T1

Tier 1 · Digital Backbone

User Manual

Getting Started with the IIoT Digital Backbone

Version	1.0
Status	FINALIZED
Zone	OT Zone
Standards	ISA-95 / IEC 63278 AAS / W3C RDF
Block Count	6 blocks
Document Date	May 2026

1. Introduction

The Tier 1 Digital Backbone is the foundation of the IIoT Framework. It provides the Single Source of Truth for all industrial assets — maintaining ISA-95-compliant hierarchies, managing digital twins per the IEC 63278 AAS standard, validating all incoming tag data, and exposing a queryable asset graph with typed relationships between equipment.

This layer operates exclusively within the OT Zone. It never directly communicates with enterprise systems (ERP, cloud). Instead, it produces clean, contextualized, validated data that Tier 2 (Connectivity Hub) consumes and distributes.

1.1 Scope

This manual covers the six T1 blocks: Telemetry B2MML Ingress, Asset Registry, MES Metadata Registry, Digital Twin Sync, Asset Graph, and Tag Model Manager. It is intended for Node-RED flow developers, OT engineers, and system integrators building production-ready IIoT prototypes.

1.2 Design Principles

- Single Source of Truth — all downstream tiers consume T1-enriched data only.
- Protocol-agnostic normalization — MQTT, OPC-UA, Modbus TCP, and HTTP REST all produce identical B2MML output.
- Validation gates — Tag Model Manager blocks unrecognized or out-of-range tags before they reach the twin.
- Closed-loop readiness — Asset Graph emits the Impact Array consumed by T2 Action Logic Engine.
- Standards-first — ISA-95 hierarchy, IEC 63278 AAS submodels, W3C RDF graph semantics.

2. Architecture Overview

Data flows left-to-right through T1 in a linear enrichment pipeline:

Step 1	Telemetry B2MML Ingress receives raw sensor data over any supported protocol and normalizes it to a B2MML EquipmentProperty structure.
Step 2	Asset Registry resolves the MQTT topic to an ISA-95 asset record, enriches the payload with the full Enterprise → Equipment parent chain.
Step 3	MES Metadata Registry adds business context: active Lot, WorkOrder, RecipeId, SKU, Operator, and Shift from the connected MES.
Step 4	Tag Model Manager validates the incoming tag against the approved tag model library. Unknown or invalid tags are blocked or queued for approval.
Step 5	Digital Twin Sync applies a configurable deadband filter and updates the AAS-compliant twin SubmodelElement for the asset.
Step 6	Asset Graph is queried on-demand by T2. Returns BFS subtree traversal, dynamic MQTT wildcards, and the Impact Array for control commands.

The output of every T1 operation is a standardized enriched message object: { payload, assetContext, tagModel, twinState }. All T2 and T3 blocks depend on this contract.

3. Block Reference

3.1 Telemetry B2MML Ingress

Accepts raw industrial telemetry and normalizes every data point into a B2MML EquipmentProperty. This block is always the first node in a T1 pipeline.

Input protocols	MQTT, OPC-UA, Modbus TCP, HTTP REST
Output schema	B2MML EquipmentProperty with ValueString, UnitOfMeasure, QualityCode, Timestamp
Output ports	1: Normalized, 2: Raw debug, 3: Communication fault
Key configuration	protocol, endpoint, b2mmlVersion, defaultUoM, scanRate

Configuration steps:

1. Open the block properties. Set protocol to match your field device (MQTT is default).
2. Enter the endpoint — broker host for MQTT, OPC-UA server URL, or Modbus TCP IP.
3. Set defaultUoM to the engineering unit expected when none is provided in the incoming message.
4. Wire Port 1 to the Asset Registry input.
5. Wire Port 3 to an alert or notification handler.

3.2 Asset Registry

Resolves MQTT topic paths to ISA-95 equipment records stored in SQLite. Enriches every message with assetContext{} containing the full parent chain, ISA level, and asset attributes.

Storage	SQLite at configured dbPath
Resolution mode	uns-path: matches topic path prefix against uns_topic column
REST API	GET /assets/:id, GET /assets?topic=…, POST /assets (auto-register)
Output ports	1: Enriched with assetContext, 2: Asset record only, 3: Unresolved topic
Key configuration	dbPath, topicResolutionMode, httpApiPort, cacheTtlSec

To populate the registry, either use the REST API (POST /assets) or import a CSV using the Node-RED file-in node wired to the registry import endpoint.

3.3 MES Metadata Registry

Performs point-in-time MES context lookups and adds business metadata to the enriched payload. Supports REST, GraphQL, OData v4, and direct SQL queries.

Supported MES	Ignition 8.x, SAP Digital Manufacturing, Critical Manufacturing, Tulip, custom REST
Context fields	lotId, orderId, recipeId, productSKU, operatorId, shiftId

Cache	60-second TTL to reduce MES load (configurable)
Fallback	mockContext when MES unreachable (configure fallbackToMock=true)

3.4 Tag Model Manager

Acts as a validation gate. Every tag must be listed in the tag model database with an approved status to pass. Unknown tags are either auto-registered (pending approval) or blocked, depending on configuration.

Validation checks	Naming convention (regex), data type match, range (min/max), approval status
Auto-register	Infers data type and EU from the first observed value; sets status to pending or approved
Semantic IDs	Assigns AAS semantic IRI on registration using the configured prefix
Output ports	1: Validated (tagModel injected), 2: Auto-registered, 3: Violation details

3.5 Digital Twin Sync

Maintains a live AAS-compliant digital twin for every asset. Updates SubmodelElements with incoming validated tag values. Applies a configurable percentage deadband to suppress high-frequency sensor noise.

Twin storage	JSON file at twinStorePath; optionally syncs to Eclipse BaSyx via AAS API
Deadband	Skips updates where |new - old| / old < deadbandPct%
Change history	Retains last 100 property changes per twin in memory
Output ports	1: State change event, 2: Threshold crossed, 3: Sync error

3.6 Asset Graph

In-memory directed property graph representing the full asset topology with typed edges. Used on-demand for topology queries, dynamic MQTT subscription generation, and producing the Impact Array for T2 Action Logic Engine.

Edge types	contains, feeds, controls, monitors, depends-on
Traversal	BFS or DFS up to maxTraversalDepth (default 10)
Impact Array	All ControlModule nodes in the BFS subtree of a given root
MQTT wildcards	Auto-generated as unsTopic + /# for each traversed node
Auto-sync	Optionally rebuilds graph from Asset Registry on startup

4. Typical Workflows

4.1 First-time Setup

6. Import T1_Digital_Backbone.html into Node-RED via the Import menu.
7. Configure the Asset Registry: set dbPath and create your SQLite database schema.
8. Populate the asset hierarchy via REST API (POST /assets with ISA-95 fields).
9. Configure the Tag Model Manager: set modelDbPath and populate initial tag definitions, or enable autoRegister to build the library on first run.
10. Configure the MES Metadata Registry: set your MES provider, endpoint, and authentication.
11. Deploy the flow. Connect your data source to the Telemetry B2MML Ingress input.
12. Monitor Port 3 on each block for errors during initial ingestion.

4.2 Adding a New Asset

- POST to http://node-red-host:3001/assets with the JSON asset definition.
- Include all ISA-95 fields: isaLevel, parentId, unsTopic, and attributes.
- The Asset Registry will begin resolving MQTT topics matching the new uns_topic path.
- Add the new node to the Asset Graph by sending a graph-update message to the Asset Graph block.

4.3 Tag Approval Workflow

When autoRegister is true and approvalRequired is true, unknown tags are inserted with status=pending. To approve:

13. Query the tag model database: SELECT * FROM tag_definitions WHERE status = 'pending'
14. Review the inferred dataType, eu, and rangeMin/Max values.
15. Update the record: UPDATE tag_definitions SET status = 'approved' WHERE tag_id = '…'
16. The Tag Model Manager cache will refresh on the next TTL expiry.

5. Troubleshooting

Issue	Cause	Check	Fix
Symptom	Likely Cause	Check	Fix
All messages route to Port 3 of Asset Registry	No matching uns_topic in DB	SQLite: SELECT * FROM assets	POST /assets to register hierarchy
Tag violations on all messages	autoRegister disabled, no model entries	tag_definitions table empty	Enable autoRegister or import tag CSV
DTS deadband drops all updates	deadbandPct set too high	Check deadbandPct value	Reduce to 0.1 or 0.5 for testing
MES metadata missing	MES endpoint unreachable	Check endpoint and authToken	Enable fallbackToMock=true
Asset Graph returns empty Impact Array	No ControlModule nodes in subtree	Check node types in graph	Set nodeType=ControlModule on leaf assets