Contents

1 Development Philosophy: Iterative Delivery at Scale

1.1 Executive Summary

This document outlines a development philosophy centered on **disciplined iteration**, **semantic versioning**, and **team coordination**. Rather than theoretical principles, it presents evidence from 3.5+ years of sustained GitHub activity demonstrating production-oriented software development.

The core thesis: **small, frequent releases with automated versioning reduce coordination overhead and deployment risk**—directly applicable to Strata Space's December 2025 and July 2026 milestones with 3 parallel squads.

1.2 Release Discipline: Evidence from Production

1.2.1 Project History

Project	Duration	Commits	Releases	Pattern
rsr-fsa	3.5+ years	34	-	Long-term financial simulation
claude-code	10 months	375	-	CLI tooling, 288 active days
scripts	6 months	205	-	macOS automation ecosystem
cc-skills	9 days	167	64	Intense sprint (7.1 releases/day)

Project	Duration	Commits	Releases	Pattern
netstrata	27 days	118	34	Responsive iteration (1.3/day)
gapless-crypto	73 days	187	10	PyPI published, CI/CD automated
alpha-forge	42 days	81	-	42 ADRs (architecture-first)

1.2.2 Key Observations

1. **Sustained activity, not recent burst**: The oldest project (rsr-fsa) dates to March 2022. This represents 3.5+ years of continuous development across financial modeling, automation, and data engineering domains.

2. **Commit discipline**: In the netstrata project, 127 of 132 commits occurred on unique dates (96.2%)—evidence of daily, disciplined work rather than bulk commits.

3. **Release velocity scales with need**: cc-skills achieved 64 releases in 9 days (7.1/day) during an intense feature sprint, while gapless-crypto maintained 10 releases over 73 days for a production PyPI package. The methodology adapts to project phase.

1.3 Commit Hygiene Examples

All projects follow Conventional Commits with semantic-release automation:

```
feat(skills): add clickhouse-architect skill for schema design
fix(validation): correct ALP codec recommendation per ClickHouse docs
docs(adr): link release notes to architecture decisions
chore(release): 2.30.0 [skip ci]
refactor(config): centralize environment variables via mise [env]
```

Why this matters for Strata Space: When 3 squads work in parallel, consistent commit conventions enable:

- Automated changelog generation (no manual release notes)
- Clear attribution of changes across modules (DMS vs Tasks vs Time Recording)
- Version tracking that supports staged rollouts

1.4 Architecture Decision Records (ADRs)

1.4.1 Evidence: 42 ADRs in alpha-forge

The alpha-forge project maintains 42 formal Architecture Decision Records documenting technical trade-offs. Sample titles:

- `2025-11-17-e2e-first-testing-strategy.md` — Quantified analysis: 2.6:1 maintenance-to-implementation ratio
- `2025-11-14-dsl-simplification-v04.md` — Breaking change management with migration path
- `2025-11-17-plugin-metadata-management.md` — Ecosystem coordination patterns

1.4.2 Why ADRs Support Squad Coordination

ADRs create **decision traceability** that benefits:

1. **Onboarding**: New team members understand *why* decisions were made, not just *what* was built
2. **Squad alignment**: When Squad A's decision affects Squad B's module, the rationale is documented
3. **Future refactoring**: Six months later, the team knows whether to preserve or change a pattern

1.4.3 Sample ADR Structure (MADR 4.0)

```
---
status: accepted
date: 2025-11-17
decision-maker: terrylica
consulted: [team members, Claude Code analysis]
research-method: Web research + codebase metrics
---

## Context and Problem Statement
[Quantified problem with metrics]

## Decision Drivers
```

```
[Prioritized criteria]

## Considered Options
[Option A, B, C with trade-offs]

## Decision Outcome
[Chosen option with rationale]
```

1.5 Application to Strata Space

1.5.1 December 2025 Milestone

Challenge: 3 squads delivering DMS, Tasks, and Time Recording simultaneously to 120+ internal users.

How iterative release discipline helps:

- Each squad releases independently when their module is ready
- Semantic versioning tracks which features are in which build
- Automated changelogs reduce Tom's coordination overhead
- Rollback granularity: if Tasks has issues, DMS/Time Recording continue

1.5.2 July 2026 Milestone

Challenge: Buildings, Asset Management, Strata Manager Dashboard—targeting 90-95% of workload in Strata Space.

How ADR discipline helps:

- Architecture decisions documented before implementation
- Breaking changes planned with migration paths
- New squad members onboard faster with decision context
- Compliance documentation generated from release history

1.5.3 Reducing "Big Bang" Risk

Ted's observation: *"120 people, hey, there's a problem here"* when encountering screen changes.

Iterative approach reduces this risk by:

- Validating each module with a subset of users before company-wide rollout
- Catching integration issues between squads early (not at final merge)
- Providing rollback points if adoption issues emerge

1.6 Verify on GitHub

Profile: github.com/terrylica

Each repository's commit history is publicly visible. Notable repositories for verification:

Repository	What to Verify
cc-skills	64 releases in 9 days, semantic-release automation
gapless-crypto-clickhouse	PyPI publishing, GitHub Actions CI/CD
netstrata	34 releases, responsive iteration to feedback

1.7 Summary

This development philosophy is not theoretical—it's demonstrated through:

- **3.5+ years** of sustained GitHub activity
- **64 releases in 9 days** when velocity is needed
- **42 ADRs** documenting architectural decisions
- **96.2% unique commit dates** showing disciplined daily work

For Strata Space, this translates to:

- Reduced coordination overhead for Tom's 3 parallel squads
- Clear version tracking for December 2025 staged rollout
- Decision traceability that accelerates July 2026 onboarding
- Risk mitigation through smaller, more frequent releases