

PUR-SAD-001

System Architecture Description

Purik Sensor System

Engineering Documentation Suite - Baseline v0.1 (Draft)

System	Purik Sensor System
Revision	0.1 Draft
Date	September 2026
Status	Controlled baseline draft; verify TBD values before release

1. Purpose and Scope

The Purik Sensor System is an experimental modular sensor-integration platform for investigating rapid technology insertion, defined interfaces, distributed sensing, machine-readable telemetry, module discovery, health reporting, and selected principles associated with a Modular Open Systems Approach (MOSA).

Scope statement: Purik is a prototype-scale engineering testbed. It is not claimed to be MOSA-compliant, an operational defense system, or a substitute for certified avionics. The architecture is intended to make integration behavior observable and measurable.

2. System Objectives

- Integrate and replace heterogeneous sensors with limited unrelated software rework.
- Separate sensing, adaptation, transport, data representation, visualization, and configuration responsibilities.
- Provide transport-independent, machine-readable telemetry.
- Support distributed sensor nodes and a ground-station role.
- Experiment with machine-readable module identity, capability, interface, power, timing, calibration, and health metadata.
- Measure integration effort, latency, availability, fault behavior, and regression impact.

3. Architectural Principles

Separation of concerns

Sensor acquisition, normalization, transport, visualization, configuration, and health monitoring should remain separable functions.

Defined boundaries

Subsystems should interact through documented, versioned interfaces rather than depend on hidden implementation details.

Replaceability

A sensor should be replaceable by another of the same functional class primarily through a new or updated adapter.

Transport independence

A sensor adapter should not need substantial redesign when data moves over Wi-Fi, NRF24, serial, or another supported transport.

Evidence discipline

Capabilities are labeled demonstrated, in progress, planned, or proposed.

4. Iteration 1 Baseline

Subsystem	Implementation	Interface	Role
Processing	Arduino UNO R4 Wi-Fi	Board-level	Acquisition, servo control, HTTP response generation
Scanning range	HC-SR04 + servo	GPIO pulse/echo + PWM	Angular ultrasonic ranging
Infrared	Analog IR sensor	A1	Raw ADC measurement
Environment	DHT11	Digital GPIO	Temperature and humidity
Local status	16x2 LCD	I2C	Boot/network/IP status
Client	Windows/Python Tkinter	HTTP/JSON over Wi-Fi	Polling and visualization

Iteration 1 data flow: sensor -> device-specific acquisition -> UNO R4 application -> JSON serialization -> HTTP/Wi-Fi -> Python client -> visualization. Current sampling is request-coupled: sensor reads occur while servicing the client request.

5. Iteration 2 Target Architecture

Element	Planned/In-progress role	Status
UNO Q ground station	Touchscreen HMI, configuration, registry, higher-level processing	In progress
UNO R4 mobile/remote nodes	Sensor adapters, local	Planned/in progress by subsystem

	timestamping, normalization, transport	
TI iWRL6432BOOST	mmWave sensing	In progress
SparkFun MAX-M10S	GNSS position/time	In progress
Slamtec A1M8	2D LiDAR	Planned
GY-521	IMU	Planned
INA260	Power telemetry	Planned
Module descriptor service	Discovery, compatibility, capability metadata	Proposed

6. Recommended Logical Stack

Application / Visualization
 Mission / Fusion Services
 Common Purik Data Model
 Transport Abstraction
 Module Adapter Layer
 Device Driver Layer
 Physical Sensor / Payload

7. Controlled Modular Boundaries

Interface ID	Boundary	Intent
PURIK-IF-01	Module to node physical/electrical	Power, connector, logic, protection
PURIK-IF-02	Driver to adapter software	Hide native device details
PURIK-IF-03	Adapter to common data model	Normalize data semantics
PURIK-IF-04	Node to transport	Separate data production from link implementation
PURIK-IF-05	Transport to ground station	Common telemetry/control contract
PURIK-IF-06	Module descriptor	Identity, capabilities, compatibility and health

8. Architecture Maturation Priorities

1. Decouple acquisition from client polling and timestamp at acquisition.
2. Formalize the common message envelope and schema versioning.
3. Implement module adapters behind a common interface.
4. Create and validate module descriptors.
5. Add transport abstraction and regression tests.
6. Establish a controlled power architecture and safe power authorization concept.
7. Maintain configuration baselines and test evidence for every release.