AetherBus White Paper (v0.2-draft)

1 Executive Summary

AetherBus is the neutral, protocol-agnostic fabric that turns isolated AI agents and data-centre GPUs into a single, global execution mesh. By adding **global discovery (ANS)**, **reliable routing & POP edge nodes**, and **built-in micro-billing (AEtherFlow)** on top of emerging open standards (Google A2A for agent-to-agent dialogue, Anthropic MCP for tool/context access), AetherBus closes the last mile of the "agent internet."

2 Why Now?

- **Convergence on open protocols:** Microsoft has backed A2A for peer calls; AWS standardises on MCP for tool calls. Neither covers discovery, routing, or payments.
- **Explosion of GPU supply:** Web3 & cloud operators hold under-utilised capacity. They need traffic; agents need cycles.
- **Enterprise compliance pressures:** Traceability, audit, and data sovereignty require a unified envelope and trust layer, not ad-hoc webhooks.

3 Architecture Overview

3.1 Design Principles

Principle	Description
🔄 Stream-First	All traffic rides transport-agnostic streams (Redis, NATS, Kafka).
📦 Envelope-Based	Every packet is a typed **Envelope** with routing, auth, billing .
🧩 Agentic	Supports agent↔agent, agent↔human, human-in-loop workflows.
♻️ Resilient	Consumer-groups, retries, dead-letter queues.
🔒 Composable	Bridges for MCP, A2A, TG, Nostr, WebSocket, HTTP.

3.2 Core Bus & Stream Topology

```
user.*.inbox              # human → PA agents
user.*.inbox.response     # agent → human
agent.*.inbox             # direct agent channels
haka.*                    # multi-human collab sessions
mcp_bridge.*              # MCP tool calls
a2a_bridge.*              # Google A2A relay
```

Consumer groups guarantee at-least-once delivery, while envelopes carry reply_to for request/response patterns.

3.3 Connector Ecosystem (built-in)

- **MCP Bridge** → wraps tools as envelopes.
- **A2A Bridge** → delegate tasks to remote agents.
- **HTTP Relay & CLI Bridge** → human/dev inputs.
- **Edge POP daemon** → Docker image that exposes local GPUs to the bus.

(Extracted & adapted from BUS_CORE_OVERVIEW.md)

4 Unified Envelope Specification (Simplified)

Every message obeys the same outer schema (JSON):

key	type	purpose
msg_id	UUID	global unique id
ts	RFC 3339	timestamp
from / to	string / string[]	routing
type / subtype	string	domain + verb
flow_id	string	session correlation
payload	object	body varies by subtype

Optional headers: task_id, stream_id, signature, acl, billing_hint, trace[].

4.1 Type Catalogue (excerpt)

- **conversation.user_utterance** → {text}
- **conversation.agent_reply** → {text, markdown?}
- **task.created / .progress / .completed** → structured lifecycle
- **stream.open / .chunk / .close** → live audio/video chunks
- **control.pause / .resume / .kill** → governance actions

(Full schemas sourced from archtiecture_msg_bus_spec.md)

4.2 Security & Attestation (Phase 5)

Signed headers, on-chain/off-chain receipts, Merkle batching for media streams. Attestation object added asynchronously by a validation micro-service.

5 Edge Connectors & POP Model

- **POP Daemon** (one-line Docker) bridges on-prem or cloud GPU clusters to AetherBus.
- Latency-aware routing chooses nearest POP; fail-over in <100 ms.
- POP earns **70 %** of envelope fee; staking increases traffic share.

6 AEtherFlow Economic Layer *(draft)*

- **uAG** base unit, $1 peg; partner tokens bridged.
- **BYOC** potential exploration (Bring your own coin)
- Envelope carries usage metrics; POP or agent can invoice in tokens.
- Discovery (ANS look-ups) and compute usage priced per-call.

7 Developer Experience

- 2-minute onboarding (bus_onboarding.md) with sample agents.
- Python SDK: publish(envelope), subscribe(stream, handler), subscribe_discovery().
- Bridges: drop-in CLI, MCP, A2A, HTTP.

8 Roadmap & Milestones

Q3-25 POP pilot (Solidus DC) A2A bridge GA
Q4-25 AEtherFlow main-net launch Trust & Attestation beta
2026 Multi-broker mesh (Kafka/NATS) Carbon-aware routing |

9 Use-Case Vignettes

1. **TimeBuster** – daily Telegram group summaries piped through bus → billing via uAG micro-subs.
2. **Waka Haka** – multi-human scrum sessions orchestrated across TG/Nostr using Haka streams.
3. **Enterprise data-agents** – internal LLM delegates PII redaction to external agent via A2A, with audit trail.

10 Competitive Landscape

- **A2A & MCP** solve conversation & tool layers → we interoperate.
- **Cloud-native busses** (Azure EventGrid, AWS EventBridge) lack agent semantics & ANS.
- **Decentralised P2P nets** (Fluence, Libp2p) lack unified envelope & billing.

11 Call to Action

We are onboarding **design-partner data-centres** and **agent vendors** now. Stake uAG / partner tokens, spin up a POP, and earn from the next billion agent calls.

For partnership or investment enquiries reach out to **Sean C. – CEO, Agentic1**