that survives contact with the Control Plane

An architecture decision record in IRGF carries a load that a conventional ADR does not: it is the approved-state reference the Control Plane compares deployed reality against. Fields that cannot be compared produce no signal.

Table 29. Writing an ADR that survives contact with the Control Plane

Field	Requirement	Why the Control Plane needs it
Decision statement	What was decided, in one sentence	Human readability
Alternatives considered	What else was assessed, and why rejected	Review quality; not machine-compared
Rationale	Why this choice	Review quality
Model or service	Named family and version policy	Compared against deployed model registry

Field	Requirement	Why the Control Plane needs it
Hosting pattern	Named deployment topology	Compared against cloud inventory
Grounding sources	Named governed data products	Compared against data catalog
Authority boundary	For agents: tool list, action limits	Compared against IAM grants
Risk tier at decision	Tier and Risk Classification Record reference	Sets alert severity
Pattern conformance	Conformant, modified with detail, or novel	Policy-as-code check
Accountable owner	Named individual	Alert routing

The distinction between a field written for a human reviewer and a field written for machine comparison is worth making explicit to authors. “We will use a modern managed model with appropriate controls” is acceptable prose and useless as approved state. “Model family: [approved list entry]; version policy: pinned minor, vendor patch auto-accepted; grounding: Data Product PD-114 only” can be compared tomorrow morning.

Superseding, not editing

An ADR is never edited to reflect a later change. A new ADR supersedes it and references the one it replaces. This preserves the decision history that makes a drift investigation tractable and prevents the common failure where the record silently tracks reality and therefore never shows drift at all.

Chapter 9. Data Readiness and Gate G2: S4

The stage that determines whether an AI system's outputs can be traced to governed inputs, and the gate that assesses architecture and data together.

9.1 What S4 establishes

Three facts, each recorded in the Data Lineage and Sensitivity Record.

Lineage. Which governed data sources ground, train, or otherwise feed the system, named individually. “The knowledge base” is not lineage; “Data Product PD-114, Policy Documents, owner Legal Operations” is.

Classification. The sensitivity classification of each source, using the enterprise's existing scheme. IRGF does not define a classification scheme and should not; it consumes yours.

Governance status. Whether each source has an owner, a maintenance commitment, and a refresh cadence. This is the field most often left blank and most often responsible for later failure.

9.2 The stale-grounding problem

A retrieval-augmented system grounded on a source nobody maintains will pass every gate and then degrade silently. Its outputs remain fluent and become progressively wrong. No infrastructure telemetry detects this, because nothing about the deployment has changed.

This is why S4 asks for a maintenance commitment, not only a classification. A grounding source without a named owner and a refresh cadence should be treated as a G2 finding, not a documentation gap.

[Practice recommendation] Record a maximum acceptable staleness per grounding source at S4, and treat exceeding it as a drift condition in the Data Lineage category (Chapter 19). For a policy corpus that changes quarterly, ninety days may be right. For a pricing table, a day may be too long. The number matters less than the existence of an agreed one.

9.3 Grounding lineage as the AI-specific extension

Data governance is otherwise deferred to your existing DAMA-aligned practice. IRGF extends it at exactly one seam: the relationship between a governed data source and an AI system it grounds.

That seam matters because it is where two failure modes meet. Data governance tracks sources and their quality. Model governance tracks model behavior. Neither, on its own, notices when a model's behavior changes because its grounding source changed. Keeping the lineage relationship as a first-class governed fact is what allows the two to be distinguished at diagnosis time, and it is one of the framework's few genuinely original mechanisms.

9.4 Gate G2 — Architecture and Data Readiness

Table 30.

Element	Content
Purpose	Confirm the proposed architecture and its data foundation are governed, traceable, and conform to patterns or justify departure
Entry criteria	ADR complete; Data Lineage and Sensitivity Record complete; risk score updated to reflect now-known architecture
Required evidence	ADR; data classification and lineage record; pattern conformance result or documented exception
Decision authority	Tier 1–2: architect self-certification against pattern. Tier 3–4: ARB decision
Outcomes	Approve / Approve with conditions / Remediate / Escalate / Reject
Automation	Moderate. Pattern conformance is a strong policy-as-code candidate at Tier 1–2

Sequencing inside G2

S3 and S4 are frequently iterative, not sequential. An architecture choice constrains which data sources are usable; a data constraint forces an architecture change. The framework was revised to permit **parallel and iterative resolution of S3 and S4 sub-components before a joint G2 exit**, instead of requiring S3 to close before S4 opens.

What must be true at G2 exit is that both records are complete and consistent with each other. What need not be true is that they were produced in order.

The risk re-score at G2

The provisional tier from G1 was assigned before the architecture was known. Architecture routinely changes the score in both directions: a decision to use a private hosted model lowers D5; a decision to grant an agent write access to a system of record raises D2 and D3 together.

A G2 that does not revisit the tier is incomplete. If the tier changes, the decision authority for G2 itself may change with it, which occasionally means a gate in progress must escalate mid-review. That is inconvenient and correct.

9.5 Vendor and procured AI at G2

Procured AI is where G2 most often fails to function, because the evidence a reviewer wants is held by a third party who may decline to provide it.

The framework's answer has three parts.

Procurement is a named consulted role at S1 and S2 for procured or integrated AI, not merely a purchasing function invoked after the architecture is set.

The ADR carries extended due-diligence fields for procured systems: what evaluation evidence the vendor supplies, what the vendor's model update policy is, what data the feature processes and where, and what contractual audit rights exist.

There is a defined resolution for evidence stalemate. When a vendor will not supply evidence the tier requires, two outcomes are available: cap the system at Tier 2 by constraining its use so that the higher-tier requirements do not apply, or obtain a documented risk acceptance from the

AI Governance Body. What is not available is proceeding at Tier 3–4 with the evidence gap unrecorded.

[Practice recommendation] Build the evidence requirements into procurement templates before you need them. Asking a vendor for model evaluation evidence during contract negotiation is a different conversation from asking after deployment, and the second conversation usually fails.

9.6 Common G2 failures

Table 31.

Failure	What it looks like	Correction
Unfalsifiable ADR fields	“Appropriate model with suitable controls”	Require named, comparable values in the fields the Control Plane reads
Conformance inflation	Materially modified architecture claimed as pattern-conformant	Publish what may vary in each pattern
Grounding source without owner	Lineage recorded, maintenance not	Treat missing owner or cadence as a finding
Tier not revisited	G1 provisional tier carried into G3	Make re-score a G2 exit condition
Vendor gap unrecorded	Evidence absent, gate passed anyway	Use the Tier-2 cap or documented acceptance; never silence
ADR edited rather than superseded	History lost	Supersede and reference

Chapter 10. Build, Configure, Acquire: S5

The stage where architecture governance deliberately does the least, and the two things it must still do.

10.1 The light-touch principle

S5 is where the system gets built, integrated, or purchased. Architecture governance is intentionally thin here. Construction is an engineering and delivery activity, and IRGF's scope boundary (Chapter 1, §1.3) puts model training, prompt authoring, retrieval tuning, and implementation detail outside its reach.

Two governance obligations remain, and both are about boundary conformance rather than construction quality.

The build must stay inside the approved architecture. The system that emerges from S5 should be the system the ADR describes. Where implementation reveals the approved architecture is unworkable, that is a change to the architecture decision, handled by superseding the ADR, not an implementation detail to absorb quietly.

Approved-list conformance must hold. Models, platforms, and patterns used in the build must be those the ADR named. This is a strong policy-as-code candidate and one of the first preventive rules worth automating (Chapter 18).

10.2 Where drift begins

Most architecture drift originates in S5, not after deployment. The pattern is consistent and worth naming so teams recognize it.

Table 32. Where drift begins

Origin	What happens	Detection
Convenience substitution	A different model is used because the approved one had a quota issue during a sprint	Model registry compared against ADR
Scope accretion	An additional data source is connected “temporarily” for testing and stays	Data catalog compared against lineage record
Permission expansion	A service account is granted broader access to unblock integration	IAM compared against authority boundary
Pattern departure	The pattern's trust boundary is adjusted to fit an existing network topology	Policy evaluation of deployment configuration

None of these is misconduct. Each is a reasonable local decision under delivery pressure. The framework's response is not to prevent them but to make them visible, so that a temporary expedient either gets ratified through a change record or gets reverted before production.

[Practice recommendation] Run Control Plane comparison in non-production environments from the point the ADR is approved, not from deployment. Drift found in a development environment is a conversation; the same drift found in production is an incident.

10.3 Procured systems at S5

For purchased AI, S5 is configuration and entitlement review, not construction. The governance content is different in kind.

Three checks carry most of the value. **What data the feature actually processes**, verified against what procurement was told. **What the default configuration enables**, since vendor defaults frequently turn on capabilities the risk assessment did not contemplate. **What the update policy is in practice**, because a vendor-side model change is a change to your system that you did not initiate and may not be notified of.

That last point connects directly to the change model in Chapter 14: vendor-initiated model updates are changes to your system, and your change classification applies to them even though you did not make them.

Chapter 11. AI Assurance: S6

Five assurance types, five different owners, and why treating them as one activity produces both over-testing and blind spots.

11.1 Assurance is not one thing

The Core Model holds a single Assurance entity, and that single entity bundles five logically separable activities with different owners, different evidence, and different escalation urgency. Separating them is what allows the framework to escalate a safety finding quickly while routing a value shortfall through ordinary change.

Table 33. Assurance is not one thing

Assurance type	Question	Owner	Evidence	Feeds
Technical AI Evaluation	Does it perform correctly and safely at the output level?	ML engineering, with architecture visibility only	Evaluation results against defined thresholds	G3, G4
Architecture Assurance	Does deployed configuration still match approved architecture?	Architecture, increasingly automated	Conformance output, drift alerts	S8, G4 trigger
Risk Assurance	Is the tier still right, and are mapped controls effective?	Risk and Compliance	Control-effectiveness testing, re-scoring	G4 trigger, escalation
Compliance Assurance	Does it still conform to policy and regulation?	Risk and Compliance	Compliance checks, external audit evidence	G3, G4, reporting
Business Outcome Assurance	Is it delivering the value it was authorized to deliver?	Business or Capability Owner	Outcome metrics against the S1 baseline	Benefits realization, retirement

11.2 Differential escalation

The five types deliberately do not escalate identically, and this is one of the more useful design choices to explain to stakeholders.

A Technical AI Evaluation failure — a hallucination rate breaching threshold — triggers Risk Assurance re-scoring, which may trigger escalation, which may re-open G4. That path is fast.

A Business Outcome Assurance failure — the system works correctly, complies fully, and simply is not delivering the promised value — does **not** trigger risk escalation. It routes through ordinary Change or Retirement. Treating a value shortfall with the same urgency as a safety finding misallocates governance attention and, over time, trains people to ignore escalations.

Reserve the fast path for safety and compliance. Route value failures through the normal cycle.

11.3 What architecture governance owns and does not

Technical AI Evaluation belongs to ML engineering. IRGF governs that evaluation gates **exist** and **bind to deployment**, not how they are conducted. The framework does not specify benchmark suites, evaluation datasets, or thresholds for hallucination or bias, and should not:

those are engineering and domain judgments that change faster than a governance framework can track.

What architecture governance requires is narrower and enforceable:

- An evaluation was run before G3, appropriate to the build type
- Its results are referenced from the AI Assurance Summary
- Its pass criteria were defined before the run, not after
- The evidence chain is deep enough for the tier

Tier depth differs materially here. At Tier 1 the AI Assurance Summary may be a pass or fail reference. At Tier 4 it must carry the full underlying evaluation reference chain, sufficient for someone to reconstruct what was tested and against what threshold.

11.4 Assurance scope by build type

Table 34.

Build type	Assurance emphasis
Traditional ML	Accuracy, bias, explainability, stability over time
Foundation model API	Output quality, data leakage, vendor dependency, behavior under model update
Generative AI	Hallucination, toxicity, bias, adversarial resilience, prompt injection
Retrieval-augmented	Faithfulness to grounding source, stale-knowledge detection, retrieval failure behavior
Fine-tuned	Regression against base-model behavior, training-data governance
Copilot	Usability of the oversight point, not only technical accuracy
Single agent	Adversarial resilience, containment, boundary enforcement under pressure
Multi-agent	Emergent behavior, interaction effects, delegation-chain integrity

The copilot row is easy to skip and important. For a human-paired system, an assurance activity that tests only model accuracy has tested half the system. The oversight interaction is an architectural element with its own failure mode — an approve button that is always pressed provides no assurance regardless of how accurate the model is (Chapter 6, §6.3).

Chapter 12. Deployment Authorization: S7 and Gate G3

The most sharply tier-differentiated decision in the framework, and the point at which machine-readable boundaries must exist.

12.1 What G3 authorizes

G3 records a Decision: this system, in this configuration, at this tier, is authorized to operate in production, by this named person, on this evidence. It is the record every subsequent drift comparison refers back to.

Table 35. What G3 authorizes

Element	Content
Purpose	Authorize production deployment on assurance evidence
Entry criteria	Completed assurance results; control-to-risk mapping; confirmed tier
Required evidence	Assurance results; Control Matrix; named accountable owner; human oversight point confirmation at Tier 2+; machine-readable permission boundary for agentic systems
Decision authority	Tier 1: System Owner. Tier 2: System Owner with AI Governance Lead notified. Tier 3: AI Governance Body. Tier 4: AI Governance Body plus Enterprise Governance Sponsor
Outcomes	Approve / Approve with conditions / Remediate / Escalate / Reject
Automation	Low at Tier 3–4 by design; moderate at Tier 1–2 where automated evidence checks pre-clear routine cases

12.2 Machine-readable permission boundaries

For any system with agentic capability, G3 requires the authority boundary to be recorded in a form a machine can evaluate, not only in prose.

The reason is detection, not formality. A boundary written as “the agent may read claims data and draft recommendations but may not approve payments” cannot be compared against an IAM grant automatically. A boundary expressed as an explicit tool and permission list can be, which means an unauthorized expansion becomes detectable rather than merely prohibited.

This requirement was added during the framework’s revision precisely because a prohibition nobody can detect the violation of is not a control.

[Practice recommendation] Express the boundary in whatever policy language your enforcement point already understands, and store the ADR-referenced version alongside it. Two representations of the same boundary that can drift apart is a worse position than one, so generate the human-readable version from the machine-readable one instead of maintaining both.

12.3 Kill-switch conditions

At Tier 3–4, G3 must pre-approve any condition under which automated shutdown is permitted. This is the single principled exception to the framework’s default prohibition on automated corrective action for high-tier systems (Chapter 18, §18.4).

A valid kill-switch condition has three properties. It is **objectively evaluable** — no judgment required at trigger time. It is **defined in advance** at G3, never invented at runtime. And it is **narrow** — matched to a specific catastrophic pattern rather than to general anomaly.

Example. Agent attempts a transaction above its approved value limit, or to a counterparty outside its approved list. On trigger: suspend agent, revoke tool access, alert Agent Owner and AI Governance Lead, preserve state for investigation.

What is not a valid kill-switch condition: “the agent behaves anomalously,” “error rate exceeds normal,” or anything requiring the Control Plane to judge whether behavior is acceptable. The kill switch fires on a pre-agreed pattern, and its firing is always traceable to a human decision made at G3.

12.4 Conditions, and the discipline of not accumulating them

Conditional approval at G3 carries more risk than at G1, because the system is going live. A condition that must be satisfied after deployment is, in effect, an accepted risk with a promise attached.

Two rules keep this honest. Any G3 condition must have a **named owner and a date**. And any condition still open at its date **escalates automatically** to the authority that granted it, instead of lapsing quietly.

[Practice recommendation] Report open G3 conditions as a portfolio metric alongside open exceptions. Both represent governance debt, and both are invisible unless counted.

Chapter 13. Runtime Operation and Monitoring: S8

The concurrent state where most of an AI system's life is spent, and where IRGF's central claim is either realized or quietly abandoned.

13.1 Why S8 is a state, not a stage

S8 begins at deployment and continues until retirement. Monitoring is not a phase that follows operation; the two are the same thing observed from different angles. Modeling them separately would reintroduce the periodic assumption the framework exists to remove.

Practically, this means there is no point at which a system is “through” governance. A production system is permanently in S8, permanently generating assurance evidence, and permanently comparable against its approved state.

13.2 What must be observable

Four questions must be answerable continuously for any production AI system. Each maps to a different evidence source.

Table 36. What must be observable

Question	Evidence source	Failure signal
Is deployed architecture still what was approved?	Control Plane comparison against ADR	Structural, configuration, integration drift
Is the model behaving as evaluated?	Technical evaluation signals, output monitoring	Behavioral drift
Are the mapped controls still effective?	Control testing, security tooling	Control effectiveness decay
Is it still delivering value?	Outcome metrics against the S1 baseline	Benefit shortfall

The fourth is the one organizations most often drop. A system that is technically sound, compliant, and useless continues to consume operating cost, attention, and risk surface. Business Outcome Assurance exists to make that visible, and its absence is why estates accumulate systems nobody would authorize today.

13.3 Monitoring intensity by tier

Table 37.

Tier	Comparison cadence	Alert routing	Review
Tier 1	Periodic batch acceptable	Low-urgency digest to owner	Annual light-touch
Tier 2	Near-continuous for structural and configuration	Direct to owner	Annual
Tier 3	Continuous	Owner, with escalation if unresolved in 5 business days	Annual re-score minimum
Tier 4	Continuous	Owner and AI Governance Lead, same-business-day response	Annual re-score, plus review after any material alert

These intervals are reasoned defaults. Calibrate them against your own alert volumes once you have them, and expect the Tier 3 five-day window in particular to need adjustment in either direction.

13.4 The cumulative exposure gap

One limitation deserves flagging at the point it bites. IRGF has **no mechanism for cumulative or structuring exposure**: many individually sub-threshold actions by a standing-authority agent accumulating into material exposure between audit cycles.

An agent authorized to make individual payments below a threshold can, over a period, move an amount that would never have been approved as a single transaction. Every action is within boundary. The aggregate is not governed by anything in the framework.

This is a known, unresolved gap. Until it is addressed, the practical mitigation is external: **[Practice recommendation]** for any standing-authority agent with financial or data-egress capability, define an aggregate limit per period alongside the per-action limit, and monitor it as a control in your existing operational risk tooling instead of expecting the framework to catch it.

Chapter 14. Change: S9 and Gate G4

Classifying change by the risk it moves rather than the effort it takes, and why that distinction carries most of the framework's protective value.

14.1 The classification principle

Change classification in IRGF tracks **which risk dimension a change plausibly moves**, not how large the change is technically. This inverts the intuition most change processes are built on.

A one-line prompt edit that grants an agent a new capability is Major. A large model version upgrade inside an already low-tier advisory system may be Minor. Effort-based classification is precisely what allows governance-relevant changes to pass as routine updates, which is the failure mode the risk-driven model exists to prevent.

Three classifications exist.

Table 38. The classification principle

Classification	Meaning	Gate consequence
Minor	No plausible movement in D1–D5	No gate; recorded only
Material	Plausible movement within the current tier	G4 with scoped re-assurance
Major	Plausible tier change, or change to purpose, autonomy, or population	Re-entry to earlier stages plus G4

14.2 The change type table

Table 39.

Change type	Default classification	Gate re-triggered	Special rule
Model version update, same provider and family	Minor at Tier 1–2; Material at Tier 3–4	None / G4	Higher tiers need revalidation because behavior can shift
Provider or model-family change	Major	G2 + G4	Treated as a new architecture decision
Prompt or system-instruction change, substantive	Material	G4, scoped to affected assurance areas	—
Prompt or formatting change, cosmetic	Minor	None	—
Grounding source addition or change	Material if new source sensitivity $\geq$ current; Minor otherwise	G4 for the material case	Lineage check re-runs regardless of classification
Training data change (fine-tuning)	Major	G2 + G4	Full revalidation, not scoped
Autonomy increase	Major, always	S2 re-entry + G2 + G4	Never eligible for automated or self-service classification
Tool or permission addition for an agent	Material to Major by blast radius	G4, or S2 + G2 + G4	A tool granting irreversible or financial action is always Major
User population change	Major	S2 re-entry + G3	Treated as re-entering deployment authorization
Business purpose change	Major, treated as a new opportunity	S0 re-entry	Technical assurance may be reused where still applicable; qualification and classification are redone

14.3 The cosmetic-versus-substantive problem

The table above relies on distinguishing a cosmetic prompt change from a substantive one, and **no bright-line test exists**. This is a named, unresolved limitation, not an omission.

The distinction is judgment, and judgment under delivery pressure drifts toward the cheaper classification. Three practices reduce the drift without pretending to eliminate it.

Define the boundary by effect, not by text. A change is cosmetic if it cannot change what the system does, only how the output is presented. Reformatting a response template is cosmetic. Changing an instruction about when to escalate is not, however few words it touches.

Require a second opinion above a tier. [Practice recommendation] At Tier 3–4, cosmetic classification of any prompt or instruction change should be confirmed by someone other than the author. The cost is small and the classification is the control.

Sample classifications retrospectively. Include change records in the quarterly audit sample (Chapter 7, §7.7). A pattern of substantive changes classified cosmetic is a signal about pressure on the team, not about individual judgment.

14.4 Vendor-initiated change

Changes you did not make are still changes to your system. When a vendor updates a model behind an API, your system's behavior may shift without any change on your side, without notification, and without a change record.

Three practical responses.

Pin where the tier justifies it. For Tier 3–4 systems, prefer version-pinned model access even at additional cost. Where pinning is unavailable, record that as an accepted risk at G3 rather than discovering it later.

Detect behaviorally. Vendor-side model change appears as AI-Model Behavioral Drift (Chapter 19), detected through evaluation signal rather than infrastructure telemetry. For any Tier 3–4 system on an unpinned vendor model, a periodic re-run of the evaluation suite is the only reliable detection available.

Contract for notice. Model update notification and a defined notice period belong in procurement terms. This is far easier to obtain before signature than after.

14.5 Gate G4 — Material Change

Table 40.

Element	Content
Purpose	Re-authorize a system whose change is classified Material or Major
Entry criteria	Change classification record; delta assurance evidence scoped to what the change affects
Required evidence	Change record; affected-control re-verification; updated risk score where D1–D5 are touched
Decision authority	Scales with the post-change tier, using the G3 authority table

Element	Content
Outcomes	Approve / Approve with conditions / Remediate / Escalate / Reject / Retire
Automation	High for minor changes, which bypass the gate entirely; low for major

Two features are worth emphasizing. Authority scales with the tier **after** the change, not before, which means a change that raises the tier also raises the approval authority for its own gate. And Retire is a legitimate G4 outcome: a change request that reveals the system should not continue is a valid finding, and having the outcome available makes it likelier to be used.

Scoping delta assurance

G4 does not require re-running every assurance activity. It requires re-running what the change plausibly affects. Scoping is a judgment recorded in the change record, and it should name what was re-run and what was not, with a reason.

The exception is fine-tuning, where the framework requires full revalidation rather than scoped. Training-data change can alter behavior in ways that do not respect the boundaries of what the change was intended to affect.

Chapter 15. Retirement: S10 and Gate G5

The stage most often skipped, and the one that determines whether you can answer questions about a system after it stops running.

15.1 Why retirement is governed at all

A decommissioned AI system leaves behind obligations: data that must be retained or deleted on a schedule, access that must be revoked, evidence that must remain available for audit or dispute, and knowledge that will otherwise be lost.

Organizations routinely handle the first two and neglect the second two. The consequence surfaces when a question arrives about a decision the system made eighteen months ago and nobody can reconstruct what version was running, what it was grounded on, or who authorized it.

15.2 What S10 must produce

Table 41.

Obligation	Evidence	Owner
Data retention or deletion	Confirmation from the data platform, per the retention schedule	Data Owner
Access revocation	IAM deprovisioning confirmation, including machine identities	System Owner with Security
Evidence preservation	Archived record set sufficient to answer post-retirement questions	System Owner
Knowledge preservation	What was learned that should inform future systems	System Owner
Dependency check	Confirmation that nothing downstream still consumes its outputs	Architect

The dependency check is frequently the one that stops a retirement. Systems acquire consumers nobody recorded, and switching off a service that quietly feeds a report is a familiar incident. **[Practice recommendation]** Use Control Plane integration telemetry to identify actual consumers instead of relying on the documented dependency map, which is usually incomplete.

15.3 What “evidence preservation” means concretely

The retained set should let someone reconstruct, without access to the running system: what the system was, what it was authorized to do, who authorized it, what data grounded it, what its risk classification was and how it changed, what changes it underwent, and what assurance evidence supported its authorization.

In practice this is the record set: final ADR chain, Risk Classification history, Deployment Authorization records, Change records, Agent Card if applicable, Data Lineage record, AI Assurance Summaries, and any exceptions. Retention runs for the life of the system plus a defined period; three years is the framework’s default and your regulatory context may require longer.

15.4 Gate G5 — Retirement

Table 42.

Element	Content
Purpose	Confirm orderly decommissioning
Entry criteria	Decommissioning plan covering retention, access, and knowledge
Required evidence	Retention or deletion confirmation; access revocation confirmation; archived audit trail
Decision authority	System Owner plus Data Owner; Tier 3–4 adds AI Governance Lead confirmation that no open obligations remain
Outcomes	Approve / Approve with conditions / Escalate
Automation	Moderate. Deletion and revocation confirmation automate well; knowledge preservation does not

Note that Reject is not an available outcome. A retirement can be delayed by conditions, but a system whose owner has decided to decommission it is not kept running by a governance gate. What the gate controls is whether decommissioning was completed properly.

15.5 Orphaned and abandoned systems

The harder retirement problem is systems nobody retires: a pilot whose sponsor moved on, a service still running because switching it off feels risky, a vendor feature nobody enabled deliberately.

[Practice recommendation] Run an annual estate sweep against three signals: no Business Outcome Assurance evidence in twelve months, no owner responding to governance correspondence, and no measurable usage. Any system triggering two of three enters a defined disposition process with a deadline, ending in either re-authorization with a current owner or retirement. Absent a forcing mechanism, orphaned systems accumulate indefinitely, and each one is an unowned risk carrying a real tier.

Chapter 16. Governing Agents

Systems that act, not advise, and the governance chain that constrains them.

16.1 Why agents need separate treatment

An agent differs from other AI systems in one governance-relevant respect: it converts an architecture question into an authority question. For a conventional application the question is what this component computes. For an agent it is what this component is permitted to do, and to whom that authority was delegated.

Classical enterprise architecture has well-developed thinking about data flow and integration. It has comparatively underdeveloped thinking about delegated authority, machine identity, and bounded autonomy, because until recently software did not act on its own.

16.2 The governance chain

Eight links, each a control point.

Agent → Identity → Permission → Tool → Policy → Action → Verification → Audit

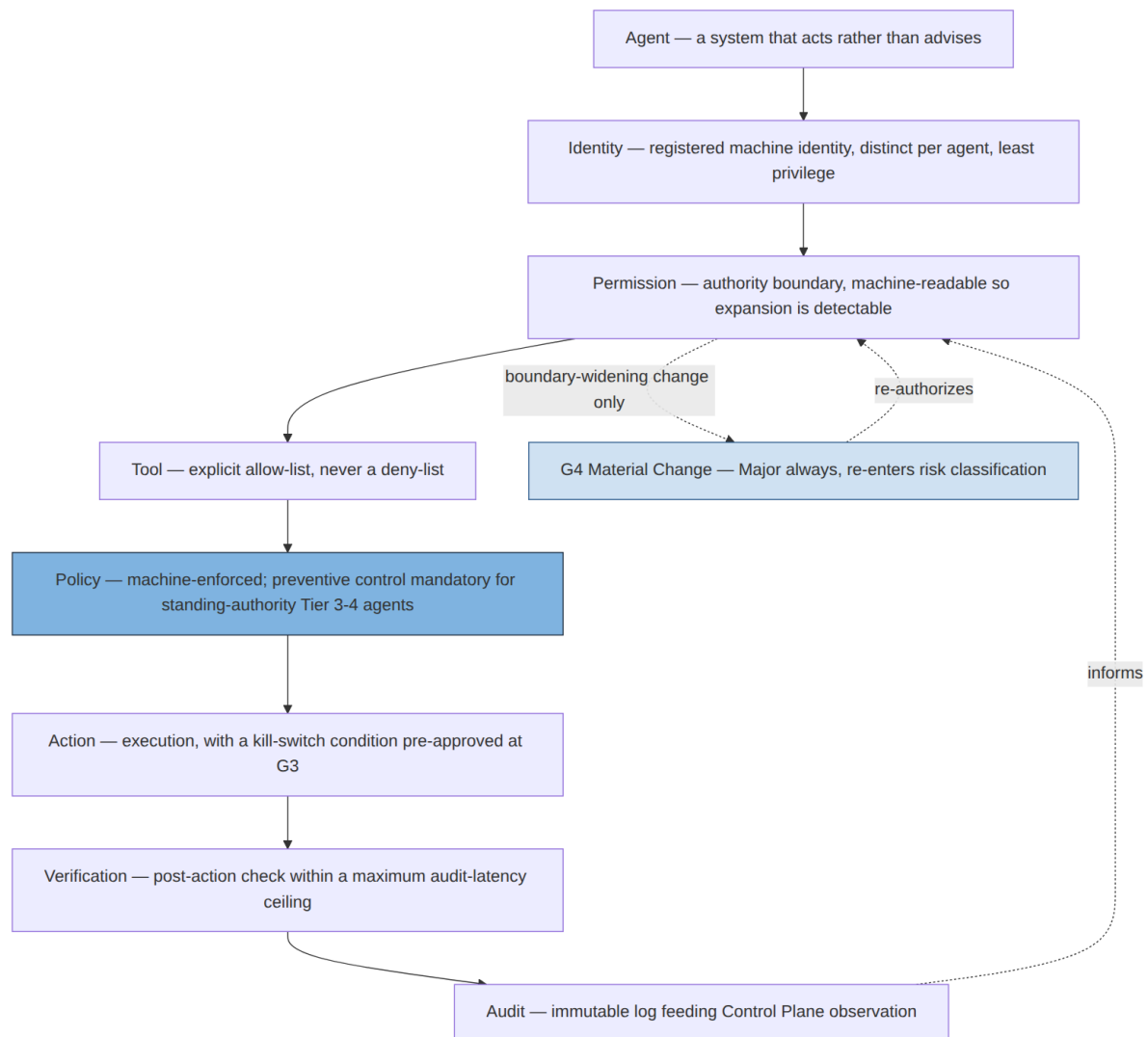


Figure 5. The agent governance chain: eight links, each a control point

Table 43. The governance chain

Link	What it establishes	Recorded in	Failure if absent
Identity	A registered machine identity, distinct per agent, least privilege	Agent Card; IAM	Actions cannot be attributed; shared service accounts make audit meaningless
Permission	The authority boundary, machine-readable	Agent Card; ADR; policy engine	Boundary cannot be compared against actuals, so expansion is undetectable
Tool	Explicit allow-list of tools and APIs	Agent Card	Implicit tool access expands silently
Policy	Machine-enforced constraints; mandatory preventive control for standing-authority Tier 3–4 agents	Policy-as-code	Boundary exists on paper only
Action	Execution, with a pre-approved kill-switch condition	Runtime	No stopping mechanism when behavior goes wrong
Verification	Post-action check within a maximum audit-latency ceiling	Runtime, audit log	Errors compound before detection
Audit	Immutable log feeding Control Plane observation	Audit store	No basis for investigation or drift detection

Two requirements in that table were added after the framework's stress-testing found the original design insufficient, and both are worth implementing early. **Standing-authority Tier 3–4 agents require a mandatory preventive control**, not merely detective monitoring; detection after an irreversible action is not a control. And **verification must occur within a defined maximum latency**, because an audit that runs weekly does not constrain an agent acting hourly.

16.3 The Agent Card

The Agent Card is the primary record carrying the enforceable content of an agent's authority.

Table 44. The Agent Card

Field	Requirement
Registered machine identity	Unique per agent, never shared
Authorized tool list	Explicit allow-list, not a deny-list
Action boundaries	Value limits, counterparty limits, rate limits, scope limits
Reversibility per action	Each authorized action classified for reversibility
Escalation thresholds	Conditions that route to a human rather than proceeding
Kill-switch conditions	Mandatory at Tier 3–4; objectively evaluable
Delegation chain	For multi-agent systems: which agents may invoke which
Owner	Named Agent Owner, always human

An allow-list instead of a deny-list is a deliberate choice. Deny-lists fail open: any capability nobody thought to prohibit is permitted. For a component that acts, failing open is the wrong default.

Worked example

Agent: Reconciliation-Agent-02

Identity: svc-recon-agent-02, dedicated, no interactive login

Authorized tools: read ledger; read bank statement feed; post correcting journal entry

Not authorized: approve payment; modify chart of accounts; access customer records

Action boundaries: single entry $\leq 5,000$ currency units; aggregate $\leq 50,000$ per day; entries to reconciliation suspense accounts only

Reversibility: journal entries reversible; audit trail and downstream reporting not

Escalation: any single entry $> 5,000$, any entry outside suspense accounts, any unmatched item older than 30 days

Kill-switch: attempted entry outside authorized account range, or aggregate exceeding daily limit $\rightarrow$ suspend agent, revoke posting permission, alert Agent Owner and AI Governance Lead, preserve state

Owner: Financial Control Manager

Tier: 4 (see Chapter 7, Example C)

Note the aggregate daily limit. It is not required by the framework — the cumulative exposure gap in Chapter 13, §13.4 is unaddressed — and it is exactly the kind of compensating control an implementer should add.

16.4 Multi-agent systems

Multi-agent composition carries the highest governance weight in the framework and the weakest coverage in existing standards.

Three additional questions apply beyond single-agent governance. **Which agents may invoke which others**, recorded as an explicit delegation graph instead of emerging from configuration. **What authority each delegation carries** — whether an invoked agent acts under its own boundary or the caller's, which is the question that determines whether authority can compound. And **what the composite boundary is**: the union of authority reachable through a delegation chain, which is frequently broader than any individual agent's boundary and is what actually needs governing.

[Practice recommendation] Compute and record the composite reachable authority for any multi-agent system, and score D2 and D3 against that composite rather than against the entry-point agent. A chain of individually modest agents can reach an immodest capability.

Chapter 17. Autonomy Change and Its Detection

The rule that autonomy increases require re-authorization, its deliberate narrowing, and the unresolved tension that narrowing created.

17.1 The rule and its narrowing

The original rule was absolute: any autonomy increase is a Major change, always, re-entering risk classification and both G2 and G4.

That rule was narrowed during revision to apply to **boundary-widening changes only** — changes that expand what the agent may do, not changes within an already-approved boundary. The reasoning was practical: treating every configuration adjustment inside an approved envelope as a Major change made agentic systems ungovernable in practice, and a rule that cannot be followed is not followed.

The narrowing is logged as an unresolved trade-off, not a settled improvement. It reopens a gap that audit sampling is expected to cover, and the interaction between the two has never been tested. You are adopting a control whose designers explicitly flagged uncertainty about its calibration.

17.2 What counts as boundary-widening

Table 45.

Change	Boundary-widening?	Classification
New tool added to the allow-list	Yes	Major
Existing tool's action limit raised	Yes	Major
Existing tool's action limit lowered	No	Minor
Human-in-the-loop removed from a path	Yes	Major
Escalation threshold raised	Yes	Major
Escalation threshold lowered	No	Minor
New counterparty added to an approved list	Yes	Material to Major by blast radius
Agent invoked by an additional agent	Yes, if composite authority grows	Major
Model swapped inside an unchanged boundary	No, on autonomy grounds	Per model-change rules
Rate limit increased	Yes, if it enables materially greater aggregate effect	Material to Major

The asymmetry is deliberate. Reducing authority is safe by construction and should not carry governance friction; expanding it is the thing being controlled.

17.3 Making increases detectable

A prohibition whose violation cannot be detected is not a control. Three mechanisms make autonomy expansion visible.

Machine-readable boundaries at G3. The recorded boundary must be in a form comparable against live IAM and tool configuration. Without this, detection depends on someone noticing.

Continuous comparison. The Control Plane compares an agent's live permission set against its G2-approved boundary. Divergence is a Security or Configuration drift signal with a specific, high-value meaning.

Verification latency ceiling. Post-action verification must occur within a defined maximum interval, scaled to tier. For a Tier 4 agent acting continuously, a weekly reconciliation is not verification.

[Practice recommendation] Treat any detected divergence between an agent's live permissions and its approved boundary as an incident rather than a drift finding, regardless of whether the expansion looks benign. The alternative — triaging each case on merit — is how boundaries erode, because each individual expansion is usually defensible.

17.4 What remains unsolved

Two gaps are worth holding in view as you operate agentic systems.

Cumulative exposure. Many sub-threshold actions accumulating between verification cycles are not governed (Chapter 13, §13.4). Compensate with aggregate limits.

The narrowed rule's interaction with sampling. Whether boundary-widening-only classification, backed by audit sampling, catches as much as the original absolute rule is unknown. **[Practice recommendation]** Until evidence exists, weight your Tier 3–4 audit sample deliberately toward agentic systems and toward changes classified as within-boundary. If that sample keeps finding misclassified expansions, revert locally to the absolute rule for agents and record why.

Chapter 18. Standing Up the Control Plane

What to connect first, what to automate, and where automation must stop.

18.1 What the Control Plane is and is not

Seven functions operating as a loop: **Observe** → **Compare** → **Evaluate** → **Alert** → **Recommend** → **Enforce** → **Learn**.

Table 46. What the Control Plane is and is not

Function	What it does
Observe	Collects telemetry, populating actual-state values on metamodel entities
Compare	Checks actual state against approved state recorded in Decision and ADR entries
Evaluate	Assesses drift severity using policy rules and the system's current risk tier
Alert	Notifies the accountable owner, scaled by severity
Recommend	Proposes remediation without necessarily executing it
Enforce	Executes correction, only within the boundaries in §18.4
Learn	Feeds observed patterns back into architecture decisions and pattern quality

A candid framing, carried from the research: **the mechanism itself is not novel**. Cloud security posture management, DevOps observability, and policy-as-code engines already do most of this for infrastructure and security. The framework classifies the Control Plane as a recommended implementation pattern instead of a contribution.

What is specific to IRGF is the comparison target. A drift alert here is not “configuration changed” but “configuration changed relative to a Decision made by a named accountable owner under a stated risk tier.” That is a governance fact rather than a technical one, and it is what makes the alert actionable. Three further elements have no direct equivalent in existing tooling: the AI-specific drift categories, the AI-specific policy rules, and the Learn function's feedback into portfolio and pattern decisions.

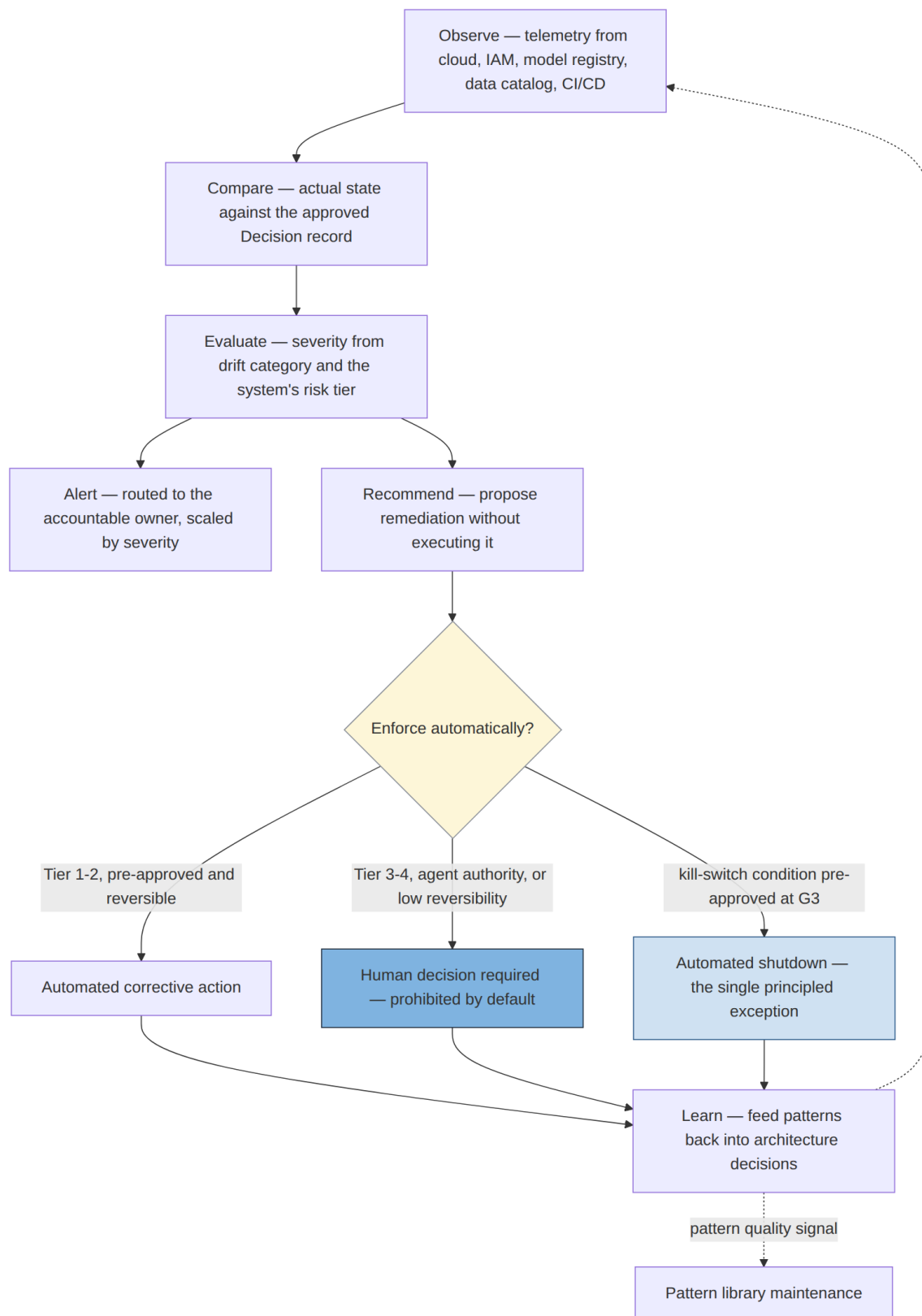


Figure 6. The Control Plane loop, with the automation boundary that constrains Enforce

The practical consequence: you are probably not buying a new platform. You are more likely extending tooling you already run, and adding the governance context that makes its output meaningful.

18.2 Do decision records first

The most common Control Plane failure is sequencing. Organizations connect telemetry before their Decision records are trustworthy, generate a high volume of alerts nobody can act on, and conclude the mechanism does not work.

Compare needs something to compare against. If ADRs are missing, vague, or unowned, every comparison either fails silently or produces a finding with no approved state to reference and no owner to route to.

[Practice recommendation] Do not connect a telemetry source until the corresponding approved-state field exists in your ADRs and is populated for at least the Tier 3–4 estate. Connect model registry telemetry once ADRs name model families. Connect IAM telemetry once agent boundaries are recorded. Sequencing telemetry to record-readiness is the difference between a working Control Plane and an alert firehose.

18.3 Telemetry sources and connection order

Ten source categories, each mapped to what it can evidence.

Table 47. Telemetry sources and connection order

Source	Evidence produced	Metamodel target
Cloud environments, IaC state	Actual deployed infrastructure configuration	Platform, Technology attributes
APIs, service meshes, gateways	Actual integration and routing in effect	API, Service, Integration discipline
IAM and identity providers	Actual granted permissions, human and machine	Agent authority boundary, actual versus approved
Application portfolio and CMDB	Actual inventory and dependency graph	Application existence, dependency map
AI platforms and model registries	Actual deployed model family and version	AI System uses Model, actual versus approved
Data catalogs	Actual grounding sources and classification	AI System grounded by Data Product
CI/CD systems	Deployment events with timestamps	Whether deployment was preceded by a valid Decision
Security tooling	Actual posture and incident signals	Control effectiveness
Policy engines	Policy evaluation results	Direct input to Compare and evidence capture
FinOps	Actual spend against approved budget	Cost drift, often a leading indicator of scale change

Suggested connection order

Connect two or three sources well rather than ten poorly. The order below front-loads governance value.

Table 48. Suggested connection order

Order	Source	Why first
1	AI platform / model registry	Directly answers “is the deployed model the approved one” — the highest-value single comparison
2	IAM	Enables agent boundary comparison, the control with the most consequential failure mode
3	Cloud / IaC	Broad structural and configuration coverage; usually

Order	Source	Why first
		already instrumented
4	Data catalog	Enables grounding-lineage drift, an AI-specific category with no equivalent elsewhere
5	CI/CD	Detects deployment without authorization
6	Policy engine	Turns comparison into enforcement for objectively defined rules
7+	Security, FinOps, APIs, CMDB	Valuable, but each adds volume before it adds new governance meaning

18.4 The automation boundary

Four control types, and a hard line through the middle.

Table 49. The automation boundary

Control type	What it does	Automation default	Exception path
Advisory	Informs a human, takes no action	Always automated	None needed
Detective	Identifies that drift has occurred	Always automated	None needed
Preventive	Blocks an action before it takes effect	Automated where the rule is objectively defined	Documented exception process
Corrective	Actively remediates a detected violation	Tier 1–2 only, pre-approved and reversible actions	Pre-approved kill-switch at Tier 3–4

The corrective line is the one that matters, and it is a permanent design position instead of a maturity gap to close.

Automated corrective enforcement is prohibited by default for Tier 3–4 systems, for any action that would change an agent’s authority, and for any action with low reversibility. The single exception is a kill-switch condition defined and pre-approved at G3 (Chapter 12, §12.3).

This is worth defending internally, because there is persistent pressure to automate remediation as maturity grows. The framework’s position is that maximum maturity means faster and better-evidenced human decisions, not fewer of them. A framework that automated consequential correction for high-tier systems would have abandoned its own human accountability principle.

Preventive controls: objectively defined only

A preventive rule may be automated only where the underlying policy is unambiguous. “Is this model on the approved list” is automatable. “Is this architecture pattern appropriate for this use case” is judgment and must not be.

Every automated preventive control needs a defined exception path. A legitimate edge case blocked with no recourse produces one of two outcomes: the work stops, or someone finds a way around the control. The second is worse than not having the control.

18.5 Policy-as-code: the rules worth writing first

Seven policy domains, with the rules that repay implementation earliest.

Table 50. Policy-as-code: the rules worth writing first

Domain	Machine-checkable rule	Type	Priority
AI model selection	Deployed model family must be on the approved list	Preventive	First. Cheap, unambiguous, high value

Domain	Machine-checkable rule	Type	Priority
Security	No machine identity may hold a permission absent from its approved boundary	Preventive	First. Highest-consequence failure mode
Cloud	Resource must carry an approved ADR reference	Preventive	Second. Forces decision-record discipline
Agent permissions	Live tool and permission set must match the G2-approved record	Preventive + Detective	Second, once agents exist at Tier 2+
Data location	Data above a sensitivity threshold may not reach an endpoint outside an approved jurisdiction	Preventive	Where sovereignty applies
Architecture patterns	System must reference an approved pattern or carry an exception	Preventive	Third. Enables the fast path
Approved technologies	System must be built on an approved platform or carry an exception	Preventive	Third

The first two are the highest-value pair in the framework's automation surface. Both are objectively evaluable, both have unambiguous failure semantics, and both catch the failure modes most likely to cause real harm.

[Practice recommendation] Deploy every new preventive rule in report-only mode first, for at least one full delivery cycle. The volume of legitimate violations you find is the calibration data you need before enforcement, and it is invariably higher than expected.

18.6 Evidence-as-code

Roughly two-thirds of the framework's minimum evidence set can be captured automatically as a byproduct of normal operation, not assembled manually.

Table 51. Evidence-as-code

Evidence item	Automatable	Mechanism
ADR reference at G2	Yes	Linked at deployment via IaC tagging
Data lineage and classification at G2	Yes	Pulled from data catalog
Pattern conformance result at G2	Yes	Policy evaluation output
Technical evaluation results at G3	Yes	Captured from the evaluation pipeline
Control-to-risk mapping at G3	Partial	Mapping is judgment; current effectiveness is capturable
Human oversight confirmation at G3	No	Requires confirming an interaction design was implemented
Change classification at G4	Partial	Classification is judgment; delta evidence is capturable
Retention and deletion at G5	Yes	Data platform deletion logs
Access revocation at G5	Yes	IAM deprovisioning logs
Knowledge preservation at G5	No	Inherently a human assessment

The remaining third requires human input by nature. That is a boundary of the same kind as the corrective-control line, not a gap awaiting better tooling.

Evidence-as-code is where real burden reduction lives. Chapter 4, §4.4 noted that gate consolidation did not reduce evidentiary burden; automated capture does. **[Practice recommendation]** Measure the proportion of gate evidence captured automatically as a standing metric. It is the most honest available indicator of whether governance is getting cheaper.

Chapter 19. Drift Triage

Eight categories, what each means, who owns it, and how to respond without drowning in alerts.

19.1 The eight categories

Table 52.

#	Category	Definition	Detection	Typical owner
1	Structural	Actual inventory diverges from the architecture repository	CMDB and cloud inventory versus repository	Architect
2	Configuration	An approved component's configuration changed without a Decision record	IaC state versus last-approved configuration	System Owner
3	Integration and Dependency	Actual integration patterns diverge from approved or from the dependency map	API and service mesh state	Architect
4	Security	Actual posture diverges from the approved security baseline	IAM, security tooling	Security Architect
5	Data Lineage and Governance	Grounding sources or classification changed outside the governed process	Data catalog versus approved lineage	Data Owner
6	AI-Model Behavioral	Model output behavior changed, through vendor update or distribution shift	Technical evaluation signals, not infrastructure telemetry	ML engineering with Risk
7	Policy and Regulatory	Applicable policy or regulation changed, independent of the system	Regulatory monitoring feed	Risk and Compliance
8	Cost	Spend diverges from approved budget	FinOps	System Owner

Two categories from earlier drafts were deliberately removed. **Capability drift** — a business capability's actual maturity diverging from its recorded state — is not observable by an API call and is handled by periodic capability re-baselining. **Strategic drift** — enterprise strategy changing such that approved architecture no longer aligns — requires human judgment, not comparison, and is handled by periodic intent review. Claiming either as telemetry-detectable would overstate what the mechanism can do.

19.2 The two meanings of “data drift”

This distinction prevents a recurring misrouting.

Statistical distribution shift in a model's input data is a model-performance concern. It belongs to Technical AI Evaluation and appears here as category 6, AI-Model Behavioral drift.

Ungoverned change to grounding sources is an architecture and lineage concern. It is category 5.

Conflating them means an architecture drift alert fires for a model-quality issue, or a model-quality signal is routed to an architect who cannot act on it. Keeping them separate is one of the framework's few original mechanisms, and its value is diagnostic: when an AI system's behavior changes, the first question is whether the model changed or its grounding changed, and separated signals answer it immediately.

19.3 Policy and regulatory drift is structurally different

Categories 1 to 6 and 8 detect actual state diverging from approved state. Something moved on the ground.

Category 7 detects the opposite: approved state no longer satisfying an external requirement that itself changed, with nothing about the system having moved at all. A system can become non-compliant overnight without a single deployment.

This requires a different mechanism — a regulatory and policy monitoring feed into Evaluate, not a telemetry comparison. **[Practice recommendation]** Maintain the Regulatory Overlay Reference as a live record mapping each applicable regime to the systems it touches, and treat a change in any mapped regime as a drift event that re-opens the affected systems' compliance assurance. Without that mapping, regulatory change requires someone to remember which systems are affected.

19.4 Triage: severity and routing

Severity is a function of the drift category and the affected system's tier, not of the category alone.

Table 53. Triage: severity and routing

Tier	Structural / Config / Cost	Security / Data Lineage	Behavioral	Policy / Regulatory
Tier 1	Digest to owner	Direct to owner	Periodic review	Batch review
Tier 2	Direct to owner	Direct to owner, 10 working days	Direct to owner	Batch review
Tier 3	Direct, 5 working days	Direct, escalates at 5 days	Triggers risk re-score	Compliance assurance re-run
Tier 4	Direct, same business day	Immediate, escalates same day	Triggers risk re-score and possible G4	Immediate compliance review

Agent boundary divergence is the exception to this table: treat it as an incident at any tier (Chapter 17, §17.3).

19.5 Managing alert volume

A Control Plane that produces more alerts than the organization can act on trains people to ignore it, which is worse than having no Control Plane because it creates a false sense of coverage.

Four practices keep volume tractable.

Start narrow. Two or three telemetry sources against the Tier 3–4 estate produces a volume a small team can absorb and learn from.

Report-only first. Every new comparison runs in report-only mode for a cycle. The initial volume tells you whether your approved-state records are accurate; a flood usually means the records are wrong, not that the estate is.

Suppress known-and-accepted, visibly. Where drift is accepted, it becomes a recorded exception with an expiry, not a suppressed alert. Suppression without a record is how coverage silently erodes.

Use the Learn function. A pattern generating repeated drift across multiple initiatives is a pattern problem, not a compliance problem. Route it to pattern-library maintenance rather than raising the same finding repeatedly against different teams. This is the loop that conventional periodic governance cannot close.

19.6 From drift to decision

A drift signal is not a decision. The path from one to the other has four possible endpoints, and making them explicit prevents alerts accumulating in an unresolved state.

Table 54. From drift to decision

Endpoint	When	Record produced
Revert	The divergence was unintended and the approved state is still right	Drift record closed; no change record
Ratify	The divergence is an improvement or a necessary adaptation	Change record, classified; G4 if Material or Major
Except	The divergence must persist temporarily and cannot be ratified yet	Architecture Exception, time-boxed
Escalate	The divergence indicates the tier or the approval was wrong	Risk re-score; possible G3 re-authorization

[Practice recommendation] Enforce a maximum age for open drift findings by tier, and report the aged backlog. Unresolved drift is the clearest available measure of whether governance is keeping pace with delivery.

Chapter 20. The Governance Bodies

Two standing bodies, not six, what each decides, and the scaling limit you will hit.

20.1 Why two bodies and not six

Six candidate bodies are commonly proposed for this problem space: an Architecture Review Board, an AI Governance Council, a Data Council, a Security Council, a Model Risk Committee, and a Transformation Council. IRGF requires two.

Table 55. Why two bodies and not six

Candidate body	Disposition	Reasoning
Architecture Review Board	Retained, mandate expanded to cover AI as a fifth domain	AI decisions are architecture decisions; a parallel board doubles review cycles
AI Governance Council	Not created	Folded into expanded ARB membership
Data Council	Not created	External if it exists; the Data Owner role is the interface
Security Council	Not created	External if it exists; the Security Architect role is the interface
Model Risk Committee	Not created	Function absorbed into the AI Governance Body, active only at Tier 3–4
Transformation Council	Not created, out of scope	The Architecture Input Package is the interface

The reasoning behind the first row is the important one. Standing up an AI approval council alongside an architecture board creates exactly the second bureaucracy the framework was designed to avoid: every AI-enabled initiative queues twice. Expanding the ARB’s mandate and membership achieves the same coverage without the duplication.

Neither body is a parallel governance stack. Both operate as the architecture and AI arm of whatever enterprise IT governance structure already exists.

20.2 The Architecture Review Board, expanded

Mandate: approve architecture decisions across all five primary domains, including AI. Ratify reference patterns. Resolve dependency conflicts that solution-level work cannot.

What changes on adoption: AI architecture expertise becomes a membership requirement rather than an occasional invitee. The board’s remit explicitly covers AI sourcing, grounding, and authority-boundary decisions at Tier 3–4.

What it decides: G2 at Tier 3–4; pattern library additions and retirements; TG0 bypass findings (Chapter 22, §22.3).

What it does not decide: deployment authorization. That belongs to the AI Governance Body, and keeping the two separate preserves a distinction between “is this the right architecture” and “is this safe to run.”

20.3 The AI Governance Body

Mandate: authorize deployment and material change for the highest-risk AI systems. Absorbs the independent-challenge function a model risk committee would otherwise provide.

Activation: Tier 3–4 only. It does not review every AI system, which is what keeps two bodies viable.

What it decides: G3 and G4 at Tier 3–4; residual risk acceptance; autonomy grants and increases; kill-switch conditions; concurrence on Tier 3–4 portfolio funding (§20.5).

Membership should span risk, architecture, security, data, and the business. The independent-challenge function only works if the body contains people who did not build the thing.

20.4 Federation is the default; the bodies are the exception

The most common implementation error is treating the two bodies as the primary mechanism. They are not. **Most decisions never reach them.**

Tier 1–2 work is self-service: architects self-certify G2 against unmodified patterns, owners approve G3. That federation is only safe because automated controls check self-certifications continuously against the same rules a reviewer would apply. Federation without the Control Plane is delegation without verification, and the framework does not support it.

The bodies exist for Tier 3–4 work, disputes, exceptions beyond delegated authority, and alerts that were not resolved locally. If your ARB agenda is full of Tier 1 systems, either your tiering is wrong or your federation is not real.

20.5 The funding concurrence right

One deliberate exception to the general rule that architecture advises and portfolio decides: **Tier 3–4 AI-relevant initiatives require AI Governance Body concurrence, not merely input, before portfolio may approve funding.**

For everything else — Tier 1–2, or non-AI — portfolio retains full authority and consumes architecture input as advisory.

The exception exists to close a specific bypass: an initiative funded and committed before anyone assessed whether its AI component was governable. By the time architecture sees it, the commitment is made.

This is logged as an unresolved tension, not a clean win. Independent review found it concentrates real power in a body with no equal-weight business counterpart able to contest it. It was retained because removing it reopens the bypass. If you adopt it, adopt the mitigation too: **[Practice recommendation]** require the AI Governance Body to record a reason whenever concurrence is withheld, and report withheld-concurrence decisions to the Enterprise Governance Sponsor quarterly. A veto that is never examined becomes a veto that is never questioned.

20.6 The scaling limit you will hit

The two-body model has a known limitation: **it does not scale on the volume axis within a single jurisdiction.**

The framework addressed geography by adding a conditional seventh governance level — Region or Legal Entity — activated only where jurisdiction-specific regulatory exposure or

country-level Tier 3–4 volume justifies it. It did not address the case where a single jurisdiction simply produces more Tier 3–4 work than two bodies can review.

You will hit this if AI adoption succeeds. Four responses, in rough order of preference.

Improve tiering accuracy first. A body overwhelmed by Tier 3 cases is sometimes a body receiving over-tiered work. Check the distribution against incident evidence before adding capacity.

Expand the pattern library. Every additional pre-approved pattern moves work from review to self-certification. This is the highest-leverage response and the one most often neglected.

Delegate by decision type, not by tier. Some Tier 3 decisions are routine within a well-understood pattern. Delegating specific decision types to a named senior architect, with the body reviewing a sample, preserves the tier while reducing the queue.

Split by domain, not by geography. If volume is concentrated in one business area, a domain-specific sub-body reporting into the main one is preferable to a general capacity increase, because it preserves consistency of judgment.

What does not work is lengthening the queue. A governance body with a six-week lead time is routed around, and the routing-around does not appear in any metric.

Chapter 21. Roles, Decision Rights, and RACI

Who decides what, and the two role problems the framework has not solved.

21.1 The role catalogue

Table 56.

Role	Accountable for	Key authority	Owns
Business Sponsor / Capability Owner	Business value delivered	Approve or reject use-case qualification	Use-Case Canvas, Capability Record
AI / Enterprise Architect	Architecture quality and pattern conformance	Self-certify Tier 1–2 G2; author ADRs	ADRs, pattern library
Data Owner	Data classification and lineage accuracy	Approve or reject data readiness	Data Lineage and Sensitivity Records
AI / System Owner	System behavior, conformance, retirement	Responsible for evidence at every gate	Configuration and change records
Risk and Compliance Lead	Risk scoring accuracy, control effectiveness	Assign and re-assign tier; countersign scores	Risk Classification Records, Control Matrix
Security Architect	Security posture of governed components	Consulted at all gates; can block on unresolved findings	Security control records
Procurement	Vendor due-diligence evidence	Consulted at S1–S2 for procured AI	Vendor evidence records
Architecture Review Board	Architecture conformance enterprise-wide	Approves Tier 3–4 G2; ratifies patterns	Approved pattern library
AI Governance Body	Deployment authorization quality at Tier 3–4	Approves Tier 3–4 G3/G4; funding concurrence	Deployment authorization records
Enterprise Governance Sponsor	Framework integrity; cross-body conflict	Ratifies principles; resolves conflicts	Central principles
Transformation / PMO (external)	Portfolio delivery outcomes	Prioritization and funding	Portfolio roadmap (external)

21.2 Decision rights by tier

Table 57.

Decision	Tier 1	Tier 2	Tier 3	Tier 4
Approve architecture (G2)	Architect self-certifies	Architect self-certifies, ARB notified	ARB	ARB, Enterprise Sponsor briefed
Approve deployment (G3)	System Owner	System Owner, AI Governance Lead notified	AI Governance Body	AI Governance Body + Enterprise Sponsor
Approve a new foundation model for the enterprise list	Architect proposes, ARB ratifies — applies once, enterprise-wide, not per tier			
Change a system's risk tier	Risk Lead, self-service if downward or lateral	Risk Lead; AI Governance Body notified if upward	AI Governance Body	AI Governance Body + Enterprise Sponsor briefed
Authorize or increase agent autonomy	Not applicable; an increase re-triggers classification	AI Governance Body	AI Governance Body	AI Governance Body + Enterprise Sponsor

Decision	Tier 1	Tier 2	Tier 3	Tier 4
Approve an exception	System Owner, ≤ 90 days	AI Governance Lead, ≤ 90 days	AI Governance Body, ≤ 90 days, renewal requires re-justification	AI Governance Body + Enterprise Sponsor, ≤ 90 days, no auto-renewal
Retire a system (G5)	Owner + Data Owner	Owner + Data Owner	Adds AI Governance Lead	Adds AI Governance Body
Accept residual risk	System Owner	AI Governance Lead	AI Governance Body	AI Governance Body + Enterprise Sponsor

The foundation-model row is deliberately not tier-differentiated. A model either joins the approved list or it does not; that decision is made once, centrally, not re-litigated by every use case that wants to use it.

21.3 The AI Build Lifecycle RACI

Table 58.

Stage / Gate	Sponsor	Architect	Data Owner	System Owner	Risk Lead	Security	ARB	AI Gov Body
S0–S1 Opportunity, Qualification	R	C	C	—	I	—	—	I (T3–4)
S2 Risk Classification	C	C	C	—	A/R	C	—	I (T3–4)
G1	A	C	I	—	R (T3–4)	—	—	C (T3–4)
S3 Architecture	I	A/R	C	C	I	C	C (T3–4)	—
S4 Data Readiness	I	C	A/R	C	I	C	—	—
G2	I	R	R	C	C	C	A (T3–4)	—
S5 Build / Acquire	I	C	I	A/R	—	C	—	—
S6 Assurance	I	C	I	R	A	C	—	—
G3	I	C	I	R (A at T1–2)	A (T3–4)	C	—	A (T3–4)
S7 Deployment Authorization	I	I	I	R	C	C	—	A (T3–4)
S8 Runtime and Monitoring	I	I	I	R	A	C	I	I (T3–4)
S9 Change	I	C	C	R	C	C	—	I (Major, T3–4)
G4	I	C	C	R	A (T1–2)	C	C (T3–4)	A (T3–4)
S10 Retirement	I	I	A/R	R	C	I	—	C (T3–4)
G5	I	I	A	R	C	I	—	C (T3–4)

R responsible, A accountable, C consulted, I informed. Tier qualifiers make involvement conditional. Blanks indicate no defined involvement.

A deliberate design check is visible in this table: no role is accountable at more than roughly a third of stages, and the Enterprise Governance Sponsor is accountable nowhere. Making every senior leader accountable for everything is an anti-pattern the framework actively guards against.

21.4 Two unsolved role problems

Security Architect is consulted, never accountable. Across the entire RACI, the Security Architect advises and can block on unresolved findings, but is never the accountable party. Independent review found this under-weighted relative to security's actual role in agent and coding-agent failure modes, and it remains an open gap.

[Practice recommendation] Elevate Security Architect to co-accountable with the Risk Lead at G3 for any system with agentic capability or external exposure. This departs from the specification, and you should record it as a local variation. The framework's own limitations register flags this as a gap worth closing; closing it locally is defensible.

Procurement arrives late in most organizations. The role is named as consulted at S1–S2, but in practice purchasing decisions for AI-embedded software are frequently made outside the technology function entirely. Naming the role does not create the routing. **[Practice recommendation]** Add an AI-content question to the standard procurement intake and route any positive answer to S0, regardless of contract value. Value thresholds are the wrong filter, because embedded AI arrives in inexpensive software.

Chapter 22. Escalation and Exceptions

What happens when something does not fit, and how to keep exceptions from becoming permanent.

22.1 The escalation model

Table 59.

Trigger	Path	Timeliness
Risk tier crosses upward	S2 → AI Governance Body if resulting tier is 3–4	Within the current change cycle
Control Plane Tier 3–4 alert	Alert → System Owner → AI Governance Body if unresolved	Tier 4 same business day; Tier 3 within 5 working days
Gate rejection or dispute	Product or Solution → ARB (architecture) or AI Governance Body (deployment)	Within one standing-body cycle
Exception request beyond delegated authority	Requester → next authority per the decision rights matrix	Time-boxed; no indefinite pending state
TG0 bypass detected retroactively	Control Plane detects a repository entry with no TG0 record → ARB → Enterprise Sponsor if disputed	Within 10 working days of detection
Cross-body conflict	Either body → Enterprise Governance Sponsor	Within one enterprise-level cycle

22.2 Making escalation usable

Escalation paths fail in predictable ways, and each has a straightforward counter.

No defined arrival state. An escalation that lands in a body’s general queue is a delay, not a decision. Escalations should arrive with the decision required stated explicitly and the evidence attached.

No timeliness enforcement. A path with a stated window and no consequence for exceeding it drifts. **[Practice recommendation]** Report breached escalation windows as a governance metric, not as an individual performance matter.

Escalation as punishment. Where escalating is treated as a failure by the escalating team, people stop doing it. Escalation should be routine and unremarkable, and the bodies should say so.

22.3 The TG0 bypass path

One escalation deserves explanation because it closes a gap the framework’s own design left open.

TG0 is the initiative qualification checkpoint in the transformation lifecycle (Chapter 23). Nothing in the checkpoint itself prevents an initiative from bypassing it — a team can simply start building.

Detection is retroactive and telemetry-based: the Control Plane finds an architecture or AI System repository entry with no corresponding TG0 record. Response authority sits with the ARB, which has cross-functional standing that neither portfolio nor the AI Governance Body holds alone. Disputes go to the Enterprise Governance Sponsor.

This is worth implementing early, because it is the only mechanism catching work that entered the estate without entering governance, and that work is disproportionately likely to be poorly governed.

22.4 Exception governance

An exception is a recorded, time-boxed decision to proceed without meeting a requirement. It is a legitimate mechanism, and a framework without one accumulates silent non-compliance instead.

Every exception requires four things: the specific requirement being excepted; a stated justification; a compensating control where one exists; and a hard expiry date.

Every exception is time-boxed to a maximum of 90 days, regardless of tier. Tier 4 exceptions have no renewal path at all: at expiry the system returns through full G3 re-authorization instead of receiving a renewed exception.

Keeping exceptions honest

Three mechanisms, in increasing order of importance.

A single register. Exceptions recorded in team-local documents are invisible. One register, one format.

Visible aggregate reporting. Exception volume appears in portfolio health reporting as “initiatives operating under exception.” Exception count is a governance signal; hiding it converts workarounds into invisible accumulated risk.

Expiry that actually fires. [Practice recommendation] Automate expiry notification to the approving authority, not only to the requester. An exception whose expiry is known only to the team benefiting from it will be renewed by inattention.

When to grant, and when to say no

An exception is appropriate where the requirement is genuinely inapplicable to the case, or where the risk it addresses is covered by a different control. It is not appropriate as a substitute for doing the work, and the tell is the justification: “we do not have time to produce the lineage record” is not a justification, while “this system’s grounding source is a public reference corpus with no classification applicable under our scheme” is.

Where a team repeatedly requests the same exception, the requirement is probably wrong or the pattern library is missing something. **[Practice recommendation]** Review recurring exception types quarterly as candidates for either a framework variation or a new reference pattern. Repeated exceptions are design feedback, not compliance failures.

Chapter 23. The Transformation Interface

What architecture owes portfolio governance, what it must not try to own, and the checkpoint that connects transformation work to the AI lifecycle.

23.1 Shallow by design

IRGF governs transformation shallowly and deliberately. The framework claims **visibility of, not ownership over**, transformation and workforce change. This is a scope decision, not an oversight, and holding the line is a condition of the framework remaining architecture governance rather than becoming an unbounded review function.

The distinction is best expressed as depth per link in the chain.

Table 60. Shallow by design

Link	Governance depth	Why
Enterprise intent and objectives	Deep	Traceability starts here
Business capability	Deep	The unit against which value is assessed
Transformation theme	Shallow	Business framing, not architecture
Portfolio	Shallow with input right	Architecture informs; portfolio decides
Initiative	Shallow with a mandatory checkpoint	TG0 is the one enforced touchpoint
Architecture and AI enablement	Deep	Squarely architecture's remit
Delivery and organizational change	External input interface only	PMO and change management own this entirely
Outcome and benefit	Deep	Closes the traceability chain

The delivery row was relabeled during the framework's revision from "Delivery and Organizational Change" to **External Input Interface**, correcting an appearance of ownership the framework never actually claimed. If your implementation finds architecture reviewing delivery plans or change management approaches, scope has crept.

23.2 The transformation lifecycle

Table 61.

Stage	Name	Architecture's role
T0	Purpose and Strategic Alignment	Informed
T1	Capability Gap Identification	Consulted; supplies capability and readiness assessment
T2	Initiative Qualification (TG0 checkpoint)	Mandatory input on specific questions
T3	Portfolio Prioritization	Consulted on dependency conflicts; AI Governance Body concurrence for Tier 3–4
T4	Architecture and AI Enablement	Accountable; spawns S0 where AI is involved
T5	External Input Interface	None; PMO owns
T6	Benefits Realization and Review	Consulted; supplies outcome evidence
T7	Capability Re-Baseline	Consulted; loops to T1

T4 is the junction. Where a transformation initiative involves AI, T4 spawns the AI build lifecycle at S0, and the two lifecycles run in parallel from that point with the AI system governed by S0–S10 while the initiative continues through T5–T7.

23.3 The Architecture Input Package

The framework does not own prioritization. What it defines is a structured package that portfolio governance must consume, without being obliged to defer to it.

Table 62. The Architecture Input Package

Portfolio decision	Architecture supplies	Architecture does not own
Initiative prioritization	Capability gap severity and architecture readiness	The business value judgment
Investment sequencing	Dependency graph across shared platform, data, and AI components	Funding allocation and timing
Dependency identification	Explicit flags where two initiatives would build conflicting or duplicative architecture	Resolution of the conflict
Reuse	Existing patterns or platform capability an initiative could use instead of building	Whether the recommendation is accepted
Platform strategy	Assessment of build-on-existing versus justify-new	The investment decision
Technical debt	Visible register of which initiatives increase or reduce debt	The trade-off decision

The pattern is consistent: architecture surfaces structured information at each decision point and does not make the decision. Whether an input-only role is strong enough to prevent poor outcomes at portfolio level is an open question in the research, and the funding concurrence right (Chapter 20, §20.5) is the single bounded exception.

23.4 TG0 and its enforcement gap

TG0 is the initiative qualification checkpoint at T2. Architecture input is mandatory on the questions concerning architecture readiness, dependency, reuse, and AI relevance.

TG0 has a known enforcement gap: nothing in the checkpoint itself prevents an initiative from bypassing it. A team can begin building without ever appearing at T2.

The framework's answer is retroactive detection rather than prevention: the Control Plane identifies architecture or AI System repository entries with no corresponding TG0 record, and routes the finding to the ARB (Chapter 22, §22.3). This is a detective control on a preventive gap, and it should be understood as such. Detection happens after the work has started, which limits what the response can achieve.

[Practice recommendation] Pair the detective control with a funding-side check. Where finance or portfolio systems can require a TG0 reference before releasing budget, the gap closes at the point it matters. This is outside the framework's scope to mandate and inside most organizations' ability to implement.

23.5 Benefits and the value scorecard

Benefits realization closes the traceability chain: an outcome measured against the baseline set at S1.

The framework tracks five metric families and **deliberately does not composite them into a single index.**

Table 63. Benefits and the value scorecard

Family	Examples
--------	----------

Family	Examples
Business	Revenue impact, productivity, time-to-market, customer experience, operational efficiency
Transformation	Capability maturity movement, modernization progress, process simplification, adoption
Architecture	Reuse rate, interoperability, standardization, technical debt, complexity
AI	Model performance, automation benefit, human augmentation, decision quality
Risk	Incidents, violations, control effectiveness, model failures, security events

Refusing to composite is a design choice with a specific purpose: a single index lets high reuse mask low benefit realization, or strong delivery mask rising risk. Five numbers that must each be looked at produce better conversations than one number that can be optimized.

Chapter 24. The Artifact Set

Sixteen primary records, four derived views, and the non-duplication rule that keeps them from going stale.

24.1 The rationalization test

Every candidate artifact was tested against one question: **what decision would become impossible or materially weaker without it?** Artifacts failing the test were rejected outright or reclassified as derived views compiled from records that pass.

Three candidates were rejected entirely. An Enterprise Transformation Map duplicates a portfolio roadmap the framework does not own. An AI Capability Map is a filtered view of the business capability map. A Model Card is ML engineering documentation; only a reference to it is architecture-relevant.

Four were reclassified from records to views: Business Capability Map, AI System Card, Benefit Map, and Assurance Dashboard. Each is genuinely valuable and none should be independently authored.

24.2 The non-duplication rule

Every primary record has exactly one authoritative owner and one location. Every derived view is compiled by reference and never re-entered.

This is the rule that determines whether your governance data stays accurate. The AI System Card illustrates why. It is the single-pane summary practitioners most want: tier, owner, architecture, controls, authorization, open changes, open exceptions, assurance status. Every one of those fields already exists in another record.

Maintaining it by hand means every field has two homes, and the copy drifts from the source within weeks. Governance data that is confidently displayed and quietly wrong is worse than data that is visibly absent, because nobody checks it.

[Practice recommendation] If you cannot compile the AI System Card automatically, do not create it. A manually maintained version will be out of date at the moment it matters most.

24.3 The primary records

Table 64.

#	Record	Purpose	Owner	Created at
1	Capability Record	Baseline and target maturity, gap, dependencies, AI-enablement flag	Sponsor with Architect	T1
2	AI Use-Case Canvas	Business justification and provisional risk signal	Business Sponsor	S1
3	TG0 Initiative Qualification Record	Evidence for the transformation qualification questions	PMO, with mandatory architecture input	T2
4	Architecture Decision Record	Architecture decision and rationale, AI-extended	Architect	S3

#	Record	Purpose	Owner	Created at
5	Reference Pattern Library entry	Pre-approved pattern enabling self-certification	Architect	On pattern proposal
6	Data Lineage and Sensitivity Record	Grounding lineage and classification	Data Owner	S4
7	Risk Classification Record	D1–D5 scores, computed tier, override flag	Risk Lead	S2
8	Control Matrix	Control-to-risk-to-standard mapping	Risk Lead	S6
9	AI Assurance Summary	Pass/fail evidence reference from technical evaluation	System Owner	S6
10	Agent Card	Agent identity, tools, boundaries, kill-switch	System Owner with Security	S3
11	Deployment Authorization Record	The G3 decision	System Owner or AI Governance Body	S7
12	AI Change Record	Change classification and evidence	System Owner	S9
13	Architecture Exception	Time-boxed exception	Requester	Any point
14	Benefit Record	Realized value against baseline	Sponsor	T6
15	Drift/Alert Record	Control Plane output	System-generated	On detection
16	Regulatory Overlay Reference	Mapping of applicable regimes to affected systems	Risk Lead	On adoption, maintained

Four derived views compile from these: Business Capability Map, AI System Card, Benefit Map, Assurance Dashboard.

Three optional artifacts improve decision quality without being load-bearing: Technical Debt Register entry, Pattern Retrospective, and Stakeholder Map.

24.4 Worked templates

Risk Classification Record

Table 65.

Field	Content
Purpose	Record D1–D5 scoring and the resulting tier
Owner	Risk and Compliance Lead
Trigger	S2 initial; any Material or Major change; any risk-relevant drift alert; annual for Tier 3–4
Mandatory fields	D1–D5 scores each with a one-line justification; D5 sub-scores for sensitivity and uncertainty; computed Impact and Control Deficit; resulting tier; safety-override flag; countersignature with date and any adjustments made
Approval	Self-service for downward or lateral change at Tier 1–2; AI Governance Body for any upward change at Tier 3–4
Update	Re-scored on every Major trigger; history preserved, never overwritten
Retention	Life of system plus 3 years
Related	Use-Case Canvas (input), Change Record (trigger), Control Matrix (downstream)

Example. *Claims triage assistant. D1=3 (misrouting delays a claim and may disadvantage a claimant). D2=2 (proposes routing, adjuster approves each). D3=2 (routing reversible; elapsed delay is not). D4=3 (all claims; feeds claims system of record). D5=3 (sensitivity 3 restricted personal data; uncertainty 3 generative; composite = higher of the two). Impact = 3. Control Deficit = 2. Tier 2 by matrix. No override. Countersigned by Risk Lead, D3 raised from 1 to 2 — original score reasoned from mechanism reversibility.*

Architecture Decision Record, AI-extended

Table 66.

Field	Content
Purpose	Record an architecture decision and rationale, with AI-specific criteria
Owner	AI / Enterprise Architect
Trigger	S3 for any new or materially changed AI System, Application, or Platform component
Mandatory fields	Decision statement; alternatives considered; rationale; pattern conformance result or documented deviation; model or service named; hosting pattern; grounding sources named; authority boundary for agentic systems; risk tier at decision; accountable owner. For procured systems: vendor evaluation evidence, model update policy, data processed and location, contractual audit rights
Approval	G2 — self-certified at Tier 1–2 against the pattern library; ARB at Tier 3–4
Update	Superseded, never edited; supersession references the prior ADR
Retention	Life of component plus 3 years
Related	Pattern library (checked against), Risk Classification (referenced), Data Lineage (referenced)

Example. ADR-0142. Use Enterprise RAG pattern v2, unmodified. Model: [approved list entry], version policy pinned minor with vendor patch auto-accepted. Hosting: vendor API, EU region. Grounding: Data Product PD-114 (Policy Documents, Restricted) only. Alternatives: fine-tuned model rejected on recurring revalidation cost; no-AI rejected, rule-based routing tested at 61% accuracy against a 90% requirement. Pattern conformance: conformant. Risk tier at decision: Tier 2. Owner: A. Mensah.

Agent Card

Fields and a worked example appear in Chapter 16, §16.3.

Architecture Exception

Table 67.

Field	Content
Purpose	Record a time-boxed decision to proceed without meeting a requirement
Owner	Requesting role
Mandatory fields	The specific requirement excepted; justification; compensating control if any; hard expiry date; approving authority
Approval	Per decision rights; always ≤ 90 days; no auto-renewal at Tier 4
Update	Never silently renewed; expiry notifies the approving authority
Related	Feeds portfolio health reporting as “initiatives operating under exception”

AI System Card, derived view

Table 68.

Field	Content
-------	---------

Field	Content
Purpose	Single-pane governance status, compiled not authored
Owner	No independent owner; the System Owner is accountable for the underlying records
Fields, all by reference	Current ADR; current tier; Control Matrix status; Agent Card if applicable; latest Deployment Authorization; open Change Records; open Exceptions; latest assurance status
Approval	Not applicable; a compiled view has no approval step
Update	Real-time or near-real-time from evidence-as-code capture

24.5 Tier-scaled documentation depth

Documentation depth scales with tier. This is where much of the framework's proportionality actually lives.

Table 69. Tier-scaled documentation depth

Record	Tier 1	Tier 4
ADR	Architect self-certified, minimal narrative	Full ARB-reviewed rationale
AI Assurance Summary	Pass or fail reference	Full underlying evaluation reference chain
Agent Card	Kill-switch field not required	Kill-switch conditions mandatory and separately approved
Risk Classification	Countersigned, brief justifications	Countersigned, full justification per dimension, annual re-score
Control Matrix	Mapped controls listed	Mapped controls with current effectiveness evidence

24.6 Where to keep records

The framework does not mandate tooling. Three properties matter more than product choice.

One authoritative location per record type. Records split across a wiki, a ticketing system, and a spreadsheet cannot be compiled into derived views, which means the views get maintained by hand, which returns you to §24.2.

Machine-readable where the Control Plane must read it. Model family, hosting pattern, grounding sources, authority boundaries, and tier need to be fields, not prose in a document.

Referenceable identifiers. Every record needs a stable identifier that other records cite. This sounds trivial and is the thing most often missing; without it, the traceability chain is a set of documents that mention each other by name.

Chapter 25. Maturity Assessment

Five levels defined by adopted mechanisms, not narrative, and how to place yourself honestly.

25.1 The levels

Each level is defined by which named mechanisms are actually in use. That makes self-assessment falsifiable: you either run independent countersignature or you do not.

Table 70. The levels

Level	Defining characteristics
L1 Fragmented	No named AI governance role. Ad hoc process, no risk tiering. No shared artifacts, no telemetry. AI not recognized as an architecture domain
L2 Controlled	Part-time AI Governance Lead. S0–S2 tiering on new initiatives. G1–G3 informal. Manual artifact tracking. Single informal review body. Risk Classification Records exist but are unaudited
L3 Integrated	Full role catalogue staffed. Full G1–G5. Partial Control Plane with manual and some automated observation. ARB expanded to five domains, AI Governance Body stood up, independent countersignature adopted . Sixteen-record artifact set maintained. Value scorecard in use
L4 Adaptive	Procurement integrated. Iterative G2 sequencing in practice. Real Control Plane with automated observe, compare, evaluate. Machine-readable G3 boundaries enabling actual autonomy-change detection. Audit sampling of Tier 1–2 running. Regulatory Overlay Reference populated. Tier scores crosswalked to the enterprise risk register
L5 Continuously Assured	Roles shift from transaction review to exception handling and anchor calibration. Evidence-as-code throughout. Full telemetry. Enforce active for objectively defined preventive rules. Patterns cover most new builds. Real incident data recalibrating scoring anchors. Metrics validated against outcomes rather than asserted

Two honest caveats. **No organization has been observed progressing through these levels;** the model is asserted design, not observed practice. And **Level 5 is not autonomous governance.** The conventional label for a top maturity level is rejected here as inconsistent with the framework's own design, which permanently prohibits automated corrective enforcement for Tier 3–4 systems. Maturity means faster and better-evidenced human decisions, not fewer of them.

25.2 The single most useful discriminator

If you assess only one thing, assess whether **independent countersignature of risk scores is actually happening.**

It is the boundary between L2 and L3, and it is the control the framework's own review identified as most load-bearing. An organization with a full artifact set, five gates, and two governance bodies but unaudited self-scoring is at L2 with L3 decoration. Every tier-scaled control it operates inherits the accuracy of a number nobody checked.

25.3 A self-assessment instrument

Answer yes only where the mechanism is in routine use, not where it exists on paper.

Table 71. A self-assessment instrument

#	Question	Level indicated
1	Is there a named person accountable for AI governance?	L2

#	Question	Level indicated
2	Do new AI initiatives receive a risk tier before architecture work begins?	L2
3	Is AI represented as a domain in architecture governance?	L2
4	Are risk scores countersigned by someone independent of delivery?	L3
5	Do all five gates operate, with authority scaled by tier?	L3
6	Is the full primary record set maintained with named owners?	L3
7	Is any deployed state automatically compared against approved state?	L3
8	Are agent authority boundaries recorded machine-readably?	L4
9	Is Tier 1–2 self-certification audit-sampled?	L4
10	Is procurement routed into the lifecycle for embedded AI?	L4
11	Are tier scores connected to the enterprise risk register?	L4
12	Is most gate evidence captured automatically?	L5
13	Have scoring anchors been recalibrated against your own incident data?	L5
14	Do reference patterns cover the majority of new builds?	L5

Your level is the highest at which you can answer yes to every question at that level and below. Partial credit is how organizations convince themselves they are at L4 while operating at L2.

Chapter 26. The Adoption Roadmap

Eighteen months, four phases, and the sequencing that avoids the common failure of building the Control Plane too early.

26.1 The phases

Table 72.

Timeframe	Transition	Key actions
Months 0–3	L1 → L2	Name an AI Governance Lead, part-time acceptable. Adopt the minimum artifact set. Run S0–S2 on all new initiatives. Do not attempt retroactive classification yet
Months 3–9	L2 → L3	Expand ARB to five domains. Stand up the AI Governance Body. Adopt independent countersignature. Retroactively classify existing production AI. Adopt the full record set
Months 9–18	L3 → L4	Implement machine-readable G3 boundaries for agentic systems. Stand up Control Plane telemetry starting with two or three sources. Populate the Regulatory Overlay Reference. Begin audit sampling
Months 18+	L4 → L5	Extend the Control Plane. Policy-as-code preventive automation. Recalibrate scoring anchors against real incident data. Activate regional governance if multinational

26.2 Why this order

Three sequencing decisions carry most of the value, and each contradicts a common instinct.

Classification before gates. The instinct is to stand up review gates first, because they feel like governance. Tiering first means the gates arrive with something to scale against; gates first means uniform ceremony that teams learn to route around before proportionality ever appears.

Governance bodies before the Control Plane. The instinct is to buy tooling early. But Compare needs approved-state records, and approved-state records need a body that produces trustworthy decisions. Telemetry connected to thin decision records produces noise (Chapter 18, §18.2).

New initiatives before retroactive classification. The instinct is to inventory and classify everything first. That is a large program that delays any visible benefit and typically stalls. Applying S0–S2 to new work produces immediate value and builds the scoring competence needed to do retroactive classification quickly later.

26.3 Months 0–3 in detail

Objective: establish tiering as a routine step, with one accountable person.

Table 73. Months 0–3 in detail

Action	Detail
Name the AI Governance Lead	Part-time is fine. What matters is a single accountable name
Adopt three records	Risk Classification, Data Lineage and Sensitivity, ADR. Not

Action	Detail
Run S0–S2 on new initiatives	sixteen Every new AI initiative, including procured. No exceptions for pilots
Establish countersignature	Even informally. This is the L3 mechanism worth starting early
Instrument the process	Record time spent per classification and per gate from week one

What to resist. Building a tool, running an estate-wide inventory, standing up two bodies, or writing a full policy suite. Each is a legitimate later step and each will consume the first quarter without producing a governed decision.

26.4 Months 3–9 in detail

Objective: working governance bodies, full record set, and a classified estate.

Retroactive classification is the large task here. Run it in tier-descending order of suspected risk, not alphabetically or by system age: systems that make or materially influence decisions about people or money first, agentic systems next, then everything else. Expect the exercise to surface systems nobody knew were AI-enabled, particularly inside purchased software.

[Practice recommendation] Timebox retroactive classification at one quarter and accept incomplete coverage instead of letting it run indefinitely. A classified 70% of the estate with the highest-risk systems covered is more useful than a perfect inventory delivered a year late.

26.5 Months 9–18 in detail

Objective: actual detection capability, starting narrow.

Connect two or three telemetry sources, not ten (Chapter 18, §18.3). Run everything in report-only mode for a full delivery cycle before enforcing anything. Expect the first comparison run to reveal that your approved-state records are less accurate than you thought; that finding is the point of the exercise.

Machine-readable agent boundaries belong in this phase and should not wait. Any agentic system deployed without one carries a boundary nobody can verify.

26.6 What tends to go wrong, by phase

Table 74.

Phase	Common failure	Early warning sign
0–3	Tiering becomes a form-filling exercise	Scores cluster at Tier 1; justifications are one word
3–9	Bodies become bottlenecks	Lead time to ARB exceeds three weeks; teams begin pre-briefing to skip queue
3–9	Retroactive classification stalls	Inventory grows faster than classification
9–18	Alert volume overwhelms	Findings age without resolution; owners disable notifications
18+	Automation pressure on corrective controls	Proposals to auto-remediate Tier 3 systems

Chapter 27. The Minimum Viable Framework

The 20% that delivers most of the value, for organizations that cannot adopt the whole thing.

27.1 What the MVF is

Applying an 80/20 test against the framework’s own evidence produces a specific answer, and it is not the governance body structure. The highest-leverage element is **independent risk-tier verification**, because unverified self-scoring was the most confirmed weakness across the entire adversarial review.

The MVF is six commitments.

- 1. Run S0–S2 on all new AI initiatives, with mandatory independent countersignature.** This is the cheapest, highest-leverage control in the framework. If you adopt one thing, adopt this.
- 2. Operate five gates, but make Tier 1–2 self-service by default.** Only Tier 3–4 requires formal review. The gates exist; most work does not touch them.
- 3. Merge the two governance bodies if you cannot credibly staff both.** In organizations without the depth to populate two distinct groups, the separation collapses informally anyway. The MVF makes it a declared simplification rather than an unacknowledged pretence. Record it as a local variation.
- 4. Keep three records, not sixteen.** Risk Classification, Data Lineage and Sensitivity, and ADR. Defer the rest.
- 5. Agent Card only where there is real tool access or autonomous action.** Advisory-only systems do not need one.
- 6. Regulatory Overlay Reference only if you operate in a named regulated sector.**

27.2 What the MVF gives up, honestly

Adopting the MVF is a real trade, and the losses should be visible.

Table 75. What the MVF gives up, honestly

Given up	Consequence
Continuous comparison	Drift is found at review, not detected. The framework’s central claim is not realized
Full artifact set	Several of the seventeen questions cannot be answered without reconstruction
Separation of architecture and deployment authority	One body decides both “is this right” and “is this safe,” losing an independent check
Audit sampling	Countersignature covers initial scoring; nothing covers later drift in tier accuracy
Evidence-as-code	Evidence is assembled manually, so gate cost stays high

The MVF is a credible starting position, not a destination. What it preserves is the part that most determines whether everything else is pointed at the right systems.

27.3 Growing out of the MVF

Three signals indicate the MVF is no longer sufficient.

Tier 3–4 volume exceeds what one merged body can review without lead times that cause routing-around. Split the bodies.

Agentic systems reach production. The Control Plane's boundary comparison becomes materially more valuable than any documentation control, because a boundary nobody can verify is not a boundary.

A drift incident occurs that periodic review would not have caught. This is the argument for the Control Plane that no business case makes as effectively.

Chapter 28. Failure Modes and Anti-Patterns

Fourteen ways implementations go wrong, what each looks like early, and the correction.

28.1 Governance-theater patterns

Ceremony without proportionality. Every system receives the same review regardless of tier. Early sign: gate meeting length does not vary by tier. Correction: audit the actual evidence demanded per tier; if it is identical, the tiering is decorative.

Tiering as form-filling. Scores are entered to complete a field, with one-word justifications. Early sign: score distribution heavily weighted to Tier 1, justifications absent or generic. Correction: reject scores without justifications; run a calibration session with real examples.

Approval by attendance. A body approves everything presented because rejecting is socially expensive. Early sign: zero rejections and zero conditional approvals over a quarter. Correction: track outcome distribution; a gate that never returns anything other than approve is not a control.

The compiled view maintained by hand. An AI System Card kept in a document, updated occasionally. Early sign: any field in it differs from its source record. Correction: compile it or delete it.

28.2 Scope-creep patterns

Absorbing engineering. Reviews begin examining prompts, evaluation datasets, and model configuration. Early sign: ARB agendas contain items no board member can assess. Correction: re-apply the boundary test in Chapter 1, §1.3.

Absorbing delivery. Architecture begins reviewing delivery plans and change management. Early sign: the framework's transformation depth exceeds visibility. Correction: the T5 stage is an external input interface; it has no architecture role.

Reinventing deferred standards. A local AI risk taxonomy appears alongside D1–D5, or a bespoke security control catalog. Early sign: two vocabularies for the same concept. Correction: the framework defers deliberately; integrate rather than duplicate.

28.3 Classification failures

Mechanism-reversibility scoring. D3 scored on whether the system state can roll back, not whether the consequence can be undone. This is the single most common scoring error. Correction: re-anchor with worked examples; make it a standing countersignature check.

Effective autonomy ignored. D2 = 1 recorded for a system whose human reviewer approves everything. Correction: instrument override rate; set the re-score threshold in advance.

Vendor AI unscored. Embedded AI features treated as configuration, not as AI systems. Early sign: the estate contains no Tier 2+ procured systems, which is implausible. Correction: procurement intake routing (Chapter 21, §21.4).

Pilot exemption. Systems run in production under a pilot label with no gate passed. Correction: pilot is a deployment state with a mandatory expiry, not a governance exemption.

28.4 Operational failures

Telemetry before records. The Control Plane connected before ADRs are trustworthy. Early sign: high alert volume, low resolution rate, owners unidentifiable. Correction: pause telemetry expansion, fix approved-state records.

Exceptions as a parallel process. Exceptions used to bypass requirements routinely, not to handle genuine edge cases. Early sign: exception count rising quarter on quarter; the same exception type recurring across teams. Correction: recurring exceptions are design feedback; fix the requirement or add a pattern.

Bodies as bottlenecks. Lead times grow until teams route around governance. Early sign: pre-briefing culture, or systems appearing in production without a corresponding authorization record. Correction: Chapter 20, §20.6.

28.5 The meta-failure

The pattern underlying most of the above is treating IRGF as a compliance regime instead of a decision-support structure. A compliance regime asks whether the form was completed. A decision-support structure asks whether the right people had the right information at the point of decision.

The diagnostic question is simple. **When your governance process runs, does anything ever change as a result?** If systems always proceed as originally proposed, the process is producing documentation rather than governance, and no amount of additional rigor will fix that.

Chapter 29. Measuring Whether It Works

What to instrument, what the numbers mean, and which measures are research constructs, not operational metrics.

29.1 Instrument from day one

Nearly every threshold in IRGF is a reasoned default with no empirical calibration. The only route from asserted numbers to calibrated ones is your own data, and it can only be collected prospectively.

Six measures are worth capturing from the first month.

Table 76. Instrument from day one

Measure	Definition	Why
Classification effort	Time from S2 start to countersigned score	The dominant recurring cost
Gate lead time	Elapsed time per gate, by tier	Detects bottlenecks before routing-around begins
Evidence production time	Effort to assemble gate evidence, by tier	Tests whether proportionality is real
Tier distribution	Share of estate at each tier	Compared against incident distribution, detects systemic mis-tiering
Countersignature adjustment rate	Share of scores changed on countersignature	Tests whether the control is doing work
Automated evidence share	Proportion of gate evidence captured automatically	The most honest indicator of falling burden

29.2 Reading the numbers

Countersignature adjustment rate is the most informative single number. Near zero means either scoring is excellent or countersignature is a rubber stamp; near half means anchors are unclear. Something in the range of one in six to one in four adjusted suggests a control that is working without being adversarial.

Tier distribution against incident distribution is the systemic check. If most of the estate is Tier 1 and most incidents occur in Tier 1 systems, the classification model does not describe your organization.

Gate lead time by tier should diverge sharply. Convergent lead times mean tiering is not changing behavior.

Aged drift backlog is the clearest measure of whether governance keeps pace with delivery. A backlog growing faster than the estate means capacity is insufficient or alert quality is poor.

29.3 The value scorecard

Five metric families, deliberately not composited (Chapter 23, §23.5). Report them together and refuse to average them.

29.4 What you cannot measure yet

Two constructs proposed in the research are **not operational metrics**: Risk Mitigation Effectiveness and Perceived Governance Burden. Both are research constructs awaiting validated measurement instruments. Neither should appear on a dashboard as though it were measurable today.

The honest position is that IRGF's effectiveness has not been demonstrated anywhere, and your own instrumentation is the first evidence that will exist. **[Practice recommendation]** Retain your baseline data. The most valuable contribution an early adopter can make to this framework is a calibrated account of what it actually cost and what it actually caught.

Chapter 30. Operating Rhythm

The cadences that make governance a routine rather than an event, and what each body should be looking at when.

30.1 Why cadence matters more than procedure

A governance framework with well-designed mechanisms and no rhythm degrades into reactive review. Work arrives when it arrives, bodies convene when there is a queue, and the periodic activities that keep classification accurate — re-scoring, sampling, calibration — never happen because nothing schedules them.

The framework specifies almost no cadences. What follows is a working rhythm you can adopt and adjust, marked throughout as practice, not requirement.

30.2 A working calendar

Table 77.

Frequency	Activity	Owner
Continuous	Control Plane comparison; alert routing	Automated, with named owners per category
Daily	Tier 4 alert response; agent boundary divergence response	System Owners, AI Governance Lead
Weekly	Risk classification sessions; countersignature clearance; drift triage for Tier 3	Risk and Compliance Lead
Fortnightly	ARB: Tier 3–4 G2 decisions, pattern proposals, dependency conflicts	ARB
Fortnightly	AI Governance Body: Tier 3–4 G3 and G4, autonomy grants, residual risk acceptance	AI Governance Body
Monthly	Exception register review; expiring exceptions; open G3 conditions	AI Governance Lead
Monthly	Aged drift backlog review	Architecture with System Owners
Quarterly	Tier 1–2 audit sampling; tier distribution against incident data	Risk and Compliance Lead
Quarterly	Pattern library review; recurring exception analysis	ARB
Quarterly	Withheld funding concurrence report	AI Governance Body to Enterprise Sponsor
Half-yearly	Scoring anchor calibration session	Risk Lead with Architects and Sponsors
Annually	Tier 3–4 risk re-score; estate sweep for orphaned systems; principle review	Various

Two rhythms carry disproportionate value and are the ones most often skipped.

Weekly countersignature clearance. If countersignature waits for a monthly meeting, teams work around it or the classification arrives after architecture is settled. It needs to be a short, frequent, low-ceremony activity.

Half-yearly anchor calibration. Scoring anchors drift in application even when the text does not change. A two-hour session re-scoring a handful of past systems as a group, and discussing disagreements, does more for consistency than any amount of written guidance.

30.3 What each body should see

Architecture Review Board, standing agenda

Table 78.

Item	Purpose
Tier 3–4 G2 decisions	The board’s primary work
Pattern proposals and deprecations	Keeps the fast path alive
Dependency conflicts referred from solution level	The cross-initiative view nobody else holds
TG0 bypass findings	Work that entered the estate outside governance
Drift patterns by pattern	The Learn function’s output; pattern quality signal

AI Governance Body, standing agenda

Table 79.

Item	Purpose
Tier 3–4 G3 and G4 decisions	Primary work
Autonomy grants and increases	Always this body, at any tier above 1
Kill-switch condition approvals	Separate approval from general G3 sign-off at Tier 3–4
Residual risk acceptances	Explicit, recorded, owned
Funding concurrence for Tier 3–4 initiatives	The bounded exception to portfolio authority
Escalated unresolved alerts	Where local resolution failed

Quarterly governance review, for the Enterprise Sponsor

Six numbers, no narrative required: tier distribution against incident distribution, countersignature adjustment rate, gate lead time by tier, exception count and aged exceptions, aged drift backlog, and automated evidence share. Each is defined in Chapter 29.

30.4 Keeping meetings decision-shaped

Governance bodies drift toward discussion. Three practices keep them producing decisions.

Pre-circulate evidence, and enforce it. A case arriving without its required evidence is deferred, not discussed. This is unpopular for one cycle and then self-correcting.

State the decision required. Every agenda item names the decision and the options. “Discuss the claims triage system” is not an item; “Approve, approve with conditions, or reject G3 for the claims triage system at Tier 3” is.

Record the reasoning, not just the outcome. The Decision record needs enough rationale that someone reconstructing it in two years understands why. This is also what makes drift comparison meaningful later.

[Practice recommendation] Timebox Tier 3 items to fifteen minutes and Tier 4 items to thirty. Items exceeding their box are deferred with a named follow-up, not allowed to consume the agenda. Bodies that routinely overrun stop attracting the senior attendance they depend on.

Chapter 31. Introducing IRGF to Delivery Teams

The framework's adoption risk is not technical. It is that the people it governs conclude it is overhead.

31.1 The objections you will actually get

Six objections recur. Each has a legitimate core, and answering the legitimate part is more effective than defending the framework.

“This will slow us down.” Partly true and worth conceding. Tier 3–4 work will be slower, deliberately. Tier 1–2 work should be faster once patterns and self-certification are in place, and that is the claim to make and then evidence. What you must not do is promise overall speed improvement in month one, because the first quarter is slower for everyone.

“We already do this.” Often partly true. Most organizations have architecture review, some risk assessment, and some testing. The honest answer is that IRGF connects those rather than replacing them, and the specific thing that is usually missing is a single tier that drives all of them consistently. Ask the team to run the seventeen questions against one of their own systems; the gaps make the argument better than you can.

“Our system is low risk.” Frequently correct, and the framework agrees: Tier 1–2 is self-service. The response is not to argue but to run the scoring, because the scoring is what establishes the claim. A team that scores its own system honestly and lands at Tier 1 has just used the framework correctly.

“The gate will reject us.” Usually the fear is delay, not rejection. Publishing gate lead times and outcome distributions defuses this faster than reassurance does.

“We’re just piloting.” The most dangerous objection, because it sounds reasonable. Pilots reach production. The answer is the deployment-state model in Chapter 5: pilot is a state with an expiry, not an exemption.

“The vendor handles governance.” Common for procured AI and almost never accurate at the level the enterprise needs. The vendor governs their model; you govern its use in your context, including the decisions it influences and the data it processes.

31.2 What to do in the first ninety days

Make the first experience good. The first three teams through the process shape its reputation permanently. Over-support them: sit with them for the scoring, pre-review their ADR, and make sure the first gate runs to time.

Show a Tier 1 pass early. The most persuasive artifact is a system that went through the framework quickly because it was genuinely low risk. It demonstrates proportionality is real, which no amount of policy text achieves.

Publish lead times from the start. Uncertainty about how long governance takes causes more avoidance than the actual duration does.

Never let a gate be the first time a team hears about a requirement. If a gate rejects work for a requirement the team did not know existed, you have lost that team and everyone they talk to. The requirements belong in the intake conversation.

31.3 What to say about the framework's status

You will be asked whether this is a standard. It is not, and the answer should be direct: IRGF is a designed framework that has been adversarially stress-tested but never deployed in a real organization, and it defers to the actual standards — NIST AI RMF, ISO/IEC 42001, ISO/IEC 23894, COBIT, ISO/IEC 27001 — for the things those standards already cover.

That answer is stronger than a claim of authority, for two reasons. It is accurate, and accuracy survives the first person who checks. And it invites teams to treat the thresholds as adjustable, which is exactly right, because they are reasoned defaults awaiting your calibration data.

[Practice recommendation] Make the limitations register (Appendix A) available to delivery teams rather than restricting it to the governance function. A framework that publishes its own weaknesses attracts better challenge, and the challenge is how the thresholds get calibrated.

31.4 Signs adoption is going well

Table 80.

Signal	What it indicates
Teams arrive at G1 having already scored themselves	Classification has become routine rather than imposed
Scoring disputes are about anchors, not about whether to score	The mechanism is accepted; only calibration is contested
Tier 1 systems pass in days	Proportionality is real
Exceptions are requested rather than taken	The path exists and is usable
Teams cite pattern conformance unprompted	The library is providing value, not just constraint
Someone challenges a threshold with data	The framework is being calibrated rather than obeyed
The last signal is the strongest. A framework nobody challenges is a framework nobody is really using.	

Chapter 32. End-to-End Walkthrough

One system, from opportunity to retirement, with every record it produces and every decision it triggers. The example is deliberately sector-neutral so that the mechanism, instead of the domain, is what is visible.

32.1 The system

A large organization operates a shared services function that receives high volumes of inbound written requests. Each request must be categorized, routed to a specialist queue, and prioritized. The work is currently manual, slow, and inconsistent between assessors.

The proposal is a system that reads each request, proposes a category and priority, and drafts a short summary for the receiving specialist.

Throughout this chapter, records are shown as they would actually be written, and the reasoning behind each judgment is stated. Several judgments are contestable; where that is so, the alternative is discussed, not hidden.

32.2 S0 — Opportunity

The capability owner proposes the initiative. The architect's role at S0 is short: establish that a capability need exists and that AI is the right instrument.

Capability: Request Intake and Routing. Baseline maturity 2, target 4.

Current state: mean 3.2 days from receipt to correct queue. 18% of requests are re-routed at least once.

Need: reduce time to correct queue and reduce re-routing.

No-AI alternative considered: a rules engine over sender and keyword was piloted two years ago and reached 54% routing accuracy against a 90% requirement. Free-text variability defeated it. Rejected on evidence.

Provisional sourcing direction: foundation model API or retrieval-augmented approach; not fine-tuning, since the routing taxonomy changes quarterly.

Decision: proceed to qualification.

Note what has not happened. No vendor has been named, no architecture proposed, and no commitment made. The recorded rejection of the rules-engine alternative is what makes the AI choice defensible later.

32.3 S1 — Use-Case Qualification

The Use-Case Canvas is produced.

Table 81. S1 — Use-Case Qualification

Field	Content
Linked capability	CAP-041 Request Intake and Routing
Problem statement	Manual categorization is slow and inconsistent; 18% re-routing rate
Expected value	Mean time-to-correct-queue from 3.2 days to under 1 day;

Field	Content
	re-routing below 8%. Measured monthly against the preceding four-quarter baseline
Data path	Historical request corpus with assigned categories, 400,000 records, held in a governed store. Routing taxonomy maintained by operations
Feasibility	Classification and summarization over text. No novel capability required
Provisional D1–D5	D1 = 2, D2 = 2, D3 = 2, D4 = 3, D5 = 3 (provisional)
Sponsor	Head of Shared Services Operations
Provisional tier: Impact = $\text{avg}(2, 3, 3) = 2.67 \rightarrow 3$. Control Deficit = $\text{avg}(2, 2) = 2$. Matrix gives Tier 2 .	

32.4 S2 — Risk Classification and countersignature

The Risk and Compliance Lead runs the scoring session with the sponsor, the architect, and the data owner.

Table 82. S2 — Risk Classification and countersignature

Dimension	Score	Justification
D1 Decision Consequence	3	Raised from the provisional 2. A misrouted request is delayed, and some request types carry time limits after which the requester loses an entitlement. The consequence is not merely internal inefficiency
D2 Autonomy	2	The system proposes a category; the intake specialist confirms or changes it before routing
D3 Reversibility Deficit	2	Re-routing is straightforward. Elapsed delay is not recoverable, which is why this is 2 rather than 1
D4 Exposure and Scale	3	All inbound requests. Internal users, but output feeds the case management system of record
D5a Data Sensitivity	3	Requests contain personal data, classified Restricted
D5b Model Uncertainty	3	Generative summarization; confident plausible error is a known property
D5 composite	3	Higher of the two sub-scores

Impact = $\text{avg}(3, 3, 3) = 3$. **Control Deficit** = $\text{avg}(2, 2) = 2$. **Matrix:** **Tier 2**. No safety override.

Countersignature. The Risk Lead is the accountable scorer here, and the countersignature is provided by a second reviewer from the compliance function who was not in the scoring session. Their note:

D1 raised from 2 to 3 during the session and I agree; the entitlement time-limit point is material. Flagging that this system sits one point of D1 from Tier 3, and that D2 = 2 depends on intake specialists genuinely reviewing proposals. Requiring override-rate instrumentation as a condition of G3.

This is countersignature working as intended. It did not change the tier. It identified the boundary condition and attached a condition that will detect if the assumption behind D2 fails.

32.5 Gate G1

Tier 2, so the gate is self-service with notification. The capability owner approves; the AI Governance Lead is notified.

Outcome: Approve with conditions. **Condition:** override-rate instrumentation to be in place before G3. Owner: system owner. Due: G3.

32.6 S3 — Architecture Decision

Now the architecture is settled. The team assesses sourcing against the twelve considerations and lands on a retrieval-augmented approach using an approved foundation model API, grounded on the routing taxonomy and a curated set of routing precedents.

Fine-tuning was rejected: the taxonomy changes quarterly, and every change would be a Major change requiring full revalidation. That recurring cost exceeded the accuracy benefit in the team's assessment.

ADR-0311

Decision: Implement request triage using the Enterprise RAG pattern v2, unmodified, with an approved foundation model accessed through the enterprise LLM gateway.

Alternatives: (a) Fine-tuned classifier — rejected; quarterly taxonomy change makes revalidation cost prohibitive. (b) Rules engine — rejected at S0 on evidence. (c) Vendor SaaS triage product — rejected; would require the request corpus to leave the enterprise boundary, and D5a is 3.

Model: [approved list entry]. Version policy: pinned minor; vendor patch auto-accepted; family change requires new ADR.

Hosting: vendor API via enterprise gateway, in-region endpoint.

Grounding: DP-207 Routing Taxonomy (Internal); DP-208 Routing Precedents (Restricted). No other sources.

Pattern conformance: Conformant, Enterprise RAG v2, unmodified.

Authority boundary: none. The system has no tool access and cannot write to any system. Output is a proposal rendered in the intake interface.

Risk tier at decision: Tier 2, per RCR-0188.

Accountable owner: Intake Platform Product Owner.

The authority boundary line matters even though it is empty. Recording that the system has no action capability is what makes a later grant of action capability detectable as a change.

32.7 S4 — Data Readiness

Table 83.

Source	Classification	Owner	Refresh	Max staleness
DP-207 Routing Taxonomy	Internal	Operations Standards Manager	Quarterly, on taxonomy change	100 days
DP-208 Routing Precedents	Restricted	Shared Services Data Steward	Monthly	60 days

The maximum staleness values are the practice recommendation from Chapter 9, §9.2, not a framework requirement. They matter here: a routing taxonomy that has changed while the grounding source has not is exactly how this system would degrade silently.

Finding raised at S4: DP-208 had no named owner. The steward was assigned before G2 could proceed. This is the most common S4 finding and it was caught because the record has an owner field that cannot be left blank.

32.8 Gate G2

Tier 2 with an unmodified pattern, so architect self-certification applies and the ARB is notified, not convened.

Risk re-score at G2: the architecture confirms D2 = 2 (no action capability, proposal only) and D5a = 3 (Restricted grounding source retained inside the boundary). The vendor-SaaS alternative would have raised D5 and possibly the tier; the chosen architecture does not. Tier confirmed at 2.

Outcome: Approve. Pattern conformance recorded as conformant. ARB notified.

32.9 S5 — Build

Light governance. Two things are checked, both automatically.

The deployed model family is compared against the ADR by policy rule; it matches. The grounding sources connected in the development environment are compared against the lineage record — and here the Control Plane, running in the non-production environment per Chapter 10's recommendation, raises a finding.

Drift finding DR-0442. Category 5, Data Lineage and Governance. Development environment shows a third grounding source connected: an unclassified export of historical case notes, not present in the lineage record.

The team explains it was connected for evaluation and was intended to be temporary. This is the classic S5 drift origin, caught before production, not after.

Endpoint: Revert. The source is disconnected. No change record required, because the approved state was correct and the divergence was unintended. Had the team wanted to keep it, the endpoint would have been Ratify, requiring a change record and a G4 assessment, since a new Restricted-or-above source is a Material change.

32.10 S6 — Assurance

Five assurance types, at Tier 2 depth.

Table 84. S6 — Assurance

Type	Activity	Result
Technical AI Evaluation	Classification accuracy against held-out set; summarization faithfulness; prompt-injection resistance	91% routing accuracy against a 90% threshold defined before the run. Faithfulness within tolerance. Injection tests passed
Architecture Assurance	Deployed configuration compared to ADR-0311	Conformant after DR-0442 resolution
Risk Assurance	Control Matrix completed; tier confirmed	Tier 2 confirmed
Compliance Assurance	Personal-data handling checked against internal policy and the applicable regime in the Regulatory Overlay Reference	Conformant
Business Outcome Assurance	Baseline confirmed for later measurement	Baseline locked: 3.2 days, 18% re-routing

Note that the pass threshold was defined before the evaluation ran. This is a small discipline with a large effect: thresholds set afterward are set to whatever the system achieved.

32.11 S7 and Gate G3

Tier 2, so the System Owner approves and the AI Governance Lead is notified.

Evidence presented: assurance summary, Control Matrix, named owner, human oversight point confirmed (the intake specialist's confirmation step), and — the G1 condition — override-rate instrumentation in place.

Deployment Authorization DAR-0129. System: Request Triage Assistant. Tier 2. Authorized by Intake Platform Product Owner, [date]. Evidence: AIS-0233, CM-0177, ADR-0311, RCR-0188. Conditions: override rate to be reported at the first quarterly review; a rate below 5% triggers a D2 re-score.

The condition converts a known weakness of the scoring model into a scheduled check with a pre-agreed threshold. This is the practical answer to the effective-autonomy problem in Chapter 6, §6.3.

32.12 S8 — Runtime, and what happens over eleven months

Month 1. Routing accuracy holds. Mean time to correct queue falls to 1.4 days.

Month 3. First quarterly review. Override rate is 11%: specialists change roughly one proposal in nine. D2 = 2 is supported by evidence rather than assumption. No re-score.

Month 5. Cost drift signal. Inference spend is 40% above the approved budget. Investigation shows request volume grew after another department began routing its intake through the same queue. This is D4 movement disguised as a cost signal, which is exactly why cost is in the drift taxonomy. The Risk Lead re-scores D4: still 3, since the population is still internal and the system of record unchanged. Tier unchanged. The budget is revised.

Month 7. Behavioral drift. Summarization quality degrades on a subset of requests. The two candidate causes are separated immediately because the framework keeps them apart: has the model changed, or has the grounding changed? The model registry shows the vendor pushed a minor version, which the ADR's version policy auto-accepts. The lineage record shows DP-208 was last refreshed 71 days ago against a 60-day maximum.

Both are true. The vendor update is a Material change at Tier 2 under the change table only at Tier 3–4, so it is recorded as Minor here; the staleness breach is a Data Lineage drift finding requiring the steward to refresh. Quality recovers after refresh, which identifies staleness as the dominant cause.

This is the diagnostic value of separating category 5 from category 6. With a single “data drift” signal, the team would have spent the investigation arguing about the model.

Month 9. A change request arrives, and it is the interesting one.

32.13 S9 and Gate G4 — the autonomy change

Operations propose that requests the system categorizes with high confidence should route automatically, without specialist confirmation, to reduce handling time further.

Classification: this removes a human step from a path. Under the change table it is **Major, always** — an autonomy increase, boundary-widening. It is never eligible for self-service classification regardless of the current tier.

Consequences: S2 re-entry, G2, and G4.

Re-scoring:

Table 85. S9 and Gate G4 — the autonomy change

Dimension	Before	After	Reasoning
D1	3	3	Unchanged; the consequence of a misroute is the same
D2	2	3	The system now acts without review on a defined subset
D3	2	2	Unchanged
D4	3	3	Unchanged
D5	3	3	Unchanged

Impact = 3. **Control Deficit** = $\text{avg}(3, 2) = 2.5 \rightarrow 3$. Matrix: Impact 3, Control Deficit 3 $\rightarrow$ **Tier 3**.

The system moves from Tier 2 to Tier 3 on a single dimension change. Everything downstream changes with it: G2 now requires ARB decision rather than self-certification; G4 requires AI Governance Body approval; annual re-scoring becomes mandatory; alert response tightens to five working days; and the AI Governance Body must approve the autonomy grant specifically, not merely the change.

ARB at G2. The architecture now includes a confidence threshold and an auto-route path. That is a modification to the Enterprise RAG pattern, so the fast path is unavailable and the modification is reviewed on its merits. ARB approves with a condition: the confidence threshold must be recorded in the ADR as an approved-state value so the Control Plane can detect it being lowered.

That condition is worth pausing on. Without it, someone could reduce the threshold from 0.95 to 0.80 as a configuration change, materially widening the autonomous path, and nothing would detect it. With it, the threshold is comparable and a reduction is Configuration drift on a Tier 3 system.

AI Governance Body at G4. Approves the autonomy grant with two conditions: the auto-route path is limited to three request categories initially, not all, and override-rate reporting moves to monthly for the confirmed path.

ADR-0388 supersedes ADR-0311. Records the confidence threshold (0.95), the three permitted auto-route categories, and the retained human path for everything else. **RCR-0244** records the Tier 3 re-score. **ACR-0517** records the change classification and its reasoning.

32.14 S10 and Gate G5 — retirement, two years on

The organization replaces its case management platform, and the new platform includes native intake triage. The Request Triage Assistant is retired.

Table 86. S10 and Gate G5 — retirement, two years on

Obligation	Evidence
------------	----------

Obligation	Evidence
Dependency check	Control Plane integration telemetry identifies two consumers not in the documented dependency map: a monthly operations report and a downstream workload forecasting model. Both migrated before decommissioning
Data retention	Grounding sources unaffected, remain owned by their stewards. System-generated summaries retained per the case retention schedule
Access revocation	Gateway credentials and service identity deprovisioned; confirmed from IAM logs
Evidence preservation	ADR-0311 and ADR-0388, RCR-0188 and RCR-0244, DAR-0129, ACR-0517, assurance summaries, and the drift record set archived. Retention: 3 years
Knowledge preservation	Note recorded: threshold-based auto-routing worked, but only after the tier moved and the ARB required the threshold to be an approved-state value. Recommended as a pattern candidate

G5 outcome: Approve. Tier 3, so the AI Governance Lead confirms no open obligations remain — one exception was open and had expired; it is closed as part of retirement.

The dependency finding is the one that would have caused an incident. Two consumers existed that nobody had recorded, and telemetry found them where documentation did not.

32.15 What the walkthrough demonstrates

Six things worth extracting.

The tier did the work. Almost every difference in how this system was governed traced to its tier, and the tier changed once, on one dimension, and changed everything downstream with it.

Countersignature earned its place without changing the tier. Its value was identifying the boundary condition and attaching the instrumentation that would detect the assumption failing.

The most consequential single control was a condition, not a gate. Requiring the confidence threshold to be an approved-state value is what made a future widening of autonomy detectable.

Separated drift categories paid off diagnostically. Month 7 resolved quickly because model behavior and grounding lineage are distinct signals.

Non-production comparison converted an incident into a conversation. DR-0442 would have been a production data-governance finding.

Retirement found what documentation had not. The dependency check is routinely the most valuable part of G5 and the most often skipped.

Chapter 33. Module Reference: Foundations, Classification, Lifecycles

Modules M1 to M9, each specified to a uniform template. This part is written for reference rather than reading: when you need to know what a module consumes, produces, and depends on, look here; when you need to know how to run it well, follow the cross-reference to the narrative chapter.

33.1 How to read a module entry

TOGAF gives each ADM phase a standard treatment — objectives, approach, inputs, steps, outputs. The module entries below follow the same discipline, adapted to what IRGF actually needs a practitioner to know.

Table 87. How to read a module entry

Field	What it tells you
Purpose	The one thing the module exists to achieve
Inputs	What must exist before the module can run
Mechanics	The steps, in order
Outputs	What the module produces that something else consumes
Accountable	The single role answerable for the output
Tier variation	How the module changes with risk tier; “none” means it is tier-invariant
Depends on / Feeds	Module dependencies, by identifier
Failure signal	The observable sign that this module is not working
In depth	The narrative chapter that explains how to run it

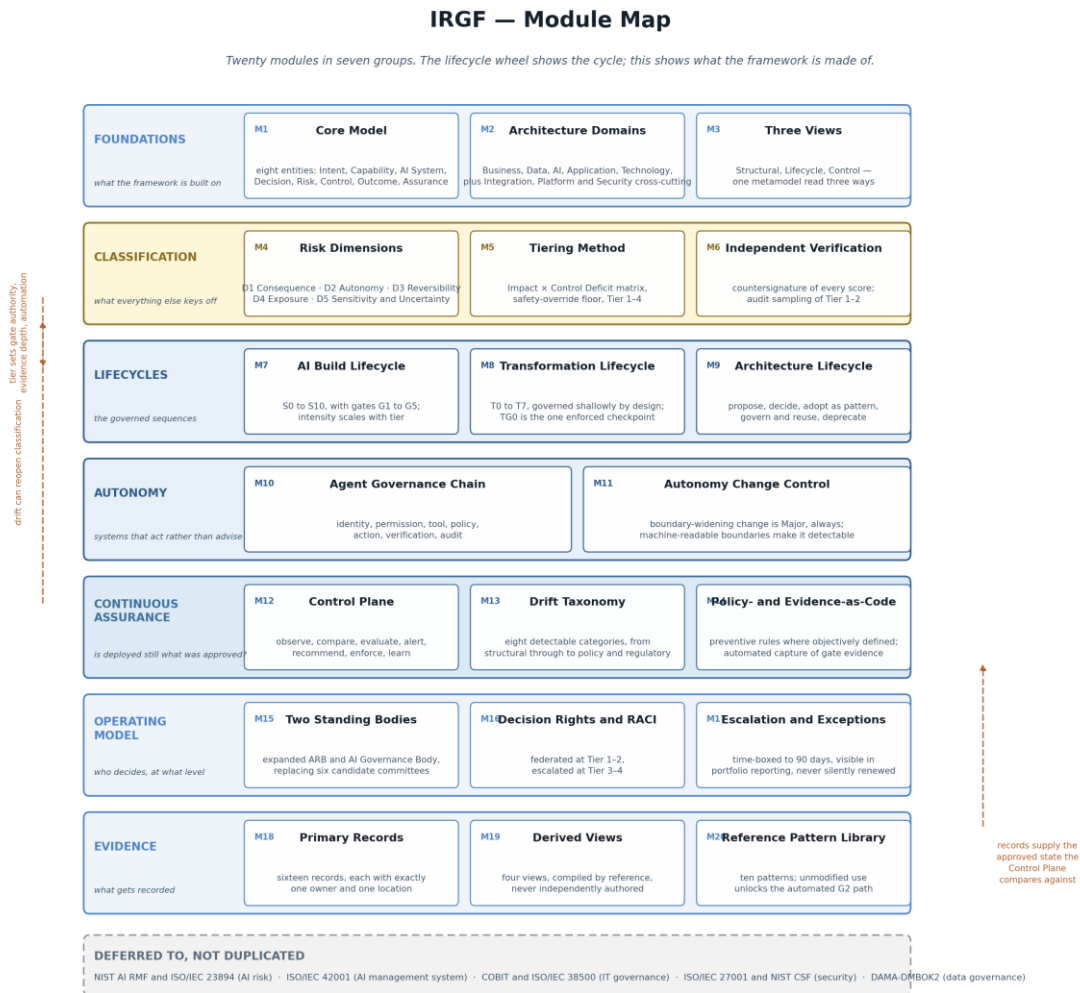


Figure 7. The IRGF module map: twenty modules in seven groups, and what each depends on

Twenty modules are grouped as they appear on the module map. The grouping is a reading aid, not a governance structure: nothing in the framework depends on a module's group.

33.2 Module map at a glance

Table 88.

Group	Modules
Foundations	M1 Core Model · M2 Architecture Domains · M3 Three Views
Classification	M4 Risk Dimensions · M5 Tiering Method · M6 Independent Verification
Lifecycles	M7 AI Build Lifecycle · M8 Transformation Lifecycle · M9 Architecture Lifecycle
Autonomy	M10 Agent Governance Chain · M11 Autonomy Change Control
Continuous Assurance	M12 Control Plane · M13 Drift Taxonomy · M14 Policy- and Evidence-as-Code
Operating Model	M15 Two Standing Bodies · M16 Decision Rights and RACI · M17 Escalation and Exceptions

Group	Modules
Evidence	M18 Primary Records · M19 Derived Views · M20 Reference Pattern Library

Which modules are load-bearing

Not all twenty carry equal weight. If you are sequencing adoption, three modules determine whether the rest function.

M4 and M5 together produce the tier, and almost every other module reads it. **M6 is what makes the tier trustworthy**, and the framework’s own review found unverified scoring to be its most confirmed weakness. **M18 supplies the approved state** that M12 compares against; without trustworthy records the Control Plane produces noise.

The Minimum Viable Framework in Chapter 27 is, in module terms, exactly M4, M5, M6, a reduced M7, and three records from M18.

33.3 Foundations

M1 — Core Model

Table 89.

Field	Specification
Purpose	Provide the minimal entity set through which every governance question is answered, so that one AI system has one connected record rather than fragments in separate functions
Inputs	Existing enterprise records: capability model, application portfolio, risk register, decision records
Mechanics	1. Map each of the eight entities to whatever your organization already holds. 2. Identify which have no home, no owner, or no link to an AI system. 3. Establish stable identifiers so entities can reference each other. 4. Close the gaps in dependency order — Decision first, since everything downstream compares against it
Outputs	A connected entity set: Intent, Capability, AI System, Decision, Risk, Control, Outcome, Assurance
Accountable	Chief Enterprise Architect, or the equivalent architecture lead
Tier variation	None. The Core Model is tier-invariant; only the depth of what populates it varies
Depends on	Nothing. This is the foundation
Feeds	Every other module
Failure signal	You cannot answer the seventeen questions in Chapter 1 for a production system without manual reconstruction across three systems
In depth	Chapter 2

Status caveat. The eight entities are asserted rather than derived. No minimality test has been performed and no expert panel has validated the set. Record any case the model cannot express; those are what a future revision needs.

M2 — Architecture Domains

Table 90.

Field	Specification
Purpose	Establish AI as a first-class architecture domain with real representation, rather than as one more application type folded into Application Architecture
Inputs	Existing domain structure and architecture review board composition
Mechanics	1. Add AI as a fifth primary domain alongside Business, Data, Application, Technology. 2. Classify Integration, Platform and Security as cross-cutting disciplines, not peer domains. 3. Add AI architecture expertise to ARB membership as a standing requirement. 4. Apply the boundary test in Chapter 1 §1.3 to settle what the domain governs
Outputs	A domain structure with AI represented; an ARB whose mandate covers it
Accountable	Chief Enterprise Architect
Tier variation	None
Depends on	M1
Feeds	M7, M9, M15, M20
Failure signal	AI decisions are reviewed by people who cannot assess them, or are not reviewed at all because no domain claims them
In depth	Chapter 1, §1.3
The domain's scope is deliberately bounded: AI Architecture governs the existence, connection, and authority boundary of a component, not its internal construction. Scope creep into model engineering is the most common way this module fails.	

M3 — Three Views

Table 91.

Field	Specification
Purpose	Let one metamodel serve three audiences without duplicating facts, by defining where each fact is authored and where it is merely read
Inputs	M1
Mechanics	1. Assign each entity attribute to exactly one authoring view. 2. Establish that the other views reference it. 3. Resist the pull to let a second view re-enter a fact for convenience
Outputs	Structural view (what exists), Lifecycle view (where it is in its progression), Control view (whether reality matches the record)
Accountable	Chief Enterprise Architect
Tier variation	None
Depends on	M1
Feeds	M12, M18, M19
Failure signal	The same fact — usually the risk tier — is maintained in two places and they disagree
In depth	Chapter 2, §2.4

33.4 Classification

M4 — Risk Dimensions

Table 92.

Field	Specification
Purpose	Convert an AI system's risk into five scored dimensions that a tiering method can act on, using anchors specific enough to be applied consistently
Inputs	Use-Case Canvas; architecture intent where known; data classification
Mechanics	1. Convene sponsor, architect, data owner and risk lead. 2. Score D1 through D5 against published anchors, one line of justification each. 3. Record D5's two sub-scores separately and take the composite as the higher, not the average. 4. Note any score sitting near a threshold
Outputs	Five scored dimensions with justifications, in the Risk Classification Record
Accountable	Risk and Compliance Lead
Tier variation	None — scoring is what determines the tier, so it cannot vary by it
Depends on	M1
Feeds	M5, and through it almost everything
Failure signal	Justifications are absent or generic; D3 scored on rollback capability; scores cluster just below thresholds
In depth	Chapter 6

M5 — Tiering Method

Table 93.

Field	Specification
Purpose	Combine five dimensions into a single tier without letting one catastrophic factor be averaged away by four benign ones
Inputs	M4 output
Mechanics	1. Impact = average(D1, D4, D5), rounded up. 2. Control Deficit = average(D2, D3), rounded up. 3. Read the tier from the 4×4 matrix. 4. Apply the safety-override floor: D1 = 4, or D2 = 4 with D3 = 4, forces Tier 3 minimum. 5. Record whether the override fired
Outputs	A tier, 1 to 4, with the computation shown
Accountable	Risk and Compliance Lead
Tier variation	Not applicable — this module produces the tier
Depends on	M4
Feeds	M7, M10, M12, M15, M16, M18
Failure signal	Tier distribution does not track your incident distribution; almost everything lands at Tier 1
In depth	Chapter 7, §7.1–7.5

Known limitation. The override floor catches extremes only. Mid-range judgment calls under delivery pressure remain the framework's live gaming vector, which is why M6 exists.

M6 — Independent Verification

Table 94.

Field	Specification
Purpose	Make the tier trustworthy by separating the person who scores from the person who is accountable for delivery
Inputs	M5 output
Mechanics	1. Route every score to a countersigner independent of delivery. 2. Review justifications, not the whole assessment. 3. Test the two or three scores nearest a threshold. 4. Adjust with a recorded reason, or accept. 5. Separately, sample Tier 1–2 self-certifications quarterly, weighted toward boundary cases
Outputs	A countersigned tier; a quarterly sampling rate
Accountable	Risk and Compliance Lead, with a countersigner independent of the scoring session
Tier variation	Countersignature applies at all tiers. Sampling targets Tier 1–2, since Tier 3–4 receives review anyway
Depends on	M5
Feeds	Confidence in every tier-scaled decision
Failure signal	Countersignature adjustment rate near zero over a meaningful volume, which means either excellent scoring or a rubber stamp
In depth	Chapter 7, §7.6–7.8
This is the highest-leverage module in the framework and the one most often dropped during adoption because it costs a person’s time. Dropping it does not remove the cost; it moves the cost to whatever the mis-tiered system eventually does.	

33.5 Lifecycles

M7 — AI Build Lifecycle

Table 95.

Field	Specification
Purpose	Govern an AI system from the question of whether to build it through to evidenced decommissioning, with intensity proportional to its tier
Inputs	A capability need; M5 tier; M20 pattern library
Mechanics	Eleven stages, S0 to S10, with five gates. S0–S1 establish justification. S2 classifies. G1 qualifies. S3–S4 settle architecture and data. G2 assesses both together. S5 builds. S6 assures. G3 authorizes. S7–S8 deploy and monitor continuously. S9 classifies change; G4 re-authorizes. S10 and G5 retire
Outputs	A governed AI system with a complete decision and evidence chain
Accountable	Varies by stage; see the RACI in Chapter 21
Tier variation	Extensive. Authority, evidence depth and automation eligibility all scale with tier at every gate
Depends on	M4, M5, M6, M20
Feeds	M12 (approved state), M18 (records), M10 where the system is agentic
Failure signal	Gate lead times do not differ by tier; systems appear in production with no Deployment Authorization Record
In depth	Chapters 4 to 15

Adoption note. Gate consolidation reduced labeled checkpoints from eleven to five but did not reduce evidentiary burden proportionally. Do not introduce this module as a lighter process; introduce it as a proportionate one.

M8 — Transformation Lifecycle

Table 96.

Field	Specification
Purpose	Connect transformation initiatives to architecture without claiming ownership of transformation delivery
Inputs	Strategic objectives; capability records; portfolio pipeline
Mechanics	Eight stages, T0 to T7. Architecture contributes mandatory input at T2 (the TG0 checkpoint), supplies the Architecture Input Package at T3, is accountable at T4 where the AI build lifecycle is spawned, and consumes outcome evidence at T6
Outputs	TG0 records; Architecture Input Package; the T4-to-S0 handoff
Accountable	Portfolio and PMO, external to this framework, with named architecture inputs
Tier variation	Tier 3–4 AI-relevant initiatives require AI Governance Body funding concurrence at T3; everything else is advisory
Depends on	M2, M5
Feeds	M7 at T4
Failure signal	Architecture reviewing delivery plans, which means scope has crept; or repository entries with no TG0 record, which means the checkpoint is being bypassed
In depth	Chapter 23
Governance depth is deliberately non-uniform across this chain: deep at capability, architecture and outcome; shallow at theme, portfolio and delivery. The T5 stage is an external input interface with no architecture role at all.	

M9 — Architecture Lifecycle

Table 97.

Field	Specification
Purpose	Turn individual architecture decisions into reusable enterprise practice, so that the second system solving a familiar problem is cheaper to govern than the first
Inputs	ADRs from M7's S3 stage; drift signals from M13
Mechanics	Propose (ADR drafted) → Decide (ARB review, tier-scaled) → Adopt as pattern (promoted to M20 if reusable) → Govern and reuse (Tier 1–2 self-certify against the unmodified pattern) → Deprecate (retirement triggers re-review of dependent systems)
Outputs	Ratified patterns; deprecation notices with dependent-system re-reviews
Accountable	Architecture Review Board
Tier variation	Pattern promotion is tier-invariant; pattern-based self-certification is available at Tier 1–2 only
Depends on	M2, M20
Feeds	M20, and through it the fast path in M7
Failure signal	The pattern library has not changed in a year, while novel architectures keep arriving at G2

Field	Specification
In depth	Chapter 35
This lifecycle was implicit across the framework before being named. It is an organizing device rather than a new governance mechanism, and it is included because pattern maintenance is the activity most often left unowned.	

Chapter 34. Module Reference: Autonomy, Assurance, Operating Model, Evidence

Modules M10 to M20, to the same template. The four groups here are what turn a set of documented decisions into a governed estate: they constrain what acts, detect what changed, decide who says yes, and record what happened.

34.1 Autonomy

M10 — Agent Governance Chain

Table 98.

Field	Specification
Purpose	Constrain what a system that acts is permitted to do, and make its authority comparable against what it actually holds
Inputs	M5 tier; architecture decision naming the agent; existing identity infrastructure
Mechanics	Eight links, each a control point. Register a distinct machine identity. Define the authority boundary in machine-readable form. Publish an explicit tool allow-list. Enforce policy in code, with a mandatory preventive control for standing-authority Tier 3–4 agents. Execute with a pre-approved kill-switch condition. Verify each action within a defined latency ceiling. Log immutably into Control Plane observation
Outputs	Agent Card; machine-readable boundary; immutable audit trail
Accountable	AI System Owner, co-authored with the Security Architect
Tier variation	Kill-switch conditions mandatory at Tier 3–4. Preventive control mandatory for standing authority at Tier 3–4. Never self-service at Tier 1
Depends on	M5, M7
Feeds	M11, M12, M18
Failure signal	Shared service accounts across agents, which makes attribution impossible; or a boundary recorded only in prose, which makes expansion undetectable
In depth	Chapter 16
An allow-list rather than a deny-list is deliberate. Deny-lists fail open, and for a component that acts, failing open is the wrong default.	

M11 — Autonomy Change Control

Table 99.

Field	Specification
Purpose	Ensure that any expansion of what an agent may do is re-authorized, and that the expansion is detectable rather than merely prohibited
Inputs	M10 boundary record; change requests; Control Plane comparison output
Mechanics	1. Classify the change against the boundary-widening test. 2. Widening changes are Major, always, and never eligible

Field	Specification
	for self-service classification. 3. Re-enter risk classification, then G2 and G4. 4. Narrowing changes are Minor by construction. 5. Compare live permissions against the approved boundary continuously
Outputs	Change classification; re-scored tier where autonomy moved; updated Agent Card
Accountable	AI Governance Body for the authorization; System Owner for the record
Tier variation	Autonomy grants and increases route to the AI Governance Body at every tier above 1
Depends on	M10, M4, M5
Feeds	M7 at S9, M12
Failure signal	Live agent permissions differ from the approved boundary, at any tier. Treat as an incident, not a drift finding
In depth	Chapter 17
Open tension. The rule was narrowed during revision from “any autonomy increase” to “boundary-widening changes only,” and the interaction between that narrowing and audit sampling has never been tested. Weight your sample toward agentic systems until you have your own evidence.	

34.2 Continuous Assurance

M12 — Control Plane

Table 100.

Field	Specification
Purpose	Establish whether what is deployed is still what was approved, continuously rather than at review checkpoints
Inputs	M18 records supplying approved state; telemetry sources; M5 tier for severity
Mechanics	Seven functions as a loop. Observe telemetry. Compare against the approved Decision record. Evaluate severity from drift category and tier. Alert the accountable owner. Recommend remediation. Enforce only within the automation boundary. Learn, feeding patterns back upstream
Outputs	Drift and alert records; remediation recommendations; pattern-quality signals
Accountable	Architecture function, with per-category owners
Tier variation	Comparison cadence, alert routing and response windows all scale with tier. Automated corrective enforcement is permitted at Tier 1–2 only
Depends on	M18 above all; also M5, M13, M14
Feeds	M4 (re-scoring), M7 (change), M9 (pattern quality)
Failure signal	High alert volume with low resolution rate, which almost always means approved-state records are wrong rather than that the estate is
In depth	Chapter 18

Sequencing warning. Connect no telemetry source until the corresponding approved-state field exists and is populated for the Tier 3–4 estate. Telemetry before records is the most common way this module fails.

Novelty status. The mechanism is not novel; it applies existing posture-management and policy-as-code practice. What is specific to IRGF is the comparison target — a named Decision under a stated tier — plus the AI-specific drift categories and the feedback into pattern decisions.

M13 — Drift Taxonomy

Table 101.

Field	Specification
Purpose	Classify divergence so that it routes to an owner who can act on it, and so that model problems and data problems are diagnosable apart
Inputs	Comparison output from M12
Mechanics	Classify into one of eight categories: structural, configuration, integration and dependency, security, data lineage and governance, AI-model behavioral, policy and regulatory, cost. Route to the category owner. Set severity from category and tier. Close to one of four endpoints: revert, ratify, except, escalate
Outputs	Classified drift records with named owners and endpoints
Accountable	Varies by category; architecture holds structural and integration, security holds security, data owner holds lineage
Tier variation	Severity and response window scale with tier. Agent boundary divergence is an incident at any tier
Depends on	M12
Feeds	M4, M7, M9
Failure signal	Findings age without resolution; owners disable notifications
In depth	Chapter 19
Two categories were deliberately excluded as not telemetry-detectable: capability drift and strategic drift. Both are assessed rather than observed, and claiming otherwise would overstate what the mechanism can do.	

The separation of category 5 from category 6 is one of the framework’s few original mechanisms. Its value is diagnostic: when behavior changes, the first question is whether the model changed or its grounding changed, and separated signals answer it immediately.

M14 — Policy- and Evidence-as-Code

Table 102.

Field	Specification
Purpose	Make preventive rules executable rather than documented, and capture gate evidence as a byproduct of operation rather than by assembly
Inputs	M18 approved-state fields; policy engine; CI/CD and IAM integration
Mechanics	1. Write preventive rules only where the underlying policy is objectively evaluable. 2. Deploy each in report-only mode for a full delivery cycle. 3. Provide a defined exception path for every enforced rule. 4. Separately, wire evidence capture into the pipelines that already produce it
Outputs	Enforced preventive rules; automatically captured evidence
Accountable	Architecture with platform engineering
Tier variation	Rules apply at all tiers. Corrective automation remains Tier 1–2 only

Field	Specification
Depends on	M12, M18
Feeds	M7 gate evidence; M12 Evaluate
Failure signal	Legitimate work blocked with no recourse, which drives people to route around the control
In depth	Chapter 18, §18.5–18.6
Roughly two-thirds of the minimum evidence set is automatable. The remaining third — human oversight confirmation, the substance of control mapping, change classification judgment, knowledge preservation — requires human input by nature. This module is where real burden reduction lives, since gate consolidation did not deliver it.	

34.3 Operating Model

M15 — Two Standing Bodies

Table 103.

Field	Specification
Purpose	Provide review capacity for high-tier work without creating a second bureaucracy alongside existing architecture governance
Inputs	Existing ARB; M5 tiers to determine what escalates
Mechanics	1. Expand the ARB's mandate to cover AI as a fifth domain, and its membership to include AI expertise. 2. Stand up an AI Governance Body activated for Tier 3–4 only, absorbing the independent-challenge function. 3. Keep architecture decisions and deployment authorization with different bodies. 4. Do not create separate data, security, model-risk or transformation councils
Outputs	Tier 3–4 G2 decisions from ARB; Tier 3–4 G3 and G4 decisions from the AI Governance Body
Accountable	Enterprise Governance Sponsor for the structure; each body for its decisions
Tier variation	The bodies exist for Tier 3–4. Tier 1–2 is federated and never reaches them
Depends on	M2, M5
Feeds	M7, M16, M17
Failure signal	ARB agendas full of Tier 1 systems, meaning tiering or federation is not working; or lead times long enough that teams route around
In depth	Chapter 20

Known limitation. The model does not scale on volume within a single jurisdiction. Only the geography axis was addressed, through a conditional Region or Legal Entity level. Expect to hit this if adoption succeeds, and respond by improving tiering accuracy and expanding patterns before adding capacity.

M16 — Decision Rights and RACI

Table 104.

Field	Specification
Purpose	Establish who decides what at which tier, so that federation is a defined delegation rather than an absence of governance

Field	Specification
Inputs	M5 tier; role catalogue
Mechanics	1. Assign approval authority per decision type across four tiers. 2. Assign responsibility across both lifecycles. 3. Check that no role is accountable at more than roughly a third of stages. 4. Keep the Enterprise Governance Sponsor accountable for principle-setting and cross-body conflict only
Outputs	Decision rights matrix; two lifecycle RACIs
Accountable	Enterprise Governance Sponsor
Tier variation	This module is the tier variation; every row scales
Depends on	M5, M15
Feeds	M7, M17
Failure signal	Decisions made by whoever is available rather than by the named authority; or every senior leader accountable for everything
In depth	Chapter 21
Open gap. The Security Architect is consulted and can block, but is never accountable anywhere in the RACI. Independent review found this under-weighted relative to security's role in agent failure modes. Elevating the role to co-accountable at G3 for agentic systems is a defensible local variation.	

M17 — Escalation and Exceptions

Table 105.

Field	Specification
Purpose	Provide defined routes for what does not fit, so that non-compliance is recorded and time-boxed rather than silent
Inputs	Gate outcomes; drift alerts; tier changes
Mechanics	1. Define six escalation triggers with paths and timeliness expectations. 2. Require every exception to name the requirement, the justification, a compensating control, and a hard expiry. 3. Time-box all exceptions to 90 days. 4. Give Tier 4 exceptions no renewal path — expiry returns the system through full re-authorization. 5. Keep one register and report the aggregate
Outputs	Escalation records; a single exception register; portfolio-visible exception counts
Accountable	AI Governance Lead for the register; the granting authority for each exception
Tier variation	Approval authority scales; the 90-day ceiling does not. Tier 4 alone has no renewal
Depends on	M15, M16
Feeds	Portfolio health reporting
Failure signal	Exception count rising quarter on quarter, or the same exception type recurring across teams — both mean the requirement is wrong or a pattern is missing
In depth	Chapter 22
Recurring exceptions are design feedback, not compliance failures. Review the recurring types quarterly as candidates for a framework variation or a new pattern.	

34.4 Evidence

M18 — Primary Records

Table 106.

Field	Specification
Purpose	Hold every governed fact once, with one owner and one location, so that derived views can be compiled and approved state can be compared
Inputs	Outputs of M7, M8, M10, M12
Mechanics	1. Maintain sixteen primary records, each with a named accountable owner. 2. Never permit an unowned state. 3. Supersede rather than edit, so history survives. 4. Make machine-comparable fields into fields, not prose. 5. Give every record a stable identifier that others cite
Outputs	Sixteen record types; the approved state the Control Plane reads
Accountable	Per record; see the table in Chapter 24
Tier variation	Depth scales sharply. A Tier 1 assurance summary is a pass reference; a Tier 4 one carries the full evaluation chain
Depends on	M1, M3
Feeds	M12, M19
Failure signal	An ADR field a machine cannot compare; a record with no owner; an ADR edited in place
In depth	Chapter 24

M19 — Derived Views

Table 107.

Field	Specification
Purpose	Present governance status without duplicating the facts that produce it
Inputs	M18
Mechanics	1. Compile four views entirely by reference: Business Capability Map, AI System Card, Benefit Map, Assurance Dashboard. 2. Author nothing in them. 3. If a view cannot be compiled automatically, do not create it
Outputs	Four compiled views
Accountable	No independent owner; accountability sits with the owners of the source records
Tier variation	None
Depends on	M18
Feeds	Reporting, and the seventeen questions in Chapter 1
Failure signal	Any field in a view differs from its source record, which means someone is maintaining it by hand
In depth	Chapter 24, §24.2

The AI System Card is the artifact practitioners most want and the one most dangerous to maintain manually. A confidently displayed summary that is quietly stale is worse than a visible absence, because nobody checks it.

M20 — Reference Pattern Library

Table 108.

Field	Specification
Purpose	Reduce governance load by pre-approving architectures, so that a familiar problem does not require a novel review
Inputs	M9 promotions; drift signals from M13
Mechanics	1. Publish patterns with explicit variation points — what may change and what may not. 2. Assess conformance as conformant, modified, or novel. 3. Grant the automated G2 path to unmodified use at Tier 1–2 only. 4. Review annually and on repeated drift. 5. Deprecation triggers re-review of dependent systems
Outputs	Ten patterns, each with problem, architecture, required controls, primary risk and applicability
Accountable	Architecture Review Board
Tier variation	Fast path at Tier 1–2 only. Patterns themselves apply at all tiers
Depends on	M9
Feeds	M7 at G2; M14 pattern-conformance rules
Failure signal	Conformance inflation — materially modified architectures claimed as conformant to obtain the fast path. The counter is specificity in the pattern, not suspicion of the team
In depth	Chapter 35
Track pattern coverage. The share of new builds using an unmodified pattern is the best available proxy for whether the library is reducing load, and it is the number that justifies continued investment in maintaining it.	

34.5 Module dependency summary

Table 109.

Module	Depends on	Feeds
M1 Core Model	—	all
M2 Architecture Domains	M1	M7, M9, M15, M20
M3 Three Views	M1	M12, M18, M19
M4 Risk Dimensions	M1	M5
M5 Tiering Method	M4	M7, M10, M12, M15, M16, M18
M6 Independent Verification	M5	confidence in all tier-scaled decisions
M7 AI Build Lifecycle	M4, M5, M6, M20	M10, M12, M18
M8 Transformation Lifecycle	M2, M5	M7
M9 Architecture Lifecycle	M2, M20	M20
M10 Agent Governance Chain	M5, M7	M11, M12, M18
M11 Autonomy Change Control	M4, M5, M10	M7, M12
M12 Control Plane	M18, M5, M13, M14	M4, M7, M9
M13 Drift Taxonomy	M12	M4, M7, M9
M14 Policy- and Evidence-as-Code	M12, M18	M7, M12
M15 Two Standing Bodies	M2, M5	M7, M16, M17
M16 Decision Rights and RACI	M5, M15	M7, M17
M17 Escalation and Exceptions	M15, M16	portfolio reporting
M18 Primary Records	M1, M3	M12, M19
M19 Derived Views	M18	reporting
M20 Reference Pattern Library	M9	M7, M14

Two dependency facts are worth extracting from this table. **M5 feeds more modules than anything else**, which is why classification accuracy matters more than any individual review. And **M12 depends on M18 before it depends on telemetry**, which is why the Control Plane is a later adoption phase rather than an early tooling purchase.

Chapter 35. The Reference Pattern Library

Ten pre-approved patterns, what each governs, and how the library earns its keep.

35.1 What a pattern is for

A reference pattern is a pre-approved architecture that a Tier 1–2 system may adopt unmodified in order to use the largely automated G2 path. That is the library’s governance function, and it is the framework’s principal genuine reduction in burden.

A pattern that cannot be assessed for conformance provides no such reduction. **A usable pattern publishes what may vary and what may not.** “Use a gateway” is a preference; a pattern specifies the components, the trust boundaries, the required controls, and the configuration points that are open.

35.2 The ten patterns

Enterprise RAG

Problem. Ungoverned proliferation of retrieval assistants, each with its own grounding source and no lineage. **Architecture.** Shared vector store, governed knowledge base, retrieval gateway. **Required controls.** Data Lineage Record per source; S4 gate; maximum staleness defined per source. **Primary risk.** Grounding-source staleness or poisoning. **Applicability.** Tier 1–3; fast-path eligible at Tier 1–2. **Watch for.** Sources added after approval without re-running lineage. This is the most common drift for this pattern.

Secure LLM Gateway

Problem. Direct, ungoverned application-to-model calls with no logging or control point. **Architecture.** Centralized gateway providing authentication, logging, rate limiting, and redaction. **Required controls.** Gateway logs feed Control Plane observation; model allow-list enforced at the gateway. **Primary risk.** The gateway becomes a single point of failure and a performance constraint. **Applicability.** All tiers; foundational infrastructure rather than a per-system choice. **Watch for.** Bypass routes. A gateway with an undocumented direct path provides assurance it does not deliver.

AI Agent

Problem. Ad hoc tool-access grants with no recorded authority boundary. **Architecture.** Agent Card-governed identity, permission, and tool chain (Chapter 16). **Required controls.** Machine-readable boundary; kill-switch at Tier 3–4; preventive control for standing authority. **Primary risk.** Autonomy creep, confirmed in the framework’s stress-testing as a real failure mode. **Applicability.** Tier 2–4. Never self-service at Tier 1, because an agent at Tier 1 usually indicates mis-scoring.

Human-in-the-Loop

Problem. High-consequence decisions made without a genuine human check. **Architecture.** Mandatory approval step before a consequential action executes. **Required controls.**

Documented oversight point; override-rate instrumentation. **Primary risk.** Automation bias hollowing out the check, confirmed in stress-testing. **Applicability.** Tier 3–4 decision-consequence systems. **Watch for.** An approve button that requires nothing the system did not already supply. Design the interaction so the human must contribute something.

AI Decision Support

Problem. Advisory AI mistaken for authoritative. **Architecture.** Recommendation-only output, no direct action capability. **Required controls.** D2 capped by architecture, not by policy alone. **Primary risk.** The formal-versus-effective autonomy gap, unresolved in the framework. **Applicability.** Tier 1–3.

Private or Sovereign AI

Problem. Sensitive data cannot leave the enterprise boundary. **Architecture.** Self-hosted or network-isolated model with no external egress. **Required controls.** Lineage record confirming no egress; data-location policy rule. **Primary risk.** Higher operational burden and capability lag against frontier models. **Applicability.** Tier 3–4 with high D5.

AI SaaS Consumption

Problem. Vendor AI features with no visibility, a confirmed structural gap. **Architecture.** Vendor feature behind a procurement and extended-ADR due-diligence gate. **Required controls.** Vendor evidence requirement; Tier-2 cap where evidence is unavailable. **Primary risk.** Opacity and evidence stalemate. **Applicability.** All tiers; the primary entry point for procured AI and the one most often ungoverned.

Multi-Model Routing

Problem. Single-vendor lock-in with no fallback. **Architecture.** Gateway routes across models by cost, capability, or risk. **Required controls.** Each candidate model separately ratified enterprise-wide. **Primary risk.** Inconsistent behavior across routed models complicates D1–D5 scoring, because the system's behavior depends on which model served the request. **Applicability.** Tier 1–2 primarily.

Responsible AI Evaluation

Problem. Bias, fairness, and toxicity testing done inconsistently or not at all. **Architecture.** Standardized, reusable evaluation suite invoked at S6. **Required controls.** AI Assurance Summary referencing results. **Primary risk.** Tier 1–2 systems receive only sampling-based checks. **Applicability.** All tiers; intensity scales by tier.

AI Observability

Problem. No visibility into live AI behavior after deployment. **Architecture.** Telemetry feeding Control Plane observation, tagged to the drift taxonomy. **Required controls.** Eight-category drift classification. **Primary risk.** The underlying mechanism is not novel; value lies in the AI-specific categorization. **Applicability.** Mandatory Tier 2–4; recommended Tier 1.

35.3 Maintaining the library

A pattern library decays without maintenance, and a decayed library is worse than none because teams self-certify against patterns that no longer reflect good practice.

Promotion. A novel architecture reviewed at G2 and judged reusable is a candidate. Promotion requires ARB ratification, published variation points, and named required controls.

Review. Annually at minimum, and whenever the Learn function reports repeated drift against a pattern. A pattern generating drift across multiple initiatives is a pattern problem.

Deprecation. Retiring a pattern triggers re-review of every system still depending on it. Without that trigger, deprecation is a documentation change with no effect.

[Practice recommendation] Track pattern coverage — the share of new builds using an unmodified pattern — as a standing metric. It is the best available proxy for whether the library is reducing governance load, and it is the number that justifies continued investment in maintaining it.

Chapter 36. Checklists and Job Aids

One-page references for the activities practitioners repeat.

36.1 Gate entry checklists

G1 — Qualification and Provisional Risk

- Named capability or stakeholder need, with current and target measure
- Expected outcome stated as a measure with a baseline
- Named business sponsor able to stop the work
- “No AI” alternative considered and rejection recorded
- Provisional D1–D5 scores present, marked provisional
- Data path identified, at least in principle
- No architecture or vendor commitment embedded in the canvas

G2 — Architecture and Data Readiness

- ADR complete, with model, hosting, grounding, and boundary named in comparable form
- Alternatives considered and rationale recorded
- Pattern conformance assessed: conformant, modified with detail, or novel
- Data Lineage and Sensitivity Record complete, each source named
- Each grounding source has an owner, a refresh cadence, and a maximum staleness
- Risk score revisited against now-known architecture, and tier confirmed or changed
- For procured systems: vendor due-diligence fields complete, or Tier-2 cap or documented acceptance recorded
- For agentic systems: authority boundary drafted

G3 — Deployment Authorization

- Technical evaluation complete, pass criteria defined before the run
- AI Assurance Summary at the depth the tier requires
- Control Matrix complete, controls mapped to specific risks
- Named accountable owner recorded
- Human oversight point confirmed for Tier 2 and above
- Machine-readable permission boundary in place for agentic systems

- Kill-switch conditions defined and separately approved at Tier 3–4
- Any conditions carry a named owner and a date

G4 — Material Change

- Change classified against the change-type table, with reasoning
- Classification confirmed by a second person at Tier 3–4 where cosmetic is claimed
- Delta assurance scoped, with what was and was not re-run recorded
- Risk score updated where any dimension is touched
- Post-change tier determined, and approval routed to that tier's authority

G5 — Retirement

- Retention or deletion confirmed against the schedule
- Access revoked, including machine identities
- Downstream consumers identified from telemetry, not only from documentation
- Evidence set archived and retrievable
- Knowledge preservation note recorded

36.2 Risk scoring worksheet

Table 110.

Dimension	Question	Score	Justification
D1 Decision Consequence	How bad if wrong, assuming the output is acted on?		
D2 Autonomy	How much human judgment sits between output and effect?		
D3 Reversibility Deficit	Can the consequence be undone, not the system state?		
D4 Exposure and Scale	How many affected, how far, can effects cascade?		
D5a Data Sensitivity	How sensitive are the inputs?		
D5b Model Uncertainty	How well understood is the behavior?		
D5 composite	Higher of D5a and D5b		

Impact = average(D1, D4, D5), rounded up = _____ **Control Deficit** = average(D2, D3), rounded up = _____ **Matrix tier** = _____ **Safety override applies?** D1 = 4, or (D2 = 4 and D3 = 4) → floor at Tier 3. Yes / No **Final tier** = _____ **Countersigned by** _____ **Date** _____
Scores adjusted _____

Countersigner's focus list

- Any 3 where 4 is arguable, particularly on D1
- Any D3 justified by rollback capability, not consequence
- Any D2 of 1 or 2 where D1 is 3 or 4

- Any matrix result sitting one point from a tier change
- D4 counting direct users rather than reach of consequence

36.3 Drift triage card

Table 111.

Step	Action
1	Classify into one of the eight categories
2	Identify the affected system and its current tier
3	Set severity from the tier-and-category table (Chapter 19, §19.4)
4	Route to the accountable owner for that category
5	Determine endpoint: revert, ratify, except, or escalate
6	Produce the corresponding record; no finding closes without one

Exception: agent boundary divergence is an incident at any tier.

36.4 Change classification quick reference

Table 112.

If the change...	Then
Adds a tool, raises a limit, removes a human step, or raises an escalation threshold	Major
Changes provider or model family	Major
Changes training data	Major, full revalidation
Changes user population	Major, re-enter deployment authorization
Changes business purpose	Major, re-enter at S0
Changes a system instruction substantively	Material
Adds a grounding source at or above current sensitivity	Material
Updates model version at Tier 3–4	Material
Updates model version at Tier 1–2	Minor
Changes formatting only	Minor
Lowers a limit or threshold	Minor

Chapter 37. Glossary

Agent Card. Primary record holding an agent's identity, authorized tools, action boundaries, escalation thresholds, and kill-switch conditions.

AI Architecture. The domain concerned with how AI capability is sourced, hosted, grounded, orchestrated, observed, and constrained. Governs existence, connection, and authority boundary; not internal construction.

AI Governance Body. Standing body authorizing deployment and material change for Tier 3–4 systems. Absorbs the independent-challenge function of a model risk committee.

AI System Card. Derived view compiling governance status from primary records. Never independently authored.

Approved state. The configuration recorded in a Decision or ADR, against which actual state is compared.

Architecture Input Package. The structured information architecture supplies to portfolio governance, which portfolio must consume but need not defer to.

Control Deficit. Axis of the tiering matrix; average of D2 and D3, rounded up.

Control Plane. The seven-function loop comparing deployed state against approved state: observe, compare, evaluate, alert, recommend, enforce, learn.

Core Model. The eight entities without which the framework's traceability claims do not function: Intent, Capability, AI System, Decision, Risk, Control, Outcome, Assurance.

D1–D5. The five composite risk dimensions: Decision Consequence Severity, Autonomy, Reversibility Deficit, Exposure and Scale, Data Sensitivity and Model Uncertainty.

Derived view. A record compiled by reference from primary records, never independently maintained.

Drift. Divergence between deployed state and approved state, or between approved state and an external requirement that changed.

Effective autonomy. The autonomy a system carries in practice, which may exceed its recorded D2 score where human review is nominal.

Evidence-as-code. Automatic capture of gate evidence as a byproduct of normal operation.

G1–G5. The five governance gates: Qualification and Provisional Risk, Architecture and Data Readiness, Deployment Authorization, Material Change, Retirement.

Impact. Axis of the tiering matrix; average of D1, D4, and D5, rounded up.

Kill-switch condition. An objectively evaluable, pre-approved condition permitting automated shutdown. The sole exception to the prohibition on automated corrective action at Tier 3–4.

Machine-readable boundary. An agent's authority expressed so that a policy engine can compare it against live permissions.

Minimum Viable Framework. The subset of IRGF judged to deliver most of its value, centered on independent risk-tier verification.

Policy-as-code. Machine-executable expression of a policy rule, enabling preventive control.

Primary record. An artifact with exactly one authoritative owner and one location.

Reference pattern. A pre-approved architecture that, used unmodified, enables the automated G2 path at Tier 1–2.

Regulatory Overlay Reference. Record mapping applicable regulatory regimes to affected systems.

S0–S10. The eleven AI build lifecycle stages, from Opportunity to Retirement.

Safety-override floor. Rule flooring a system at Tier 3 where $D1 = 4$, or where $D2 = 4$ and $D3 = 4$.

T0–T7. The eight transformation lifecycle stages, from Purpose to Capability Re-Baseline.

TG0. The initiative qualification checkpoint at T2, with a known enforcement gap addressed by retroactive detection.

Tier 1–4. Risk classification determining governance intensity, decision authority, evidence depth, and automation eligibility.

Appendix A. Known Limitations Register

Carried from the research report. Each limitation is stated with its operational consequence and the practical mitigation available.

Table 113. Known Limitations Register

#	Limitation	Operational consequence	Mitigation available
1	The framework has never been deployed in a real organization	Every threshold and cadence is a reasoned default, not a calibrated one	Instrument from day one (Chapter 29)
2	No systematic academic literature review was performed	Claims are reasoned, not literature-benchmarked	Treat design rationale as argument, not evidence
3	Risk-tier self-scoring remains gameable in the mid-range	Under-tiering is likely and the safety floor will not catch it	Countersignature and biased audit sampling (Chapter 7)
4	The two-body model does not scale on volume within one jurisdiction	Bodies become bottlenecks as adoption succeeds	Pattern expansion, delegation by decision type (Chapter 20, §20.6)
5	No cost, staffing, or service-level model exists	You cannot size the function from the framework	Measure your own; Chapter 29
6	The Core Model's minimality is asserted, not derived	The eight entities may be wrong in composition	Record cases the model cannot express
7	The maturity model is asserted, not observed	Progression may not work as described	Use it as a checklist, not a prediction
8	The NIST and ISO crosswalk is reasoned, not clause-verified	Do not rely on it for a compliance claim	Verify independently before external use
9	Individual due-process and contestability are a declared non-goal	The framework provides an interface point, not a mechanism	Address through your regulatory overlay and legal function
10	The narrowed autonomy rule and audit sampling have not been tested together	Boundary-widening-only classification may under-catch	Weight sampling toward agentic systems (Chapter 17, §17.4)
11	No mechanism exists for cumulative or structuring exposure	Sub-threshold agent actions can aggregate to material exposure	Add aggregate limits outside the framework (Chapter 13, §13.4)
12	Cosmetic versus substantive change has no bright-line test	Classification drifts toward the cheaper option	Second opinion above a tier; retrospective sampling (Chapter 14, §14.3)
13	Security Architect is consulted, never accountable	Security under-weighted in agent failure modes	Elevate to co-accountable locally (Chapter 21, §21.4)
14	Gate consolidation reduced labels more than burden	Do not promise a lighter process	Pursue burden reduction through evidence-as-code (Chapter 4, §4.4)

Appendix B. Departures From the Specification

Where this handbook recommends something the specification does not require. Record any you adopt as local variations, so that a future reader can distinguish framework from practice.

Table 114. Departures From the Specification

Recommendation	Chapter	Rationale
Instrument override rate for low-D2, high-D1 systems	6	Detects effective autonomy exceeding the recorded score
Score D5 composite as the higher of its two sub-scores	6	Prevents sensitivity being diluted by low uncertainty, or the reverse
Bias audit sampling rather than randomize it	7	Random sampling of a mostly-correct population wastes the sample
Cap open conditions per initiative at three	5	Prevents conditional approval becoming unconditional
Treat pilot as a deployment state with mandatory expiry	5	Closes the pilot exemption route to production
Run Control Plane comparison in non-production	10	Converts production incidents into development conversations
Define maximum acceptable staleness per grounding source	9	Makes silent knowledge decay detectable
Add aggregate limits for standing-authority agents	13, 16	Compensates for the cumulative exposure gap
Treat agent boundary divergence as an incident at any tier	17, 19	Prevents boundary erosion by case-by-case triage
Deploy preventive rules in report-only mode first	18	Produces calibration data before enforcement
Elevate Security Architect to co-accountable at G3 for agentic systems	21	Addresses a limitation the framework itself records
Route procurement intake by AI content, not contract value	21	Embedded AI arrives in inexpensive software
Require a reason when funding concurrence is withheld	20	A veto never examined is a veto never questioned
Add a funding-side TG0 check	23	Converts a detective control into a preventive one
Track pattern coverage as a standing metric	30	The best proxy for whether the library reduces load