Appendix: Beyond the First Quarter

The Mandate chapter covers what to do this quarter. This appendix covers the years that follow — when the investment pays back, what to build versus buy along the way, and how the operating rhythm scales as the organization grows.

The framework is the same. The cadence and the controls are not.

ROI TIMELINE AND PAYBACK SIGNALS

Cost per accepted change is the trailing indicator. Escape rate is the leading indicator — it typically drops four to six weeks before cost per accepted change does. A realistic timeline:

- **Months 1–3:** Cost per accepted change *rises* as verification work catches up to volume. This is expected; do not panic.
- **Months 4–6:** Cost stabilizes as gates absorb the load reviewers were carrying. Escape rate falls.
- **Months 7–12:** Cost per accepted change drops below the pre-investment baseline. Velocity *and* quality both improving.
- **Year two:** Compound returns. The freed capacity gets redirected into deeper work — refactoring, capability building, the things engineers want to do. BCG's "5% generate value from AI" finding becomes accessible.

If month 6 still shows rising cost with no escape-rate improvement, diagnose: usually weak intent clarity, weak gates, or unmeasured rework. The Triangle tells you which.

BUILD VS BUY VS OPEN-SOURCE

Capability	Build / Buy / OSS	Notes
Tier 0 static analysis	OSS	Linters, type checkers, secret scanners – no reason to build
Tier 1 contract gates	OSS for protocol (Protobuf, OpenAPI), build for business contracts	
Tier 2 invariant gates	Build	Invariants are domain-specific

Capability	Build / Buy / OSS	Notes
Tier 3 policy gates	Mix	Security scanning is buy/OSS; permission boundaries are build
Tier 4 behavioral gates	Build, with bought observability	Anomaly detection is hard to buy as a generic product
Tier 5 review	LLM-as-judge: buy (Anthropic / OpenAI APIs); human review: process	Agent framework
Buy / OSS	Claude Code, Cursor, Goose, Codex — minimal reason to build a framework	Agent permissions
Build	Constraints are organization-specific; sandbox via existing isolation tools	

QUARTERLY RHYTHM BY SIZE

The weekly scorecard and quarterly tier-map review scale with the organization. The rhythm itself does not change; the ownership does.

- **Smaller teams:** one scorecard, one tech lead owns metrics, gates are part of the sprint.
- **Mid-size organizations:** per-team scorecards, manager roll-up monthly, gates owned by a platform team.
- **Larger organizations:** governance committee, per-stage metrics, gates as a dedicated function with SLAs to product teams.

WHAT THE REFERENCE COMPANIES ACTUALLY USE

The Mandate chapter trims the rollout sequence to a five-bullet summary. This is the longer version, with the published examples that map to each step.

Company	Scale	Key Result	Reference
Dropbox	3,500 engineers, 550K+ files indexed	35% more PRs, 40% faster to production, reduced failures	Cursor case study
Airbnb	Test migration project	18-month project completed in 6 weeks	Cretik blog, 2025
Ramp	Hyperspeed development team	Parallel sessions via MCP integration	Anthropic case study
Zapier	800 employees	89% AI adoption, 800+ internal agents	Anthropic case study
Rakuten	Enterprise deployment	2x faster issue resolution	OpenAI case study

Six companies have published rollout patterns that converge on the same sequence:

Start with organic adoption on well-suited services. Let early adopters find the tools, but guide where they start. Swarmia's research recommends beginning with well-documented, actively maintained services owned by teams with strong engineering practices — not a blanket rollout. Dropbox let Cursor adoption start organically before formalizing anything. Zapier expanded from grassroots usage to company-wide access.

Formalize with champions. Identify power users and give them a role. Dropbox created "AI champions." Stripe ran onboarding labs and shared Cursor Rules across 3,000 engineers.

Remove friction. Pre-install tools. Share configurations. Eliminate signup barriers. Stripe preinstalled Cursor for every engineer. Ramp connected Claude Code to test frameworks and observability via MCP so the tools worked inside existing workflows from day one.

Connect to existing infrastructure. The tools should plug into what you already have. Spotify built Honk on top of Fleet Management and Backstage. Ramp connected to existing test and observability systems via MCP.

Add verification alongside adoption, not after. Spotify added deterministic verifiers and an LLM-as-judge before opening Honk to wider use. Stripe required human review on every merge from day one. Rakuten paired speed with safety guardrails from the start.

Track outcomes, not adoption. Adoption percentage is an activity metric. Cost per accepted change tells you whether adoption is producing delivery.

This sequence maps directly to the eight decisions in the Mandate chapter. The convergent stack across these companies — single-agent loops, sandboxed environments, verification before human review, structured context files, mature CI/CD foundations — is detailed in Chapter 8.

A NOTE ON WHAT THIS APPENDIX IS NOT

This is not a procurement guide. The verification infrastructure described here is built primarily by the engineering organization, with bought components where they fit. If the appendix reads as if it is recommending a specific vendor for any tier, that is unintentional — the fitness of any particular tool is downstream of whether the organization has built the infrastructure to use it well.

If you take only one thing from this appendix: **the investment is multi-year, and the payback shows up in the second year, not the first.**