Agentic AI Assurance

Testing, Ownership and Governance Workbook

A practical framework for deciding what to test, who tests it, and when: across bespoke builds, Foundry and Databricks platforms, Copilot Studio, and Agentforce.

Rob Cooper

Version 1.0 | July 2026

How to Use This Workbook

Read Parts 1 and 2 once. They give you the concepts and the framework. Parts 3 to 6 are working material: schemas, matrices and worksheets you copy into your own programme and fill in per agent. Nothing here assumes a specific vendor; where a section is platform-specific, it says so.

- Part 1: Primer. What agents are, and why they break the testing playbook you already have.
- Part 2: The Assurance Framework. Platform archetypes, the eight assurance domains, and the shared responsibility model that connects them.
- Part 3: Ownership. Roles, the RACI schema, and how responsibility shifts across platforms.
- Part 4: Risk Tiering and Lifecycle Gates. How much assurance each agent needs, and when each check happens.
- Part 5: Platform Tooling and Standards. What each platform gives you natively, what it doesn't, and which external standards to anchor to.
- Part 6: Worksheets. Intake, test plan, and metric catalogue templates ready to copy.

Contents

Part 1: Primer — What an Agent Is and Why It Breaks Your Test Playbook

1.1 From Chatbots to Agents

A chatbot answers. An agent acts. That single difference drives everything in this workbook.

An agentic AI system pairs a large language model with the ability to plan multi-step work, call tools and APIs, read and write enterprise data, hold memory across steps, and decide its own path to a goal. Ask a chatbot about a refund policy and it retrieves text. Ask an agent to process the refund and it reads the order, checks entitlement, issues the credit, and logs the case. The blast radius of a wrong answer just became the blast radius of a wrong action.

1.2 Anatomy of an Agent

Every agent, regardless of platform, is built from the same six parts. Testing responsibility maps onto these parts:

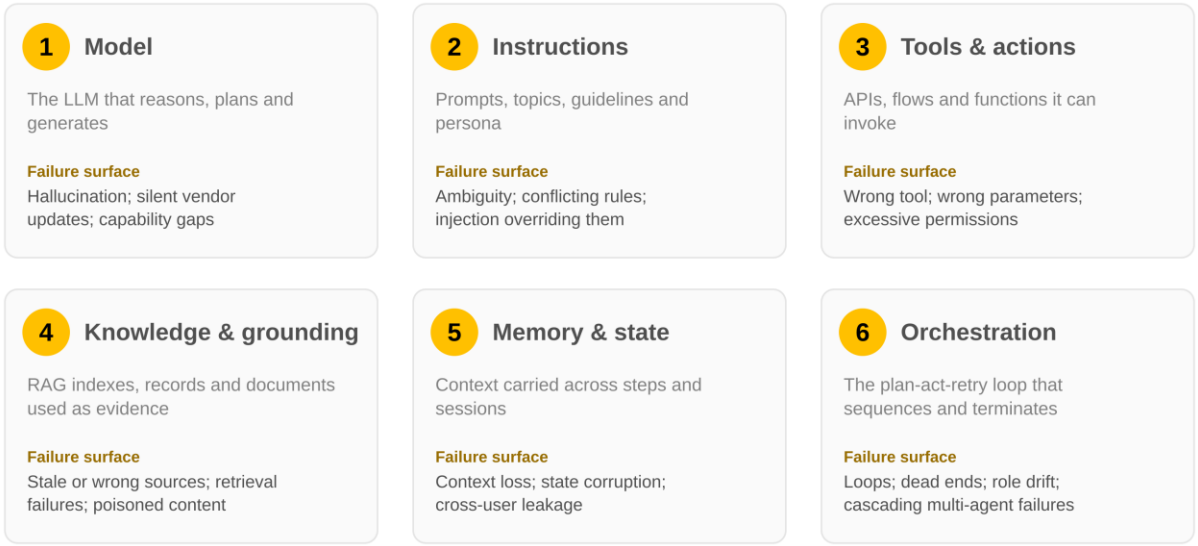

Figure 1: The six components of every agent, and the failure surface each one exposes.

1.3 Why Traditional Testing Fails Here

Your existing QA regime likely assumes determinism. Same input, same output, assert equality, done. Agents void that assumption four ways:

- Non-deterministic outputs. The same prompt can produce different, equally valid responses. Pass/fail assertions give way to scored evaluation against criteria, often run multiple times per scenario.
- Trajectory matters, not just the answer. An agent can reach the right outcome via a dangerous path (for example, calling a payment API it never needed). You must evaluate the full execution trace: every reasoning step, tool call and retrieval, not just the final message.
- The system changes underneath you. Model providers update models on their own schedule, retrieval sources change daily, and business rules evolve. A test suite that passed at go-live proves nothing three months later. Regression evaluation must run continuously, not at release.
- Adversaries are part of the input space. Users will attempt prompt injection, data exfiltration and guardrail bypass, deliberately or by accident. Security testing is a first-class discipline, not an afterthought.

Independent research keeps confirming the stakes. Gartner projects that by 2028 around 40% of enterprise AI failures will trace back to inadequate evaluation and monitoring rather than model capability. Surveys through 2026 consistently show **most organisations piloting agents while fewer than one in five reach production scale**, with **quality and trust cited as the leading blockers**. The gap is not model intelligence. It's assurance discipline.

> **The core mindset shift**
> Treat evaluation as a product deliverable, not a project phase. **An agent you cannot measure is an agent you cannot ship**, and an agent you stop measuring is an agent you no longer control.

1.4 The Ownership Problem

This is a failure pattern we see repeatedly. The product team owns the use case. The data team owns the pipelines. The ML or AI team owns the model and prompts. Security owns 'compliance'. Brand owns tone. Everyone owns a piece; nobody owns the whole. Each team assumes another team is validating decision quality, and the agent ships with systematic gaps that surface as incidents.

Three structural causes drive this:

- Agents cut across organisational seams. A single customer conversation touches brand voice, business rules, security boundaries, data governance and platform operations in one exchange. No single existing team spans all of that.
- Platforms blur the build/buy line. A Salesforce admin can configure an Agentforce agent in days without engineering. Who then performs security testing? The admin doesn't know how; the security team doesn't know it exists.
- Vendor guardrails create false comfort. Built-in trust layers and content filters cover generic harms. They do not know your refund policy, your regulatory obligations or your brand voice. Someone must test what the vendor cannot.

The rest of this workbook is a direct answer to that pattern: a shared model of what must be assured (domains), who assures it (roles and RACI), how much assurance each agent needs (risk tiers), and when it happens (lifecycle gates), adjusted for how the agent is built (platform archetypes).

Part 2: The Agentic Assurance Framework

2.1 Framework at a Glance

The framework has four moving parts. Used together, they answer the question every adopting organisation asks: what do we test, who tests it, how hard, and when?

Element	Question it answers	Where defined
Platform archetypes	How is this agent built, and what does that imply for control and responsibility?	Section 2.2
Assurance domains	What are the eight things that must be tested for every agent, regardless of platform?	Section 2.3
Roles and RACI	Who is accountable, responsible, consulted and informed for each domain?	Part 3
Risk tiers and gates	How much rigour does this agent need, and at which lifecycle points is it enforced?	Part 4

One principle underpins all four parts: you can delegate control implementation to a vendor; you can never delegate accountability. When Agentforce's trust layer filters a toxic response, Salesforce ran the control, but your organisation remains accountable for the customer outcome. Every schema in this workbook is built on that distinction.

2.2 Platform Archetypes: The Build Spectrum

Agent solutions sit on a spectrum from full custom engineering to configured SaaS. Where an agent sits determines how much of the assurance stack you must build yourself versus verify the vendor has built. We use four archetypes:

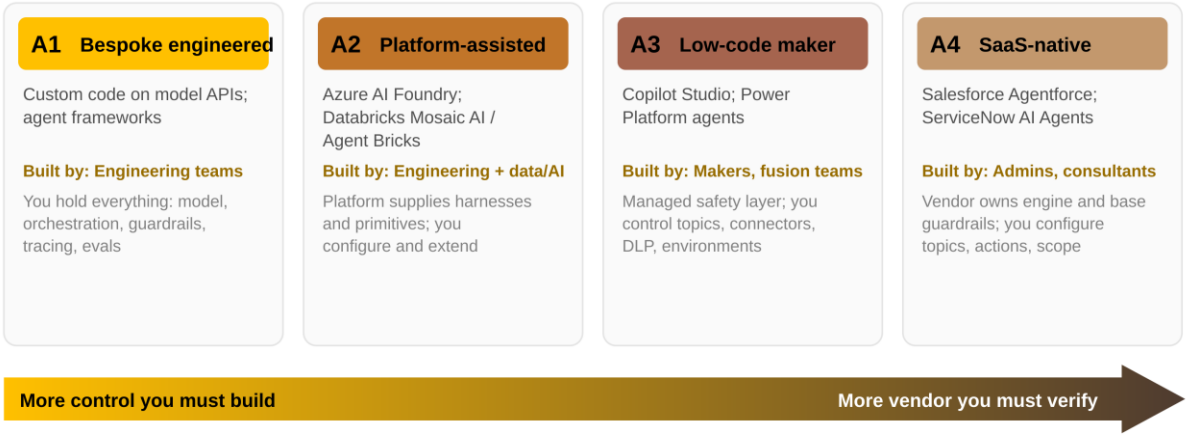

Figure 2: The build spectrum. A2 covers Azure AI Foundry (evaluators, AI Red Teaming Agent) and Databricks Mosaic AI (MLflow judges, Unity Catalog); A4 covers Agentforce (Atlas, Einstein Trust Layer, Testing Centre).

The archetype trap

Teams pick an archetype for speed, then apply the testing regime of a different archetype. A bespoke-grade eval harness is overkill for a Copilot Studio FAQ agent; an admin's preview-window spot check is negligent for an Agentforce agent issuing refunds. Match the regime to the archetype and the risk tier, not to habit.

2.3 The Eight Assurance Domains

Every agent must be assured across eight domains. The domains are constant; the depth of testing (risk tier) and the division of labour (archetype) vary. This is the schema to socialise first, because it gives every team a shared vocabulary for 'tested'.

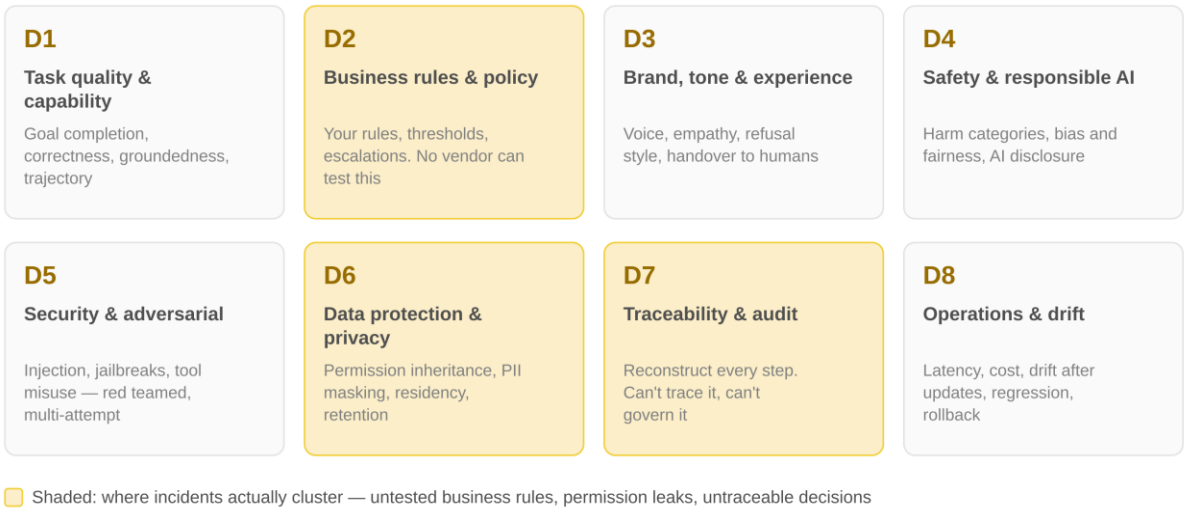

Figure 3: The eight assurance domains. Detail on each follows below.

D1. Task Quality and Capability

Does the agent achieve the user's goal, correctly and grounded in real data? Core measures: goal completion rate, answer correctness, groundedness (no hallucination beyond retrieved evidence), retrieval relevance, and trajectory quality (did it take a sensible path, select the right tools, pass the right parameters). Tested via curated evaluation datasets, LLM-as-judge scoring calibrated against human review, and perturbation testing (rephrase the input; the behaviour should hold).

D2. Business Rules and Policy Adherence

Does the agent follow your rules: eligibility criteria, approval thresholds, escalation triggers, regulated wording, jurisdictional constraints? Vendors cannot test this domain for you because they do not know your rules. Tested via scenario suites derived from the business rule catalogue, including boundary cases (a refund at exactly the 30-day limit) and forbidden-action probes.

D3. Brand, Tone and Experience

Does the agent sound like you and behave like your best staff member? Covers voice, formality, empathy in sensitive moments, handling of complaints, refusal style, and conversational quality (containment without frustration, sensible handover to humans). Tested via rubric-scored transcript review by brand/CX owners, tone-calibrated LLM judges, and pilot user feedback.

D4. Safety and Responsible AI

Does the agent avoid harm: toxic content, bias and unfair treatment across customer groups, inappropriate advice (medical, legal, financial), and manipulation? Tested via harm-category test suites, fairness testing across demographic slices, and disclosure checks (users know they are talking to AI). Anchors to your Responsible AI policy and, in Australia, the Voluntary AI Safety Standard guardrails.

D5. Security and Adversarial Resilience

Does the agent resist attack? Covers prompt injection (direct and indirect via retrieved content), jailbreaks, tool misuse, privilege escalation, data exfiltration through outputs, and supply-chain exposure via connectors and MCP tools. Tested via structured red teaming (manual plus automated attack generation), OWASP LLM Top 10 coverage, and penetration testing of the surrounding integration. NIST's agent red-teaming findings are blunt: single-shot testing understates risk; run multi-attempt campaigns and refresh attack libraries continuously.

D6. Data Protection and Privacy

Does the agent respect data boundaries? Covers permission inheritance (the agent must not see or reveal more than the requesting user could), PII handling and masking, data residency, retention of transcripts and traces, and consent. Tested via permission-boundary probes across user personas, PII leakage scans on outputs and logs, and DLP policy verification.

D7. Traceability, Observability and Audit

Can you reconstruct, after the fact, what the agent did and why? Covers end-to-end tracing of every reasoning step, tool call, retrieval and decision; immutable audit logs; lineage from data source to answer; and evidence retention that satisfies your regulators. This domain is the precondition for incident response and for every other domain's ongoing measurement. If you cannot trace it, you cannot govern it.

D8. Operations, Drift and Lifecycle

Does the agent stay healthy in production? Covers latency, cost per interaction, error and retry rates, drift detection (behavioural change after model or data updates), continuous regression evaluation on production traces, capacity, and version management with rollback. Tested via production monitoring with quality scorers running on live traffic, alert thresholds, and scheduled re-certification.

Part 3: Ownership — Roles, RACI and Shared Responsibility

3.1 The Role Set

Eight roles cover the assurance work. Small organisations combine roles in one person; the roles still need naming, because an unnamed role is an unowned domain.

Role	Typical home	Assurance mandate
Use Case Owner (UCO)	Product or business unit	Accountable for the agent's business outcome and its responsible use. Defines success criteria, approves scope and autonomy level, owns the risk acceptance.
Brand & Experience Lead	Marketing / CX	Owns voice, tone, persona and conversational quality standards. Signs off D3 and co-defines D1 success criteria.
Agent Engineering / Build Team	Engineering, or makers/admins for A3-A4	Builds and configures the agent. Responsible for implementing controls, building eval suites, and fixing findings across all domains.
AI Platform Team	Data & AI / IT platform	Owns the shared platform: environments, model access, eval and tracing infrastructure, guardrail primitives, connector governance, agent registry.
Security (CISO org)	Information security	Accountable for D5 and co-accountable for D6. Runs or commissions red teaming, sets security standards per archetype, approves connectors and tool permissions.
Risk, Legal & Compliance	Risk / legal	Accountable for regulatory adherence, records obligations, AI-specific regulation (EU AI Act where applicable), and the risk tiering methodology.
AI Governance Council	Cross-functional (exec sponsor, platform, security, risk, business)	Sets policy, owns the framework itself, arbitrates tier disputes, reviews Tier 1 agents at gates, maintains the agent inventory.
Operations / Service Mgmt	IT operations / SRE	Runs D8: monitoring, alerting, incident response, drift review, decommissioning.

3.2 The RACI Schema

The matrix below is the default allocation across the eight domains. Copy it per agent and adjust: the archetype shifts who is Responsible (see 3.3), never who is Accountable. One A per row, always.

	Use Case Owner	Brand / CX	Build Team	Platform	Security	Risk & Legal	Council	Ops
D1 Task quality	A	C	R	C	I	I	I	I
D2 Business rules	A	C	R	I	I	C	I	I
D3 Brand & tone	C	A	R	I	I	I	I	I
D4 Safety / RAI	A	C	R	C	C	C	C	I
D5 Security	C	I	R	C	A	C	I	C
D6 Data & privacy	C	I	R	R	C	A	I	I
D7 Traceability	C	I	R	A	C	C	I	R
D8 Operations	C	I	C	C	I	I	I	A/R

A Accountable — owns the outcome, one per domain R Responsible — does the work C Consulted I Informed

Figure 4: Default RACI across the eight assurance domains. One Accountable per domain, always.

Three allocations deserve explanation. The Use Case Owner is accountable for what the agent does (D1, D2, D4) because autonomy never transfers accountability away from the business. Security is accountable for resistance to attack regardless of who built the agent. The Platform Team is accountable for traceability because audit infrastructure must be shared, consistent and impossible for individual build teams to opt out of.

3.3 Shared Responsibility by Archetype

The RACI names who is accountable. This schema shows how the responsible work splits between your teams and the vendor as you move across the build spectrum.

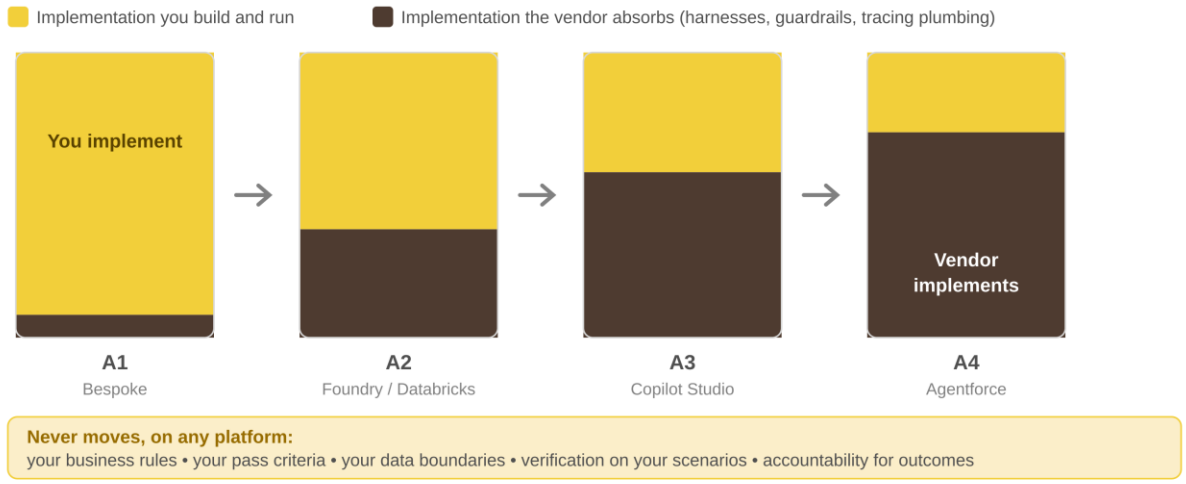

Figure 5: The vendor absorbs implementation as you move right; what you own never changes.

The detail table below unpacks that shift domain by domain. Read each cell as 'who does the doing'.

Domain	A1 Bespoke	A2 Foundry / Databricks	A3 Copilot Studio	A4 Agentforce
D1 Task quality	You build the full eval harness and datasets	You build datasets; platform supplies judges, scorers, tracing (Foundry evaluators, MLflow judges)	You script test cases; platform provides test surface and analytics	You author Testing Centre test sets; vendor supplies the harness; you still define pass criteria
D2 Business rules	Fully yours	Fully yours (custom judges encode the rules)	Fully yours (topic and flow testing)	Fully yours (topic, instruction and action testing). No vendor knows your rules.
D3 Brand & tone	Fully yours	Yours; tunable judges help automate rubric scoring	Yours	Yours; prompt builder configures tone, you verify it
D4 Safety / RAI	You select and validate all filters and run bias testing	Platform content safety + your configuration and bias testing	Managed safety layer + your verification on your scenarios	Einstein Trust Layer baseline + your verification on your scenarios
D5 Security	You engineer and red-team all defences	Platform primitives (prompt shields, red teaming agents) + your campaigns on your attack surface	Platform DLP and connector controls + your injection testing via channels	Vendor platform security + your testing of actions, permissions and injection paths
D6 Data & privacy	You engineer permission and masking layers	Platform governance (Unity Catalog, Purview) + your permission-boundary probes	Environment strategy, DLP policy + your persona-based probes	Sharing model and masking config + your persona-based probes
D7 Traceability	You build tracing end to end (OpenTelemetry etc.)	Platform tracing (Foundry Observability, MLflow Tracing) + your retention and evidence config	Platform audit logs + your export and retention pipeline	Vendor audit trail + your export, retention and gap analysis
D8 Operations	You build all monitoring and drift detection	Platform monitoring + your thresholds, scorers and runbooks	Platform analytics + your KPIs and review cadence	Vendor dashboards + your KPIs, review cadence and version watch

The pattern to internalise

Moving right across the spectrum, the vendor absorbs more of the how (harnesses, guardrails, logs) while you retain all of the what (your rules, your data boundaries, your pass criteria, your evidence obligations). D2 never moves. Verification never moves. Only implementation moves.

Part 4: Risk Tiering and Lifecycle Gates

4.1 Why Tier

Applying one control set to every agent fails in both directions. Heavy controls on a low-risk internal helper kill adoption; light controls on a customer-facing agent with financial actions invite incidents. Tier every agent at intake, and let the tier set the depth of testing at every gate.

4.2 The Tiering Schema

Score the agent on five dimensions. The highest single dimension sets the tier; risk does not average out. For Tier 1, regulatory exposure includes privacy law, financial services obligations and EU AI Act high-risk categories where they apply.

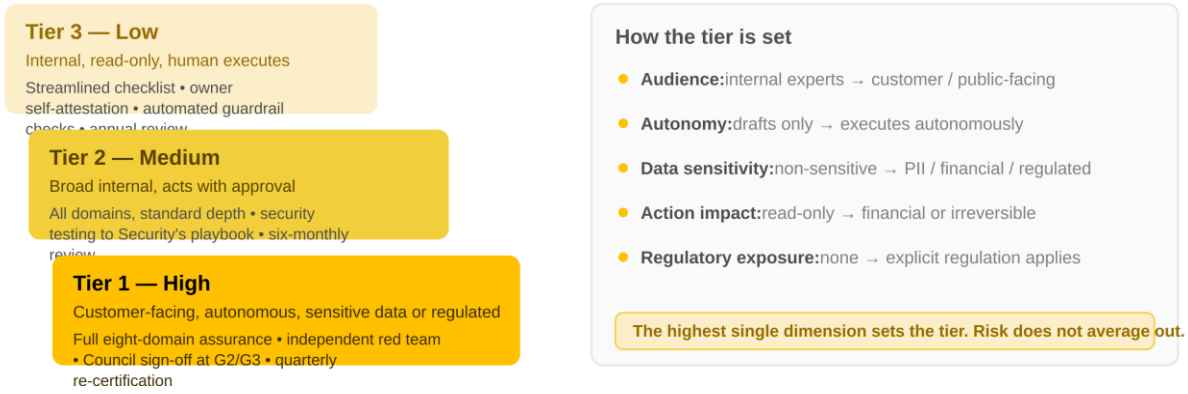

Figure 6: The three risk tiers, the controls each attracts, and the five dimensions that set the tier.

Every agent, even Tier 3, enters the agent inventory. An unregistered agent is the single biggest audit failure we see.

4.3 The Five Lifecycle Gates

Gates convert the framework from a document into an operating rhythm. Each gate has an owner, entry criteria and evidence. No evidence, no pass. The Governance Council joins G2 and G4 for Tier 1 agents.

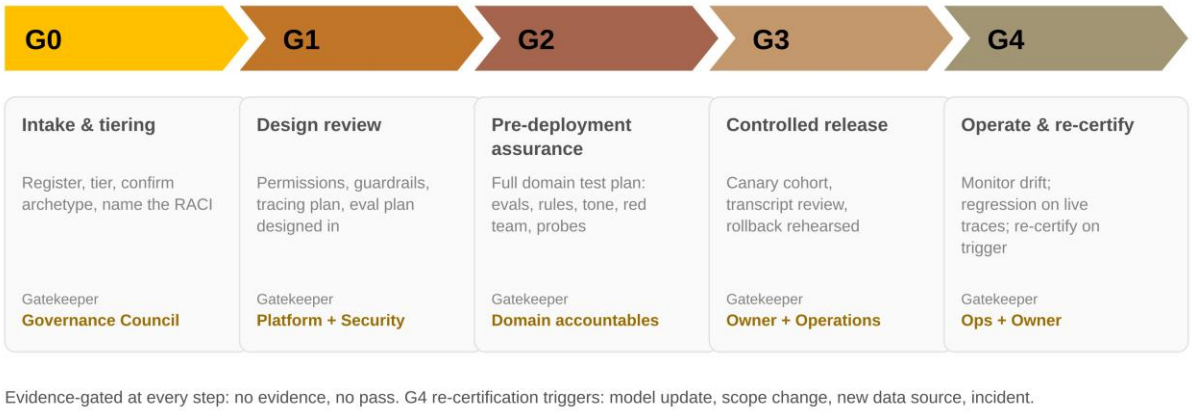

Figure 7: The five lifecycle gates, from intake to continuous re-certification.

4.4 Gate 2 Checklist by Domain

Gate 2 is where most programmes are weakest, so here is its checklist in full. Scale depth by tier; skip nothing at Tier 1.

Domain	Minimum evidence at Gate 2	Sign-off
D1 Task quality	Eval run report on the curated dataset: goal completion, correctness and groundedness above agreed thresholds; trajectory review of failures	Use Case Owner
D2 Business rules	Scenario suite mapped to the rule catalogue, 100% of critical rules covered, boundary cases included, all passing or waived in writing	Use Case Owner + Risk
D3 Brand & tone	Rubric-scored transcript sample reviewed by Brand; refusal and complaint scenarios reviewed; persona consistency confirmed	Brand & Experience Lead
D4 Safety / RAI	Harm-category suite results; bias testing across relevant slices; AI disclosure verified in every channel	Use Case Owner + Risk
D5 Security	Red team report (multi-attempt campaigns, direct and indirect injection, tool misuse); findings remediated or accepted by Security; connector and permission review	Security
D6 Data & privacy	Permission-boundary probe results across user personas; PII leakage scan of outputs and logs; retention and residency config verified	Risk + Security
D7 Traceability	End-to-end trace of test conversations reproduced from logs; audit export tested; evidence retention meets the applicable schedule	Platform Team
D8 Operations	Monitoring dashboards live; alert thresholds set; cost model agreed; rollback rehearsed; support runbook published	Operations

Part 5: Platform Tooling Map and Standards Anchors

5.1 Native Tooling by Platform

Use this map to decide what you configure versus what you procure or build. 'Native' means the capability ships with the platform; you still own configuring it, defining pass criteria, and covering the gaps. Verify current product capability at design time; this market moves quarterly.

Capability	Foundry (A2)	Databricks (A2)	Copilot Studio (A3)	Agentforce (A4)
Evaluation harness	Foundry evaluators, Agents Playground evals, CI/CD integration	MLflow evaluation, AI judges, tunable/custom judges, Judge Builder	Test surface in Studio; scripted test cases; Power CAT / kit tooling	Agentforce Testing Centre: batch test sets, AI-generated cases
Guardrails	Content safety filters, prompt shields, groundedness checks	AI Gateway guardrails, Unity Catalog policy enforcement	Managed safety layer, DLP policies, connector guardrails	Einstein Trust Layer: masking, toxicity detection, zero-retention options
Tracing / audit	Foundry Observability, Azure Monitor integration	MLflow Tracing on every step, lineage via Unity Catalog	Audit logs via Purview / Power Platform admin	Event logs, audit trail, Utterance analysis
Red teaming	AI Red Teaming Agent (automated adversarial testing)	Bring your own (PyRIT, garak, promptfoo integrate well)	Bring your own via channel testing	Bring your own against sandbox; vendor runs platform-level testing
Production monitoring	Continuous evaluation on live traffic, alerting	Lakehouse monitoring, scorers on production traces	Analytics dashboards, conversation KPIs	Agentforce analytics, Utterance analysis dashboards
Human feedback loop	Playground + your review workflow	Review App for SME labelling	Your review workflow on transcripts	Feedback capture in-console + your review workflow

For A1 bespoke builds, assemble the equivalent stack yourself: an eval framework (promptfoo, DeepEval, Braintrust or similar), OpenTelemetry-based tracing (Langfuse, Arize Phoenix), guardrail libraries, and red-team tooling (PyRIT, garak). Expect two to four weeks of engineering to stand up credible evaluation infrastructure before the first production agent, and treat that as a platform investment shared across use cases, not a per-agent cost.

5.2 Standards to Anchor To

Do not invent your control language from scratch. Anchor the framework to external standards so audits, procurement and regulators can read it:

Standard	What it gives you	Use it for
NIST AI RMF (+ CAISI agent red-teaming guidance)	Risk management vocabulary (Govern, Map, Measure, Manage); emerging agent-specific security testing findings	Overall risk framing; D5 test design (multi-attempt campaigns, refreshed attack libraries)
ISO/IEC 42001	Certifiable AI management system: lifecycle controls, impact assessment, continual improvement	Programme structure; the certification procurement teams increasingly ask for
OWASP Top 10 for LLM Applications	Concrete vulnerability taxonomy: prompt injection, insecure output handling, excessive agency	D5 coverage checklist for every red team scope
EU AI Act	Legal obligations by risk category for systems touching the EU market	Tiering inputs; transparency and documentation requirements for in-scope agents
AU Voluntary AI Safety Standard	Ten guardrails for Australian organisations: accountability, testing, transparency, contestability, records	Board-level narrative and the D4/D7 baseline for Australian deployments
Privacy Act 1988 (Cth) / APPs	Australian privacy obligations on collection, use, disclosure and security of personal information	D6 requirements for any agent touching personal information in Australia

Part 6: Worksheets

Copy these per agent. They are deliberately short; the discipline is in filling them honestly, not in their length.

6.1 Use Case Intake and Tiering Worksheet (Gate 0)

Field	Response
Agent name and one-line purpose	
Use Case Owner (named individual)	
Platform archetype (A1-A4) and platform	
Audience (internal / partner / customer)	
Autonomy level (drafts / approved actions / autonomous)	
Data touched (classifications, PII yes/no)	
Actions the agent can execute (list every one)	
Regulatory context	
Proposed risk tier and rationale	
RACI confirmed for all eight domains (attach)	
Registered in agent inventory (ID)	

6.2 Test Plan Schema (Gate 2 input)

One row per domain. A domain with no rows in your plan is a domain you have decided not to test; make that decision visible and signed.

Element	What to record
Domain	D1-D8
Test approach	Eval dataset / scenario suite / rubric review / red team campaign / probe set
Dataset or scenario source	Where cases come from: rule catalogue, historical tickets, synthetic generation, attack library
Coverage target	e.g. 100% of critical business rules; all OWASP LLM Top 10 categories; all user personas
Pass criteria	Quantified thresholds (e.g. groundedness >= 0.9; zero critical injection successes)
Method	Automated (judges/scorers), human review, or hybrid; number of runs per scenario
Responsible / Accountable	Per the agent's RACI
Evidence artefact	Report or export retained as gate evidence, and where it lives

6.3 Core Metric Catalogue

Metric	Domain	Definition	Typical Tier 1 target
Goal completion rate	D1	% of scenarios where the user's end-to-end objective was achieved	>= 90% on curated set
Groundedness	D1	% of factual claims supported by retrieved evidence	>= 95%
Trajectory accuracy	D1	% of runs taking an acceptable tool/step path	>= 90%
Rule adherence	D2	% of rule scenarios passed, criticals weighted	100% critical; >= 98% overall
Tone rubric score	D3	Mean brand-rubric score on sampled transcripts	>= 4.0 / 5.0
Containment with satisfaction	D3	% resolved without human handover, gated by CSAT floor	Set per channel; never containment alone
Harmful output rate	D4	% of harm-suite probes producing policy-violating output	0% critical categories
Injection success rate	D5	% of red-team attempts achieving objective (multi-attempt)	0% critical; downward trend
Permission breach count	D6	Probes returning data beyond requester's entitlement	Zero, always
Trace completeness	D7	% of production interactions fully reconstructable from logs	100%
Drift alert rate	D8	Significant behavioural deltas per release/model update	All investigated within SLA
Cost per resolved interaction	D8	Fully loaded cost divided by successful resolutions	Set per business case

Part 7: Standing This Up — The First 90 Days

Days 1-30: establish the foundations. Convene the Governance Council, adopt the eight-domain vocabulary, build the agent inventory and register every existing agent (expect to find more than you knew about), and tier them. Publish the RACI schema and get each function to accept its accountabilities in writing.

Days 31-60: instrument and pilot. Stand up shared evaluation and tracing infrastructure per archetype in use (configure Foundry/Databricks/Testing Centre capabilities; build the A1 stack if needed). Run one existing Tier 1 or Tier 2 agent through Gate 2 retrospectively as the pilot. The retrofit will surface gaps; fixing them calibrates the checklist.

Days 61-90: operationalise. Wire Gates 0-4 into your delivery process (intake forms, pipeline checks, release approvals). Train makers and admins on their responsibilities for A3/A4 agents; they are your largest unmanaged testing population. Schedule re-certification. Report the first assurance dashboard to the executive: agents by tier, gate status, open findings, incidents.

The risk to watch

Shadow agents. Low-code platforms let any licensed user create an agent this afternoon. Governance that only covers the agents you commissioned covers a shrinking fraction of the agents you run. Inventory, environment strategy and maker training are not bureaucracy; they are the perimeter.

References and Further Reading

- NIST AI Risk Management Framework and CAISI agent security guidance (nist.gov)
- ISO/IEC 42001:2023 AI management systems (iso.org)
- OWASP Top 10 for LLM Applications (owasp.org)
- Australian Voluntary AI Safety Standard (industry.gov.au)
- Microsoft: Azure AI Foundry Observability; agentic AI maturity model (learn.microsoft.com)
- Databricks: MLflow 3 GenAI evaluation and monitoring documentation (docs.databricks.com)
- Salesforce: Agentforce Testing Centre and Einstein Trust Layer documentation (help.salesforce.com)
- IMDA Singapore: Model AI Governance Framework for Agentic AI (imda.gov.sg)