University of Tasmania
GPO Box 367
Hobart 7001
Australia

imos.org.au

May 2026

## IMOS NESP 5.9

# AMSA Vessel Tracking Optimised Data

### Product Technical Document

Version 1.1

Karishma Khanna
Australian Ocean Data Network AODN / IMOS

Australia's Integrated Marine Observing System (IMOS) is enabled by the National Collaborative Research Infrastructure Strategy (NCRIS). It is operated by a consortium of institutions as an unincorporated joint venture, with the University of Tasmania as Lead Agent.

IMOS acknowledges the Traditional Custodians and Elders of the land and sea on which we work and observe, and recognise them as Australia's first marine scientists and carers of sea Country. We pay our respects to Aboriginal and Torres Strait Islander peoples past and present.

# 1. Version History

| Version | Date | Comments | Author |
|---|---|---|---|
| v1.0 | January-2026 | Initial Release | Karishma, K |
| v1.1 | May 2026 | Minor Changes | Galindo, T |
| | | | |

## Citation

Karishma, K. (2026). AMSA Vessel Tracking Optimised Data Product Technical Document. Version 1.1. Integrated Marine Observing System.

## Copyright/Creative Commons Licence

# Table of Contents


**1. Publication Details**

1.1 Revision History

1.2 Citation

1.3 License

**2. Background**

2.1 Overview

Basis of the AMSA Vessel Tracking Data Product

Data source

2.2 Source Data Characteristics

Delivery structure

Common inconsistencies accounted for

2.3 Output Data Products

Validated dataset (HIVE partitioned Parquet)

Rejected dataset (Per source file)

**3. Methodology**

3.1 Extract

3.2 Transform

A) Column standardisation

B) Coordinate recovery

C) Timestamp normalisation

D) Enrichment

Geospatial Enrichment

3.3 Validation Rules (Waterfall)

3.4 Schema Enforcement

3.5 Load Strategy

Rejected output

HIVE partition write

3.6 Output Schema (Summary)

Validated dataset

Rejected dataset

3.7 Configuration

Global config (`amsa_config.json`)

**4. Flow Advantages**

# 2. Background

## 2.1 Overview

AMSA publishes monthly Vessel Tracking spatial datasets as downloadable archives. The raw deliveries are provided as **nested ZIP files** containing **Shapefiles**. This work converts those monthly Shapefile deliveries into a consistent, analytics-ready Parquet dataset with:

- cleaned and standardised coordinates
- consistently typed UTC timestamps
- a strict output schema
- an H3 spatial index for fast spatial grouping and joins
- a separate rejected-records output with explicit rejection reasons

### Basis of the AMSA Vessel Tracking Data Product

AMSA (Australian Maritime Safety Authority) vessel tracking data is critical for maritime domain awareness, search and rescue operations, and environmental monitoring. However, the raw monthly Shapefile deliveries present several challenges:

- nested archive structures require manual extraction
- inconsistent timestamp formats across historical files
- coordinate columns may vary or require derivation from geometry
- no built-in spatial indexing for efficient queries

The goal of this aggregated data product is to transform these monthly deliveries into a unified, analytics-ready dataset that is optimised for spatial queries and time-series analysis.

### Data source

`https://www.operations.amsa.gov.au/spatial/DataServices/DigitalData`

The source package also includes a metadata PDF describing the dataset fields and spatial reference details. That metadata is used as the reference for schema and validation expectations.

## 2.2 Source Data Characteristics

### Delivery structure

Monthly deliveries follow this structure:

- **Outer ZIP** (monthly package)
  - contains an **Inner ZIP**
    - contains **Shapefiles** ( `.shp` + `.dbf` + `.shx` + `.prj` , etc.)

---

### Common inconsistencies accounted for

To keep the pipeline robust across historical files and any future variations, the processing logic includes defensive handling for scenarios that are robust to future changes in monthly geospatial files:

- archive naming conventions may vary (month abbreviation vs full month name, ordering of month/year)
- timestamps provided in different string formats
- coordinate columns may be present ( `LAT` , `LON` ) or coordinates may need to be derived from geometry
- duplicate records appearing in the source extracts

---

## 2.3 Output Data Products

Two outputs are generated:

### Validated dataset (HIVE partitioned Parquet)

Records are written to a HIVE partition path per source file: `datauplift/amsa/dataset/year={year}/source={source_file_name}/0.parquet`

Each partition file contains:

- validated, standardised records
- `H3_INDEX` at H3 resolution 4 (configurable via `amsa_config.json`)
- `AUSTRALIAN_MARINE_REGIONS_TAGS` containing relevant Australian marine region classifications
- audit fields ( `SOURCE_FILE_NAME` , `PROCESSED_DATE` )

Idempotency is achieved by overwriting the same partition path when a month is reprocessed, rather than a load-remove-append cycle.

### Rejected dataset (Per source file)

Records that fail validation are written separately:

- written per processed source file
- includes `REJECTION_REASON`
- used for traceability and diagnostics

# 3. Methodology

The AMSA Vessel Tracking Data Product relies on the standard Extract, Transform, and Load (ETL) methodology orchestrated with Prefect.

## 3.1 Extract

**Objective:** read monthly archives and convert Shapefile content into Arrow tables.

Steps:

1. Download outer ZIP from S3 into memory ( `BytesIO` )
2. Parse Year/Month from the outer ZIP filename using resilient regex patterns
3. Extract the inner ZIP to a temporary directory
4. Extract Shapefile contents using:
    - standard `zipfile` extraction first
    - fallback `stream_unzip` for archives that fail standard extraction
5. Read `*.shp` using `pyogrio.read_dataframe(..., use_arrow=True)`
6. Convert geometry to WKB for safe transport and downstream processing
7. Convert to `pyarrow.Table` and return extracted items

**Extract output:**

`(shapefile_name, arrow_table, year, month)` for each Shapefile found.

## 3.2 Transform

**Objective:** standardise columns, recover coordinates, normalise time, enrich, validate, and enforce a strict schema.

### A) Column standardisation

- uppercase all column names
- map known misspellings to standard names, e.g.:
  - `TMESTAMP`, `TIMETAMP`, `TIMESAMP` → `TIMESTAMP`
  - `LATITUDE` → `LAT`
  - `LONGITUDE` → `LON`

### B) Coordinate recovery

- if `LAT/LON` exist, use them
- if missing or null, derive from geometry:
  - decode WKB geometry to X/Y (LON/LAT)
- round coordinates to 5 decimal places

### C) Timestamp normalisation

- parse `TIMESTAMP` using a coalesced multi-format strategy:
  - date-only
  - 24-hour datetime
  - 12-hour datetime with AM/PM
- normalise to `timestamp[ms, UTC]`

### D) Enrichment

- `SOURCE_FILE_NAME` (lineage key)
- `PROCESSED_DATE` (UTC processing timestamp)
- `H3_INDEX` computed using configured H3 resolution
- `AUSTRALIAN_MARINE_REGIONS_TAGS` containing relevant Australian marine region classifications

---

### Geospatial Enrichment

The output dataset is enriched with the following geospatial columns:

- `H3_INDEX` - A hexadecimal string representing an H3 polygon at resolution 4 (configurable via `amsa_config.json`), derived from the `LAT` and `LON` coordinates
- `AUSTRALIAN_MARINE_REGIONS_TAGS` - A `|` separated tag column featuring common Australian marine regions

---

## 3.3 Validation Rules (Waterfall)

Validation is applied sequentially. Each rejected set is preserved and labelled.

1. **Duplicate detection**
    - rule: full-row first-distinct check
    - reason: `Duplicate`
2. **Missing coordinates**
    - rule: `LAT` and `LON` must be non-null
    - reason: `Missing Coordinates`
3. **Invalid global coordinate ranges**
    - rule: `-90 <= LAT <= 90` and `-180 <= LON <= 180`
    - reason: `Invalid Global Coords`
4. **Configured region bounds (SAR bounding box)**
    - rule: coordinates must be within configured bounds
    - reason: `Outside SAR Region`
5. **Non-marine location detection**
    - rule: coordinates identified as land using global_land_mask. This check is applied on:
        - Australia Land : `-10 < LAT < -45` and `110 < LON < 156`
        - Antartica Land : `LAT < -68`
    - reason: `Non-Marine Locations`

Rejected rows from all stages are concatenated into a single rejected DataFrame and written to S3.

---

## 3.4 Schema Enforcement

A strict output schema is defined via a Pydantic wrapper around a PyArrow schema. Before load:

- missing expected columns are added as nulls
- columns are ordered consistently
- the dataset is cast to the target Arrow schema
- the final typed table is rehydrated back to Polars for writing

This ensures stable typing across months, years, and reruns.

---

## 3.5 Load Strategy

### Rejected output

Rejected records are written as Parquet (Snappy) files per source:

- `.../{year}/{source_stem}_rejected.parquet`

---

### HIVE partition write

The validated dataset is written directly to a HIVE partition path:

`datauplift/amsa/dataset/year={year}/source={source_file_name}/0.parquet`

Idempotency is guaranteed by overwriting the same partition path when a month is reprocessed. There is no load-remove-append cycle; re-running a month simply overwrites the existing partition.

---

## 3.6 Output Schema (Summary)

### Validated dataset

The validated master Parquet includes the following fields:

| name | type | nullable | description |
|---|---|---|---|
| CRAFT_ID | string | **false** | Unique vessel identifier |
| LAT | float64 | **false** | Latitude in decimal degrees |
| LON | float64 | **false** | Longitude in decimal degrees |
| TIMESTAMP | timestamp[ms, UTC] | **false** | Observation timestamp (UTC) |
| COURSE | float64 | true | Vessel course in degrees |
| SPEED | float64 | true | Vessel speed |
| TYPE | string | true | Vessel type classification |
| SUBTYPE | string | true | Vessel subtype classification |
| LENGTH | float64 | true | Vessel length in metres |
| BEAM | float64 | true | Vessel beam in metres |
| DRAUGHT | float64 | true | Vessel draught in metres |
| H3_INDEX | string | **false** | H3 spatial index (resolution 4, configurable via `amsa_config.json`) |
| AUSTRALIAN_MARINE_REGIONS_TAGS | string | true | Pipe-separated marine region tags |
| SOURCE_FILE_NAME | string | **false** | Original source file (lineage key) |
| PROCESSED_DATE | timestamp[ms, UTC] | **false** | ETL processing timestamp |

### Rejected dataset

Rejected output includes all source fields plus:

| name | type | nullable | description |
| --- | --- | --- | --- |
| REJECTION_REASON | string | **false** | Reason for rejection (e.g., `Duplicate`, `Missing Coordinates`) |

## 3.7 Configuration

The pipeline is configuration-driven and validated using Pydantic.

### Global config (`amsa_config.json`)

Controls:

- input/output/rejected S3 locations
- bounding box for region validation
- H3 resolution

Example:

```
{
  "input":   { "block_name": "processing-bucket", "path":
"karishma.khanna/AMSA/Processing/Extracted" },
  "output":  { "block_name": "optimised-bucket",  "path":
"karishma.khanna/AMSA/Optimised" },
  "rejected":{ "block_name": "error-bucket",      "path":
"karishma.khanna/AMSA/Rejected" },
  "geospatial": {
    "lon_min": 75.0,
    "lon_max": 163.0,
    "lat_min": -86.6,
    "lat_max": -2.0,
    "h3_resolution": 4
  }
}
```

# 4. Flow Advantages

**Reproducibility** the original AMSA monthly Shapefile archives require manual extraction and geospatial software to process. The ETL procedure makes this process reliable and replicable.

**Space Efficiency** raw monthly Shapefile deliveries (~~2.5GB per month) are consolidated into yearly Parquet files (~~65MB per year with Snappy compression), optimising storage and access.

**Time Efficiency** scheduled updates allow users to access up-to-date aggregated data without re-implementing the extraction and transformation steps.

**Data Quality** waterfall validation ensures coordinate integrity, regional bounds compliance, and land/marine classification with full traceability of rejected records.

**Idempotent Partition Write** the HIVE partition write strategy allows safe reprocessing of individual months without data duplication. Re-running a month simply overwrites the same partition path.

**Cloud-native workflows** reduce I/O overhead and enable seamless integration with AWS S3 storage.