transferData

Service Description

Abstract
This document describes the transferData service, which enables sensor data collection and measurement transfer between systems within the Arrowhead Framework. The service provides abstract operations for submitting sensor measurements, retrieving transfer records, and managing data transfers independently of implementation protocols.

Contents

1 Overview

This document describes the transferData service, which enables systems to submit sensor measurements and retrieve transfer records within the Arrowhead Framework. The service abstracts data transfer operations, allowing sensor systems to transmit measurements to data consumers without requiring knowledge of underlying protocols or storage mechanisms.

The rest of this document is organized as follows. In Section 2, we describe the abstract message operations provided by the service. In Section 3, we end the document by presenting the data types used by the mentioned operations.

1.1 Significant Prior Art

The transferData service builds upon established IoT data exchange patterns and sensor measurement standards. Key influences include:

SenML (Sensor Measurement Lists) - RFC 8428: Provides semantic structure for sensor data representation, including measurement types, units, and timestamps.

REST architectural principles: Operations follow resource-oriented design with standard CRUD patterns adapted for sensor data lifecycle management.

Arrowhead Framework service patterns: Conforms to Arrowhead service registry, authorization, and orchestration mechanisms for industrial automation systems.

1.2 How This Service Is Meant to Be Used

The transferData service operates in sensor data collection scenarios within Arrowhead local clouds:

Sensor data submission: Producer systems (sensors, IoT devices) invoke the Transfer operation to submit measurements. Each submission includes sensor identification, measurement type, value with units, and temporal information.

Historical data retrieval: Consumer systems invoke GetTransfer to retrieve specific measurement records by identifier, enabling audit trails and historical analysis.

Transfer lifecycle management: Systems invoke DeleteTransfer to cancel pending transfers or remove records, supporting data retention policies.

Typical workflow:

1. Sensor system discovers transferData service via Service Registry
2. Authorization system validates sensor credentials
3. Sensor invokes Transfer operation with measurement data
4. Service responds with unique transfer identifier
5. Consumer systems retrieve data via GetTransfer using transfer identifiers
6. System administrators manage data lifecycle via DeleteTransfer

The service decouples data producers from consumers, enabling flexible orchestration patterns where multiple consumers can access sensor data without direct producer-consumer coupling.

1.3 Important Delimitations and Dependencies

Delimitations:

The transferData service handles individual sensor measurements, not bulk time-series operations. Systems requiring batch uploads or historical aggregation must invoke Transfer multiple times or implement batch extensions.

The service does not mandate storage duration, data retention policies, or archival mechanisms. These concerns belong to service provider implementations.

Real-time streaming is outside scope. The service follows request-response patterns suitable for periodic measurements, not continuous data streams.

Data transformation, unit conversion, or semantic reasoning are not provided. Measurements are stored and retrieved as submitted.

Dependencies:

Service Registry (mandatory): Service discovery requires functional Service Registry for consumers to locate transferData providers.

Authorization System (mandatory): Proper certificate-based authentication via Arrowhead Authorization System ensures only authorized sensors can submit data and authorized consumers can retrieve it.

Orchestration System (recommended): Dynamic consumer-provider binding benefits from Orchestration System, though static configuration is possible.

Time synchronization: Sensor systems must maintain accurate time sources for measurement timestamps. Clock skew degrades temporal data quality.

2 Service Interface

This section describes the abstract operations of the transferData service. Each subsection names an operation, an input type (in parentheses), and an output type (preceded by colon). Input and output types are only denoted when accepted or returned by the interface.

All abstract data types are defined in Section 3.

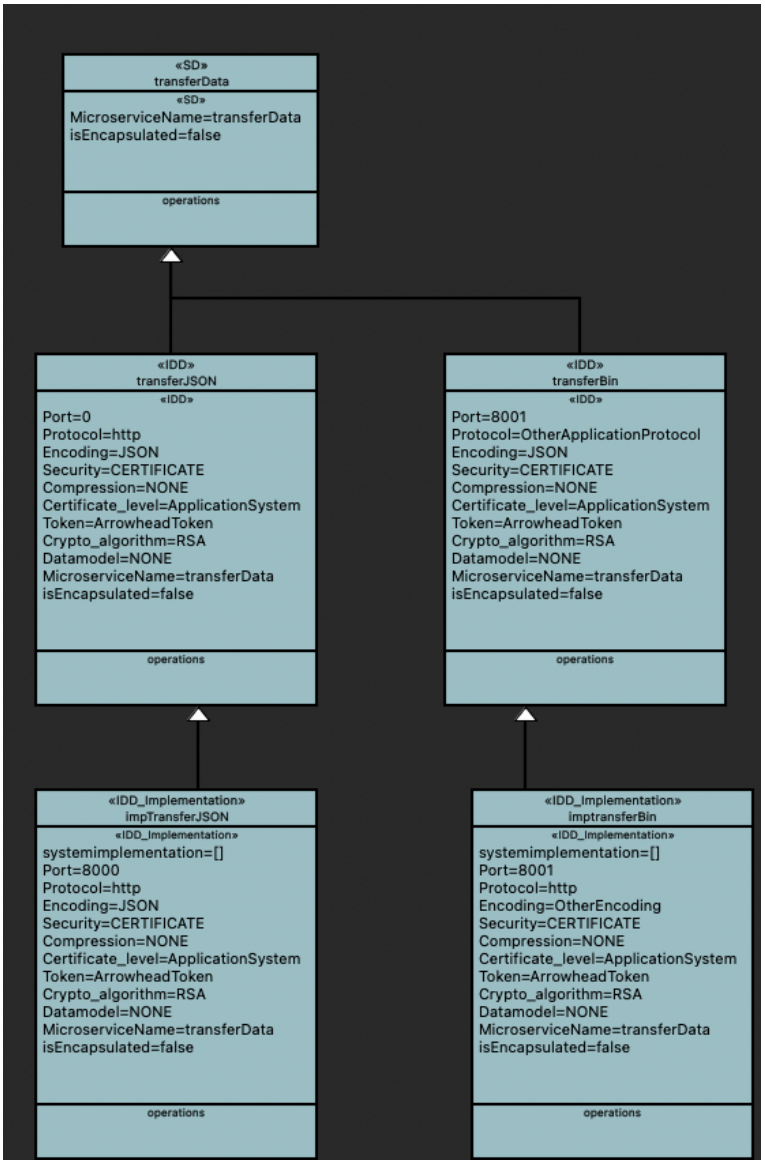

Figure 1: SysML block diagram of transferData service and its operations

The following interface operations are available.

2.1 operation Transfer (TransferRequest) : TransferResponse

The Transfer operation submits sensor measurement data from a provider system. The operation accepts a measurement containing sensor identification, temporal information, measurement type, numerical value, unit specification, and optional metadata. The service validates the measurement structure and returns a unique transfer identifier enabling later retrieval. The operation is idempotent when invoked with identical sensor identifiers and timestamps within implementation-defined time windows.

2.2 operation GetTransfer (TransferId) : TransferRecord

The GetTransfer operation retrieves a complete transfer record using its unique identifier. The operation returns all measurement data including current processing status. Implementations may support historical queries beyond active transfer lifecycles, though retention duration is implementation-specific. The operation fails if the specified identifier does not exist or the requesting system lacks authorization for the record.

2.3 operation DeleteTransfer (TransferId) : StatusCodeKind

The DeleteTransfer operation cancels a pending transfer or removes a completed transfer record. The operation is idempotent; repeated invocations on the same identifier succeed without error. Implementations may restrict deletion to transfer owners or authorized administrators. The operation returns success status codes defined by the implementing protocol.

2.4 operation Echo () : StatusCodeKind

The Echo operation provides service availability verification. The operation requires no input and returns a success status when the service is operational. This enables health monitoring and service discovery validation without side effects.

3 Information Model

Here, all data objects that can be part of the transferData service are listed in alphabetic order. Each subsection describes one type of object beginning with the *struct* keyword, denoting a collection of named fields with explicit data types.

Section 3.5 defines primitive types used throughout this model.

3.1 struct TransferRequest

This structure is used to submit sensor measurement data for transfer.

Field	Type	Description
sensorId	Name	Unique identifier of the sensor producing the measurement. Must be consistent across invocations from the same sensor.
timestamp	DateTime	Moment when the measurement was captured. Must represent actual measurement time, not transmission time.
measurementType	Name	Classification of the measured phenomenon (e.g., temperature, humidity, pressure, velocity).
value	Number	Numerical measurement value. Precision and range are implementation-defined.
unit	Name	Unit of measurement specification (e.g., celsius, percent, pascal, meters-per-second).
metadata	Metadata	Optional additional context such as sensor location, calibration data, or accuracy specifications.

3.2 struct TransferResponse

This structure is returned after successful transfer submission.

Field	Type	Description
transferId	Id	Unique identifier assigned to this transfer. Used for subsequent GetTransfer and DeleteTransfer operations.
status	Status	Current transfer processing status. Indicates whether the measurement was accepted, is pending validation, or encountered errors.
timestamp	DateTime	Moment when the service accepted the transfer request. Distinct from the measurement timestamp.
sensorId	Name	Echo of the submitting sensor identifier for correlation purposes.

3.3 struct TransferRecord

This structure represents a complete transfer record including measurement data and processing metadata.

Field	Type	Description
transferId	Id	Unique identifier for this transfer record.
sensorId	Name	Unique identifier of the source sensor.
timestamp	DateTime	Original measurement capture time.
measurementType	Name	Classification of the measured phenomenon.
value	Number	Numerical measurement value.
unit	Name	Unit specification for the measurement.
status	Status	Current processing status of this transfer.
metadata	Metadata	Optional additional measurement context.

3.4 struct Metadata

An unordered collection of key-value pairs providing additional context for measurements. Keys and values are both text strings. Common uses include sensor location identifiers, calibration coefficients, accuracy specifications, or environmental conditions during measurement.

3.5 Primitives

Types mentioned throughout this document that are assumed available to service implementations. Concrete interpretations must be provided by IDD documents implementing this service.

Type	Description
DateTime	Pinpoints a specific moment in time with sufficient precision for sensor measurements. Implementations must specify timezone handling and precision.
Id	Unique identifier distinguishing transfer records. Implementations choose appropriate types (integers, UUIDs, etc.).
Metadata	Collection of key-value pairs where both keys and values are text strings.
Name	Short text identifier intended to be both human and machine-readable. Typically under 64 characters.
Number	Numerical value supporting sensor measurement ranges. Implementations must specify precision (floating-point, fixed-point, integer).
Status	Enumeration of transfer processing states. Implementations define specific values (e.g., accepted, pending, completed, failed, cancelled).
StatusCodeKind	Protocol-specific status indicator for operation success or failure.
TransferId	Alias for Id used specifically for transfer record identification.

4 Revision History

4.1 Amendments

No.	Date	Version	Subject of Amendments	Author
1	2025-10-22	4.4.1	Initial draft	Rasmus Tengstedt

4.2 Quality Assurance

No.	Date	Version	Approved by
1	-	4.4.1	Pending