# Sensor Data Collection System

## System of Systems Description

**Abstract**
This document describes the Sensor Data Collection System of Systems (SoS), which provides end-to-end sensor measurement capture, transfer, and storage capabilities within an Arrowhead Framework local cloud. The SoS orchestrates producer systems (sensors), data transfer systems, storage systems, and consumer systems to enable industrial IoT data collection workflows with decoupled service interactions.

# Contents

# 1 Overview

This document describes the Sensor Data Collection System of Systems (SoS), which enables industrial sensor measurements to flow from producer systems through transfer services to storage systems and ultimately to consumer systems within an Arrowhead Framework local cloud. The SoS architecture decouples sensor systems from data consumers through abstract service interfaces, enabling flexible orchestration patterns and supporting multiple concurrent consumers without direct producer-consumer coupling.

The rest of this document is organized as follows. In Section 1.1, we reference major prior art capabilities of the SoS. In Section 1.2, we describe the intended usage of the SoS. In Section 1.3, we describe fundamental properties provided by the SoS. In Section 1.4, we describe delimitations of capabilities of the SoS. In Section 2, we describe the microsystems (abstract level with references to their SysDs) which constitute the SoS. In Section 3, we describe the security capabilities of the SoS.

## 1.1 Significant Prior Art

The Sensor Data Collection SoS builds upon established patterns in industrial automation and IoT architectures:

**Arrowhead Framework Core Systems:** The SoS relies on Service Registry, Authorization, and Orchestration systems as foundational infrastructure for service discovery, access control, and system-of-systems coordination.

**SenML (RFC 8428):** Sensor Measurement Lists specification provides semantic foundation for measurement representation including types, units, and temporal information.

**REST and Service-Oriented Architecture:** The SoS follows resource-oriented design patterns with loose coupling between producers and consumers through abstract service interfaces.

**Industrial IoT Edge-Cloud Patterns:** The architecture implements edge data collection with local cloud orchestration, supporting industrial requirements for latency, reliability, and local autonomy.

## 1.2 How This SoS Is Meant to Be Used

The Sensor Data Collection SoS operates in industrial automation scenarios requiring systematic sensor data acquisition:

**Manufacturing Process Monitoring:** Production line sensors (temperature, pressure, vibration) continuously submit measurements. The SoS routes data to storage systems for historical analysis and real-time monitoring dashboards.

**Environmental Control Systems:** HVAC sensors in factory environments submit climate data. The SoS enables building management systems to retrieve measurements for control decisions while archiving data for energy optimization analysis.

**Quality Assurance Workflows:** Inspection stations submit measurement data during production. The SoS provides quality systems with immediate access to inspection records while maintaining traceability archives.

**Machine Learning Training Data Collection:** Sensor measurements are accumulated in blob storage systems specifically for ML model training purposes. When management systems (controllers) specify training cycle requirements, the accumulated measurement datasets are transferred from blob storage to separate ML training local clouds. This enables centralized model training on aggregated sensor data while maintaining operational separation between production data collection and ML infrastructure.

**Typical Workflow:**

1. Sensor systems register with Service Registry upon startup
2. Authorization system validates sensor credentials and access rights
3. Orchestration system configures sensor-to-transfer service connections
4. Sensor systems periodically invoke transferData service with measurements
5. Transfer systems validate and forward data to storage systems
6. Blob storage systems accumulate measurement datasets for ML training purposes

7. Controller systems specify training cycle requirements based on management policies
8. Upon training cycle initiation, blob storage transfers accumulated datasets to ML training local cloud
9. Consumer systems discover and retrieve measurement data via orchestrated services
10. Lifecycle management systems invoke deletion operations per retention policies

The SoS enables flexible deployment scenarios where multiple consumer systems access sensor data without requiring sensors to maintain awareness of consumers. Orchestration patterns support both push-based data flows (sensors proactively submit) and pull-based patterns (consumers query storage).

## 1.3 SoS Functionalities and Properties

### 1.3.1 Functional Properties of the SoS

**Measurement Capture:** Sensor systems submit individual measurements including sensor identification, temporal information, measurement type, numerical value, unit specification, and optional metadata.

**Data Transfer:** Transfer systems accept measurements from producers, validate structure and authorization, assign unique identifiers, and route data to storage systems.

**Historical Retrieval:** Consumer systems retrieve specific measurement records by identifier, supporting audit trails, historical analysis, and quality traceability.

**Lifecycle Management:** Administrative systems manage data retention by canceling pending transfers or removing completed records according to policy.

**Service Discovery:** All systems leverage Service Registry for dynamic discovery of available services, supporting resilient operation under system additions and removals.

**Authorization Enforcement:** Authorization system validates all service interactions, ensuring only credentialed sensors submit data and only authorized consumers access records.

### 1.3.2 Configuration of SoS Properties

**Sensor Registration:** Each sensor system configures its unique sensor identifier, measurement types, units, and submission frequency. Configuration typically occurs during system commissioning.

**Transfer Routing:** Orchestration system configures which storage systems receive data from which transfer systems, enabling segmentation by measurement type, sensor location, or security zone.

**Retention Policies:** Storage systems configure data retention duration, archival strategies, and deletion triggers. Policies vary by measurement type and regulatory requirements.

**Consumer Authorization:** Authorization rules define which consumer systems access which measurement types or sensor identifiers, supporting role-based access control.

### 1.3.3 Data Stored by the Individual Microsystems

**Sensor Systems:** Minimal state - typically only maintain last successful submission timestamp for failure recovery.

**Transfer Systems:** Temporary transfer records during processing, including pending validation queue and recently submitted measurements for idempotency checking.

**Storage Systems:** Complete measurement archives including all fields from original submissions, transfer metadata (identifiers, acceptance timestamps), and indexing structures for retrieval.

**Blob Storage Systems:** Accumulated measurement datasets organized for ML training purposes. Data is structured to facilitate batch transfer to training local clouds. Blob storage maintains datasets until training cycles complete and data is confirmed transferred.

**Controller Systems:** Training cycle specifications including dataset requirements, trigger conditions (time-based, data volume thresholds, quality metrics), and inter-cloud transfer configurations. Controllers track training cycle status and coordinate with blob storage for dataset transfers.

**Authorization System:** Access control policies mapping sensor identifiers and consumer system identifiers to permitted measurement types.

**Orchestration System:** Service routing rules defining producer-transfer-storage-consumer connection patterns.

### 1.3.4 Non-Functional Properties

**Security:** All service interactions operate over TLS-secured channels with X.509 certificate authentication. Authorization checks precede all operations. Measurement data integrity is maintained through cryptographic transport protection.

**Latency:** Typical measurement submission completes within 100-500ms depending on network conditions and storage backend. Retrieval operations complete within similar timeframes for recently submitted data.

**Reliability:** The SoS tolerates individual system failures through stateless operation patterns. Sensors retry failed submissions. Storage replication provides data durability.

**Scalability:** Horizontal scaling is achieved through multiple instances of transfer and storage systems with orchestration-based load distribution. The architecture supports thousands of concurrent sensor systems.

**Energy Consumption:** Sensor systems minimize energy usage through periodic submission patterns rather than continuous connections. Lightweight protocols reduce transmission overhead on battery-powered sensors.

### 1.3.5 Stateful or Stateless

**Sensor Systems:** Stateless for measurement submission - each invocation is independent. Minimal state for failure recovery (last submission timestamp).

**Transfer Systems:** Mostly stateless - transfer operations are idempotent within time windows. Temporary state for duplicate detection.

**Storage Systems:** Stateful - maintain complete measurement archives with historical records.

**Core Systems (Service Registry, Authorization, Orchestration):** Stateful - maintain service registrations, authorization rules, and orchestration patterns across system lifecycle.

## 1.4 Important Delimitations

**Delimitations:**

The Sensor Data Collection SoS handles individual sensor measurements through discrete service invocations. Bulk data upload, batch operations, and high-frequency streaming (¿10 Hz) are outside scope. Systems requiring such capabilities must implement protocol-specific extensions.

Real-time control loops are not directly supported. The request-response architecture introduces latency unsuitable for sub-100ms control requirements. The SoS targets monitoring and analysis rather than closed-loop control.

Data transformation, unit conversion, aggregation, and semantic reasoning are not provided by the SoS. Measurements are transferred and stored as submitted. Consumer systems requiring transformed data must implement their own processing.

The SoS does not mandate specific storage duration, retention policies, or archival mechanisms. These decisions are implementation-specific and vary by deployment requirements.

Inter-cloud data transfer for ML training is within scope through blob storage systems coordinated by controllers. However, general-purpose cross-cloud data federation, replication, or synchronization beyond training dataset transfer is not addressed. The SoS operates primarily within a single Arrowhead local cloud with specific extensions for ML training cloud integration.

Measurement quality validation beyond structural checks (required fields, data types) is not performed. Semantic validation (range checking, outlier detection, calibration verification) is the responsibility of consumer systems.

**What the SoS Solves:**

- Decoupled sensor-to-consumer data flow enabling flexible system composition
- Standardized measurement representation across heterogeneous sensor types
- Service-oriented access control and authorization for industrial environments
- Historical measurement retrieval supporting audit and analysis workflows
- Dynamic service discovery enabling system additions without reconfiguration
- Systematic accumulation of training datasets in blob storage for ML model development

- Controlled inter-cloud transfer of training data from production to ML infrastructure
- Management-driven training cycle coordination through controller systems

**What the SoS Does Not Solve:**

- Real-time streaming protocols for high-frequency continuous data
- Data transformation, aggregation, or semantic processing
- Long-term archival strategies and data lifecycle governance
- General-purpose cross-cloud data federation beyond ML training dataset transfer
- Quality validation and outlier detection beyond structural checks
- Protocol-specific optimizations for constrained devices
- ML model training, inference, or deployment (handled by separate training local cloud)

# 2 Services

This section describes consumed and produced services at the SoS level, indicating how microsystems interact through abstract service interfaces.

## 2.1 Produced Services

The Sensor Data Collection SoS produces services consumed by external systems:

### 2.1.1 transferData Service

**Description:** Core service enabling sensor measurement submission, retrieval, and lifecycle management.
**Reference:** See transferData Service Description (SD) v4.4.1 for complete specification.
**Operations:**

- Transfer: Submit sensor measurement data
- GetTransfer: Retrieve transfer record by identifier
- DeleteTransfer: Cancel or remove transfer records
- Echo: Service health verification

**Produced By:** Transfer systems (microsystem component of SoS)
**Consumed By:** External sensor systems (producers) and consumer systems (data analysts, monitoring dashboards, quality systems)

### 2.1.2 storageMeasurement Service

**Description:** Persistent storage service for measurement archives.
**Reference:** storageMeasurement Service Description (SD) - to be specified
**Operations:**

- Store: Persist measurement record
- Retrieve: Query measurements by sensor, type, or time range
- Delete: Remove measurement records

**Produced By:** Storage systems (microsystem component of SoS)
**Consumed By:** Transfer systems (for persistence) and consumer systems (for historical queries)

### 2.1.3 blobStorage Service

**Description:** Specialized storage service for accumulating ML training datasets with inter-cloud transfer capabilities.
**Reference:** blobStorage Service Description (SD) - to be specified
**Operations:**

- AccumulateData: Add measurement data to training dataset
- InitiateTransfer: Begin inter-cloud training dataset transfer
- GetDatasetStatus: Query dataset size, completeness, and transfer status
- ClearDataset: Remove transferred or obsolete training data

**Produced By:** Blob storage systems (microsystem component of SoS)
**Consumed By:** Transfer systems (for training data accumulation), controller systems (for training cycle coordination), and ML training local cloud systems (for dataset retrieval)

#### 2.1.4 trainingControl Service

**Description:** Management interface for specifying and coordinating ML training cycles.
**Reference:** trainingControl Service Description (SD) - to be specified
**Operations:**

- SpecifyTrainingRequirements: Define dataset requirements and trigger conditions
- InitiateTrainingCycle: Trigger dataset transfer and training process
- GetTrainingStatus: Query training cycle progress and completion
- CancelTrainingCycle: Abort in-progress training coordination

**Produced By:** Controller systems (microsystem component of SoS)
**Consumed By:** Management systems (for training cycle specification) and blob storage systems (for transfer coordination)

## 2.2 Consumed Services

The Sensor Data Collection SoS depends on Arrowhead core services:

#### 2.2.1 ServiceRegistry Service

**Description:** Arrowhead mandatory core service enabling dynamic service discovery.
**Reference:** Arrowhead Framework ServiceRegistry SD
**Operations:**

- Register: Systems register their provided services
- Unregister: Systems remove service registrations
- Query: Systems discover available services by definition and metadata

**Consumed By:** All SoS microsystems (sensors, transfer, storage, consumers)

#### 2.2.2 Authorization Service

**Description:** Arrowhead mandatory core service enforcing access control policies.
**Reference:** Arrowhead Framework Authorization SD
**Operations:**

- GetAuthorizationRules: Systems retrieve applicable access control rules
- CheckAuthorization: Service providers validate consumer authorization

**Consumed By:** Transfer systems and storage systems (for authorization enforcement)

#### 2.2.3 Orchestration Service

**Description:** Arrowhead mandatory core service coordinating service interactions.
**Reference:** Arrowhead Framework Orchestration SD
**Operations:**

- RequestOrchestration: Systems request orchestrated service connections
- StoreOrchestrationRules: Administrators configure orchestration patterns

**Consumed By:** All SoS microsystems for dynamic service binding

### 2.2.4 InterCloudGateway Service

**Description:** Arrowhead inter-cloud communication service enabling secure data transfer between local clouds.
**Reference:** Arrowhead Framework InterCloudGateway SD
**Operations:**

- InitiateInterCloudTransfer: Begin secure data transfer to remote local cloud
- GetTransferStatus: Query inter-cloud transfer progress
- AuthorizeRemoteAccess: Validate cross-cloud authorization tokens

**Consumed By:** Blob storage systems (for training dataset transfer to ML training local cloud)

# 3 Security

The Sensor Data Collection SoS operates exclusively in Arrowhead secure mode. All systems require valid X.509 certificates compliant with the Arrowhead certificate profile.

## 3.1 Security Model

### 3.1.1 Protocol Security

**Transport Protection:** All service interactions occur over TLS 1.3 or higher with mutual authentication. Certificate validation enforces Arrowhead naming conventions.

**Supported Protocols:** HTTP/HTTPS for RESTful implementations, CoAP/CoAPS for constrained devices, with mandatory encryption for all variants.

### 3.1.2 Authentication

**System Authentication:** X.509 certificates identify all systems. Certificate Common Names follow Arrowhead system naming: `<systemname>.<cloudname>.<operator>.arrowhead.eu`

**Certificate Validation:** Transfer and storage systems validate client certificates against Authorization system rules before processing operations.

**Certificate Lifecycle:** Systems obtain certificates during provisioning. Certificate renewal follows Arrowhead onboarding procedures.

### 3.1.3 Authorization

**Service-Level Authorization:** Authorization system maintains access control lists defining:

- Which sensor systems may invoke Transfer operations
- Which consumer systems may invoke GetTransfer operations
- Which administrative systems may invoke DeleteTransfer operations

**Data-Level Authorization:** Policies optionally restrict access by measurement type or sensor identifier, enabling fine-grained control.

**Authorization Enforcement:** Transfer systems check authorization before accepting measurements. Storage systems check authorization before returning records.

### 3.1.4 Data Protection

**In Transit:** TLS encryption protects all measurement data during transfer operations. Certificate-based authentication prevents unauthorized submission.

**At Rest:** Storage systems implement encryption for persistent measurement archives. Encryption keys are managed per local cloud security policies.

**Data Integrity:** Cryptographic transport protection ensures measurements cannot be modified in transit. Storage systems maintain integrity through checksums or cryptographic hashes.

### 3.1.5 Operational Security

**Insecure Mode:** The SoS does NOT support insecure (unencrypted, unauthenticated) operation. All deployments require Arrowhead secure mode.

**Certificate Revocation:** Systems check certificate validity against Certificate Authority revocation lists. Revoked certificates immediately lose service access.

**Audit Logging:** Transfer and storage systems log all operations including system identifiers, timestamps, and operation results for security auditing.

**Non-Compliant Certificates:** Systems rejecting non-Arrowhead-compliant certificates prevents unauthorized access from improperly provisioned systems.

# 4 Revision History

## 4.1 Amendments

| No. | Date | Version | Subject of Amendments | Author |
|---|---|---|---|---|
| 1 | 2025-10-22 | 1.0.0 | Initial draft | Rasmus Tengstedt |

## 4.2 Quality Assurance

| No. | Date | Version | Approved by |
|---|---|---|---|
| 1 | - | 1.0.0 | Pending |