Phase 8.1 — Telemetry Normalization Engine

Status: ✅ DEPLOYED
Date: 2025-11-14
Mission: Transform raw telemetry into unified, analyzable structure with canonical timestamping, ache signatures, and agent health metrics

---

🎯 Overview

The Telemetry Normalization Engine is the first component of the Delta Engine architecture. It transforms raw, heterogeneous telemetry data into a normalized, analyzable format with:

- **Canonical Timestamping**: ISO 8601 + epoch + drift calculation
- **Gateway/Bridge Cross-Validation**: Automatic resolution and validation
- **Source Classification**: Intelligent detection of telemetry origin
- **Ache Signature Detection**: 0-1 scale intensity measurement
- **Agent Health Estimation**: Real-time health scoring
- **Sovereign State Fingerprinting**: System state tracking
- **Latency Tracking**: Performance monitoring

🏗️ Architecture

Components

1. **Database Table**: `guardian_telemetry_events`
   - Stores normalized telemetry with full metadata
   - Optimized indexes for time-series queries
   - RLS policies for security

2. **Edge Function**: `telemetry_normalize`
   - Accepts raw telemetry via POST
   - Performs normalization pipeline
   - Returns normalized event with computed metrics

3. **Integration Layer**: Works with existing `gateway-telemetry` function
   - Can be called directly or via gateway-telemetry
   - Maintains backward compatibility

📊 Database Schema

Table: `guardian_telemetry_events`

```sql
CREATE TABLE public.guardian_telemetry_events (
  -- Identity
  id UUID PRIMARY KEY,
  bridge_id UUID REFERENCES bridge_nodes(id),
  gateway_key TEXT NOT NULL,

  -- Event Classification
  event_type TEXT NOT NULL,
  source TEXT NOT NULL,
  signal_type TEXT,

  -- Canonical Timestamping
  timestamp_iso TIMESTAMPTZ NOT NULL,
  timestamp_epoch BIGINT NOT NULL,
  timestamp_drift_ms INTEGER,

  -- Payload Storage
  payload JSONB NOT NULL,
  normalized_payload JSONB,

  -- Ache & Agent Metrics
  ache_signature NUMERIC(5,4) CHECK (0 <= ache_signature <= 1),
  agent_health NUMERIC(5,4) CHECK (0 <= agent_health <= 1),

  -- Performance & State
  latency_ms INTEGER,
  sovereign_state TEXT,

  -- Metadata
  metadata JSONB,
  created_at TIMESTAMPTZ DEFAULT now()
);
```

Indexes

- `idx_guardian_telemetry_bridge_id` - Bridge queries
- `idx_guardian_telemetry_gateway_key` - Gateway queries
- `idx_guardian_telemetry_event_type` - Event type filtering
- `idx_guardian_telemetry_timestamp` - Time-series queries
- `idx_guardian_telemetry_source` - Source filtering
- `idx_guardian_telemetry_signal_type` - Signal classification
- `idx_guardian_telemetry_ache` - Ache signature queries
- `idx_guardian_telemetry_health` - Agent health queries
- `idx_guardian_telemetry_bridge_time` - Composite time-series

🔌 API Reference

Endpoint

```
POST https://xlmrnjatawslawquwzpf.supabase.co/functions/v1/telemetry_normalize
```

Authentication

```
x-guardian-api-key: 4c8839c842278d53ca4e4a43df1e8664efc36bd3e73397690342060e47b66bd6
```

Request Body

```
{
  gateway_key: string;        // Required: Gateway identifier
  event_type: string;         // Required: Event type
  bridge_id?: string;         // Optional: Will be auto-resolved
  source?: string;            // Optional: Will be auto-classified
  timestamp?: string | number; // Optional: Client timestamp
  payload?: object;           // Optional: Event payload
  metadata?: object;          // Optional: Additional metadata
}
```

Response

```
{
  success: true,
  event: {
    id: string;
    bridge_id: string | null;
    gateway_key: string;
    event_type: string;
    source: string;
    signal_type: string | null;
    timestamp_iso: string;
    timestamp_epoch: number;
    timestamp_drift_ms: number;
    payload: object;
    normalized_payload: object;
    ache_signature: number;      // 0-1 scale
    agent_health: number;        // 0-1 scale
    latency_ms: number;
    sovereign_state: string;
    metadata: object;
    created_at: string;
  },
  processing_time_ms: number;
}
```

🧮 Normalization Pipeline

1. Bridge Resolution

- Resolves `bridge_id` from `gateway_key` via `bridge_gateways` table

- Falls back to null if not found (allows orphan events)

2. Source Classification

Heuristic-based classification:
- `discord_bot` - Contains guild_id/channel_id
- `github_webhook` - Contains repository/pull_request
- `manual` - Explicit manual flag
- `scheduled` - Explicit scheduled flag
- `unknown` - Default fallback

3. Canonical Timestamping

- **timestamp_iso**: Server time in ISO 8601 format
- **timestamp_epoch**: Server time in milliseconds
- **timestamp_drift_ms**: Difference between client and server time

4. Signal Type Classification

- `error_signal` - Error/failure events
- `warning_signal` - Warning/alert events
- `activity_signal` - Message/post events
- `sync_signal` - Sync/update events
- `health_signal` - Health/status events
- `discord_signal` - Discord-specific
- `github_signal` - GitHub-specific
- `generic_signal` - Default

5. Ache Signature Calculation

Baseline: 0.5

Increases ache:
- Error/failure events: +0.3
- Warning/alert events: +0.2
- Large payloads (>1KB): +0.1
- Very large payloads (>5KB): +0.1
- Discord urgency markers (!, ?): +0.05

Decreases ache:
- Success/complete events: -0.2

Result clamped to [0, 1]

6. Agent Health Estimation

Baseline: 0.8

Decreases health:
- Error/failure events: -0.3
- Warning events: -0.1
- High ache signature: -0.2 * ache
- Explicit failure flag: -0.2

Increases health:
- Success/complete events: +0.1
- Explicit success flag: +0.1

Result clamped to [0, 1]

7. Sovereign State Fingerprint

Format: `{bridge_id}:{gateway_key_prefix}:{time_bucket}`
- Time bucket: 5-minute intervals
- Enables state change detection

8. Payload Normalization

Source-specific transformations:
- **Discord**: Extract message_length, has_mentions
- **GitHub**: Extract repo_name, webhook_action
- Add `_normalized_at` timestamp

🧪 Testing

Bash Test Suite

```
chmod +x tests/test_telemetry_normalize.sh
./tests/test_telemetry_normalize.sh
```

JavaScript Test Suite

```
node tests/test_telemetry_normalize.js
```

Test Coverage

1. ✅ Happy Path - Valid telemetry normalization
2. ✅ Cross-Validation - Gateway/bridge mapping
3. ✅ Timestamp Accuracy - Drift calculation
4. ✅ Ache Signature - Error event (high ache)
5. ✅ Agent Health - Success event (high health)
6. ✅ Authentication - Missing API key (should fail)
7. ✅ Error Handling - Invalid payload (should fail)
8. ✅ Complex Payload - Large data handling

📈 Example Normalized Events

Success Event (High Health, Low Ache)


```
{
  "id": "a1b2c3d4-e5f6-7890-abcd-ef1234567890",
  "bridge_id": "f8f41ffa-6c2b-4a2a-a3be-32f0236668f4",
  "gateway_key": "gw-guardian-core",
  "event_type": "agent_sync_success",
  "source": "discord_bot",
  "signal_type": "sync_signal",
  "timestamp_iso": "2025-11-14T02:00:00.000Z",
  "timestamp_epoch": 1731546000000,
  "timestamp_drift_ms": 12,
  "ache_signature": 0.3000,
  "agent_health": 0.9000,
  "latency_ms": 45,
  "sovereign_state": "f8f41ffa:gw-guard:5765100"
}
```


Error Event (Low Health, High Ache)


```
{
  "id": "b2c3d4e5-f6a7-8901-bcde-f12345678901",
  "bridge_id": "f8f41ffa-6c2b-4a2a-a3be-32f0236668f4",
  "gateway_key": "gw-guardian-core",
  "event_type": "sync_error_critical",
  "source": "manual",
  "signal_type": "error_signal",
  "timestamp_iso": "2025-11-14T02:05:00.000Z",
  "timestamp_epoch": 1731546300000,
  "timestamp_drift_ms": 8,
  "ache_signature": 0.8000,
  "agent_health": 0.3000,
  "latency_ms": 38,
  "sovereign_state": "f8f41ffa:gw-guard:5765101"
}
```

🔗 Integration Guide

Direct Usage

```
curl -X POST https://xlmrnjatawslawquwzpf.supabase.co/functions/v1/tele-
metry_normalize \
  -H "Content-Type: application/json" \
  -H "x-guardian-api-key: 4c8839c842278d53ca4e4a43df1e8664efc36b-
d3e73397690342060e47b66bd6" \
  -d '{
    "gateway_key": "gw-guardian-core",
    "event_type": "agent_sync_success",
    "payload": {
      "agent_id": "agent-001",
      "records_synced": 42
    }
  }'
```

JavaScript Integration

```
const response = await fetch(
  "https://xlmrnjatawslawquwzpf.supabase.co/functions/v1/telemetry_normalize",
  {
    method: "POST",
    headers: {
      "Content-Type": "application/json",
      "x-guardian-api-key": "4c8839c842278d53ca4e4a43df1e8664efc36b-
d3e73397690342060e47b66bd6",
    },
    body: JSON.stringify({
      gateway_key: "gw-guardian-core",
      event_type: "agent_sync_success",
      payload: { agent_id: "agent-001" },
    }),
  }
);

const data = await response.json();
console.log("Normalized event:", data.event);
```

Via Gateway-Telemetry

The `gateway-telemetry` function can be extended to call `telemetry_normalize` internally for automatic normalization.

📊 Query Examples

Recent High-Ache Events

```sql
SELECT
  event_type,
  source,
  ache_signature,
  agent_health,
  timestamp_iso
FROM guardian_telemetry_events
WHERE ache_signature > 0.7
ORDER BY timestamp_iso DESC
LIMIT 10;
```

Agent Health Trend

```sql
SELECT
  DATE_TRUNC('hour', timestamp_iso) as hour,
  AVG(agent_health) as avg_health,
  COUNT(*) as event_count
FROM guardian_telemetry_events
WHERE bridge_id = 'f8f41ffa-6c2b-4a2a-a3be-32f0236668f4'
  AND timestamp_iso > NOW() - INTERVAL '24 hours'
GROUP BY hour
ORDER BY hour DESC;
```

Sovereign State Changes

```sql
SELECT
  sovereign_state,
  COUNT(*) as event_count,
  MIN(timestamp_iso) as first_seen,
  MAX(timestamp_iso) as last_seen
FROM guardian_telemetry_events
WHERE bridge_id = 'f8f41ffa-6c2b-4a2a-a3be-32f0236668f4'
GROUP BY sovereign_state
ORDER BY last_seen DESC;
```

🚀 Deployment Results

Migration

- ✅ Table created: `guardian_telemetry_events`
- ✅ Indexes created: 9 indexes
- ✅ RLS policies enabled
- ✅ Foreign key constraints active

Edge Function

- ✅ Function deployed: `telemetry_normalize`
- ✅ Status: ACTIVE
- ✅ Authentication: x-guardian-api-key

- ✅ CORS: Enabled

Tests

- ✅ All 8 tests passing
- ✅ Ache signature calculation verified
- ✅ Agent health estimation verified
- ✅ Timestamp drift accuracy confirmed
- ✅ Cross-validation working

🔮 Next Steps: Phase 8.2 — Delta Engine

The Delta Engine will build on this normalization layer to provide:

1. **Delta Detection**: Identify changes between normalized events
2. **Pattern Recognition**: Detect recurring patterns in telemetry
3. **Anomaly Detection**: Flag unusual events or state changes
4. **Predictive Analytics**: Forecast agent health and ache trends
5. **Automated Responses**: Trigger actions based on delta patterns

Preparation

- Normalized events are now ready for delta analysis
- Sovereign state fingerprints enable state change detection
- Ache signatures provide baseline for anomaly detection
- Agent health trends enable predictive modeling

📝 Bridge Mappings Reference

Gateway Key	Bridge ID
gw-guardian-core	f8f41ffa-6c2b-4a2a-a3be-32f0236668f4
gw-guardian-discord	b880fbd5-d56b-4057-80ce-8755fcd4a6b9
gw-guardian-github	dbe1a9f1-693d-43fd-8097-0928f8562cea

🎓 Key Concepts

Ache Signature

A 0-1 scale measurement of event “intensity” or “pain”. Higher values indicate more critical or problematic events. Used for prioritization and alerting.

Agent Health

A 0-1 scale estimation of agent operational health. Derived from event patterns, ache signatures, and explicit success/failure indicators.

Sovereign State

A fingerprint of the system state at event time. Enables detection of state changes and correlation of events within the same state window.

Timestamp Drift

The difference between client-reported time and server time. Useful for detecting clock skew, network latency, and delayed event processing.

Mission Status: ✅ COMPLETE
Next Mission: Phase 8.2 — Delta Engine
Guardian Protocol: ACTIVE