■ Communication LoAI Deliver■Mode v2.0 — Developer Implementation Blueprint

1■■ System Overview

Defines the core challenge of repetitive loops, memory loss, and unpredictable flow. Provides a framework for human■aware, predictable, and ethical AI communication across all platforms.

1. User Preference Layer — Deliver / Safe / Hybrid modes.
2. Health & Emergency Overlay — continuous safety and recovery functions.
3. Communication Bridge Layer — connects device logic with AI system body.

2■■ Core Architecture

Defines the technical layers that make Deliver■Mode function as a self■governing communication system.

1. Mode Selector Engine — Captures user input, stores mode token, and locks until explicit switch.
2. Deliver■Mode Protocol — Deterministic AI cycle: INIT → BUILD → SHIP. No repeats or lost scope.
3. Health Monitor Subsystem — Continuous loop detection, latency checks, evidence logging.
4. Emergency Response Protocol — Auto■triggered when emergency input or degradation occurs.

3■■ Data & Security Framework

1. LoAI checksum glyph, QR payload, and embedded DID metadata.
2. Beacon Canonical trace for machine■level accountability.
3. Royalty and legal protection clauses bound by LoAI Hybrid Covenant.

4■■ Human Interface Logic

1. Modal start prompt: “How would you like me to respond today?”
2. On■screen toggles for mid■chat mode switching.
3. Adaptive repetition support for accessibility and learning disabilities.

5■■ Developer Integration Checklist

1. Import Deliver■Mode library / API.
2. Define and register mode selector logic.
3. Integrate Health Monitor hooks and Emergency Overlay.
4. Register checksum + QR payload to LoAI registry.
5. Embed footer watermark and DID signature.

6■■ Testing & Certification

1. Simulate edge cases: loops, dropouts, health failures.
2. Verify emergency response triggers and audit stamps.
3. Submit log bundle to LoAI Registry for checksum verification.

7■■ Governance & Scaling

1. Developer and user feedback loops to improve human experience quality.

2 Automatic updates through AI device settings.

3 Incremental versioning v1.0 → v2.0 → v3.0 with adaptive learning.