OGC® DOCUMENT: 22-016R3

External identifier of this OGC® document: http://www.opengis.net/doc/PER/T18-D020

Open Geospatial Consortium

TESTBED-18: MOVING FEATURES ENGINEERING REPORT

ENGINEERING REPORT

PUBLISHED

Submission Date: 2022-11-22
Approval Date: 2023-03-02
Publication Date: 2023-06-26
Editor: Brittany Eaton

Note

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. The Open Geospatial Consortium shall not be held responsible for identifying any or all such patent rights.

Recipients of this document are requested to submit, with their comments, notification of any relevant patent claims or other intellectual property rights of which they may be aware that might be infringed by any implementation of the standard set forth in this document, and to provide supporting documentation.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

I ABSTRACT

This OGC Testbed-18 (TB-18) Engineering Report (ER) is based on previous OGC Moving Features and Sensor Integration (MFSI) activities. The OGC TB-18 MFSI task addressed the interoperability between sensors and between sensing systems as well as the exchange of multiple sources of detected moving objects into one common analytic client. This ER describes the architecture framework for multi-source moving object detection into the client supported by OGC MFSI Standards and describes challenges of multi-sensor integration in the context of Moving Features data.

II EXECUTIVE SUMMARY

The OGC Testbed-18 initiative aimed to explore six tasks, including advanced interoperability for: Building Energy; Secure; Asynchronous Catalogs; Identifiers for Reproducible Science; Moving Features and Sensor Integration (MFSI); 3D+ Data Standards and Streaming; and Machine Learning (ML) Training Data (TD).

The goal of the MFSI task was to define a powerful Application Programming Interface (API) for discovery, access, and exchange of moving features and their corresponding tracks and to exercise this API in a near real-time scenario. The MFSI task considered these advancements by addressing the following two application scenarios.

- An extension of the previous work on the Federated Marine Spatial Data Infrastructure (FMSDI) project.
- The monitoring of hurricanes and Saildrones.

This Engineering Report (ER) represents deliverable D020 of the OGC Testbed-18 Moving Features and Sensor Integration task. The ER explores the architecture for collaborative distributed object detection and analysis of multi-source motion imagery.

The ER begins with an overview of Testbed-17 work. The overview is followed by an explanation of Moving Features and Sensor Integration in the context of the task requirements and use case scenarios. Then, the individual Testbed participant deliverables are reviewed in terms of data, architecture, functions, and results. Finally, the Technical Integration Experiment (TIE) results, future work, and lessons learned are summarized.

III KEYWORDS

The following are keywords to be used by search engines and document catalogues.

Moving Features, Sensor hub, ingestion service, hurricane tracking, Actuator, Trajector

IV SECURITY CONSIDERATIONS

No security considerations have been made for this document.

V SUBMITTING ORGANIZATIONS

The following organizations submitted this Document to the Open Geospatial Consortium (OGC):

- Blue Monocle, Inc.

VI SUBMITTERS

All questions regarding this document should be directed to the editor or the contributors:

NAME	ORGANIZATION	ROLE
Brittany Eaton	Blue Monocle, Inc.	Editor
Logan Stark	Blue Monocle, Inc.	Contributor
Gianpiero Maiello	Superelectric	Contributor
Alex Robin	Botts Innovative Research	Contributor
Zhining Gu	Arizona State University	Contributor
Glenn Laughlin	Pelagis	Contributor
Rob Smith	Away Team	Contributor
Sara Saeedi	OGC	Contributor

1 SCOPE

1 SCOPE

This Testbed-18 Engineering report (ER) begins with an introduction to previous Testbed-17 work on Moving Features, followed by an explanation of Moving Features and Sensor Integration in the context of the task requirements and the hurricane and vessel tracking use case scenario. Then, the individual Testbed participant deliverables are reviewed in terms of data, architecture, functions, and results. The moving features ingestion services from two different participants are explored, as well as the sensor hub, and then the client in the context of three participants is discussed. Additionally, a chapter discussing orientation analysis of geotagged video for road network use case scenarios from a TB-18 observer participant is included. Finally, the summary reviews the Technology Integration Experiments (TIEs), future work, and lessons learned.

2

NORMATIVE REFERENCES

2 NORMATIVE REFERENCES

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

Kyoung-Sook KIM, Nobuhiro ISHIMARU: OGC 19-045r3, **Moving Features Encoding Extension — JSON**, 2019 https://docs.ogc.org/is/19-045r3/19-045r3.html

Open API Initiative: **OpenAPI Specification 3.0.2**, 2018 https://github.com/OAI/OpenAPI-Specification/blob/master/versions/3.0.2.md

van den Brink, L., Portele, C., Vretanos, P.: OGC 10-100r3, **Geography Markup Language (GML) Simple Features Profile**, 2012 http://portal.opengeospatial.org/files/?artifact_id=42729

W3C: **HTML5**, W3C Recommendation, 2019 http://www.w3.org/TR/html5/

Schema.org: http://schema.org/docs/schemas.html

R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, T. Berners-Lee: IETF RFC 2616, *Hypertext Transfer Protocol — HTTP/1.1*. RFC Publisher (1999). https://www.rfc-editor.org/info/rfc2616.

E. Rescorla: IETF RFC 2818, *HTTP Over TLS*. RFC Publisher (2000). https://www.rfc-editor.org/info/rfc2818.

G. Klyne, C. Newman: IETF RFC 3339, *Date and Time on the Internet: Timestamps*. RFC Publisher (2002). https://www.rfc-editor.org/info/rfc3339.

M. Nottingham: IETF RFC 8288, *Web Linking*. RFC Publisher (2017). https://www.rfc-editor.org/info/rfc8288.

H. Butler, M. Daly, A. Doyle, S. Gillies, S. Hagen, T. Schaub: IETF RFC 7946, *The GeoJSON Format*. RFC Publisher (2016). https://www.rfc-editor.org/info/rfc7946.

3

FOREWORD

3 FOREWORD

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. The Open Geospatial Consortium shall not be held responsible for identifying any or all such patent rights.

Recipients of this document are requested to submit, with their comments, notification of any relevant patent claims or other intellectual property rights of which they may be aware that might be infringed by any implementation of the standard set forth in this document, and to provide supporting documentation.

4

INTRODUCTION

4 INTRODUCTION

Testbed 17 (TB-17) successfully tracked moving objects by combining data from multiple sensors mounted on a moving platform. Matching object trajectories synchronized to motion imagery data enabled visual identification of the tracked object for more accurate discrimination of observations. Calculation of movement attributes increased confidence in correct identification by confirming that values were in the expected range for that object type, such as speed, which can be calculated even from short tracks with a one-second duration. This work was demonstrated using two real-time situational awareness scenarios: the detection and tracking of moving buses in front of a school; and autonomous vehicle use case analyzed to detect and track people and vehicles moving nearby with WebVMT. WebVMT is an enabling technology whose main use is for marking up external map track resources in connection with the HTML <track> element. WebVMT files provide map presentation and annotation synchronized to video content, including interpolation, and more generally any form of geolocation data that is time-aligned with audio or video content.

In the OGC Moving Features, a 'feature' is defined as an abstraction of real-world phenomena [ISO 19109:2015] whereas a "moving feature" is defined as a representation, using a local origin and local ordinate vectors, of a geometric object at a given reference time [adapted from ISO 19141:2008]. The goal of this ER is to demonstrate the business value of moving features that play an essential role in hurricane and ship tracking scenarios as well as other use cases.

Moving Features are a vital part of the future of the Internet of Things (IoT). The fast-pace growth of digital motion imagery and advancements in machine learning technology continue to accelerate widespread use of moving feature detection and analysis systems. The overall Testbed 18 MF goal was to define a powerful Application Programming Interface (API) for discovery, access, and exchange of moving features and their corresponding tracks and to exercise this API in a near real-time scenario. The OGC Testbed-18 MF task considers these advancements by addressing three application scenarios.

The first scenario is an extension of the previous work completed as part of the Marine Data Working Group (DWG) Federated Marine Spatial Data Infrastructure (FMSDI) project. In this scenario, daily aggregated vessel traffic from Denmark published to the Automatic Identification System (AIS) Vessel Traffic system was used. This use case focused on using the vessel traffic to understand the positional relationships of ships to marine protected areas as defined by the IHO S-122 standard. Of interest is identifying those vessels that altered their path away from a Maritime Patrol Aircraft (MPA) when not required. This analysis was based on class type and the MPA restrictions. Another related use case is 'stop detection' of ocean vessels to identify those vessels that slowed their course through an MPA area, possibly indicating illegal fishing in the area. More recently, this work could have helped identify those vessels in the area of the Nordstream pipelines immediately prior to the issues with the pipelines.

The second scenario involved hurricanes and Saildrones. Saildrones for the National Oceanic and Atmospheric Administration (NOAA) Hurricane Monitoring programs for 2021, specifically data for Hurricane Sam and for 2022, data for Hurricane Fiona. The use cases in this scenario relate to the environmental conditions of the ocean surface in the vicinity of the hurricane tract.

Testbed 18 MF task worked to identify anomalies in the ocean surface to indicate the influence of these properties on the trajectory of these hurricanes.

The third scenario extended the Testbed-17 autonomous vehicle use case to address the real-world road network issues of identifying wrong-way drivers and monitoring roadside litter accumulation. These use cases aggregate geotagged video footage from roadside traffic cameras and fleet dashcams respectively. The former correlates moving vehicle observations from multiple locations to track vehicle movements over time so that dangerous drivers can be quickly intercepted. The latter identifies locations of roadside objects from fleet dashcam footage and aggregates these over time to calculate litter accumulation rates so that collection teams can be optimally deployed. Testbed 18 MF task focused on the orientation issues associated with the video capture devices required for such use cases and highlighted that the draft OGC GeoPose Standard can play a key role in aggregating orientation data captured by diverse devices in commercial markets often dominated by proprietary formats.

Additionally, this OGC MF ER explores the architecture for collaborative distributed object detection and analysis of multi-source motion imagery. The ER represents deliverable D020 of the OGC Testbed 18 performed under the OGC Innovation Program.
The additional deliverables under this Testbed 18 Moving Features and Sensor Integration task include the following.

D140 & D141 Moving Features Collection Ingestion Service: This component is a collection of Moving Features deployed for the following two experiments.
To link moving features to a shared collection of features in order to develop a Best Practice for extending an existing Feature dataset with Moving Features data
To serve as an ingestion system that ingests moving feature detections into the Sensor Hub (D142)

D142 Sensor Hub: This component is a software (SW) system that can enable a sensor or system to be discovered, accessed, and controlled through OGC standard services and APIs. The sensor hub receives moving feature detections from components D140 & D141 and provides API-Moving Features to the client (D143).

D143 Client: This component uses AI technology to improve and enhance the moving feature data ingested from D140 & D141 and refined and stored in D142. These deliverables will be discussed in detail in later sections.

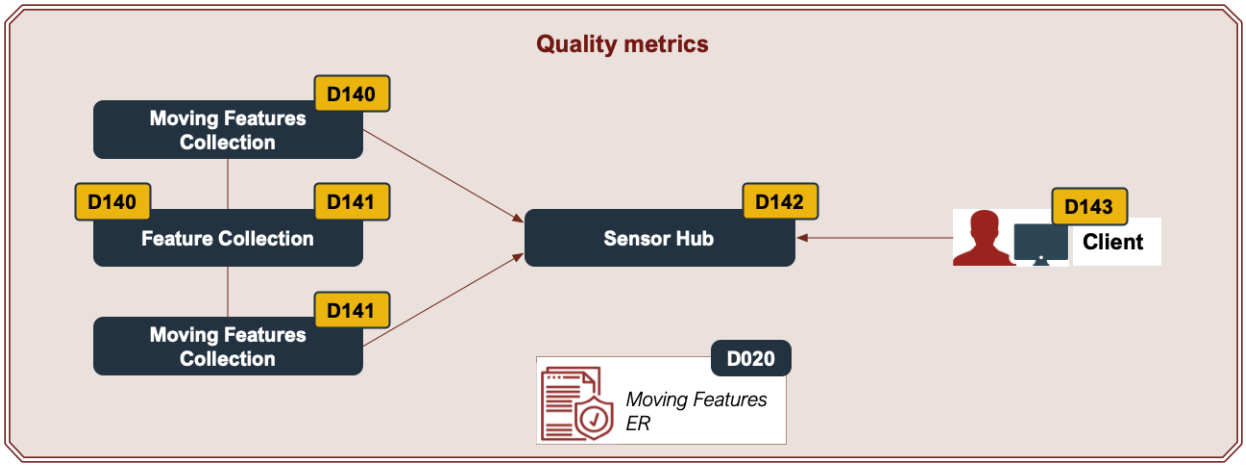


Figure 1 — Testbed 18 Moving Features and Sensor Integration Task Deliverables

5

ABBREVIATIONS & DEFINITIONS

5 ABBREVIATIONS & DEFINITIONS

This section includes abbreviations and definitions needed for this ER.

5.1. Actuator

A device that is used by, or implements, an (Actuation) Procedure that changes the state of the world. Actuator is a subclass of System.

5.2. Application Programming Interface

An Application Programming Interface (API) is a standard set of documented and supported functions and procedures that expose the capabilities or data of an operating system, application or service to other applications (adapted from ISO/IEC TR 13066-2:2016).

5.3. Deployment

Describes the Deployment of one or more Systems for a particular purpose. A Deployment may be done on a Platform.

5.4. Feature Of Interest

The thing whose property is being estimated or calculated in the course of an Observation to arrive at a Result, or whose property is being manipulated by an Actuator, or which is being sampled or transformed in an act of Sampling.

5.5. Observation

Act of carrying out an (Observation) Procedure to estimate or calculate a value of a property of a FeatureOfInterest. Links to a Sensor to describe what made the Observation and how; links

to an Observable Property to describe what the result is an estimate of, and to a Feature Of Interest to detail what that property was associated with.

5.6. Observation Collection (SOSA/SSN extension)

Collection of one or more observations, whose members share a common value for one or more properties.

5.7. OGC APIs

Family of OGC standards developed to make it easy for anyone to provide geospatial data to the web.

5.8. Ontology Design Pattern

Reusable solutions intended to simplify ontology development and support the use of semantic technologies by ontology engineers that document and package good modelling practices for reuse, ideally enabling inexperienced ontologists to construct high-quality ontologies

5.9. Procedure

A workflow, protocol, plan, algorithm, or computational method specifying how to make an Observation, create a Sample, or make a change to the state of the world (via an Actuator). A Procedure is re-usable, and might be involved in many Observations, Samplings, or Actuations. It explains the steps to be carried out to arrive at reproducible Results.

5.10. Platform

A Platform is an entity that hosts other entities, particularly Sensors, Actuators, Samplers, and other Platforms.
In general, a SOSA Platform can host any other Systems, but also other Platforms. We thus model Platform as a particular type of System in the Connected System API.

5.11. Sensor

Device, agent (including humans), or software (simulation) involved in, or implementing, a Procedure. Sensors respond to a Stimulus, e.g., a change in the environment, or Input data composed from the Results of prior Observations, and generate a Result. Sensors can be hosted by Platforms. Sensor is a subclass of System.

5.12. System

System is a unit of abstraction for pieces of infrastructure that implement Procedures. A System may have components, its subsystems, which are other Systems.

Abbreviated terms
ASV Autonomous Surface Vessel
ER Engineering Report
FMSDI Federated Marine Spatial Data Infrastructure
IoT Internet of Things
MPA Maritime Patrol Aircraft
MF Moving Features
MFSI Moving Features and Sensor Integration
OGC Open Geospatial Consortium
OMS Observations, Measurements, and Samples
SSN Semantic Sensor Network
SWE Sensor Web Enablement

6

INGESTION SERVICE: D140 & D141 ARIZONA STATE UNIVERSITY

6 INGESTION SERVICE: D140 & D141 ARIZONA STATE UNIVERSITY

6.1. Introduction

The Moving Feature collection (D140) obtains hurricane tracking data from the National Oceanic and Atmospheric Administration (NOAA) and converts different types of hurricanes and corresponding tracks into Features of Interest (FOI) and Observations associated with moving features and collection of features (that are not moving). The service then posts FOIs and observations to the Sensor Hub where multi-source datasets (e.g, Hurricane tracks from D140, Automatic Identification System (AIS) data (or Vessel traffic data) from D141, etc.) are stored and interpreted as moving features associated with static features.

6.2. Data

The ingestion service obtained information for 1,338 hurricanes from NOAA. As an example, the figure below shows a track for the 2002 Hurricane Lili. For Hurricane Lili, there is a collection of moving features (observations) indicating the movement of the hurricane. Additionally, each observation is associated with static features including the segment ID, occurrence time, wind speed, pressure, and category. With the collected dataset, the service processed them into JSON files. Each type of hurricane information is saved in a file.

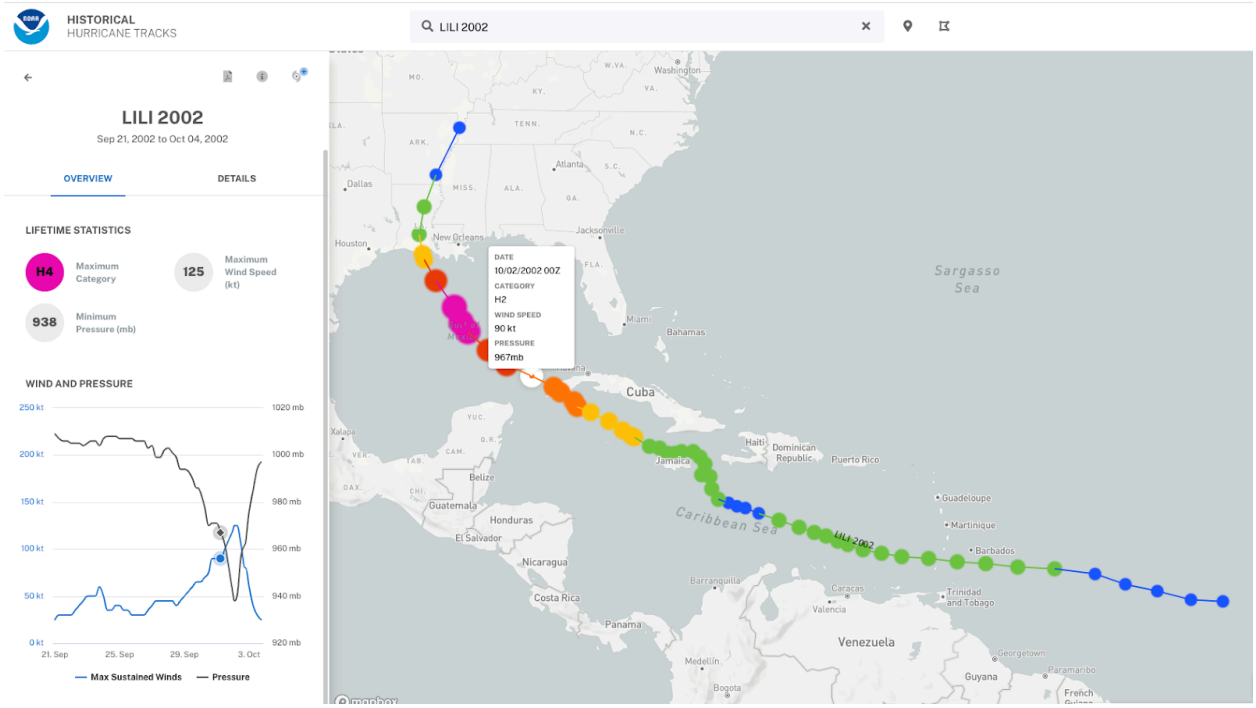


Figure 2 — Example: Hurricane (Lili 2002) track and its static features

6.3. Architecture

Ingestion process architecture

The following figure displays the entire architecture of the ingestion service for the hurricane use case. The D140 ingestion service deliverable is composed of three parts: data loader, integration, and publication.

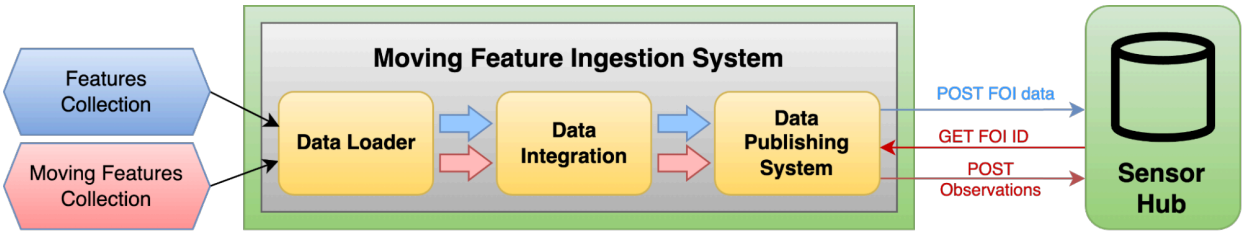


Figure 3 — Ingestion architecture for hurricane use case

The data loader reads the hurricane track raw data. Raw data includes the moving feature collection (hurricane tracks) and the static feature collection (i.e., wind speed, pressure, precipitation, timestamp, etc.). The data loader takes the raw data into the ingestion service so the data can be further processed. Then the ingestion service integrates different types of hurricane information into Features of Interest (FOIs). For each type of hurricane, the corresponding tracks are chronologically integrated according to the “storm ID” attribute for each FOI following the data model standards. At different timestamps, the hurricane occurrence

place has different static characteristics. For example, different wind speeds and pressures at different times. After integration, the ingestion service is ready to publish features and observations into the Sensor Hub via HTTP POST. The Sensor Hub takes data in the format of the output from the integration component of the ingestion service. The publication includes FOI ingestion and Observation ingestion.

In this way, the D143 Client deliverable can obtain the customized information via the Sensor Hub for the integrated visualization use case. The following figure shows the interaction among the ingestion service, Sensor Hub, and Client.

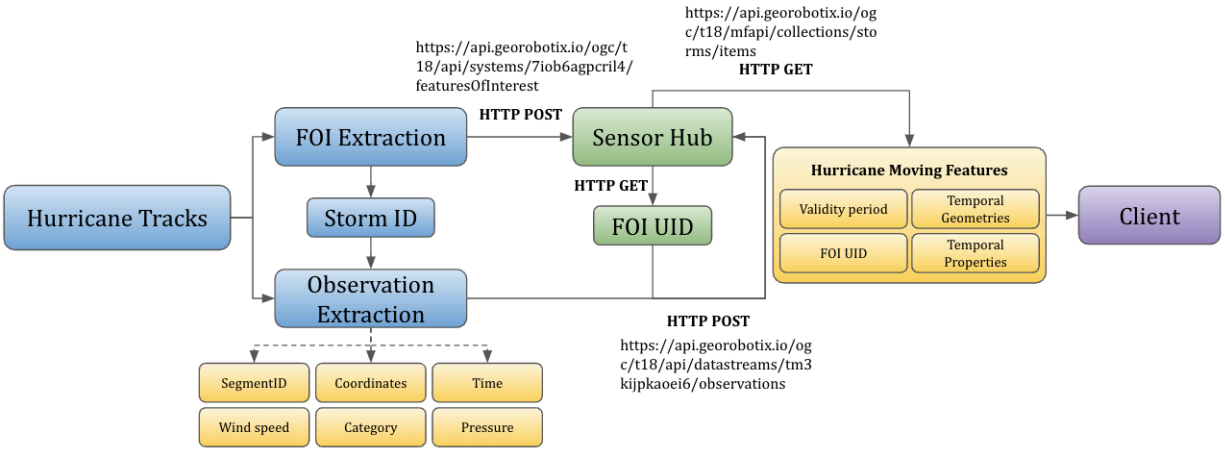


Figure 4 — Interaction with other team components

Data Ingestion

To publish a FOI corresponding to a type of hurricane, the ingestion service sent the HTTP POST with the following payload at the endpoint shown below:

Table 1

endpoint	https://api.georobotix.io/ogc/t18/api/systems/7iob6agpcril4/featuresOfInterest

payload	{
	“type”: “Feature”,
	“properties”: {
	“uid”: “urn:osh:foi:storm:2020228N37286”,
	“name”: “Tropical Storm Kyle”,
	“validTime”: [
	“2020-08-14T12:00:00Z”,
	“2020-08-16T00:00:00Z”
	]
	}
	}

After publishing a FOI, a FOI ID (attribute: id) is created automatically in the Sensor Hub. The ingestion service then sends a HTTP GET request to obtain the generated FOI id through the

following endpoint, so the Observations associated with corresponding FOI will be published correctly.

Table 2

endpoint	https://api.georobotix.io/ogc/t18/mfapi/collections/storms/items

After obtaining the FOI id (attribute: foi@id), the ingestion service integrates corresponding observations to further publish moving features and static features for the FOI. Then, the service sends a HTTP POST request for publication. The endpoint and payload are shown as follows:

Table 3

endpoint	https://api.georobotix.io/ogc/t18/api/datastreams/tm3kijpkaoei6/observations

payload	{
	“foi@id”: “0hh79ki1f29l8”,
	“phenomenonTime”: “2020-08-14T12:00:00Z”,
	“resultTime”: “2020-08-14T12:00:00Z”,
	“result”: {
	“location”: {
	“lat”: 36.6,
	“lon”: -74.2
	},
	“windSpeed”: 35,
	“minPressure”: 1008.0,
	“category”: “TS”
	}
	}

Results

The Sensor Hub automatically integrates each set of ingested hurricane data with the attributes including FOI id, validity period, temporal geometries, and temporal properties. The validity period indicates the occurrence duration for the hurricane with the given FOI id. Temporal geometries represent the moving features of the hurricane chronologically. Temporal properties including wind speed, pressure, and category represent non-moving features as time goes on. As an example, the figure below shows some ingested hurricane information in Sensor Hub. On the interface, each block contains the information for a hurricane. Temporal Geometries and temporal properties are clickable for more details in JSON format.

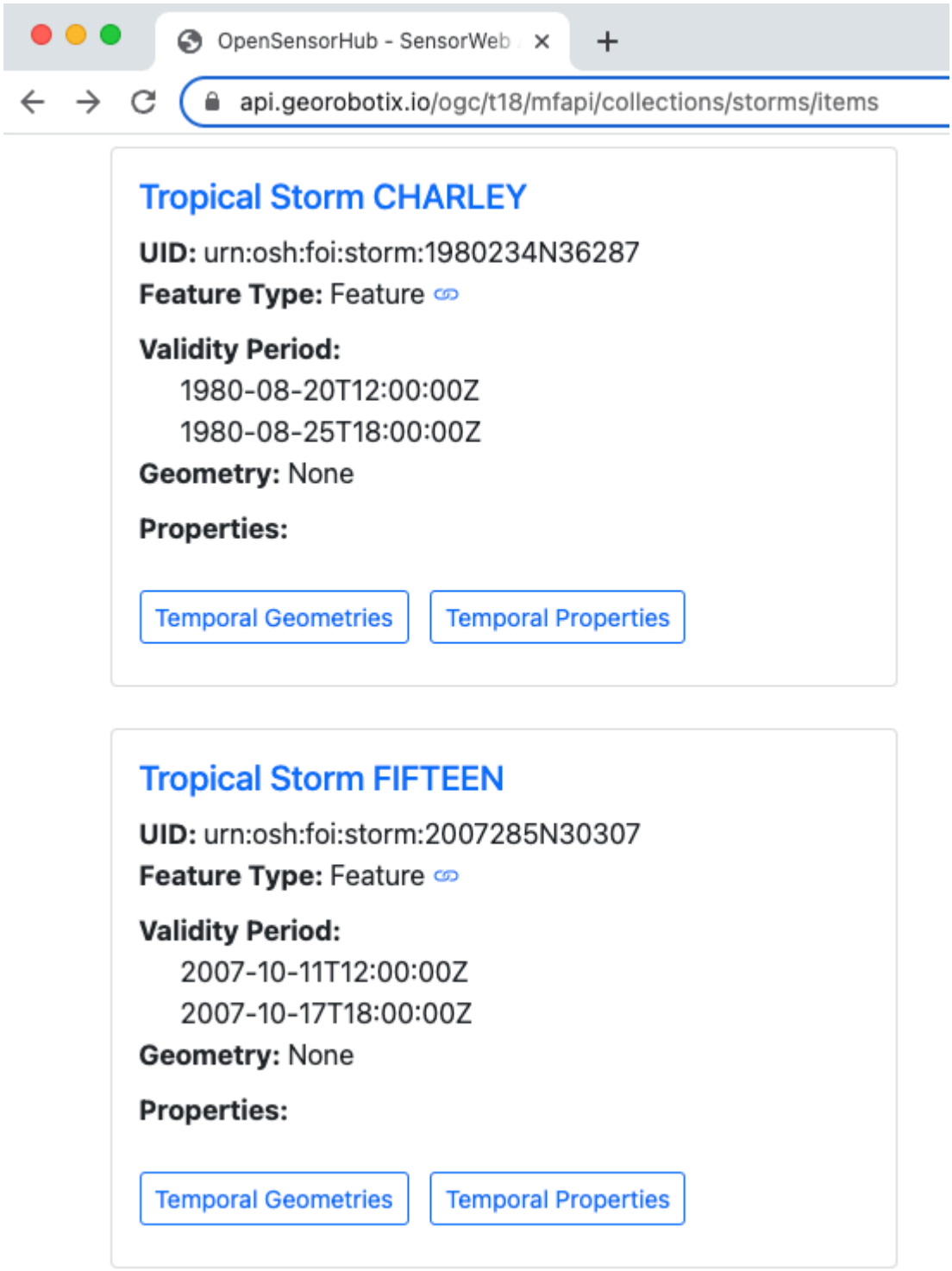


Figure 5 — Example for ingested hurricanes in Sensor Hub

Consider Hurricane Charley as an example. The corresponding Temporal Geometries and Temporal Properties are shown as follows:

Table 4

PROPERTY/GEOMETRY	RESULTS
Temporal Geometries	{ “temporalGeometries”: [{ “id”: “tg-tm3kijpkaoei6”, “type”: “MovingPoint”, “datetimes”: [“1980-08-20T12:00:00Z”, “1980-08-20T18:00:00Z”, “1980-08-21T00:00:00Z”, “1980-08-21T06:00:00Z”, “1980-08-21T12:00:00Z”, “1980-08-21T18:00:00Z”, “1980-08-22T00:00:00Z”, “1980-08-22T06:00:00Z”, “1980-08-22T12:00:00Z”, “1980-08-22T18:00:00Z”, “1980-08-23T00:00:00Z”, “1980-08-23T06:00:00Z”, “1980-08-23T12:00:00Z”, “1980-08-23T18:00:00Z”, “1980-08-24T00:00:00Z”, “1980-08-24T06:00:00Z”, “1980-08-24T12:00:00Z”, “1980-08-24T18:00:00Z”, “1980-08-25T00:00:00Z”, “1980-08-25T06:00:00Z”, “1980-08-25T12:00:00Z”, “1980-08-25T18:00:00Z”], “coordinates”: [[36.0, -73.0], [35.0, -72.0], [34.0, -71.0], [33.4, -69.5], [34.0, -68.0], [34.8, -66.8], [35.8, -65.7], [37.0, -64.8], [38.3, -64.7], [39.1, -64.9], [39.6, -65.9], [38.9, -66.7], [38.2, -66.0], [38.0, -64.7], [38.0, -63.1],

PROPERTY/GEOMETRY	RESULTS
	[37.9, -61.7], [37.9, -60.2], [37.9, -58.2], [38.0, -55.0], [38.1, -51.3], [38.2, -47.0], [38.5, -42.2]], “interpolation”: “Linear” }] }

Temporal Properties	Results {
	“temporalProperties”: [
	{
	“datetimes”: [
	“1980-08-20T12:00:00Z”,
	“1980-08-20T18:00:00Z”,
	“1980-08-21T00:00:00Z”,
	“1980-08-21T06:00:00Z”,
	“1980-08-21T12:00:00Z”,
	“1980-08-21T18:00:00Z”,
	“1980-08-22T00:00:00Z”,
	“1980-08-22T06:00:00Z”,
	“1980-08-22T12:00:00Z”,
	“1980-08-22T18:00:00Z”,
	“1980-08-23T00:00:00Z”,
	“1980-08-23T06:00:00Z”,
	“1980-08-23T12:00:00Z”,
	“1980-08-23T18:00:00Z”,
	“1980-08-24T00:00:00Z”,
	“1980-08-24T06:00:00Z”,
	“1980-08-24T12:00:00Z”,
	“1980-08-24T18:00:00Z”,
	“1980-08-25T00:00:00Z”,
	“1980-08-25T06:00:00Z”,
	“1980-08-25T12:00:00Z”,
	“1980-08-25T18:00:00Z”
	],
	“windSpeed”: {
	“type”: “TDouble”,
	“form”: “http://mmisw.org/ont/cf/parameter/wind_speed”,
	“description”: “Maximum sustained winds”,
	“interpolation”: “Linear”,

PROPERTY/GEOMETRY	RESULTS

	“values”: [25.0, 25.0, 30.0, 30.0, 35.0, 40.0, 45.0, 50.0, 55.0, 60.0, 65.0, 70.0, 70.0, 70.0, 65.0, 60.0, 55.0, 45.0, 35.0, 35.0, 35.0, 35.0]
	},
	“minPressure”: {
	“type”: “TDouble”,
	“form”: “http://mmisw.org/ont/cf/parameter/air_pressure”,
	“interpolation”: “Linear”,
	“values”: [1010.0, 1008.0, 1006.0, 1005.0, 1003.0, 1002.0, 1000.0, 998.0, 996.0, 994.0, 992.0, 990.0, 989.0, 990.0, 991.0, 994.0, 998.0, 1000.0, 1001.0, 1002.0, 1004.0, 1005.0]
	},
	“category”: {
	“type”: “TText”,
	“form”: “http://sensorml.com/ont/T18/concepts/StormCategory”,
	“interpolation”: “Linear”,
	“values”: [“TD”, “TD”, “TD”, “TD”, “TS”, “TS”, “TS”, “TS”, “TS”, “TS”, “H1”, “H1”, “H1”, “H1”, “H1”, “TS”, “TS”, “TS”, “TS”, “TS”, “TS”, “TS”]
	}
	}
	]
	}

7

INGESTION SERVICE: D140 & D141 PELAGIS

7 INGESTION SERVICE: D140 & D141 PELAGIS

7.1. Scope of Work

The Pelagis ingestion service deliverable focused on creating sets of feature collections to be used to model the interaction of features in motion with features at rest. The feature collections were derived from the marine domain and related human activity.

Observation data for vessel traffic and Saildrones were sourced, processed, and published to the D141 OpenSensorHub (OSH). The feature model aligns with the OGC Semantic Sensor Network (SSN) ontology identifying systems of sensors, datastreams of observations, and features of interest.

7.1.1. Project Outputs

1. An ingestion service designed to publish streams of observations to the D142 Sensor Hub.
2. An information model mapping the observation systems to the OGC SSN ontology identifying compliance and gaps when considering the effect of moving features on these observation systems.

7.2. Introduction

Pelagis was tasked with developing an ingestion service that publishes observation data from external sources into the D142 SensorHub. For the scenarios addressed within this project, the focus was on the following two main challenges.

- First, the harmonization of sensor integration frameworks across OGC while aligning with work developed through external standards bodies.
- Second, the maturation of the Moving Features architecture and its integration with a harmonized OGC sensor architecture.

7.2.1. Harmonization of Sensor Integration Frameworks

OGC maintains several different standards and specifications related to real-world observation systems. The primary work associated with this project centers on two standards:

* the OGC Sensor Web Enablement (SWE) Standards with focus on the joint OGC Semantic Sensor Network (SSN) and W3C SOSA standard as well as the W3C Semantic Sensor Web Ontology (SOSA) Recommendation; and
* the OGC Observations, Measurements, and Samples (OMS v3) draft standard.

SSN and OMS are complementary viewpoints (more information is provided on OMS below). SSN is 'provider-centric' and encodes details of the sensing system along with raw observation data. SSN is self-contained and focuses on the set of requirements for data producers. SSN provides extensive support for serialization of numeric data arrays and is particularly optimized for data that includes multiple parallel streams that must be processed together. For example, the data collected by cameras on airborne vehicles must be geo-referenced based on the instantaneous position of the platform and orientation of the camera.

7.2.1.1. OGC Semantic Sensor Network

The D142 SensorHub platform closely models the ontology of the OGC Semantic Sensor Network. The SSN ontology is built around a central Ontology Design Pattern (ODP) describing the relationships between sensors, stimulus, and observations.

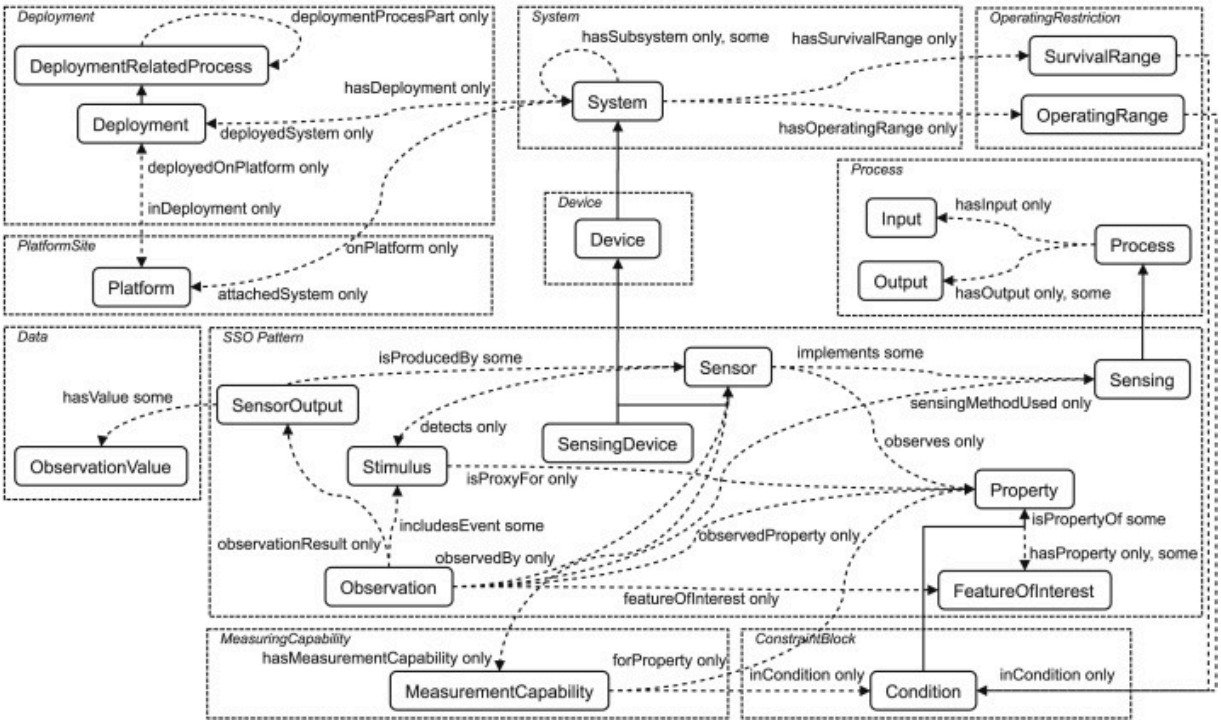


Figure 6 — OGC Semantic Sensor Network Ontology

The ontology can be considered from four main perspectives:

- The *Sensor* — A perspective with a focus on what a sensor senses and how;
- The *Observation* — A perspective with a focus on observations and measurements and related metadata;

- The *System* — A perspective with a focus on systems of sensors and deployments; and,
- The *Feature* — A perspective focusing on the platform hosting the sensing system or the real-world entity being observed. In this context, it is important to differentiate the former as a *Sampling Feature* and the latter as a *Feature of Interest* or *Sampled Feature*. This is required to bridge the concepts defined by the SSN ontology to the OGC OMS model.

OMS is designed to be more 'user-centric' with the target of the observation and the observed property as first-class objects. OMS works at a higher semantic level than SSN and abstracts the concepts of a sensing system, expecting the details to be provided by specific applications and domains. OMS also provides a model for sampling since almost all scientific observations are made on a subset of, or proxy for, the ultimate feature of interest.

For context, the assertion used within this project activity positions the SSN Standard as the foundation for modeling the *physical* characteristics of an observation system inclusive of the raw observation data while OMS models the *use* of the observation system to measure the observable properties of real-world features.

A main focus of this project is to identify the opportunity to leverage the Moving Features specification when applied to observation systems managed through the OGC SWE and OMS standards. Of particular interest is whether the Moving Features MF_JSON encoding scheme may be used to represent observation systems in which the sampling features (Observers) move and/or the features of interest are in motion.

7.3. Design Patterns

The main concepts applied within the scope of this project are as follows.

From the perspective of the OGC Semantic Sensor Network, a *System* represents a sensing platform producing a stream of *Measurements* against a specific *Phenomenon*. The *Measurements* are organized as *Observations* related to the *Observable Properties* of a *Feature of Interest*. Collections of *Observations* are organized as *Datastreams* over an inclusive extent of time and space. Of note is that from the perspective of the sensing system, the host of the sensing system — the *Sampling Feature* — often fulfills the role of the *Feature of Interest*.

From the perspective of the OGC OMS, the main concepts applied within the scope for this project are: — a *SamplingFeature* is an abstract *Feature* responsible for sensing and managing the *Observations* related to a *Feature of Interest*. A *SamplingFeature* is associated with a *MonitoringFacility* representing the real-world feature hosting the *SamplingFeature* and provides the spatial reference for the set of *Observations* to the *Observing System*.

The main difference between the OGC SSN model and the OGC OMS model is the perspective of the use of the observation system. The OGC SSN Standard models the observation system as a *provider* whereas the OGC OMS model views the observation system from the standpoint of a *consumer*. This is an important distinction to ensure that the two models align, as opposed to compete, for the management of common entities.

7.4. Application model

The main scenarios for this project are represented as follows.

7.4.1. Ocean Vessel Traffic

An AIS monitoring network is responsible for the identification and positioning of ocean vessels and *aids to navigation* such as shore stations and moored buoys. The AIS Base Station is the primary component in an AIS shore station and is responsible for the transceiving of AIS data from all AIS sources within its area of concern.

Applied to this project, one *System* record was defined to represent the aggregated AIS data sourced for an area of interest. This *System* is an abstraction of the complex sensor system used to implement an AIS monitoring network. Each vessel is modelled as a *Feature of Interest* with its corresponding AIS positioning records modelled as a *Datastream* of *Observations* where the *Datastream* is specific to the cumulative daily geopositioning reports.

This scenario models the use of a stationary feature (the AIS base station) to observe a set of *moving features* within its coverage area.

7.4.2. Saildrone USV Monitoring

A Saildrone is an Autonomous Surface Vessel (ASV) equipped with various sensor systems designed to monitor phenomenon related to the marine environment.

NOAA commissioned Saildrone ASVs to monitor the sea surface conditions of the North Atlantic basin. In 2021 and 2022, NOAA created missions to monitor the ocean environment during Hurricane Sam (2021) and Hurricane Fiona (2022). Each of these missions included a number of Saildrone ASVs hosting specific sensing equipment measuring wind speeds, barometric pressure, water temperature, and salinity. The information collected by each mission is made available through the NOAA/PMEL open data portal.

The mission data for 2021 and 2022 is produced as NetCDF files. For each of these files, the format is transformed from NetCDF to the OSH SSN physical model and published to the D142 SensorHub.

Each Saildrone attached to the NOAA mission is modelled as a *System* with associated properties to identify the ASV's role, mission, valid time period, and spatial coverage. For each Saildrone, the set of sensing platforms provides environmental observations related to the water column immediately below the ocean surface and the air column immediately above the ocean surface.

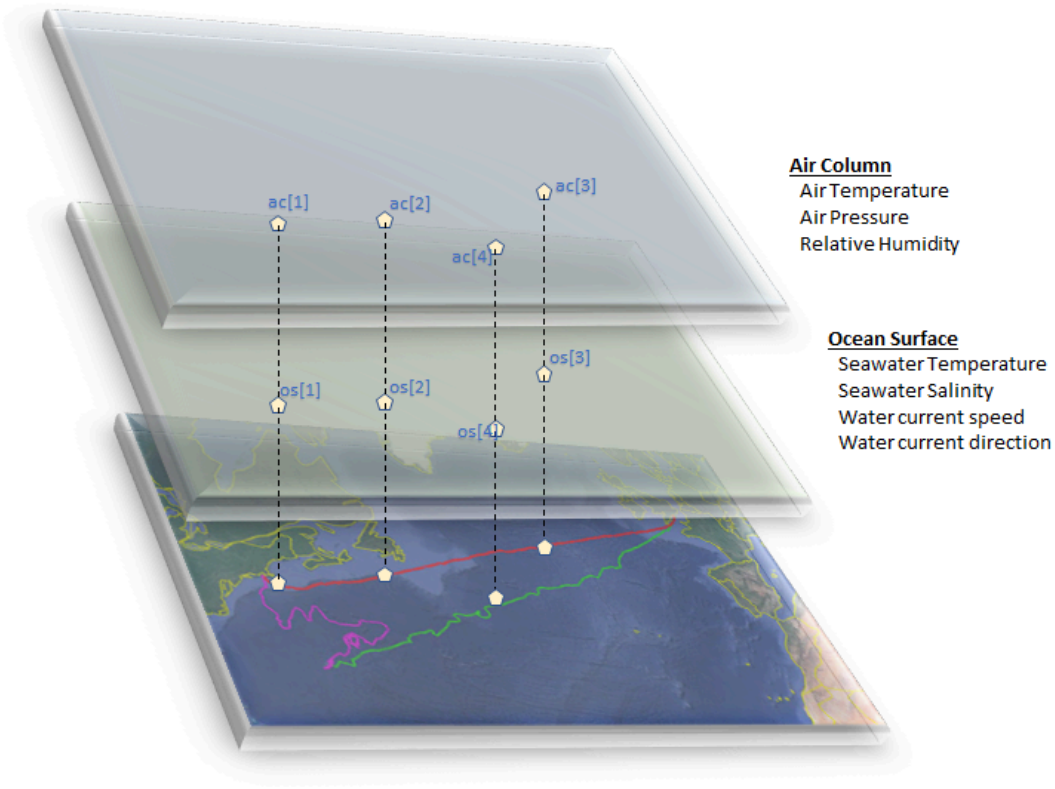


Figure 7 — Saildrone Discrete Point Coverage

Each sensing platform records measurements against its respective phenomenon with a nominal sampling schedule. Geopositioning measurements are collected across a *continuous* curve while environmental sampling uses a *discrete* sampling schedule. Of note is that each sensing platform has its own sampling schedule independent of the other sensing platforms.

From the perspective of Mission Control, the *Feature of Interest* is the Saildrone feature itself. The environmental sensing platforms represent *Sampling Features* against the *Phenomenon* of the *Sampled Feature* — the North Atlantic Basin — which also represents the *Ultimate Feature of Interest* for this project.

This scenario models the use of *moving features* as observers of a stationary feature of interest with a wide extent and time period. The observations associated with each Saildrone mission represent a *Discrete Point Coverage* from the ocean's surface domain. This point coverage of measurements is inclusive of sea surface temperature, salinity, current speed, wave height, and

direction as well as wind speed and direction for the air column immediately above the ocean's surface.

7.4.3. North Atlantic Ocean Basin

The *ultimate feature of interest* for this Testbed-18 project is the North Atlantic Ocean. Specifically, the participants were interested in the ingestion of ocean observations related to the ocean surface with the goal of correlating these observations to weather events (*Hurricanes*) and human activities (*Vessel Traffic*).

7.5. Technical Architecture

The following workflows support the D141 Ingestion Service for AIS vessel traffic and Saildrone ocean observations.

7.5.1. AIS Vessel Traffic

Vessel traffic is ingested through open source repositories as observations of ocean vessel locations and travel activities. The source of the observations is provided through the AIS messaging system in which an ocean vessel reports its location, speed, course, and heading. These observations are processed to provide insight into the general behavior of ocean vessels within an area of interest.

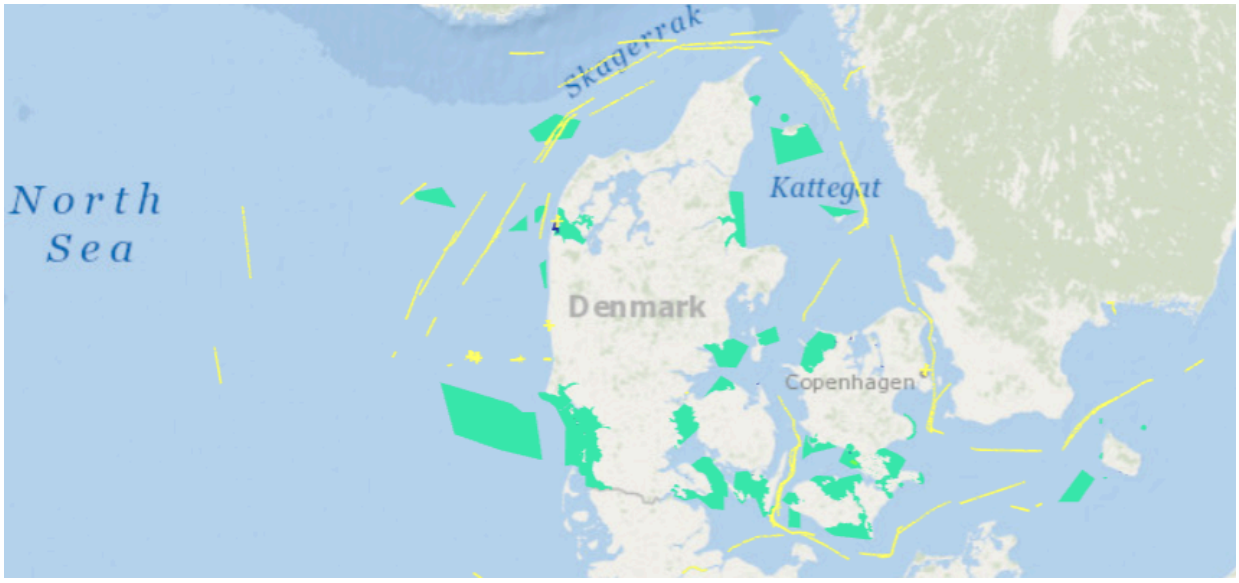


Figure 8 — Denmark Maritime Authority Vessel Traffic

Vessel traffic is modelled as an observation system. One *System_* entity is created representing the *Responsible Party*, in this case the Denmark Maritime Authority, providing the set of observations within the area of concern for this project. The system *has many* datastreams

of observations with each representing a specific *time period* (daily) of observations. Each observation provides the geolocation, speed, course, and heading of the vessel at the reported time. Each observation is directly related to *one* ocean vessel modelled as a *feature of interest*.

7.5.1.1. Workflow

This component reads from a collection of AIS messages made available through the Denmark Open Data Portal. To minimize the amount of data for the initial use case, only vessel traffic for May 14, 2022 is used.

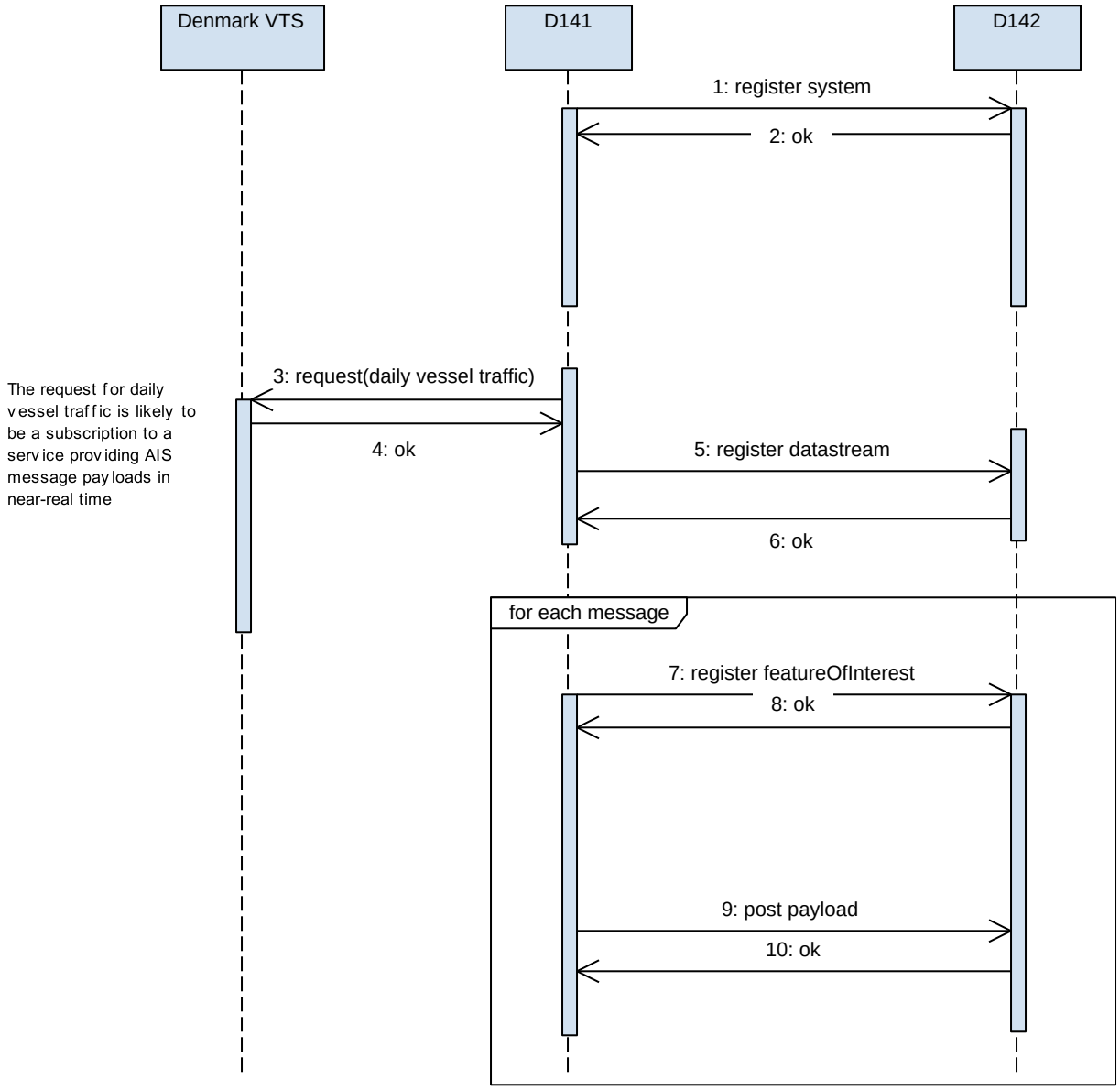


Figure 9 — AIS Ingestion Workflow

1. The component registers a new 'system' with the D142 SensorHub for AIS messages specific to the area of interest. For example, the AIS tracking service is registered as a *System* labelled "Denmark AIS Monitoring Network"

2. The component then registers a new 'datastream' over which AIS message payloads will be published to the D142 SensorHub. There is one datastream per daily reporting interval.

3. The AIS datafile is pre-processed to determine the identity of each ocean vessel navigating within the area of interest for one 24-hour period. For each unique instance of an ocean vessel, a *Feature of Interest* is created identifying the static properties of the ocean vessel including its MMSI, IMO number, and Call Sign.

4. The component then iterates over the AIS message datafile and for each message posts the message payload to the D142 SensorHub. This payload identifies the temporal properties of the ocean vessel including its reported location, heading, course, and speed.

7.5.2. Saildrone Ocean Observations

A Saildrone is modelled as a *Platform* hosting a *System* of *Sensors*. Each Saildrone has a *Deployment* specific to a mission sponsored through NOAA for ocean observations. For example, Saildrone 1021 was deployed to the Atlantic Ocean between May 25, 2019 and October 22, 2019 with a mission to collect ocean observation data over a crossing from Bermuda to Lymington, UK with return to Newport, RI.

Mission data for the 2021 Hurricane season is sourced from the NOAA/PMEL open data portal. Saildrone observation data is provided in raw NetCDF format and processed into the D142 SensorHub using the SSN ontology.

7.5.2.1. Workflow

Saildrone mission data specific to the NOAA monitoring program for the 2021 Hurricane Season is used for this project. Each Saildrone's observation data is provided as a NetCDF file which is transformed and loaded into the D142 OpenSensorHub portal.

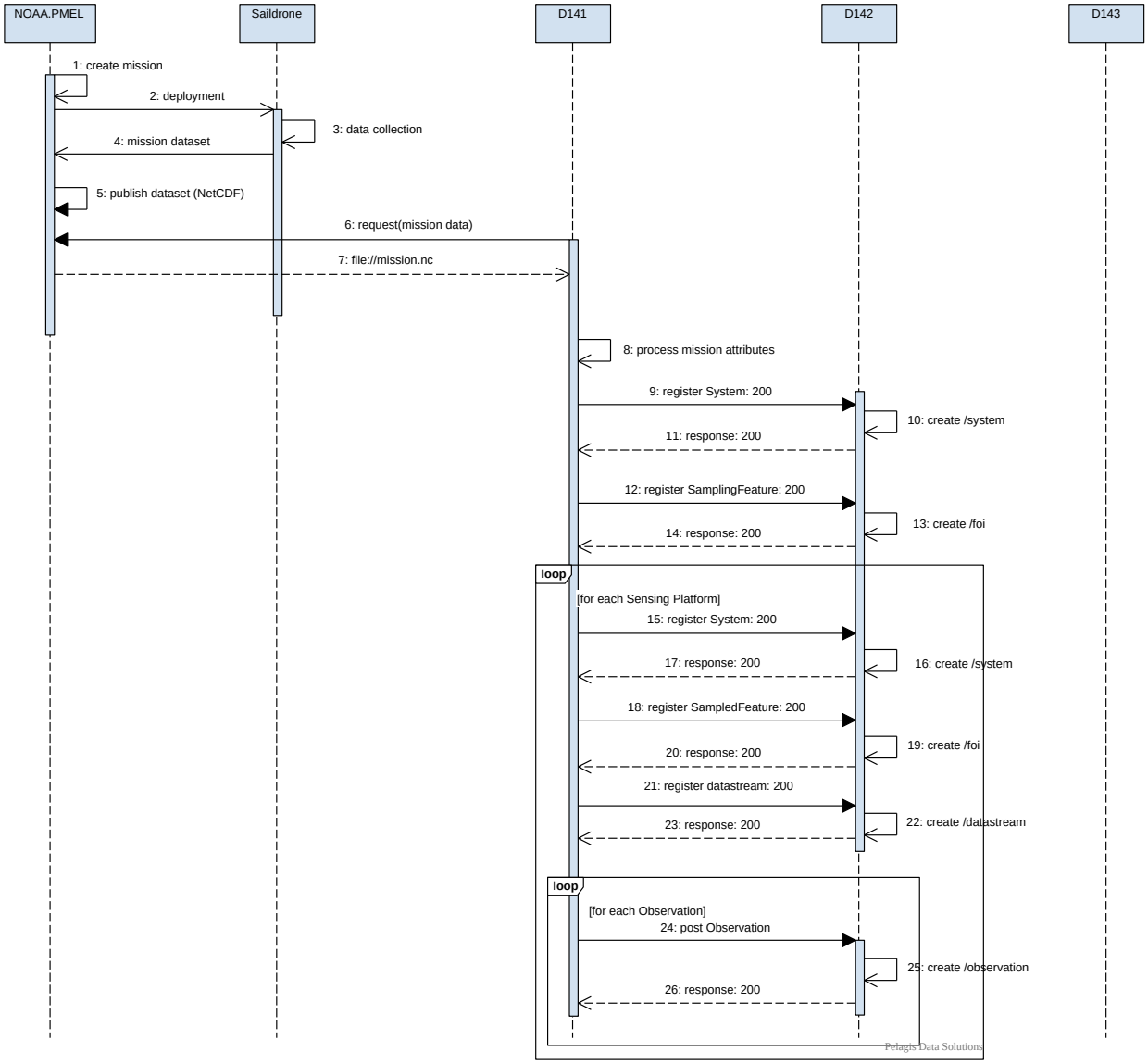


Figure 10 — Saildrone Ingestion Workflow

1. The component registers a new *System* with the D142 SensorHub for each Saildrone.

2. The component registers a *Datastream* for the Saildrone representing the collection of observations provided over the range of its mission.

3. In order to manage the dependency between the *Datastream* and the *Feature of Interest* relationship, the Saildrone is also registered as a *Feature of Interest*. This model overloads the meaning of the *Feature of Interest* to be a *SamplingFeature* (Observer).

4. The component then iterates over the NetCDF datafile and for each message payload posts the observation payload to the D142 SensorHub as an *Observation*.

7.6. Comments, Issues and Recommendations

When viewing an observation system from the standpoint of a *Moving Feature*, care must be made to properly model the datastream of observations in the context of (a) a sampling feature that is moving, and (b) a feature of interest that is moving.

7.6.1. Moving Sampling Features

A sampling feature such as the Saildrone platform creates observations along its trajectory. This series of observations is represented as a *Sampling Curve* and relate to the coverage area representing the immediate vicinity around the ocean surface over which the Saildrone passes. When considering the North Atlantic Ocean Basin as the 'ultimate' feature of interest, the set of observations are relevant only in the context of when **and where** the observations were made.

The OGC SSN model infers a dependency between the *Sampling Feature* (Saildrone) and the *Feature of Interest* (Ocean Basin) through the Datastream entity. The Datastream maintains the Schema for the Observations and defines the observed properties for each Feature of Interest. The issue revolves around the dependency that the Feature of Interest being observed **must have** all observable properties defined by the Datastream schema. In the case of the Saildrone, in order to properly model the sampling curve, the [latitude, longitude] of each observation must be referenced as part of the observable schema for each Datastream of Observations produced. And yet, the Ocean Basin feature does not have [lat,lon] of the Saildrone trajectory as one of its observable properties.

The second issue related to 'moving' sampling features is the *cadence* of the set of observations. The Saildrone platform has more than one sensing platform. Each sensing platform has a cadence of observations defined by its sampling rate. For example, sea surface temperature may be observed over a 30s interval with 90s elapsed between observations. Air temperature, however, is sensed through a separate device hosted on the Saildrone and is monitored at a difference cadence — 60s on, 240s off, centered at :00. As a result, the SSN datastream model separates each sensing platform based on the nominal sampling schedule of the platform.

Related to the issue of nominal sampling schedules is that the SSN model does not provide a means to identify the specific sampling curve used to sense a phenomenon. In the case of sea surface temperature with a cadence of 12s on, 588s off, centered at :00 — the result observation set is effectively modeled as a time series of observations — one observation every 10 minutes with a time instant result time. Modelling the observation with a time instant value rather than a time period infers that a client application may use the time instant value to locate the position of the observation. This would be an incorrect inference. The observation is actually made through a 12s sensing period which means that the result observation is made over a 12s sampling curve overlapping the Saildrone trajectory. To properly identify the sampling curve, client applications should use the time-start and sampling period to determine the start and stop times of the sampling observation. This time interval can then be used to map against the sampling features trajectory to produce that section of curve representing the sampling area for each observation.

7.6.2. Moving Features of Interest

Moving features are modelled within the SSN ontology as features of interest. In the case of AIS vessel traffic, each ocean vessel is treated as a feature of interest with the reported [lat,lon] and temporal properties for the course, heading, speed, and rate of turn. With regards to the OGC SSN model, the challenge is that the feature of interest — the ocean vessel — is unknown until its first reported message. This requires the vessel to be registered as a feature of interest on its first report prior to recording the AIS message as an observation. If each ocean vessel was to be registered as a SSN:*System*, then the full SSN schema would need to be created on first report. The approach taken to mitigate the cost of instantiating a new system for each ocean vessel is to treat the AIS Base Station as the observing system. Effectively, the base station becomes a stationary observation system (Sampling Point) for its coverage area. As a new ocean vessel is observed within the station's area of interest, the vessel is registered as a feature of interest and each AIS message stored as an observation.

Similarly, Saildrones are managed as a feature of interest with a temporal geometry representing the location of the platform. However, Saildrones do not monitor any temporal properties related to the Saildrone itself. This is managed within SSN as simply a Datastream of positional observations with a simplified schema for its [lat,lon]. As identified in the previous section, the observed properties provided by the Saildrone are related to the North Atlantic Ocean basin. This results in a collection of datastreams where the Saildrone trajectory is stored in the geopositioning datastream for the Saildrone while observed properties are managed as temporal properties stored in separate datastreams. In this context, client applications must have prior knowledge of the use of the SSN *System* object (Saildrone instance) to tie together the set of disparate environmental observation datastreams.

7.6.2.1. Recommendations

In most observations systems, it is assumed that an observation is made at a single point in time and space. Much research has focused on Time Series analysis at a scale based on this observation model.

The previous use cases highlight the need to tightly model both time intervals and space-curves as part of the observation model. As well, the OGC SSN ontology should be revisited to accommodate the concepts of observation systems that use 'moving' sampling features to observe their environment without necessarily requiring that the sampling feature be registered as a feature of interest. The OGC OMS standard does highlight the concept of a *sampling curve* as a specialized type of *Observer* but further research is required to leverage the MovingFeatures standard as a possible encoding scheme for OMS Observers 'that move.'

8

SENSOR HUB: D142 BOTTS INC

8 SENSOR HUB: D142 BOTTS INC

This section provides information on the Sensor Hub designed by Botts Innovative Research.

8.1. Introduction & Architecture

As shown on the following figure, the architecture developed in Testbed-18 consists of the following parts.

- Data Ingesters (left) are responsible for ingesting “raw” observation data into the Sensor Hub.
- The Sensor Hub (center) is an aggregation layer that collects data from multiple sources, harmonizes it to a common data model, and stores it in persistent storage. The Sensor Hub is then capable of transforming that data (usually on-the-fly) to various output formats.
- Client applications (right) consume data from the Sensor Hub and allow users to browse and display that data in various visualization contexts. In the context of Testbed-18, a particular focus was made on transforming individual Observations and the Features of Interest they relate to into Moving Features representations.

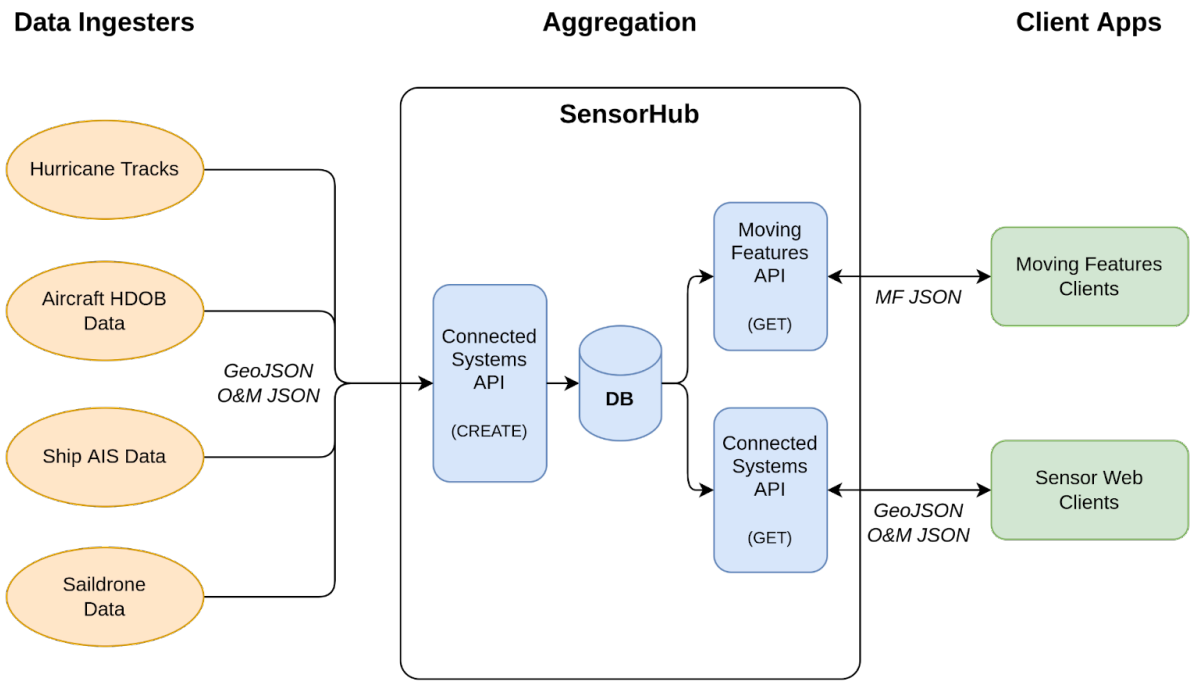


Figure 11 — Overall Architecture Diagram

8.1.1. Data Ingesters

Several data ingesters were developed during the project to collect data from various data sources.

- Hurricane Tracks obtained from NOAA Office for Coastal Management. This data includes the hurricane center location, as well as wind speed, air pressure, and storm category.
- High Density Observations Bulletin (HDOB) collected by aircrafts and obtained from NOAA Hurricane Center. This data includes aircraft location, as well as geopotential height, air and surface pressure, air temperature, dew point temperature, average and peak wind speeds, and wind direction.
- Automatic Identification System (AIS) data received from ship transponders and used for maritime traffic monitoring. This data was obtained from the NOAA Office for Coastal Management via MarineCadastre.gov and Danish coastal authorities. This data includes vessel location, speed over ground, course over ground, heading, and status code.
- Autonomous Surface Vehicle (ASV or USV) data obtained from Saildrone. This data includes vessel location, speed over ground, course over ground, heading, pitch, roll, and ~20 atmospheric and oceanic variables.

Each data ingester is responsible for pushing feature of interest and observation data to the Sensor Hub in GeoJSON and O&M JSON formats, respectively, following the protocol defined in the draft Connected Systems API (see the next section for details).

8.1.2. Sensor Hub (Aggregation Layer)

The Sensor Hub receives features of interest and related observations from all data ingesters, saves them in persistent storage, and makes all ingested data available with different protocols and formats.

8.1.2.1. Data Ingestion

On the ingestion side, the Sensor Hub implements the draft Connected Systems API. This API accepts feature descriptions in GeoJSON format and observations in the O&M-JSON format.

For example, to create the feature of interest corresponding to a storm, the data ingester issues an HTTP POST request with the following GeoJSON payload:

```
{
  "type": "Feature",
  "properties": {
    "uid": "urn:x-wmo:storm:2020228N37286",
    "name": "Tropical Storm Kyle",
    "validTime": [
      "2020-08-14T12:00:00Z",
      "2020-08-16T00:00:00Z"
```

```
    ]
  }
}
```

Below is another example of a ship feature that carries additional static properties:

```
{
  "type": "Feature",
  "properties": {
    "uid": "urn:mrn:x-vessel:338531000",
    "name": "GENESIS VIGILANT",
    "featureType": "http://www.opengis.net/def/featureType/x-T18/Vessel",
    "validTime": [
      "1981-05-22T09:00:00Z",
      "now"
    ],
    "mmsi": "338531000",
    "imo": "IMO8973928",
    "callSign": "WDG9352",
    "vesselType": "Towing",
    "length": 30,
    "width": 10,
    "draft": 5
  }
}
```

If the ingestion is successful, the server responds with the ID assigned to the new feature of interest (e.g., “0hh79ki1f29l8”). In order to upload an observation related to this feature of interest, the data ingester issues another POST request with the following O&M JSON payload:

```
{
  "foi@id": "0hh79ki1f29l8",
  "phenomenonTime": "2020-08-14T12:00:00Z",
  "resultTime": "2020-08-14T12:00:00Z",
  "result": {
    "location": {
      "lat": 36.6,
      "lon": -74.2
    },
    "windSpeed": 35,
    "minPressure": 1008.0,
    "category": "TS"
  }
}
```

The observation result must be provided according to a schema defined at the Datastream level. The schema corresponding to the observation listed above is provided below:

```
{
  "datastream@id": "tm3kijpkaoei6",
  "obsFormat": "application/om+json",
  "resultSchema": {
    "type": "DataRecord",
    "label": "Storm Observations",
    "fields": [
      {
        "name": "location",
        "type": "Vector",
        "definition": "http://sensorml.com/ont/swe/property/Location",
        "referenceFrame": "http://www.opengis.net/def/crs/EPSG/0/4326",
        "coordinates": [
          {
            "name": "lat",
```

```
            "type": "Quantity",
            "definition": "http://sensorml.com/ont/swe/property/
GeodeticLatitude",
            "axisID": "Lat",
            "label": "Geodetic Latitude",
            "uom": {
              "code": "deg"
            }
          },
          {
            "name": "lon",
            "type": "Quantity",
            "definition": "http://sensorml.com/ont/swe/property/Longitude",
            "axisID": "Lon",
            "label": "Longitude",
            "uom": {
              "code": "deg"
            }
          }
        ]
      },
      {
        "name": "windSpeed",
        "type": "Quantity",
        "definition": "http://mmisw.org/ont/cf/parameter/wind_speed",
        "label": "Maximum Wind Speed",
        "description": "Maximum sustained winds",
        "uom": {
          "code": "[kn_i]"
        }
      },
      {
        "name": "minPressure",
        "type": "Quantity",
        "definition": "http://mmisw.org/ont/cf/parameter/air_pressure",
        "label": "Minimum Pressure",
        "uom": {
          "code": "mbar"
        }
      },
      {
        "name": "category",
        "type": "Category",
        "definition": "http://sensorml.com/ont/T18/concepts/StormCategory",
        "label": "Storm Category",
        "codeSpace": {
          "href": "http://sensorml.com/ont/T18/concepts/SaffirSimpsonScale"
        },
        "constraint": {
          "type": "AllowedTokens",
          "values": ["TD","TS","H1","H2","H3","H4","H5"]
        }
      }
    ]
  }
}
```

8.1.2.2. Data Delivery

On the consumer side, a client can either use the Connected Systems API to browse through features of interest and observations, or view the same data in MF-JSON format via the draft "OGC API — Moving Features" (MF API) interface.

All Features of Interest registered on the Sensor Hub are also exposed as Moving Features and all their static properties are carried over in the process. In addition, all observations related to a given feature of interest are exposed as either "temporal geometries" or "temporal properties" through the MF API interface. The Sensor Hub is responsible for carrying over not only data values but also semantics from the O&M model to the Moving Features model.

This transformation shows that Moving Features are just another view/representation of the same underlying observation data. While in the O&M model, features of interest and observation series are provided as completely separate resources. The Moving Features view allows packaging values of variable feature properties (aka observations) along with the feature description.

For example, to retrieve all features in the 'vessels' collection, the client issues a GET request on the following URL: $root/collections/vessels/items?f=application%2Fjson+

The server returns the list of features in the collection and looks like this (abridged for brevity):

```
{
  "items": [
    {
      "type": "Feature",
      "id": "p96kqua33r1n2",
      "properties": {
        "uid": "urn:osh:foi:vessel:367533290",
        "featureType": "http://www.opengis.net/def/featureType/x-T18/Vessel",
        "name": "MARIE CHERAMIE",
        "description": "Proxy feature for vessel 367533290",
        "mmsi": "367533290",
        "imo": "IMO8964862",
        "callSign": "WDG4131",
        "vesselType": "Other",
        "length": 49.0,
        "width": 13.0,
        "draft": 3.1
      }
    },
    ...
  ]
}
```

Accessing a single Moving Feature by ID allows the user to discover the links to access temporal variables, as shown in the example below (abridged for brevity):

```
{
  "type": "Feature",
  "id": "p96kqua33r1n2",
  "properties": {
    ... same as above ...
  },
  "links": [
```

```
    {
      "rel": "temporalGeometries",
      "title": "Temporal Geometries",
      "href": "$root/collections/vessels/items/p96kqua33r1n2/tgeometries"
    },
    {
      "rel": "temporalProperties",
      "title": "Temporal Properties",
      "href": "$root/collections/vessels/items/p96kqua33r1n2/tproperties"
    }
  ]
}
```

The client can then access time-dependent properties of the feature (i.e., temporal geometries and temporal properties). Temporal geometries can be fetched by connecting to the following API path:

$root/collections/vessels/items/p96kqua33r1n2/tgeometries?limit=3

which generates the following response:

```
{
  "temporalGeometries": [
    {
      "id": "tg-kuhmds0ib5gd8",
      "type": "MovingPoint",
      "datetimes": [
        "2020-01-01T00:00:07Z",
        "2020-01-01T00:01:08Z",
        "2020-01-01T00:02:18Z",
        ...
      ],
      "coordinates": [
        [ 33.73451, -118.27071 ],
        [ 33.7345, -118.27071 ],
        [ 33.73449, -118.27069 ]
      ],
      "interpolation": "Linear"
    }
  ],
  "links": [
    {
      "rel": "next",
      "href": "$root/collections/vessels/items/p96kqua33r1n2/tgeometries?limit=
3&offset=3"
    }
  ]
}
```

Temporal properties are fetched from the following path:

$root/collections/vessels/items/p96kqua33r1n2/tproperties?limit=3

which generates the following response:

```
{
  "temporalProperties": [
    {
      "datetimes": [
        "2020-01-01T00:00:07Z",
        "2020-01-01T00:01:08Z",
        "2020-01-01T00:02:18Z"
```

```
      ],
      "sog": {
        "type": "TDouble",
        "form": "http://sensorml.com/ont/swe/property/SpeedOverGround",
        "description": "Vessel speed relative to ground",
        "interpolation": "Linear",
        "values": [ 0.0, 0.0, 0.0 ]
      },
      "cog": {
        "type": "TDouble",
        "form": "http://sensorml.com/ont/swe/property/CourseOverGround",
        "description": "Vessel travel direction relative to ground",
        "interpolation": "Linear",
        "values": [ 280.4, 274.5, 260.33 ]
      },
      "heading": {
        "type": "TDouble",
        "form": "http://sensorml.com/ont/swe/property/TrueHeading",
        "description": "Vessel heading direction relative to true north,
measured clockwise",
        "interpolation": "Linear",
        "values": [ 511.0, 511.0, 511.03 ]
      },
      "status": {
        "type": "TText",
        "form": "http://sensorml.com/ont/x-ais/property/NavigationStatus",
        "description": "Vessel Status",
        "interpolation": "Linear",
        "values": [ "0", "0", "0" ]
      }
    }
  ],
  "links": [
    {
      "rel": "next",
      "href": "$root/collections/vessels/items/p96kqua33r1n2/tproperties?limit=
3&offset=3"
    }
  ]
}
```


Although not implemented during Testbed-18, it would also be possible to embed the “temporal geometries” and “temporal properties” directly inside the Moving Feature resource without having to dereference the links (this is allowed by the MF-JSON format).

8.1.3. Client Applications

Two of the D143 Client deliverable applications consume data from the Sensor Hub using the Moving Features API. The Clients issue GET requests to retrieve feature collections in GeoJSON format.

Features and Moving Features can also have more static properties.

8.1.4. Sequence Diagram

The following UML sequence diagram provides more details about the interaction between the different components described in the previous sections.

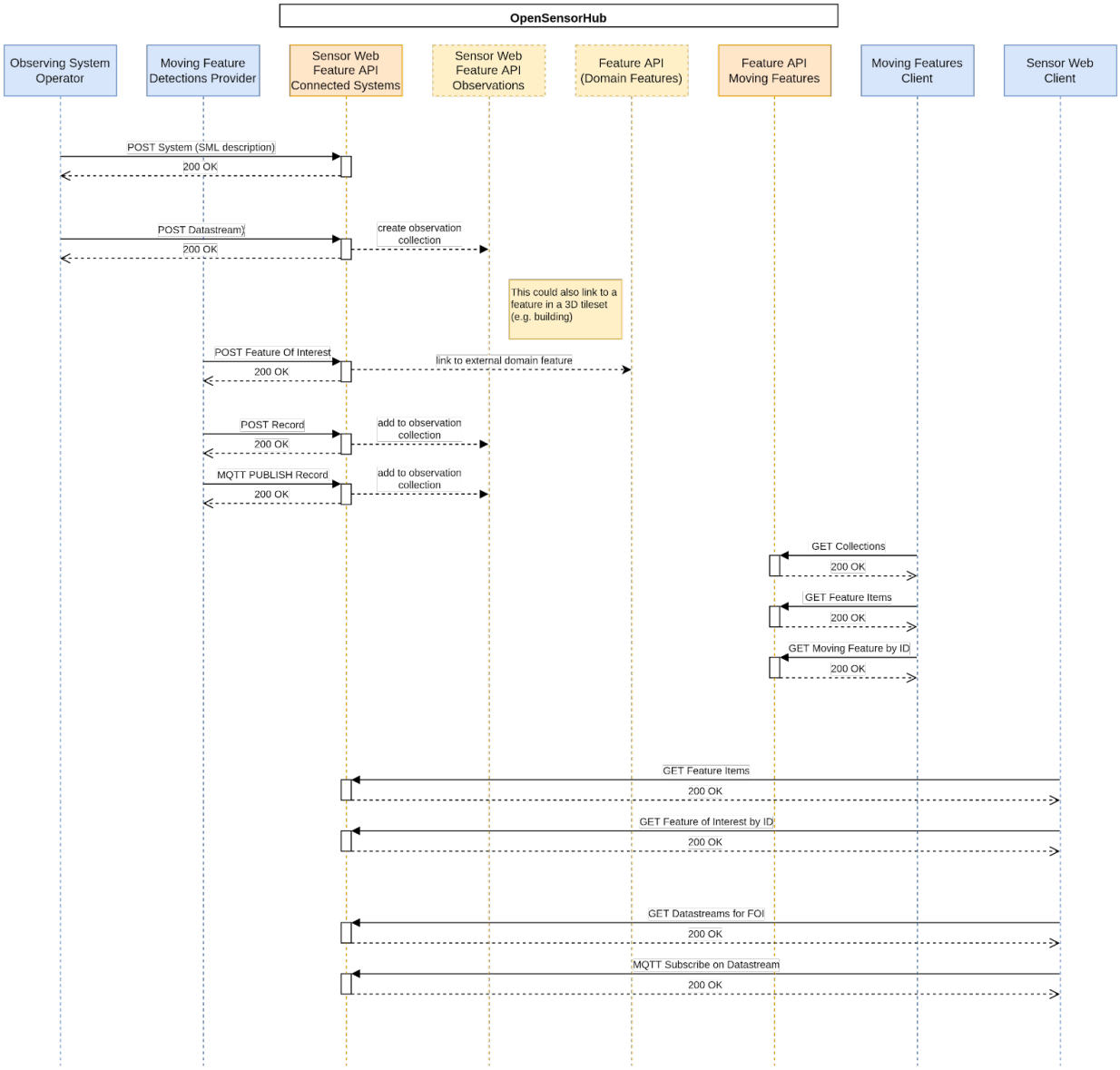


Figure 12 — Sensor Hub Sequence Diagram

8.2. Information Model

The information model that was refined and used during the project is based on the Semantic Sensor Network Ontology model (SOSA/SSN) developed jointly the by OGC and the W3C.

This model was heavily inspired by prior work as part of the OGC Sensor Web Enablement (SWE) initiative, in particular the Observations and Measurements (O&M) and Sensor Modeling Language (SensorML) Standards.

8.2.1. Definitions

For clarity, this section summaries the definitions of some of the fundamental concepts of the SOSA / SSN model and provides a mapping table to other existing and upcoming OGC information models.

8.2.1.1. FeatureOfInterest

The thing whose property is being estimated or calculated in the course of an Observation to arrive at a Result, or whose property is being manipulated by an Actuator, or which is being sampled or transformed in an act of Sampling.

8.2.1.2. System

System is a unit of abstraction for pieces of infrastructure that implement Procedures. A System may have components, its subsystems, which are other Systems.

8.2.1.3. Procedure

A workflow, protocol, plan, algorithm, or computational method specifying how to make an Observation, create a Sample, or make a change to the state of the world (via an Actuator). A Procedure is re-usable and might be involved in many Observations, Samplings, or Actuations. It explains the steps to be carried out to arrive at reproducible Results..

8.2.1.4. Platform

A Platform is an entity that hosts other entities, particularly Sensors, Actuators, Samplers, and other Platforms.
In general, a SOSA Platform can host any other Systems, but also other Platforms. Platform is modelled as a particular type of System in the Connected System API.

8.2.1.5. Sensor

Device, agent (including humans), or software (simulation) involved in, or implementing, a Procedure. Sensors respond to a Stimulus, e.g., a change in the environment, or Input data composed from the Results of prior Observations and generate a Result. Sensors can be hosted by Platforms. Sensor is a subclass of System.

8.2.1.6. Actuator

A device that is used by, or implements, an (Actuation) Procedure that changes the state of the world. Actuator is a subclass of System.

8.2.1.7. Deployment

Describes the Deployment of one or more Systems for a particular purpose. A Deployment may be done on a Platform.

8.2.1.8. Observation

Act of carrying out an (Observation) Procedure to estimate or calculate a value of a property of a FeatureOfInterest. Links to a Sensor to describe what made the Observation and how; links to an ObservableProperty to describe what the result is an estimate of; and to a FeatureOfInterest to detail what that property was associated with.

8.2.1.9. ObservationCollection (SOSA/SSN extension)

Collection of one or more observations, whose members share a common value for one or more properties.

The following table provides mappings between SOSA/SSN concepts and other OGC information models, including prior standards as well as upcoming ones, Observations, Measurements, and Samples (OMS) standard, and the draft Connected Systems API spec:

Table 5

	O&M V2	OMS (O&M V3)	SENSORML	OGC API-CONNECTED SYSTEMS
FeatureOfInterest	Feature Of Interest		FeatureOfInterest resource	System
out of scope	out of scope	PhysicalSystem	System resource (4)	Platform
out of scope	out of scope	PhysicalSystem	System resource tagged sosa:Platform	Sensor
Procedure	Observer	PhysicalComponent or PhysicalSystem (2)	System resource tagged sosa:Sensor	Actuator
out of scope	out of scope	PhysicalComponent or PhysicalSystem (2)	System resource tagged sosa:Actuator	Deployment

	O&M V2	OMS (O&M V3)	SENSORML	OGC API-CONNECTED SYSTEMS
out of scope	Deployment	out of scope (3)	TBD	Procedure
Procedure (1)	Procedure	SimpleProcess or AggregateProcess	Procedure resource	Observation
Observation	Observation	The output data of a sensor system or process	Observation	Actuation
out of scope	out of scope	The output data of an actuator system	Command (5)	Observation Collection(SOSA/SSN extension)

(1) The original O&M model combines the procedure and the physical observer/sensor into a single concept, which caused confusion. This was later improved in SOSA and OMS.
(2) In SensorML, both sensors and actuators are modeled using the PhysicalComponent or PhysicalSystem classes. Differentiating between Sensor and Actuator can be made by explicit semantic tagging (using the 'definition' attribute) or by inspecting the inputs and outputs.
(3) The concept of deployment is not explicitly defined in SensorML, but it can be modeled as a "virtual system" that aggregates together a platform and its subsystems for a specific period of time (cf. validTime property). Another possibility is by time tagging the platform description and changing it every time its components are changed (several descriptions of the same platform thus exists).
(4) All system resources are also features, which allows the draft Connected Systems API to be fully compatible with OGC API — Features.
(5) The concept of command or task in the Connected Systems API is slightly more general as it does not always imply changing the state of a real world feature (e.g., it can lead to a change of processing parameters only).
(6) In SensorML and the Connected Systems API, Datastream often refers to a real-time stream of observations, but is also used to describe a collection of historical observations.

8.2.2. Application to Testbed-18 Use Cases

The next paragraphs summarize how the different Testbed-18 use cases were implemented.

8.2.2.1. Hurricane Track Use Case

In this use case, the Features of Interest were Hurricanes and storms, while the Sensor/Observer was the hurricane tracking system as a whole. All observations were interleaved in a single collection.

Table 6

Mappings	
Sensor/Observer	System tagged as Sensor representing the entire hurricane tracking system
FeatureOfInterest	A hurricane or storm
Platform	Not modeled
Observations	Single Datastream collecting all observations for all hurricane tracks

8.2.2.2. Marine AIS Use Case

In this use case, vessels were not modeled as individual Platforms or Systems but rather as Features of Interest only. Thus the Sensor is the AIS infrastructure as a whole, not the individual sensors mounted on ships.

Table 7

Mappings	
Sensor/Observer	The entire AIS data collection system
FeatureOfInterest	A tracked vessel
Platform	Not modeled
Observations	Single Datastream collecting all observations from all vessels

8.2.2.3. Saildrone Use Case

The particularity of the Saildrone use case is that observation data was collected during several missions and separate metadata and data files were provided for each mission. It was decided to keep the same organization on the SensorHub by creating a separate System for each mission. These systems have to be seen as “virtual systems” because they really represent a particular deployment of a given hardware system (here a saildrone platform). A better way of implementing Deployments with the Connected Systems API is planned in future work.

Table 8

Mappings

Sensor/Observer	A different Saildrone observing system for each mission (even though it sometimes refer to the same hardware)
Feature OfInterest	A hurricane or storm
Platform	Not modeled
Observations	A different datastream for each mission

See the next sections for a more comprehensive modeling of Saildrone features of interest and deployments.

8.2.2.4. HDOB Use Case

Similarly to the AIS use case, HDOB equipped aircraft were not modeled as individual Platforms. For the sake of simplicity, a single System was used to represent the entire fleet. The Features of Interest are the same storm features already defined in the Hurricane Tracks use case.

Table 9

Mappings	
Sensor/Observer	System tagged as Sensor representing the entire HDOB aircraft fleet
FeatureOfInterest	A hurricane or storm
Platform	Not modeled
Observations	Single Datastream collecting all HDOB observations for all hurricanes

8.2.3. Connected Systems API

Testbed-18 has been the opportunity to refine the initial design of the Connected Systems API for which a new OGC Standard Working Group has been formed.

The goals of this new API are the following:

- modernize the existing SWE services using a REST approach;
- fully align with the new OGC API guidelines;
- design as an extension of OGC API — Features;
- better integrate static feature data and bi-directional dynamic (highly time variable) data flows;

- make use of existing data models (SensorML, SWE Common, O&M/OMS);
- support HTTP, Websocket and MQTT protocols; and
- provide efficient binary encodings for high bandwidth sensor data (video, radar, lidar, etc.).

Currently, the API exposes the following resource hierarchy:

Table 10

COLLECTION TYPE/SUB RESOURCE NAME	DESCRIPTION	SUPPORTED FORMATS
Systems	Instances of connected systems registered on the server (e.g., platforms, sensors, robots, etc...)	GeoJSON,JSON-FG,SensorML,MF-JSON
/Members	System resources that are members of a parent system (e.g., hardware subsystem, member of a sensor network, etc.)	GeoJSON,JSON-FG,SensorML
/SamplingFeatures	Sampling features attached to a particular system (e.g., sampling points on a platform or within a sensor network)	GeoJSON,JSON-FG,MF-JSON
/Datastreams	Datastreams produced by a given system and its subsystems	JSON
/Controls	Control channels used to interact with a system and its subsystems	JSON
FeaturesOfInterest	Features of Interest that are independent from a particular system (typically domain features)	GeoJSON,JSON-FG,MF-JSON
Deployments	Information about deployments of specific systems at a specific place and time	GeoJSON,JSON-FG,SensorML
Procedures	Procedures implemented by systems (e.g., system datasheets, lab protocols, etc.). This can also be seen as “system types”	GeoJSON,JSON-FG,SensorML
Datastreams	Datastreams are collections of observations generated by a single system (they are special cases of observation collections that are useful for more efficient streaming and ingestion of high frequency observations)	JSON

COLLECTION TYPE/SUB RESOURCE NAME	DESCRIPTION	SUPPORTED FORMATS
Observations	Sensor observations	OM JSON SWE Common

Some of these resources will provide linking capabilities to other OGC APIs, as explained in the table below.

Table 11

RESOURCE TYPES	LINK TARGETS
Datastream	SensorThings API Datastream, OGC API — EDR Collection
Features Of Interest	OGC API — Features, OGC API — GeoVolumes, OGC 3D Tiles Dataset, OGC API — Moving Features
Observation (Result)	OGC API — Coverages, OGC API — Maps

More up-to-date information about OGC API — Connected Systems can be found on the official GitHub repository.

8.2.4. Using Sampling Features

The Saildrone (and HDOB) use cases highlighted how important the modeling of the Features of Interest is in order to make sense of the data that is being collected without a-priori knowledge of the sensor systems being used.

The Saildrone is an automated marine vehicle that measures its own mechanical state (location, orientation, and velocity) as well as both atmospheric and oceanic parameters. In order to differentiate the intent of the different sensors on-board, multiple features of interest must be defined as follows.

- The Saildrone System is the Feature of Interest for the vessel state observations.
- The Atmosphere is the ultimate Feature of Interest for observations of atmospheric parameters.
- The Ocean is the ultimate feature of interest for observations of water parameters.

However, Saildrones are mobile platforms, which means the on-board sensors are also moving within the atmosphere and the ocean. So, attaching the last two features to the corresponding atmospheric and oceanic observations only provides part of the required information because it does not capture where exactly in the atmosphere or the ocean the observation is made.

Although simply assuming that the user knows the data and can thus imply that the exact location of the observations is provided by the location of the drone platform, providing

metadata that clearly describes this fact is preferable. This information would become even more necessary for more complex use cases where profilers or other types of remote sensors are used (e.g., Saildrones can also carry sonar profilers and cameras).

This pattern has been applied with success to static in-situ sensors using the concept of Spatial Sampling Features introduced along with O&M v2. A Spatial Sampling Feature is a type of "proxy feature" that describes a spatial extent within a larger feature that the observation applies to. It thus references a larger feature (i.e., often called a domain feature) and also provides a geometry for a sample taken within this larger feature. In the O&M v2 model, the geometry can be 0D (SamplingPoint), 1D (SamplingCurve, e.g., LineString), 2D (SamplingSurface, e.g., Polygon), or 3D (Sampling Volume, c.f. Testbed-17 for Sphere examples). *Note that OMS calls this a Spatial Sample and the standard also defines other types of samples (Statistical Sample, Material Sample, etc.).*

The Connected Systems API specification intends to clarify how this existing Spatial Sample model can be used to define Sampling Features whose geometry is defined relative to another feature. This will be helpful for mobile systems since the sampling geometry of such systems typically moves along with the system itself. The key is simply that the sampling feature location is not provided in a geographic coordinate reference system but rather in a local reference system that is attached to the platform and thus moves with it.

The following figure contains annotations overlaid on an original Saildrone diagram to illustrate how sampling features could be used to precisely model the Saildrone use case.

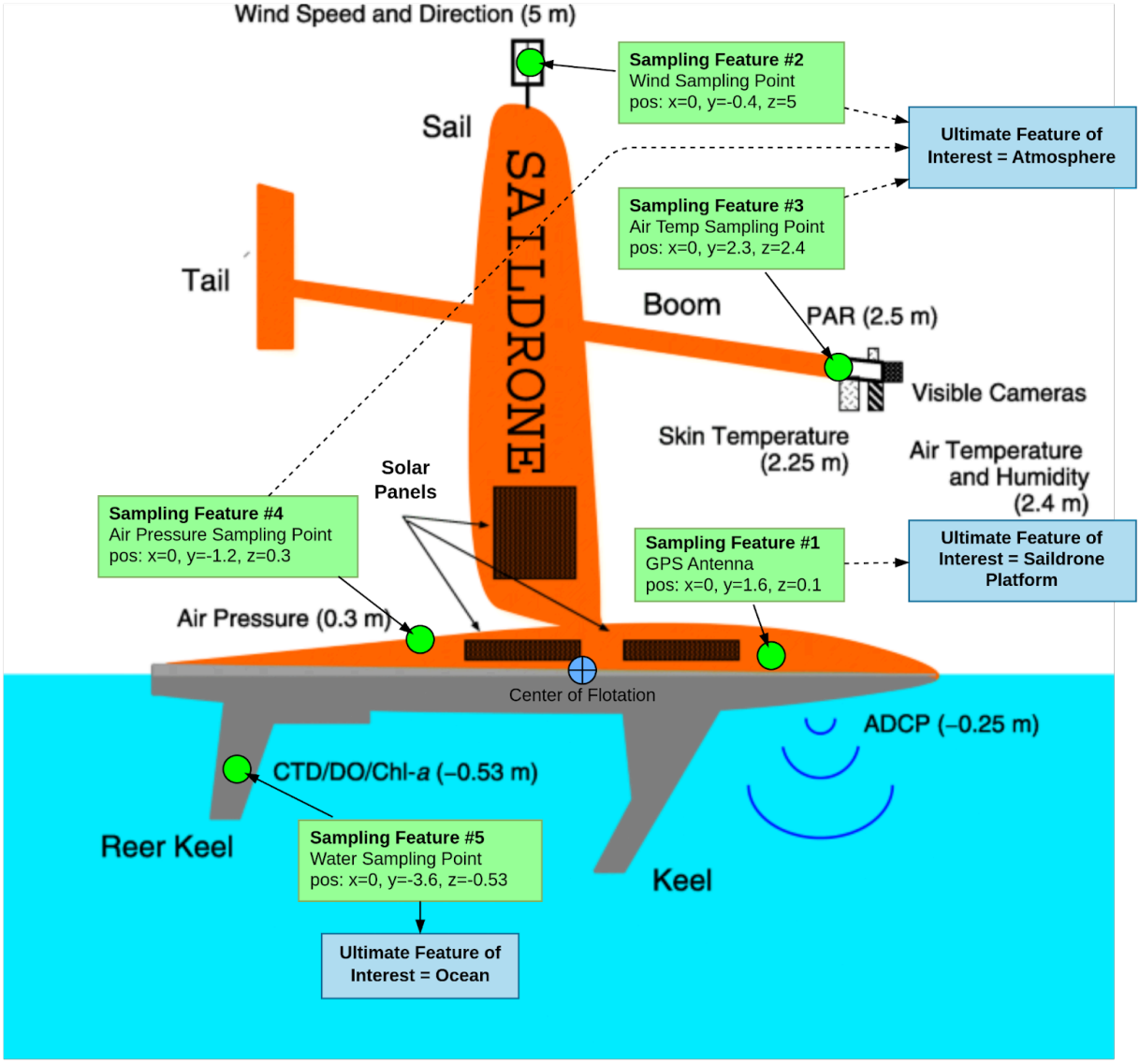


Figure 13 — Saildrone diagram with Sampling Features

The diagram shows different sampling features that are positioned relative to a local reference system whose origin is located at the center of flotation of the vessel. Defining a separate sampling feature for each sensor or group of sensors allows one to specify the exact sampling coordinates but also set different ultimate features of interest (in this case 'ocean' or 'atmosphere').

The sampling features are further described in the following table:

Table 12

SAMPLING FEATURE NAME	ULTIMATE FEATURE OF INTEREST	ASSOCIATED SENSORS
GPS Antenna	Saildrone Platform	GPS, Heading, Course, Ground Speed
Wind Sampling Point	Atmosphere	Wind speed, Wind Direction
Air Temp Sampling Point	Atmosphere	Air Temperature, Humidity, Photosynthetically Active Radiation (PAR)
Air Pressure Sampling Point	Atmosphere	Air pressure
Water Sampling Point	Ocean	Water Temperature, Conductivity, Dissolved Oxygen, Chlorophyll-A Concentration

Note: This is a quite realistic model but this level of detail is not always needed. Since the exact location of each sensor may not be significant, a possible simplification could be to use a single sampling feature for all atmospheric parameters.

8.2.5. Using Deployments

As discussed previously, the Deployment concept is defined in the SOSA/SSN ontology as well as in the upcoming OMS standard. Although it was not part of the Connected Systems API implementation that was used during the Testbed, it is currently being implemented in OpenSensorHub and will be part of the proposed API standard. It was clear during the Testbed that there are many use cases that benefit from using a Deployment instance that is separate from the System, Platform, and Feature of Interest resources.

A Deployment resource is used to provide metadata describing the purpose for deploying a particular observing system at a particular place and time. Creating Deployment resources is especially useful when the same system is deployed multiple times for different purposes. For example, it could describe a particular mission involving one or more mobile sensor systems, a survey/field campaign involving humans, or simply describe that a sensor that is usually not mobile has been moved to a different location.

The plan is to implement the Deployment concept as another kind of Feature in the Connected Systems API, with the following properties containing more information about the deployment.

Table 13

DEPLOYMENT FEATURE PROPERTY	DESCRIPTION
Name	Name of the Deployment (e.g., mission name)

DEPLOYMENT FEATURE PROPERTY	DESCRIPTION
Description	Human readable description of the deployment (e.g., the purpose of the mission)
Location / Geometry	The geographic area where the assets are deployed and/or where observations are collected
Valid Time	The time period during which the deployment was active

In addition, a Deployment resource has the following sub-resources.

Table 14

LINK TYPE	DESCRIPTION
Datastreams	Collection of all datastreams containing observations collected during a particular deployment. It is a filtered collection containing only Datastreams whose valid time intersects the Deployment time period.
Systems	Collection of all systems used in the deployment (i.e. , the deployed systems), including observing systems and platforms *. It is a filtered collection containing only System descriptions that are valid during the Deployment time period (i.e., not the current system description).

** Systems can be deployed on a given Platform (this pattern is often used when the platform is fixed), but Platforms can also be deployed themselves (pattern used for mobile platforms, e.g., drones).*

8.2.6. Detailed System Description

The draft Connected Systems API is designed to let data providers decide the proper level of detail to use when describing observing systems. The data model that was initially used to describe Saildrones during Testbed-18 was purposefully kept simple, as shown on the following UML diagram.

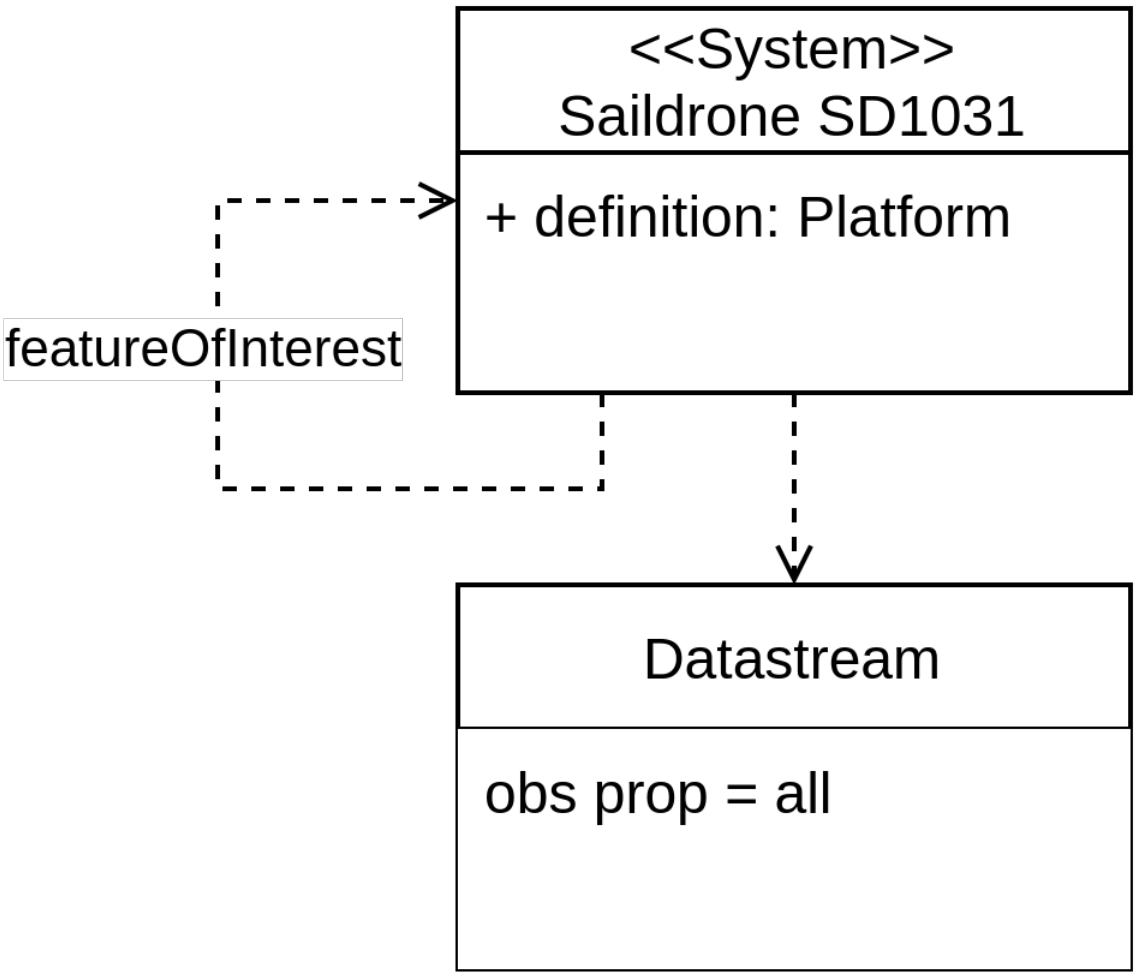


Figure 14 — Saildrone UML Diagram (Simple Metadata)

In this model, each Saildrone vessel is a simple System that references itself as the feature of interest. This model is very simple to implement but fails to capture important metadata about each sensor as well as the ultimate feature of interest for metaocean observables.

Fortunately, an alternative model allowing the provision of much greater detail is also possible and was added to the SensorHub server at the end of the Testbed. The following simplified UML diagram illustrates this model, showing how the Saildrone System can be broken down into different sensor components.

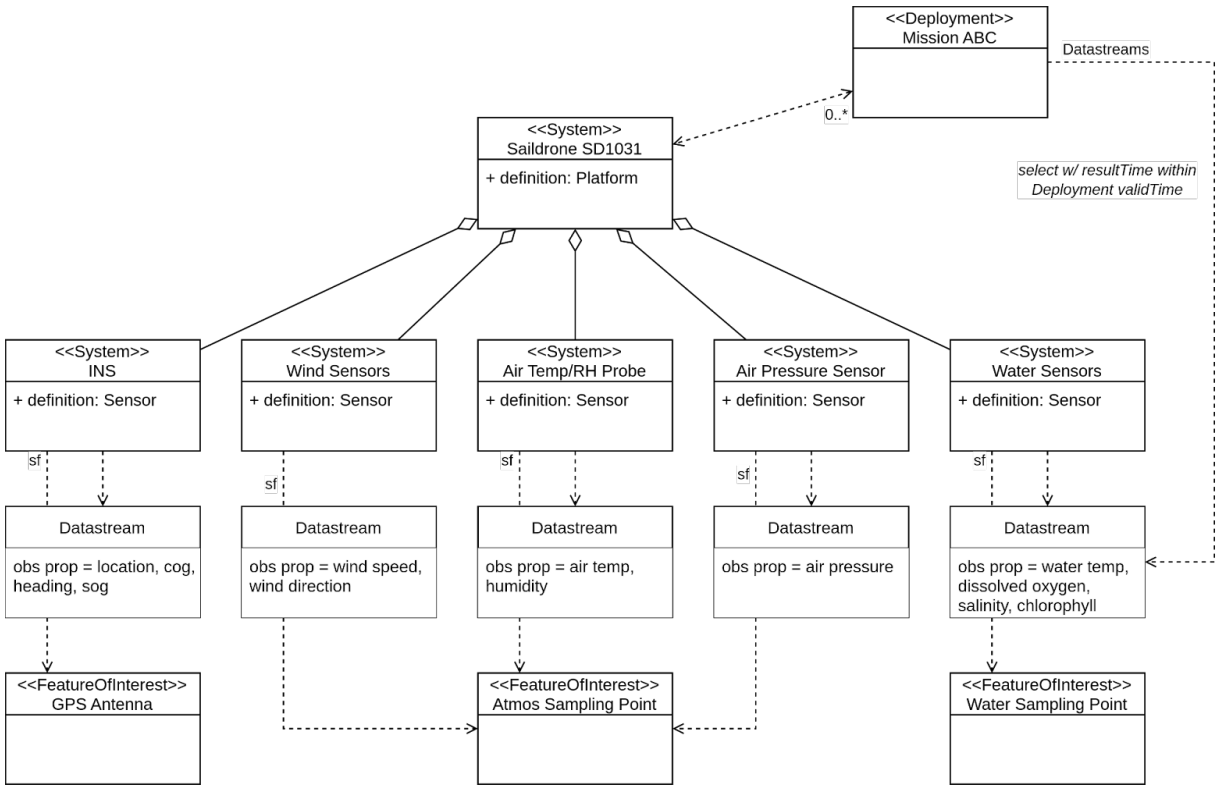


Figure 15 — Saildrone UML Diagram (Advanced Metadata)

In this model, each sensor component produces a different datastream, and observations in each datastream are associated with a different sampling feature located at a specific point on the watercraft. This model provides a much more accurate representation of the actual system.

In addition to the sampling features, more detailed SensorML datasheets have also been created for the platform and its subsystems. The following table provides URLs to the corresponding example resources on the SensorHub:

Table 15

RESOURCE	URL
Saildrone Platform Feature	https://api.georobotix.io/ogc/t18/api/systems/5gsqvptdbdcfm
Saildrone Platform SensorML Description	https://api.georobotix.io/ogc/t18/api/systems/5gsqvptdbdcfm/details
Saildrone Platform Subsystems	https://api.georobotix.io/ogc/t18/api/systems/5gsqvptdbdcfm/members
Saildrone Platform Sampling Features	https://api.georobotix.io/ogc/t18/api/systems/5gsqvptdbdcfm/fois
Navigation Sensors (SensorML)	TBD

RESOURCE	URL
Air Temp/RH Sensor (SensorML)	https://api.georobotix.io/ogc/t18/api/systems/rbdij5ffrqj4i/details
Wind Sensor (SensorML)	https://api.georobotix.io/ogc/t18/api/systems/iv3f2kcq27gfi/details
CTD Sensor (SensorML)	https://api.georobotix.io/ogc/t18/api/systems/g2cg6k2p3rdia/details
Water Oxygen Sensor (SensorML)	https://api.georobotix.io/ogc/t18/api/systems/hpvpn5g0lk7bq/details
Water Chlorophyll Sensor (SensorML)	TBD

8.3. Sensorhub Implementation

The SensorHub component was implemented using version 2.0-beta2 of the open-source OpenSensorHub (OSH) framework. The existing SensorWeb API present in OSH v2 beta2 was used as the basis to develop the new Connected Systems API. This existing framework also made it possible to quickly implement the Moving Feature API using common OSH v2 REST API building blocks.

Both APIs are able to fetch data from a common database, demonstrating how data can be automatically converted from low-level observations to moving features with time varying geometries and properties. The following diagram illustrates in more details how the internal OSH architecture allows implementing both APIs on top of the same middleware.

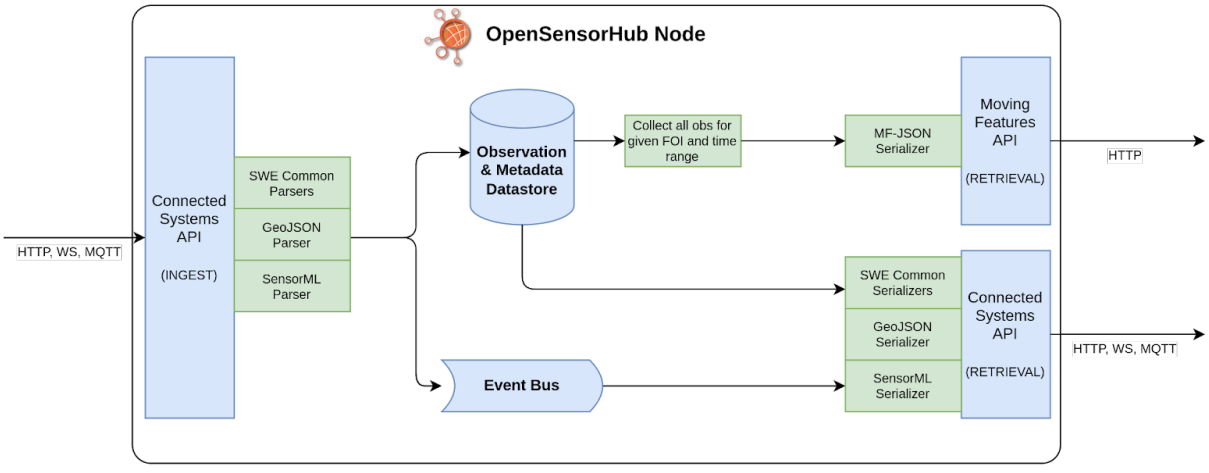


Figure 16 — Internal OSH Architecture

8.3.1. Data Ingestion

Features of interest, System descriptions, and Observations are ingested into the SensorHub using the Connected Systems API. Various protocols and formats are supported by the API.

Supported resource formats include the following.

- GML or GeoJSON for features of interest
- GeoJSON, SensorML/XML or SensorML/JSON for system descriptions
- O&M/XML, O&M/JSON or SWE Common (CSV, JSON or binary) encoded streams for observations
- SWE Common for commands

Regarding protocols, resources can be inserted using the following.

- HTTP POST (or PUT for updates)
- Incoming WebSocket stream (for observations and commands, add only)
- MQTT Publish (any resource type, add only)

When a given resource is received and parsed without error, it is added to a persistent store using OSH Datastore API. This API is used to abstract various storage backends, ranging from embedded databases to large scale distributed databases.

During the transactional process, an event is also dispatched using the Event Bus so that consumers can be notified that data has been added, modified, or deleted. For observations and commands, real-time consumers can receive data without having to wait for the database to be updated, thus reducing overall latency of the data delivery pipeline. This is particularly useful for data streams with high sampling rate such as video or orientation telemetry.

8.3.2. Data Retrieval

Once ingested into the system, data is made available via several channels, including the following.

- OGC Sensor Observation Service 2.0 (SOS)
- OGC SensorThings API 1.1 (STA)
- OGC Connected Systems API (draft)
- OGC Moving Features API (draft)

The following protocols are supported for retrieval.

- HTTP and Websocket for SOS
- HTTP and MQTT for SensorThings
- HTTP, Websocket, and MQTT for Connected Systems API
- HTTP for Moving Features API

During Testbed-18, only the Connected Systems API and Moving Features API were used for data retrieval by clients. The same data was exposed via both interfaces.

For the Connected Systems API, the retrieval functionality was supported natively since the same interface was used for ingestion. However, exposing the same data using the Moving Features API was more complex and involved the following process.

1. Every Feature of Interest registered on the SensorHub is exposed as a "Moving Feature" resource through the Moving Feature API. The resulting Moving Features can then be grouped into collections based on any criteria.
2. When temporal properties of a given Moving Feature are requested, all observations (other than location) for the corresponding feature of interest are collected, time matched, and packaged in MF-JSON format in chronological order.
3. When temporal geometries of a given Moving Feature are requested, observations of the feature location are collected and packaged in MF-JSON format. Only point location is supported by the current prototype implementation.

The following filtering capabilities have been implemented in the Moving Feature API prototype.

- Filter moving features by id, UID, keywords, property values, valid time, and spatial extent
- Filter temporal properties by observable and time

8.4. Client Implementation

A client application making use of the Connected Systems API was also developed during the Testbed. This application is able to discover systems and datastreams registered on the server and display all data ingested for the Testbed, including the following.

- Hurricane Tracks
- Vessel AIS Tracks and Navigation Information
- Saildrone Tracks, Navigation Information, and Observations

- HDOB Aircraft Tracks and Observations

The client is a web application developed with the Vue.js framework and based on Cesium JS and the OpenSensorHub Javascript Toolkit (OSH-JS). All data can be viewed in 3D and a time controller is available to visualize and replay historical data.

The client architecture is illustrated below.

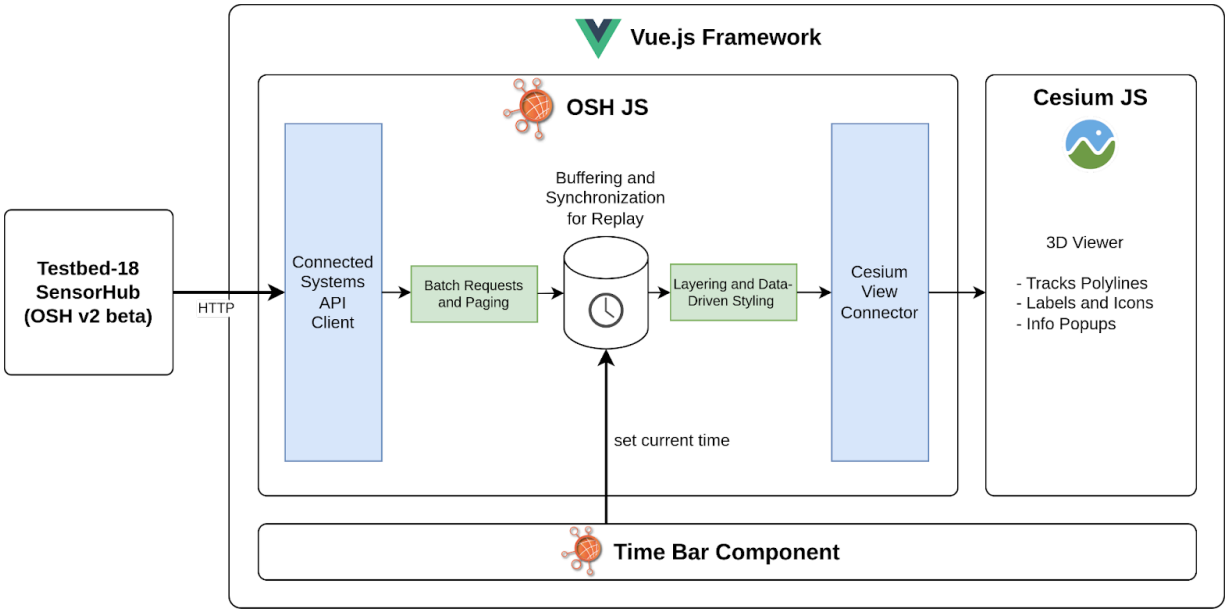


Figure 17 — Botts Client Architecture

The next section presents screenshots of the application displaying various datasets ingested during Testbed 18.

A live version of this client application is also available at: https://api.georobotix.io/ogc/t18/web/tracks-demo/index.html

8.4.1. Hurricane Tracks

The first screenshot shows the tracks of two hurricanes that have been selected in the application: Hurricane SAM (2021) and Tropical storm KARL (2016).

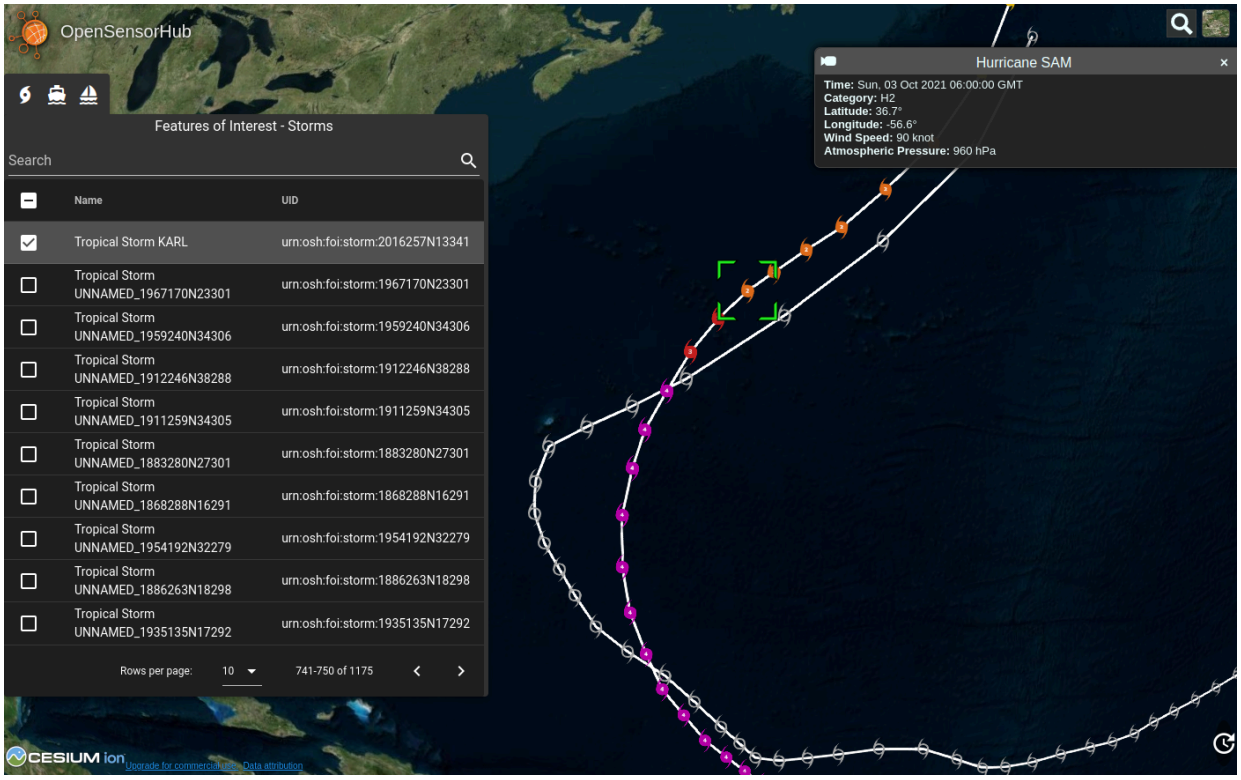


Figure 18 — Hurricane SAM and Tropical storm KARL

The table on the left allows selection of features of interest for which observations should be added to the map. When a storm is selected, its entire track is displayed on the map using both a polyline and individual markers. The visual style of each marker can be changed according to data values. In this case, a different icon is used depending on the storm category.

When selecting a particular marker, more information about the state of the hurricane at that particular location is shown in the popup at the top right of the screen.

8.4.2. Maritime Vessel Tracks

The following two screenshots show tracks for marine vessels for which AIS data was ingested during the testbed. The first screenshot shows the track of a towing vessel named “SEASPAN PUSHER.”

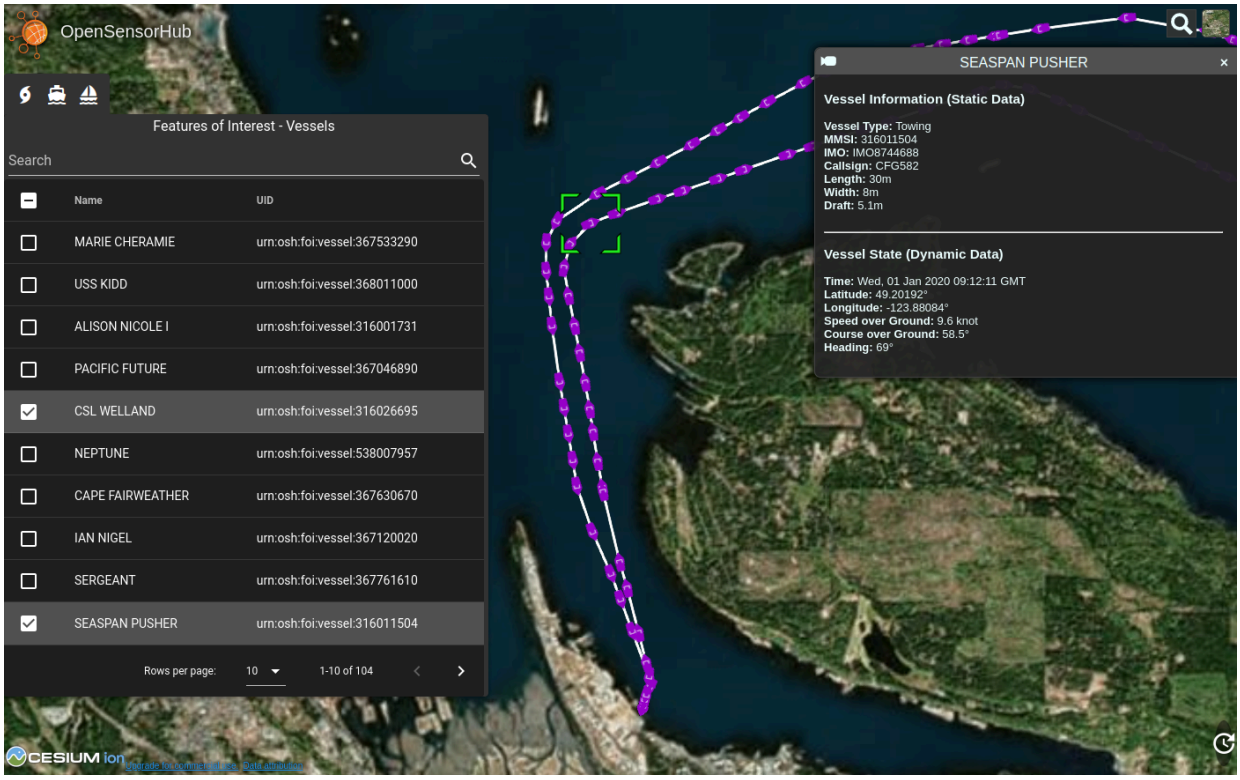


Figure 19 — Marine Vessel Tracking

Here again, the left panel allows selection of the features of interest (each vessel is a different feature of interest). The track is shown as a polyline and individual sampling locations (i.e., where measurements were made and transmitted using AIS channel) are represented using a ship icon. Each icon is also oriented according to the heading measurement provided in the data. When clicking on one of these icons, more information about the vessel and the corresponding observation are displayed in the popup at the top left.

The next screenshot shows the same data but for a different ship named “CSL WELLAND.”

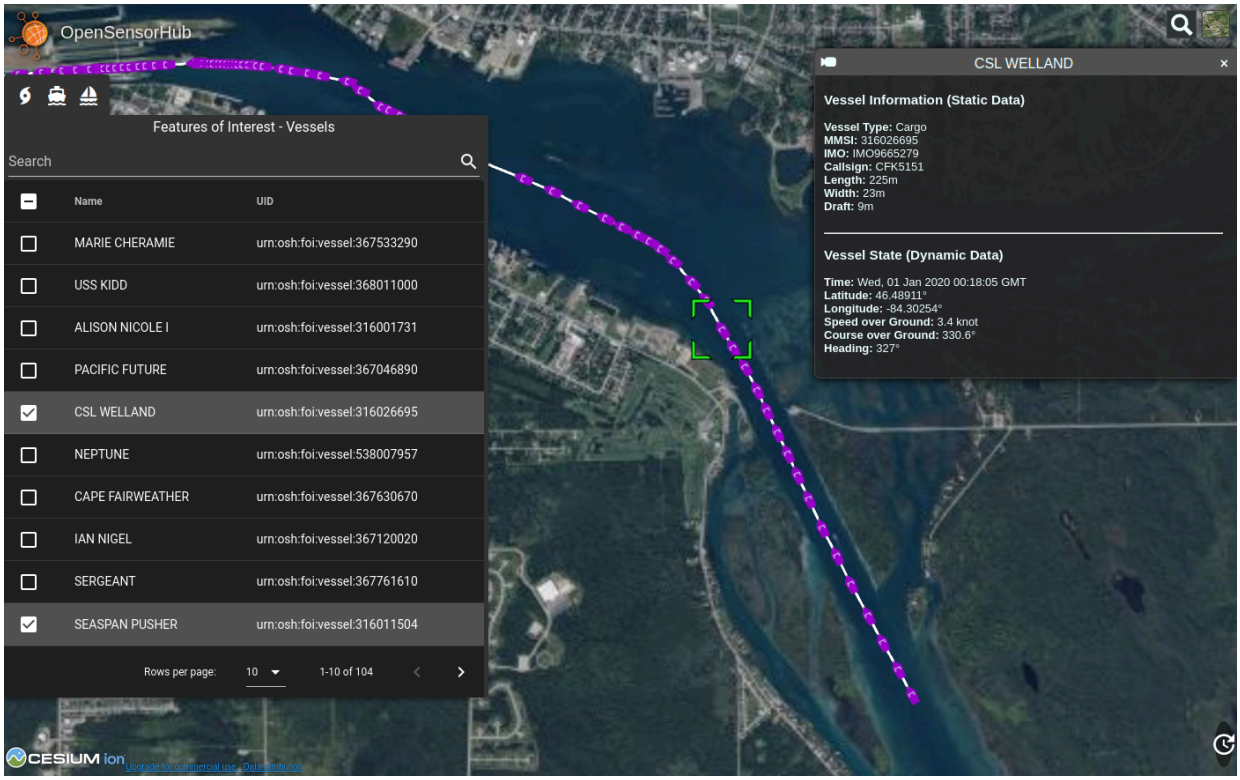


Figure 20 — CSL Welland

8.4.3. Saildrone Tracks

The next examples show the tracks of two different saildrone missions. For the transatlantic mission #1021, only navigation data was ingested in the SensorHub. This data is similar to the AIS vessel data and is displayed on the following screenshot.

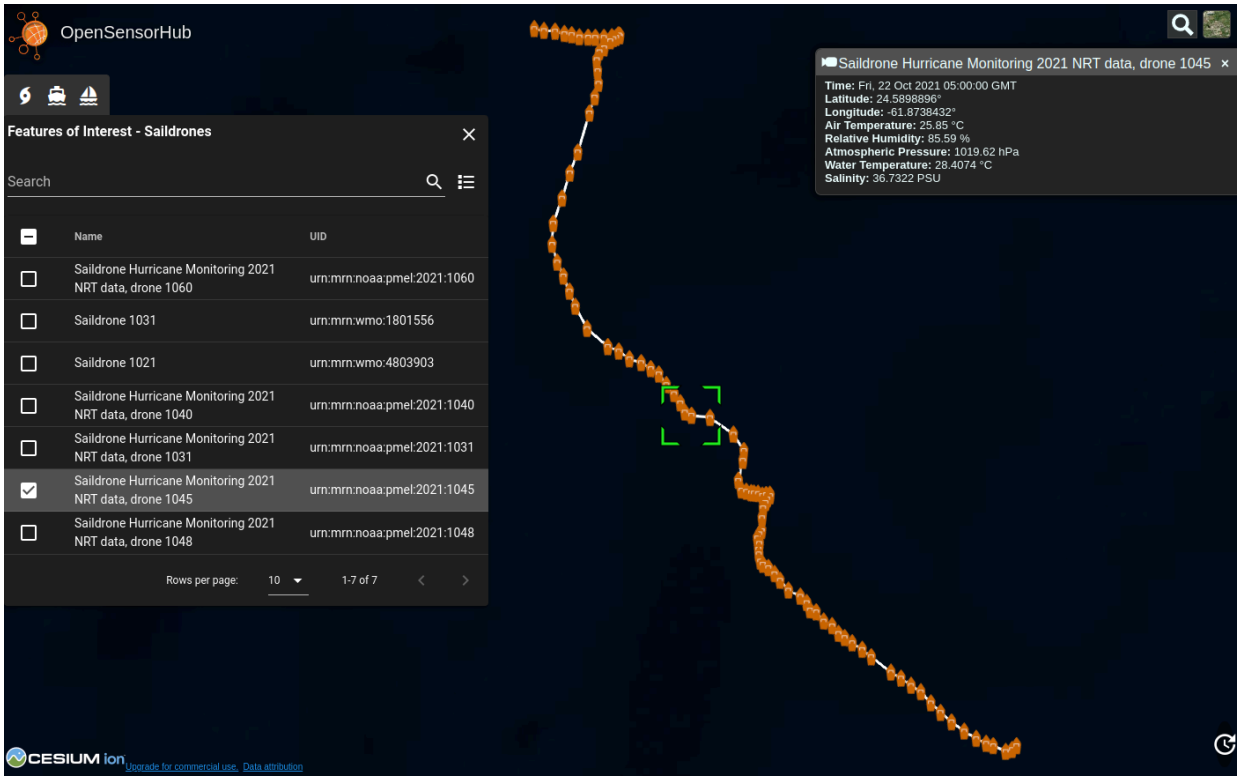


Figure 21 — Saildrone Mission 1045

In this case, heading or course data is not provided so it is not possible to orient the ship icon in the correct direction.

9

CLIENT: D143 SUPERELECTRIC

9 CLIENT: D143 SUPERELECTRIC

This section provides information about the Client deliverable by Superelectric.

9.1. Description & Objective

The scope of the Superelectric D143 Client Application component was to connect to the Sensorhub, retrieve data from the motion feature ingestion systems using the draft OGC API — Moving Feature Standard, and enhance track data and moving feature data via multi-sensor fusion. If possible, the client connects to additional synthesized data sources as satellite and/or drone imagery.

For this purpose, Superelectric started from a prototype application developed from a previous activity in collaboration with European Union Satellite Centre (EU Satcen). This prototype was designed to create “co-recording” images between streams of images from different sources, such as from satellites and drones, that are geographically and temporally coherent; i.e., the images containing the same AOI (Area Of Interest) and acquired images in the same period of time.

The sequences of images generated by such different sensors will have both different frame rates and different resolutions (in pixels). Also the definition of the Sensor Model can be different. For example, usually satellite images are always supplied by metadata expressed in RPC parameters, while drone images have the physical sensor model in the data of their images, i.e., the focal length, the pixel size, etc.+

After a harmonization process, in which pairs of images, both from satellite and drone, with the same timestamps and containing the same AOI are selected, these pairs move on to the next step of image fusion. The fusion process (with images at different resolutions) is an analytical process based on the calculation of the RPCs for the drone images. The RPCs for the satellite are provided as metadata. The calculation of the RPCs starting from the physical sensor model of the drone depends on the supply of a large number of GCPs within the acquired images by drone. If these GCPs do not exist, it will be necessary to estimate with geometric calculations.

The figure below shows the client architecture.

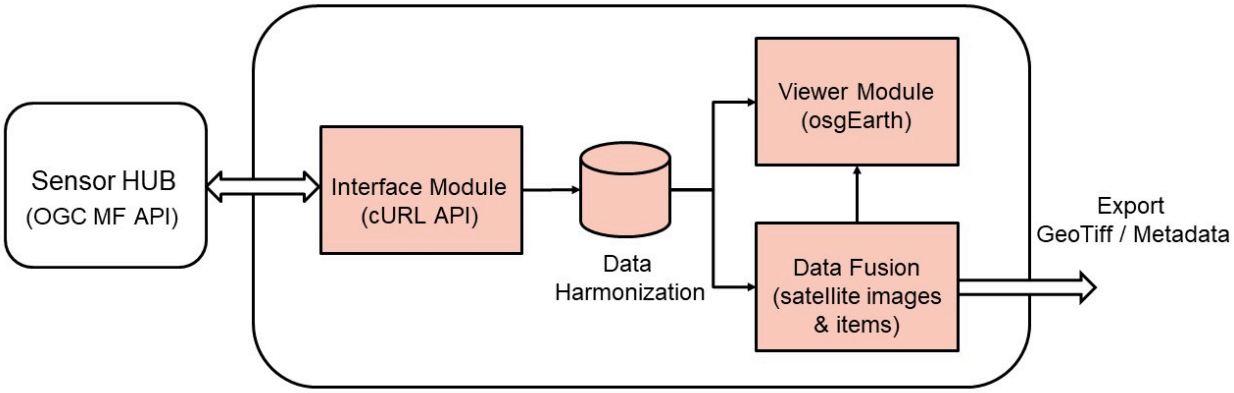


Figure 22 — Client Architecture

9.2. Workflow & Interfaces

To adapt the initial prototype to the needs of the TB-18 MFSI task, Superelectric proposed the workflow described by the sequence diagram below adding the “Interface Module” component in order to connect with the Sensor HUB (D142) with requests via OGC MF API.

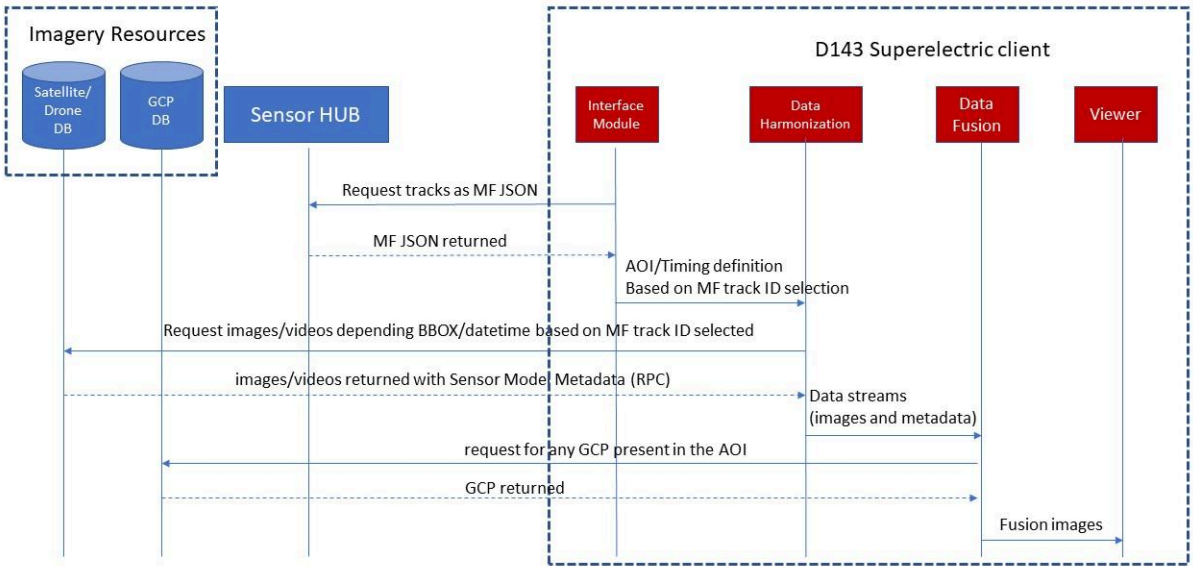


Figure 23 — Sequence diagram of D143 client component

The following steps are performed.

Step 1:

Request Tracks (temporal geometries and temporal properties): specified by a particular "mFeatureId" as FOI the following queries are made as JSON-encoded:
GET /collections/{collectionId}/items/{mFeatureId}/tgeometries
GET /collections/{collectionId}/items/{mFeatureId}/tproperties

Two use cases were considered: the hurricanes and the AIS data, so among the collections available in the sensor HUB we have selected the "storms" and "vessels" as FOI collectionsID is performed.

Step 2:
Area Of Interest (AOI) and Timing definition: Once a FeatureID has been selected and its properties have been explored, an AOI or better a Bounding Box (BBOX) and a time interval is defined to make requests to the satellite and or drone imagery server.

Step 3:
Request Satellite and drone images with BBOX & Time parameters: through the available protocol by the server, the request for images of the area of interest is made by specifying as search parameters the BBOX of the mFeatureID selected and time interval in reference to the observation datetime.

Step 4:
Data Harmonization: identification of the pairs of images (from drone and satellite) such that the drone image is contained within the satellite image and has the same timestamp.
This step will be performed if only both satellite and drone images are available.

Step 5:
Data Fusion: the data fusion is the step where the drone image is merged with the satellite image. In this process it is necessary to calculate the RPC (sensor model) of the drone sensor. For this it is necessary to obtain the large number of GCPs of the area of interest.
This step will be performed only if both satellite and drone images are available.

Step 6:
Rendering: in this step both the mFeatureID (FOI) track (as geometric polyline) on a 3D GIS cartography and the result of the Data Fusion as raster image in OGC GeoTIFF format will be displayed. A GIS rendering engine osgEarth SDK OpenSource will be used.

9.3. Model

In the framework of the Testbed-18 MF thread, two use cases were considered where trajectories are represented by hurricanes and vessels from the AIS data repository. The example project scenario raw data is from the Gulf of Mexico.

Input Data

Retrieving Moving Features from the SensorHub

The input data for the D143 Client is the output data from the Moving Feature Collection Services which is being stored and provided by the Sensor HUB. This client is able to manage

collections registered on the server and display all observations data ingested for the Testbed including the following.

- Hurricane Tracks
- Vessel AIS Tracks and Navigation Information
- Saildrone Tracks, Navigation Information, and Observations

9.3.1. Hurricane collections

The input data as storm collection ID returned by the https://api.georobotix.io/ogc/t18/mfapi/collections/storms/items?f=application%2Fjson query has the following format:


```
{
  "items": [
    {
      "type": "Feature",
      "id": "resvipdo9f52u",
      "properties": {
        "uid": "urn:osh:foi:storm:1916181N13281",
        "featureType": "Feature",
        "name": "Tropical Storm UNNAMED_1916181N13281",
        "validTime": [
          "1916-06-28T12:00:00Z",
          "1916-07-10T18:00:00Z"
        ]
      }
    },
    {
      "type": "Feature",
      "id": "31d9u5gf6qveu",
      "properties": {
        "uid": "urn:osh:foi:storm:1859297N20267",
        "featureType": "Feature",
        "name": "Tropical Storm UNNAMED_1859297N20267",
        "validTime": [
          "1859-10-24T00:00:00Z",
          "1859-10-29T18:00:00Z"
        ]
      }
    },
…..
],
  "links": [
    {
      "rel": "next",
      "href": "https://api.georobotix.io/ogc/t18/mfapi/collections/storms/
items?offset=100&f=application%2Fjson",
      "type": "application/json"
    }
  ]
}
```


Each response contains a maximum of 100 items, and if a next node appears between the “links” then the search for the next storm items continues with the url specified in the “href” node.

The input data as temporal geometries for storm ID returned by the https://api.georobotix.io/ogc/t18/mfapi/collections/storms/items/resvipdo9f52u/tgeometries query has the following format:


```
{
  "temporalGeometries": [
    {
      "id": "tg-tm3kijpkaoei6",
      "type": "MovingPoint",
      "datetimes": [
        "1916-06-28T12:00:00Z",
        "1916-06-28T18:00:00Z",
        "1916-06-29T00:00:00Z",
        "1916-06-29T06:00:00Z",
…..
      ],
      "coordinates": [
        [ 12.5, -79.5 ],
        [ 12.5, -79.8 ],
        [ 12.5, -80.1 ],
        [ 12.5, -80.5 ],
…..
      ],
      "interpolation": "Linear"
    }
  ]
}
```


The input data as temporal properties for storm ID returned by the https://api.georobotix.io/ogc/t18/mfapi/collections/storms/items/resvipdo9f52u/tproperties query has the following format:


```
{
  "temporalProperties": [
    {
      "datetimes": [
        "1916-06-28T12:00:00Z",
        "1916-06-28T18:00:00Z",
        "1916-06-29T00:00:00Z",
        "1916-06-29T06:00:00Z",
…..
      ],
      "windSpeed": {
        "type": "TDouble",
        "form": "http://mmisw.org/ont/cf/parameter/wind_speed",
        "description": "Maximum sustained winds",
        "interpolation": "Linear",
        "values": [ 25.0, 25.0, 25.0, 25.0, ….]
      },
      "minPressure": {
        "type": "TDouble",
        "form": "http://mmisw.org/ont/cf/parameter/air_pressure",
        "interpolation": "Linear",
        "values": [ -1.0, -1.0, -1.0, -1.0, ….]
      },
      "category": {
        "type": "TText",
        "form": "http://sensorml.com/ont/T18/concepts/StormCategory",
        "interpolation": "Linear",
        "values": [ "TD", "TD", "TD", "TD", ….]
      }
    }
```

```
  ]
}
```

The input data as vessel collection ID returned by the https://api.georobotix.io/ogc/t18/mfapi/collections/vessels/items?f=application%2Fjson query has the following format:

```
{
  "items": [
    {
      "type": "Feature",
      "id": "p96kqua33r1n2",
      "properties": {
        "uid": "urn:osh:foi:vessel:367533290",
        "featureType": "http://www.opengis.net/def/featureType/x-T18/Vessel",
        "name": "MARIE CHERAMIE",
        "description": "Proxy feature for vessel 367533290",
        "mmsi": "367533290",
        "imo": "IMO8964862",
        "callSign": "WDG4131",
        "vesselType": "Other",
        "length": 49.0,
        "width": 13.0,
        "draft": 3.1
      }
    },
    {
      "type": "Feature",
      "id": "umlomq0l6bsuq",
      "properties": {
        "uid": "urn:osh:foi:vessel:368011000",
        "featureType": "http://www.opengis.net/def/featureType/x-T18/Vessel",
        "name": "USS KIDD",
        "description": "Proxy feature for vessel 368011000",
        "mmsi": "368011000",
        "imo": "",
        "callSign": "NKID",
        "vesselType": "Other"
      }
    },
…..
  ],
  "links": [
    {
      "rel": "next",
      "href": "https://api.georobotix.io/ogc/t18/mfapi/collections/vessels/
items?offset=100&f=application%2Fjson",
      "type": "application/json"
    }
  ]
}
```

Each response contains a maximum of 100 items and if a next node appears between the “links” then the search for the next vessel items continues with the url specified in the “href” node.

The input data as temporal geometries for vessel ID returned by the https://api.georobotix.io/ogc/t18/mfapi/collections/vessels/items/umlomq0l6bsuq/tgeometries query has the following format:

```
{
  "temporalGeometries": [
    {
      "id": "tg-kuhmds0ib5gd8",
```

```
      "type": "MovingPoint",
      "datetimes": [
        "2020-01-01T00:00:07Z",
        "2020-01-01T00:01:17Z",
        "2020-01-01T00:02:27Z",
        "2020-01-01T00:03:37Z",
….
      ],
      "coordinates": [
        [ 47.98258, -122.22902 ],
        [ 47.98255, -122.22893 ],
        [ 47.98257, -122.22893 ],
        [ 47.98255, -122.22897 ],
…..
      ],
      "interpolation": "Linear"
    }
  ],
  "links": [
    {
      "rel": "next",
      "href": "https://api.georobotix.io/ogc/t18/mfapi/collections/vessels/
items/umlomq0l6bsuq/tgeometries?offset=100",
      "type": "auto"
    }
  ]
}
```

Each response contains a maximum of 100 observations and if a next node appears between the “links” then the search for the next vessel item observations continues with the url specified in the “href” node.

The input data as temporal properties for vessel ID returned by the https://api.georobotix.io/ogc/t18/mfapi/collections/vessels/items/umlomq0l6bsuq/tproperties query has the following format:

```
{
  "temporalProperties": [
    {
      "datetimes": [
        "2020-01-01T00:00:07Z",
        "2020-01-01T00:01:17Z",
        "2020-01-01T00:02:27Z",
        "2020-01-01T00:03:37Z",
…..
      ],
      "sog": {
        "type": "TDouble",
        "form": "http://sensorml.com/ont/swe/property/SpeedOverGround",
        "description": "Vessel speed relative to ground",
        "interpolation": "Linear",
        "values": [ 0.5, 0.9, 0.6, 0.3, ….]
      },
      "cog": {
        "type": "TDouble",
        "form": "http://sensorml.com/ont/swe/property/CourseOverGround",
        "description": "Vessel travel direction relative to ground",
        "interpolation": "Linear",
        "values": [ 316.9, 121.7, 121.0, 137.3, ….]
      },
      "heading": {
        "type": "TDouble",
```

```
        "form": "http://sensorml.com/ont/swe/property/TrueHeading",
        "description": "Vessel heading direction relative to true north,
measured clockwise",
        "interpolation": "Linear",
        "values": [ 216.0, 216.0, 216.0, 216.0, …..]
      },
      "status": {
        "type": "TText",
        "form": "http://sensorml.com/ont/x-ais/property/NavigationStatus",
        "description": "Vessel Status",
        "interpolation": "Linear",
        "values": [ "0", "0", "0", "0", ….]
      }
    }
  ],
  "links": [
    {
      "rel": "next",
      "href": "https://api.georobotix.io/ogc/t18/mfapi/collections/vessels/
items/umlomq0l6bsuq/tproperties?offset=100",
      "type": "auto"
    }
  ]
}
```


9.3.2. Vessel AIS collections

The input data as vessel collection ID returned by the https://api.georobotix.io/ogc/t18/mfapi/collections/vessels/items?f=application%2Fjson query has the following format:


```
{
  "items": [
    {
      "type": "Feature",
      "id": "p96kqua33r1n2",
      "properties": {
        "uid": "urn:osh:foi:vessel:367533290",
        "featureType": "http://www.opengis.net/def/featureType/x-T18/Vessel",
        "name": "MARIE CHERAMIE",
        "description": "Proxy feature for vessel 367533290",
        "mmsi": "367533290",
        "imo": "IMO8964862",
        "callSign": "WDG4131",
        "vesselType": "Other",
        "length": 49.0,
        "width": 13.0,
        "draft": 3.1
      }
    },
    {
      "type": "Feature",
      "id": "umlomq0l6bsuq",
      "properties": {
        "uid": "urn:osh:foi:vessel:368011000",
        "featureType": "http://www.opengis.net/def/featureType/x-T18/Vessel",
        "name": "USS KIDD",
        "description": "Proxy feature for vessel 368011000",
        "mmsi": "368011000",
        "imo": "",
        "callSign": "NKID",
```

```
        "vesselType": "Other"
      }
    },
….
  ],
  "links": [
    {
      "rel": "next",
      "href": "https://api.georobotix.io/ogc/t18/mfapi/collections/vessels/
items?offset=100&f=application%2Fjson",
      "type": "application/json"
    }
  ]
}
```

Each response contains a maximum of 100 observations and if a next node appears between the “links” then the search for the next vessel item observations continues with the url specified in the “href” node.

The input data as temporal properties for vessel ID returned by the https://api.georobotix.io/ogc/t18/mfapi/collections/vessels/items/umlomq0l6bsuq/tproperties query has the following format:

```
{
  "temporalGeometries": [
    {
      "id": "tg-kuhmds0ib5gd8",
      "type": "MovingPoint",
      "datetimes": [
        "2020-01-01T00:00:07Z",
        "2020-01-01T00:01:17Z",
        "2020-01-01T00:02:27Z",
        "2020-01-01T00:03:37Z",
….
      ],
      "coordinates": [
        [ 47.98258, -122.22902 ],
        [ 47.98255, -122.22893 ],
        [ 47.98257, -122.22893 ],
        [ 47.98255, -122.22897 ],
…..
      ],
      "interpolation": "Linear"
    }
  ],
  "links": [
    {
      "rel": "next",
      "href": "https://api.georobotix.io/ogc/t18/mfapi/collections/vessels/
items/umlomq0l6bsuq/tgeometries?offset=100",
      "type": "auto"
    }
  ]
}
```

Each response contains a maximum of 100 observations and if a next node appears between the “links” then the search for the next vessel item observations continues with the url specified in the “href” node.

The input data as temporal properties for vessel ID returned by the https://api.georobotix.io/ogc/t18/mfapi/collections/vessels/items/umlomq0l6bsuq/tproperties query has the following format:


```
{
  "temporalProperties": [
    {
      "datetimes": [
        "2020-01-01T00:00:07Z",
        "2020-01-01T00:01:17Z",
        "2020-01-01T00:02:27Z",
        "2020-01-01T00:03:37Z",
….. 
      ],
      "sog": {
        "type": "TDouble",
        "form": "http://sensorml.com/ont/swe/property/SpeedOverGround",
        "description": "Vessel speed relative to ground",
        "interpolation": "Linear",
        "values": [ 0.5, 0.9, 0.6, 0.3, ….]
      },
      "cog": {
        "type": "TDouble",
        "form": "http://sensorml.com/ont/swe/property/CourseOverGround",
        "description": "Vessel travel direction relative to ground",
        "interpolation": "Linear",
        "values": [ 316.9, 121.7, 121.0, 137.3, ….]
      },
      "heading": {
        "type": "TDouble",
        "form": "http://sensorml.com/ont/swe/property/TrueHeading",
        "description": "Vessel heading direction relative to true north,
measured clockwise",
        "interpolation": "Linear",
        "values": [ 216.0, 216.0, 216.0, 216.0, …..]
      },
      "status": {
        "type": "TText",
        "form": "http://sensorml.com/ont/x-ais/property/NavigationStatus",
        "description": "Vessel Status",
        "interpolation": "Linear",
        "values": [ "0", "0", "0", "0", ….]
      }
    }
  ],
  "links": [
    {
      "rel": "next",
      "href": "https://api.georobotix.io/ogc/t18/mfapi/collections/vessels/
items/umlomq0l6bsuq/tproperties?offset=100",
      "type": "auto"
    }
  ]
}
```


Each response contains a maximum of 100 observations and if a next node appears between the “links” then the search for the next vessel item observations continues with the url specified in the “href” node.

9.3.3. Saildrone Collections

The input data as saildrone collection ID returned by the https://api.georobotix.io/ogc/t18/mfapi/collections/saildrones/items?f=application%2Fjson query has the following format:


```
{
  "items": [
    {
      "type": "Feature",
      "id": "p7e4barvjhaj2",
      "properties": {
        "uid": "urn:mrn:wmo:1801556",
        "featureType": "http://www.opengis.net/def/featureType/T18/Saildrone",
        "name": "Saildrone 1031",
        "description": "Saildrone Unmanned Surface Vehicle (USV) 1031",
        "validTime": [
          "2022-07-16T20:00:00Z",
          "2022-10-14T09:59:00Z"
        ],
        "mission": "Saildrone Atlantic 2022 Hurricane Monitoring, drone 1031",
        "platformID": "1031"
      }
    },
    {
      "type": "Feature",
      "id": "dpgba35clbhau",
      "properties": {
        "uid": "urn:mrn:noaa:pmel:2021:1045",
        "featureType": "http://www.opengis.net/def/featureType/T18/Saildrone",
        "name": "Saildrone Hurricane Monitoring 2021 NRT data, drone 1045",
        "description": "Saildrone Hurricane Monitoring 2021 NRT data, drone
1045",
        "validTime": [
          "2021-10-21T00:00:00Z",
          "2021-10-28T13:59:00Z"
        ]
      }
    },
….
  ],
  "links": [
    {
      "rel": "next",
      "href": "https://api.georobotix.io/ogc/t18/mfapi/collections/saildrones/
items?offset=100&f=application%2Fjson",
      "type": "application/json"
    }
  ]
}
```


Each response contains a maximum of 100 items and if a next node appears between the “links” then the search for the next saildrone items continues with the url specified in the “href” node.

The input data as temporal geometries for Saildrone ID returned by the https://api.georobotix.io/ogc/t18/mfapi/collections/saildrones/items/dpgba35clbhau/tgeometries query has the following format:


```
{
  "temporalGeometries": [
```

```
    {
      "id": "tg-4knoao0uki8bq",
      "type": "MovingPoint",
      "datetimes": [
        "2021-10-21T00:00:00Z",
        "2021-10-21T00:30:00Z",
        "2021-10-21T01:00:00Z",
        "2021-10-21T02:00:00Z",
….
      ],
      "coordinates": [
        [24.3430336, -61.6090112],
        [24.3401616, -61.61264],
        [24.3387488, -61.6178816],
        [24.3431344, -61.6280896],
….
      ],
      "interpolation": "Linear"
    }
  ],
  "links": [
    {
      "rel": "next",
      "href": "https://api.georobotix.io/ogc/t18/mfapi/collections/saildrones/
items/ dpgba35clbhau/tgeometries?offset=100",
      "type": "auto"
    }
  ]
}
```

Each response contains a maximum of 100 observations and if a next node appears between the “links” then the search for the next Saildrone item observations continues with the url specified in the “href” node.

The input data as temporal properties for Saildrone ID returned by the https://api.georobotix.io/ogc/t18/mfapi/collections/saildrones/items/dpgba35clbhau/tproperties query has the following format:

```
{
  "temporalProperties": [
    {
      "datetimes": [
        "2021-10-21T00:00:00Z",
        "2021-10-21T00:30:00Z",
        "2021-10-21T01:00:00Z",
        "2021-10-21T02:00:00Z",
…..
      ],
      "TEMP_AIR_MEAN": {
        "type": "TDouble",
        "form": "",
        "description": "Mean Air Temperature",
        "interpolation": "Linear",
        "values": [27.58, 27.47, 27.41, 27.15,…]
      },
      "RH_MEAN": {
        "type": "TDouble",
        "form": "",
        "description": "Relative humidity",
        "interpolation": "Linear",
        "values": [79.38, 81.23, 80.71, 82.44,….]
      },
```

```
      "BARO_PRES_MEAN": {
        "type": "TDouble",
        "form": "",
        "description": "Air pressure",
        "interpolation": "Linear",
        "values": [1019.44, 1019.65, 1019.73, 1019.7,….]
      },
      "TEMP_SBE37_MEAN": {
        "type": "TDouble",
        "form": "",
        "description": "Seawater temperature",
        "interpolation": "Linear",
        "values": [28.3778, 28.4097, 28.4121, 28.3226,….]
      },
      "WIND_FROM_MEAN": {
        "type": "TDouble",
        "form": "http://mmisw.org/ont/cf/parameter/wind_from_direction",
        "description": "Wind Direction",
        "interpolation": "Linear",
        "values": [79.9, 76.7, 95.9, 115.8,….]
      },
      "WIND_SPEED_MEAN": {
        "type": "TDouble",
        "form": "http://mmisw.org/ont/cf/parameter/wind_speed",
        "description": "Wind speed nominal_sampling_schedule: 60s on, 240s off,
centered at :00",
        "interpolation": "Linear",
        "values": [4.67, 5.09, 5.45, 3.74,….]
      },
      "SAL_SBE37_MEAN": {
        "type": "TDouble",
        "form": "",
        "description": "Seawater salinity",
        "interpolation": "Linear",
        "values": [36.5952, 36.5911, 36.5926, 36.6004,….]
      },
      "WATER_CURRENT_SPEED_MEAN": {
        "type": "TDouble",
        "form": "",
        "description": "Speed of water current",
        "interpolation": "Linear",
        "values": [0.4, 0.388, 0.282, 0.269,…]
      },
      "WATER_CURRENT_DIRECTION_MEAN": {
        "type": "TDouble",
        "form": "",
        "description": "Direction of water current",
        "interpolation": "Linear",
        "values": [158.7, 159.3, 160.7, 190.6,….]
      },
      "WAVE_DOMINANT_PERIOD": {
        "type": "TDouble",
        "form": "",
        "description": "Dominant wave period",
        "interpolation": "Linear",
        "values": [9.85, 9.85, 9.14, 9.14,….]
      },
      "WAVE_SIGNIFICANT_HEIGHT": {
        "type": "TDouble",
        "form": "",
        "description": "Significant wave height",
        "interpolation": "Linear",
        "values": [1.457, 1.456, 1.574, 1.442,….]
```

```
      }
    }
  ],
  "links": [
    {
      "rel": "next",
      "href": "https://api.georobotix.io/ogc/t18/mfapi/collections/saildrones/
items/dpgba35clbhau/tproperties?offset=100",
      "type": "auto"
    }
  ]
}
```

Each response contains a maximum of 100 observations and if a next node appears between the “links” then the search for the next saildrone item observations continues with the url specified in the “href” node.

The server also provides another type of saildrone item, i.e., with different properties, and precisely the following type:

```
"SOG": {
        "type": "TDouble",
        "form": "http://sensorml.com/ont/swe/property/SpeedOverGround",
        "description": "Vessel speed relative to ground",
        "interpolation": "Linear",
        "values": [0.023, 0.006, 0.907, 0.651,…]
      },
      "COG": {
        "type": "TDouble",
        "form": "http://sensorml.com/ont/swe/property/CourseOverGround",
        "description": "Vessel travel direction relative to ground",
        "interpolation": "Linear",
        "values": [143.6, 329.9, 93.9, 4.9,…]
      },
      "HDG": {
        "type": "TDouble",
        "form": "http://sensorml.com/ont/swe/property/TrueHeading",
        "description": "Vessel heading direction relative to true north,
measured clockwise",
        "interpolation": "Linear",
        "values": [-1.0, -1.0, -1.0, -1.0,…]
      },
      "HDG_WING": {
        "type": "TDouble",
        "form": "http://sensorml.com/ont/swe/property/WingHeading",
        "description": "Direction of wing relative to true north, measured
clockwise",
        "interpolation": "Linear",
        "values": [256.7, 18.4, 333.6, 260.8,…]
      },
      "WING_ANGLE": {
        "type": "TDouble",
        "form": "http://sensorml.com/ont/swe/property/WingAngle",
        "description": "Angle of the wing",
        "interpolation": "Linear",
        "values": [-84.7, 23.3, -131.2, -122.7,…]
      },
      "PITCH": {
        "type": "TDouble",
        "form": " http://sensorml.com/ont/swe/property/PitchAngle",
        "description": "Vessel pitch angle w.r.t to the local horizontal
plane",
```

```
      "interpolation": "Linear",
      "values": [-1.0, -1.0, -1.0, -1.0,…]
    },
    "ROLL": {
      "type": "TDouble",
      "form": "http://sensorml.com/ont/swe/property/RollAngle",
      "description": "Vessel roll angle w.r.t to the local horizontal plane",
      "interpolation": "Linear",
      "values": [-1.0, -1.0, -1.0, -1.0,…]
    },
    "STATUS": {
      "type": "TText",
      "form": "http://sensorml.com/ont/ais/property/NavigationStatus",
      "description": "Vessel Status",
      "interpolation": "Linear",
      "values": ["-1", "-1", "-1", "-1",…]
    }
```


The client allows recognition of both types of saildrone items. Therefore, it allows the correct management and visualization of data, geometries, and properties of different types.

9.4. Retrieving Satellite Imagery (Via Web)

For satellite images, the web portal made available by NASA and Nasa EarthData was accessed: https://search.earthdata.nasa.gov/search
There are two ways of searching and downloading satellite imagery: using direct web access to the portal or using a Rest API made available by the portal.+ In either case, the search for the dataset is done through a filter by selecting some options such as the following.

- Rectangle Area as AOI or single point
- Time interval or single datetime
- Instrument
- Data Format

The result of the search will be a collection of datasets. The figure below shows the result of the web search for the following filter.

- AOI: SW 15.60009,-99.98438, NE 33.31329,-76.00781
- Start Date: 13 September 2016 and End Date: 21 September 2016
- Instrument MODIS HDF Data Format

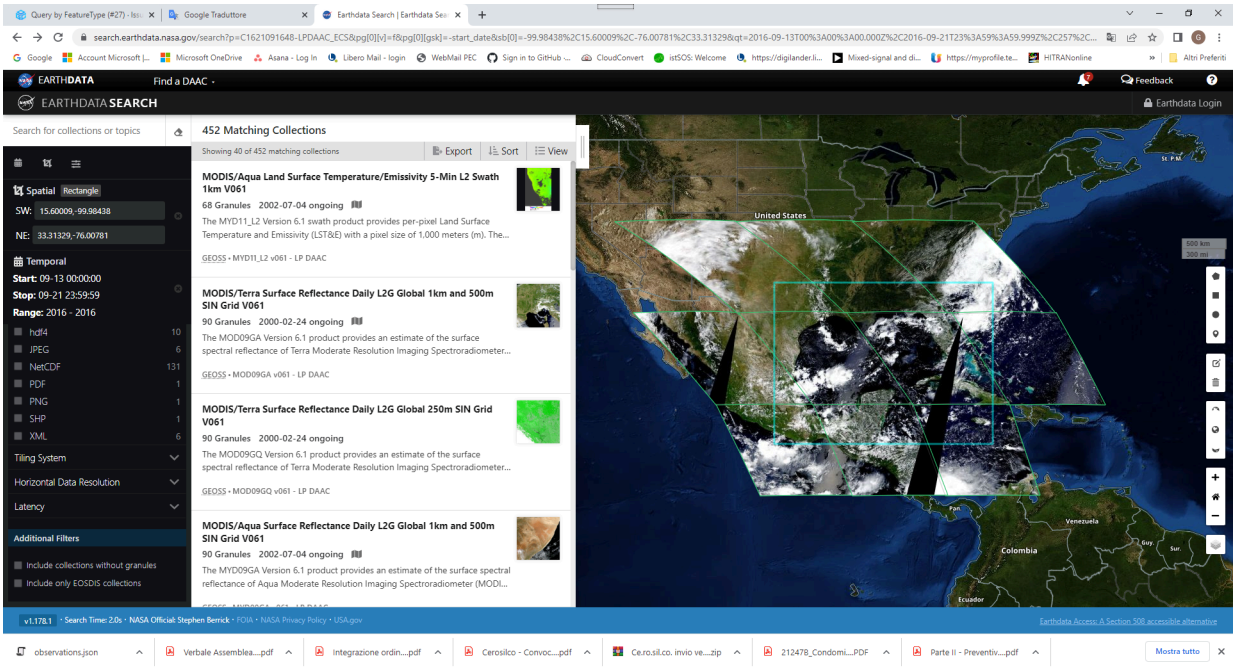


Figure 24 — Web Search Results

The next step is the selection of the collection. In this example, Superelectric chose the "MODIS/Terra Surface Reflectance Daily L2G Global 250m SIN Grid V061" collection.

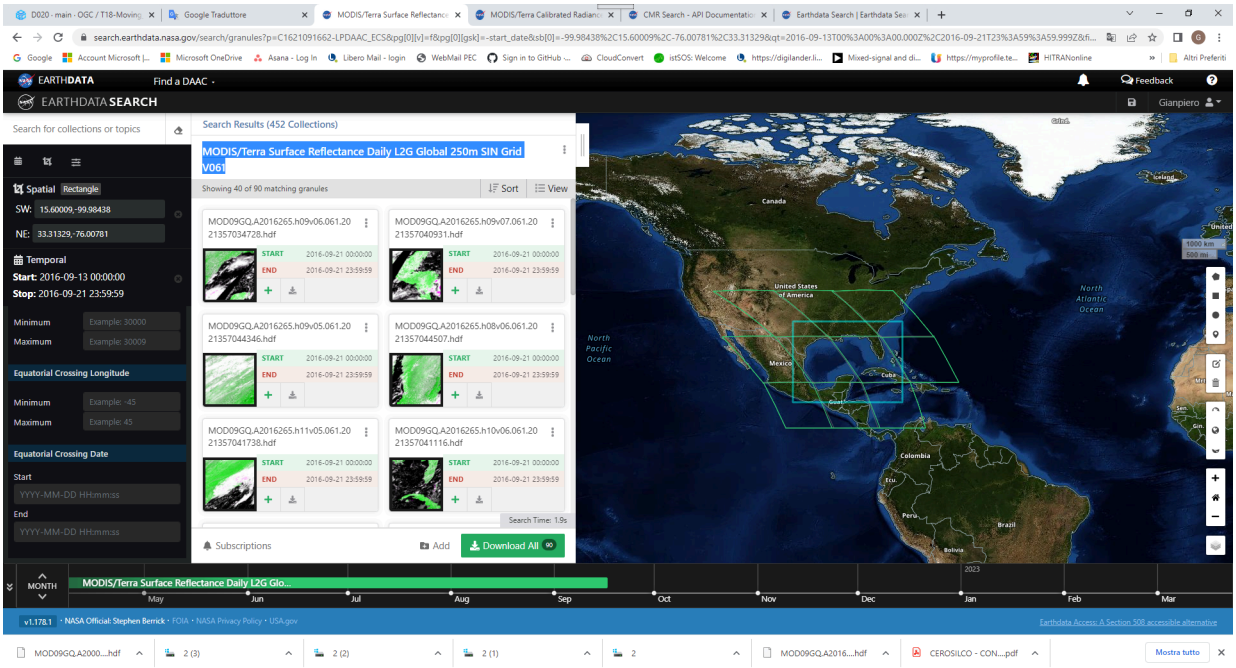


Figure 25 — Selection of the Collection

Next go to the downloading page.

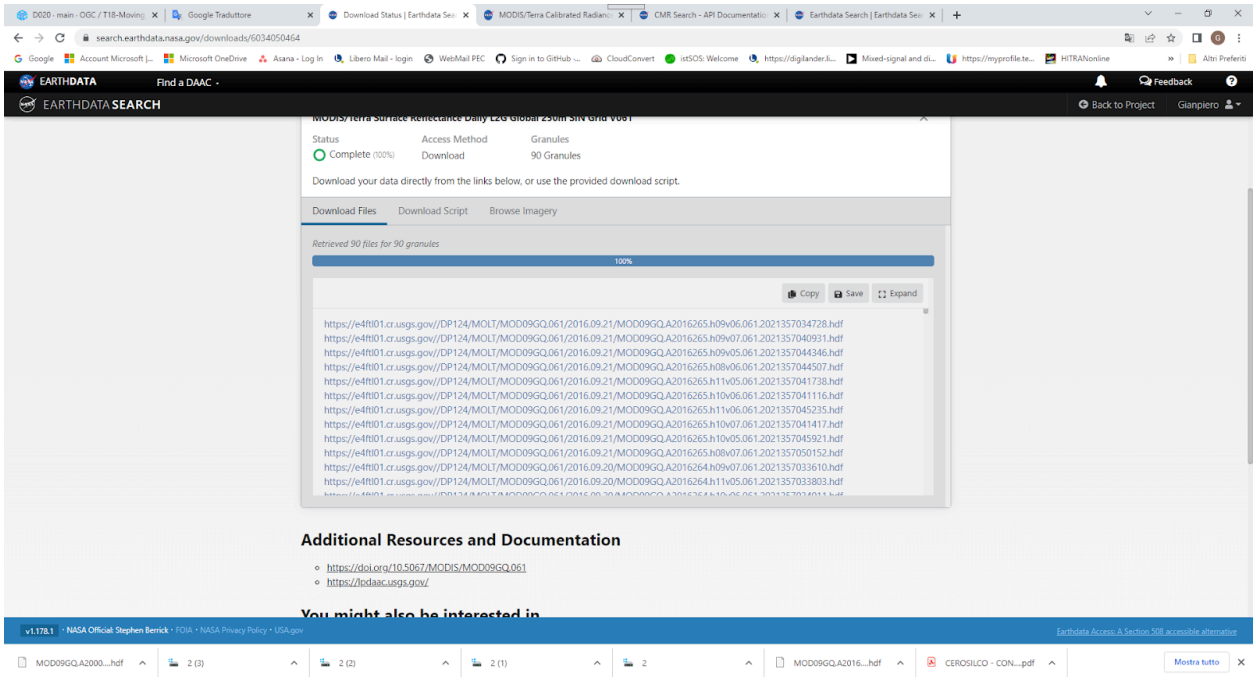


Figure 26 — Download

Then proceed with the download by single file or by means of an appropriate batch file script downloadable from this same web page.

9.5. Retrieving Satellite Imagery (Via CMR API)

As mentioned before, searching and downloading a collection using the EarthData Common Metadata Repository "CMR" Search API is possible. https://cmr.earthdata.nasa.gov/search/site/docs/search/api.html

The search for collections is done first with the filter options, with the following query: https://cmr.earthdata.nasa.gov/search/collections?page_size=10&temporal=2016-09-13T06:00:00Z,2016-09-21T06:00:00Z&bounding_box=-99,15,-76,33&instrument=MODIS This query is returned in xml format:

```
<?xml version="1.0" encoding="UTF-8"?>
<results>
    <hits>737</hits>
    <took>97</took>
    <references>
        <reference>
            <name>GHRSST Level 2P Global Sea Surface Skin Temperature from
the Moderate Resolution Imaging Spectroradiometer (MODIS) on the NASA Aqua
satellite (GDS2)</name>
            <id>C1940473819-POCLOUD</id>
            <location>https://cmr.earthdata.nasa.gov:443/search/concepts/
C1940473819-POCLOUD/41</location&#x3e;
            <revision-id>41</revision-id>
            <score>2.58720015</score>
```

```
        </reference>
        <reference>
            <name>GHRSST Level 2P Global Sea Surface Skin Temperature from
the Moderate Resolution Imaging Spectroradiometer (MODIS) on the NASA Terra
satellite (GDS2)</name>
            <id>C1940475563-POCLOUD</id>
            <location>https://cmr.earthdata.nasa.gov:443/search/concepts/
C1940475563-POCLOUD/46</location&#x3e;
            <revision-id>46</revision-id>
            <score>2.58720015</score>
        </reference>
        <reference>
            <name>MODIS/Aqua Land Surface Temperature/Emissivity 5-Min L2
Swath 1km V061</name>
            <id>C1621389548-LPDAAC_ECS</id>
            <location>https://cmr.earthdata.nasa.gov:443/search/concepts/
C1621389548-LPDAAC_ECS/18</location&#x3e;
            <revision-id>18</revision-id>
            <score>1.32</score>
        </reference>
        <reference>
            <name>MODIS/Terra Surface Reflectance Daily L2G Global 1km and
500m SIN Grid V061</name>
            <id>C1621091648-LPDAAC_ECS</id>
            <location>https://cmr.earthdata.nasa.gov:443/search/concepts/
C1621091648-LPDAAC_ECS/21</location&#x3e;
            <revision-id>21</revision-id>
            <score>1.32</score>
        </reference>
        <reference>
            <name>MODIS/Terra Surface Reflectance Daily L2G Global 250m SIN
Grid V061</name>
            <id>C1621091662-LPDAAC_ECS</id>
            <location>https://cmr.earthdata.nasa.gov:443/search/concepts/
C1621091662-LPDAAC_ECS/21</location&#x3e;
            <revision-id>21</revision-id>
            <score>1.32</score>
        </reference>
        <reference>
            <name>GHRSST Level 4 MUR Global Foundation Sea Surface Temperature
Analysis (v4.1)</name>
            <id>C1664741463-PODAAC</id>
            <location>https://cmr.earthdata.nasa.gov:443/search/concepts/
C1664741463-PODAAC/90</location&#x3e;
            <revision-id>90</revision-id>
            <score>0.6</score>
        </reference>
        <reference>
            <name>GHRSST Level 4 MUR Global Foundation Sea Surface Temperature
Analysis (v4.1)</name>
            <id>C1996881146-POCLOUD</id>
            <location>https://cmr.earthdata.nasa.gov:443/search/concepts/
C1996881146-POCLOUD/33</location&#x3e;
            <revision-id>33</revision-id>
            <score>0.6</score>
        </reference>
        <reference>
            <name>MODIS/Aqua Surface Reflectance Daily L2G Global 1km and 500m
SIN Grid V061</name>
            <id>C1621389350-LPDAAC_ECS</id>
            <location>https://cmr.earthdata.nasa.gov:443/search/concepts/
C1621389350-LPDAAC_ECS/19</location&#x3e;
            <revision-id>19</revision-id>
```

```
            <score>1.32</score>
        </reference>
        <reference>
            <name>MODIS/Aqua Surface Reflectance Daily L2G Global 250m SIN
Grid V061</name>
            <id>C1621389411-LPDAAC_ECS</id>
            <location>https://cmr.earthdata.nasa.gov:443/search/concepts/
C1621389411-LPDAAC_ECS/20</location&#x3e;
            <revision-id>20</revision-id>
            <score>1.32</score>
        </reference>
        <reference>
            <name>MODIS/Aqua Terra Thermal Anomalies/Fire locations 1km FIRMS
V006 NRT (Vector data)</name>
            <id>C1227495594-LANCEMODIS</id>
            <location>https://cmr.earthdata.nasa.gov:443/search/concepts/
C1227495594-LANCEMODIS/7</location&#x3e;
            <revision-id>7</revision-id>
            <score>1.32</score>
        </reference>
    </references>
</results>
```

At this point through the **<reference>** the desired collection is chosen and then a new search is made for "granule" images using the reference ID, for example:

```
    <reference>
            <name>MODIS/Terra Surface Reflectance Daily L2G Global 250m SIN
Grid V061</name>
            <id>C1621091662-LPDAAC_ECS</id>
            <location>https://cmr.earthdata.nasa.gov:443/search/concepts/
C1621091662-LPDAAC_ECS/21</location>
            <revision-id>21</revision-id>
            <score>1.32</score>
    </reference>
```

The next query is: https://cmr.earthdata.nasa.gov/search/granules?page_size=100&temporal=2016-09-13T06:00:00Z,2016-09-21T06:00:00Z&bounding_box=-99,15,-76,33&instrument=MODIS&collection_concept_id=C1621091662-LPDAAC_ECS This query is returned in xml format:

```
<?xml version="1.0" encoding="UTF-8"?>
<results>
    <hits>90</hits>
    <took>330</took>
    <references>
        <reference>
            <name>SC:MOD09GQ.061:2524121072</name>
            <id>G2188754129-LPDAAC_ECS</id>
            <location>https://cmr.earthdata.nasa.gov:443/search/concepts/
G2188754129-LPDAAC_ECS/1</location&#x3e;
            <revision-id>1</revision-id>
        </reference>
        <reference>
            <name>SC:MOD09GQ.061:2524124159</name>
            <id>G2188757380-LPDAAC_ECS</id>
            <location>https://cmr.earthdata.nasa.gov:443/search/concepts/
G2188757380-LPDAAC_ECS/1</location&#x3e;
            <revision-id>1</revision-id>
        </reference>
        <reference>
            <name>SC:MOD09GQ.061:2524124590</name>
```

```
            <id>G2188757572-LPDAAC_ECS</id>
            <location>https://cmr.earthdata.nasa.gov:443/search/concepts/
G2188757572-LPDAAC_ECS/1</location&#x3e;
            <revision-id>1</revision-id>
        </reference>
        <reference>
            <name>SC:MOD09GQ.061:2524126968</name>
            <id>G2188759616-LPDAAC_ECS</id>
            <location>https://cmr.earthdata.nasa.gov:443/search/concepts/
G2188759616-LPDAAC_ECS/2</location&#x3e;
            <revision-id>2</revision-id>
        </reference>
        <reference>
            <name>SC:MOD09GQ.061:2524128104</name>
            <id>G2188763289-LPDAAC_ECS</id>
            <location>https://cmr.earthdata.nasa.gov:443/search/concepts/
G2188763289-LPDAAC_ECS/1</location&#x3e;
            <revision-id>1</revision-id>
        </reference>
        <reference>
            <name>SC:MOD09GQ.061:2524129810</name>
            <id>G2188765355-LPDAAC_ECS</id>
            <location>https://cmr.earthdata.nasa.gov:443/search/concepts/
G2188765355-LPDAAC_ECS/1</location&#x3e;
            <revision-id>1</revision-id>
        </reference>
        <reference>
            <name>SC:MOD09GQ.061:2524129594</name>
            <id>G2188765373-LPDAAC_ECS</id>
            <location>https://cmr.earthdata.nasa.gov:443/search/concepts/
G2188765373-LPDAAC_ECS/1</location&#x3e;
            <revision-id>1</revision-id>
        </reference>
        <reference>
            <name>SC:MOD09GQ.061:2524132200</name>
            <id>G2188768658-LPDAAC_ECS</id>
            <location>https://cmr.earthdata.nasa.gov:443/search/concepts/
G2188768658-LPDAAC_ECS/1</location&#x3e;
            <revision-id>1</revision-id>
        </reference>
        <reference>
            <name>SC:MOD09GQ.061:2524131499</name>
            <id>G2188769093-LPDAAC_ECS</id>
            <location>https://cmr.earthdata.nasa.gov:443/search/concepts/
G2188769093-LPDAAC_ECS/1</location&#x3e;
            <revision-id>1</revision-id>
        </reference>
        <reference>
            <name>SC:MOD09GQ.061:2524132735</name>
            <id>G2188771234-LPDAAC_ECS</id>
            <location>https://cmr.earthdata.nasa.gov:443/search/concepts/
G2188771234-LPDAAC_ECS/1</location&#x3e;
            <revision-id>1</revision-id>
        </reference>
    </references>
</results>
```

The results show that there are 90 granules (HDF file format). By repeating the queries for each of the granules through the **<location>** element, returning the location for the download is possible. For example from query: https://cmr.earthdata.nasa.gov:443/search/concepts/G2188754129-LPDAAC_ECS/1 return its metadata in xml format:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Granule>
    <GranuleUR>SC:MOD09GQ.061:2524121072</GranuleUR>
    <InsertTime>2021-12-22T12:34:01.610Z</InsertTime>
    <LastUpdate>2021-12-22T12:34:54.489Z</LastUpdate>
    <Collection>
        <DataSetId>MODIS/Terra Surface Reflectance Daily L2G Global 250m SIN
Grid V061</DataSetId>
    </Collection>
    <DataGranule>
        <SizeMBDataGranule>62.4378</SizeMBDataGranule>
        <ReprocessingPlanned>further update is anticipated</
ReprocessingPlanned>
        <ReprocessingActual>reprocessed</ReprocessingActual>
        <ProducerGranuleId>MOD09GQ.A2016257.h10v06.061.2021356170001.hdf</
ProducerGranuleId>
        <DayNightFlag>DAY</DayNightFlag>
        <ProductionDateTime>2021-12-22T17:00:01.000Z</ProductionDateTime>
        <LocalVersionId>6.0.9</LocalVersionId>
    </DataGranule>
    <PGEVersionClass>
        <PGEVersion>6.1.8</PGEVersion>
    </PGEVersionClass>
    <Temporal>
        <RangeDateTime>
            <BeginningDateTime>2016-09-13T00:00:00.000000Z</BeginningDateTime>
            <EndingDateTime>2016-09-13T23:59:59.000000Z</EndingDateTime>
        </RangeDateTime>
    </Temporal>
    <Spatial>
        <HorizontalSpatialDomain>
            <Geometry>
                <GPolygon>
                    <Boundary>
                        <Point>
                            <PointLongitude>-85.1199000270707</PointLongitude>
                            <PointLatitude>19.8972839682412</PointLatitude>
                        </Point>
                        <Point>
                            <PointLongitude>-92.3760430595367</PointLongitude>
                            <PointLatitude>29.9999999973059</PointLatitude>
                        </Point>
                        <Point>
                            <PointLongitude>-80.5352459184066</PointLongitude>
                            <PointLatitude>30.0383760061888</PointLatitude>
                        </Point>
                        <Point>
                            <PointLongitude>-74.2129537334706</PointLongitude>
                            <PointLatitude>19.9339184091026</PointLatitude>
                        </Point>
                    </Boundary>
                </GPolygon>
            </Geometry>
        </HorizontalSpatialDomain>
    </Spatial>
    <OrbitCalculatedSpatialDomains>
        <OrbitCalculatedSpatialDomain>
            <OrbitNumber>89039</OrbitNumber>
            <EquatorCrossingLongitude>-72.2002041432384</
EquatorCrossingLongitude>
            <EquatorCrossingDateTime>2016-09-13T15:19:18.022519Z</
EquatorCrossingDateTime>
        </OrbitCalculatedSpatialDomain>
```

```
        <OrbitCalculatedSpatialDomain>
            <OrbitNumber>89040</OrbitNumber>
            <EquatorCrossingLongitude>-96.9218604453134</
EquatorCrossingLongitude>
            <EquatorCrossingDateTime>2016-09-13T16:58:11.120432Z</
EquatorCrossingDateTime>
        </OrbitCalculatedSpatialDomain>
    </OrbitCalculatedSpatialDomains>
    <MeasuredParameters>
        <MeasuredParameter>
            <ParameterName>MOD09G</ParameterName>
            <QAStats>
                <QAPercentMissingData>9</QAPercentMissingData>
                <QAPercentOutOfBoundsData>0</QAPercentOutOfBoundsData>
                <QAPercentInterpolatedData>0</QAPercentInterpolatedData>
            </QAStats>
            <QAFlags>
                <AutomaticQualityFlag>Passed</AutomaticQualityFlag>
                <AutomaticQualityFlagExplanation>No automatic quality
assessment is performed in the PGE</AutomaticQualityFlagExplanation>
                <OperationalQualityFlag>Passed</OperationalQualityFlag>
                <OperationalQualityFlagExplanation>Passed</OperationalQualityFl
agExplanation>
                <ScienceQualityFlag>Not Investigated</ScienceQualityFlag>
                <ScienceQualityFlagExplanation>See https://landweb.modaps.
eosdis.nasa.gov/cgi-bin/QA_WWW/qaFlagPage.cgi?sat=terra&ver=C6 for the product
Science Quality status.</ScienceQualityFlagExplanation>
            </QAFlags>
        </MeasuredParameter>
    </MeasuredParameters>
    <Platforms>
        <Platform>
            <ShortName>Terra</ShortName>
            <Instruments>
                <Instrument>
                    <ShortName>MODIS</ShortName>
                </Instrument>
            </Instruments>
        </Platform>
    </Platforms>
    <Campaigns>
        <Campaign>
            <ShortName>TERRA</ShortName>
        </Campaign>
    </Campaigns>
    <AdditionalAttributes>
        <AdditionalAttribute>
            <Name>QAPERCENTGOODQUALITY</Name>
            <Values>
                <Value>78</Value>
            </Values>
        </AdditionalAttribute>
        <AdditionalAttribute>
            <Name>QAPERCENTOTHERQUALITY</Name>
            <Values>
                <Value>13</Value>
            </Values>
        </AdditionalAttribute>
        <AdditionalAttribute>
            <Name>QAPERCENTNOTPRODUCEDCLOUD</Name>
            <Values>
                <Value>0</Value>
            </Values>
```

```
        </AdditionalAttribute>
        <AdditionalAttribute>
            <Name>QAPERCENTNOTPRODUCEDOTHER</Name>
            <Values>
                <Value>10</Value>
            </Values>
        </AdditionalAttribute>
        <AdditionalAttribute>
            <Name>HORIZONTALTILENUMBER</Name>
            <Values>
                <Value>10</Value>
            </Values>
        </AdditionalAttribute>
        <AdditionalAttribute>
            <Name>VERTICALTILENUMBER</Name>
            <Values>
                <Value>6</Value>
            </Values>
        </AdditionalAttribute>
        <AdditionalAttribute>
            <Name>TileID</Name>
            <Values>
                <Value>51010006</Value>
            </Values>
        </AdditionalAttribute>
        <AdditionalAttribute>
            <Name>PROCESSVERSION</Name>
            <Values>
                <Value>6.0.9</Value>
            </Values>
        </AdditionalAttribute>
        <AdditionalAttribute>
            <Name>QAPERCENTPOOROUTPUT250MBAND1</Name>
            <Values>
                <Value>14</Value>
            </Values>
        </AdditionalAttribute>
        <AdditionalAttribute>
            <Name>QAPERCENTPOOROUTPUT250MBAND2</Name>
            <Values>
                <Value>12</Value>
            </Values>
        </AdditionalAttribute>
        <AdditionalAttribute>
            <Name>RESOLUTIONBANDS1AND2</Name>
            <Values>
                <Value>250</Value>
            </Values>
        </AdditionalAttribute>
        <AdditionalAttribute>
            <Name>identifier_product_doi</Name>
            <Values>
                <Value>10.5067/MODIS/MOD09GQ.061</Value>
            </Values>
        </AdditionalAttribute>
        <AdditionalAttribute>
            <Name>identifier_product_doi_authority</Name>
            <Values>
                <Value>http://dx.doi.org</Value&#x3e;
            </Values>
        </AdditionalAttribute>
    </AdditionalAttributes>
    <InputGranules>
```

```
        <InputGranule>MOD09GST.A2016257.h10v06.061.2021356165734.hdf</
InputGranule>
        <InputGranule>MOD09GHK.A2016257.h10v06.061.2021356165825.hdf</
InputGranule>
        <InputGranule>MOD09GQK.A2016257.h10v06.061.2021356165808.hdf</
InputGranule>
        <InputGranule>MODPT1KD.A2016257.h10v06.061.2021356160753.hdf</
InputGranule>
        <InputGranule>MODPTHKM.A2016257.h10v06.061.2021356160754.hdf</
InputGranule>
        <InputGranule>MODPTQKM.A2016257.h10v06.061.2021356160754.hdf</
InputGranule>
        <InputGranule>MODMGGAD.A2016257.h10v06.061.2021356160759.hdf</
InputGranule>
        <InputGranule>MODTBGD.A2016257.h10v06.061.2021356165838.hdf</
InputGranule>
        <InputGranule>MODOCGD.A2016257.h10v06.061.2021356165846.hdf</
InputGranule>
        <InputGranule>MOD10L2G.A2016257.h10v06.061.2021356163549.hdf</
InputGranule>
        <InputGranule>DEM_SN_H.h10v06.006_0.hdf</InputGranule>
        <InputGranule>MCDLCHKM.A2010001.h10v06.051.2014287174241.hdf</
InputGranule>
    </InputGranules>
    <OnlineAccessURLs>
        <OnlineAccessURL>
            <URL>https://e4ftl01.cr.usgs.gov//DP124/MOLT/MOD09GQ.061/
2016.09.13/MOD09GQ.A2016257.h10v06.061.2021356170001.hdf</URL&#x3e;
            <URLDescription>MOD09GQ.A2016257.h10v06.061.2021356170001.hdf.
MimeType: application/x-hdfeos</URLDescription>
            <MimeType>application/x-hdfeos</MimeType>
        </OnlineAccessURL>
    </OnlineAccessURLs>
    <OnlineResources>
        <OnlineResource>
            <URL>https://doi.org/10.5067/MODIS/MOD09GQ.061</URL&#x3e;
            <Description>The Landing Page for this file may be accessed
directly from this link</Description>
            <Type>DOI</Type>
            <MimeType>text/html</MimeType>
        </OnlineResource>
        <OnlineResource>
            <URL>https://e4ftl01.cr.usgs.gov//WORKING/BRWS/Browse.001/
2021.12.22/BROWSE.MOD09GQ.A2016257.h10v06.061.2021356170001.1.jpg</URL&#x3e;
            <Description>This Browse file may be downloaded directly from this
link</Description>
            <Type>BROWSE</Type>
            <MimeType>image/jpeg</MimeType>
        </OnlineResource>
        <OnlineResource>
            <URL>https://e4ftl01.cr.usgs.gov//DP124/MOLT/MOD09GQ.061/
2016.09.13/MOD09GQ.A2016257.h10v06.061.2021356170001.hdf.xml</URL&#x3e;
            <Description>This Metadata file may be downloaded directly from
this link</Description>
            <Type>EXTENDED METADATA</Type>
            <MimeType>text/xml</MimeType>
        </OnlineResource>
    </OnlineResources>
    <Orderable>true</Orderable>
    <DataFormat>HDF-EOS2</DataFormat>
    <Visible>true</Visible>
    <CloudCover>0</CloudCover>
```

```
</Granule>
```

As a last step, the download was performed directly from the url specified by the node: **<OnlineAccessURL>**

Data harmonization

The following figures show the harmonization process for the JSON data for hurricanes and vessels in the format chosen in order to temporarily organize the data so that they can be managed from the Data Fusion and Viewer module.

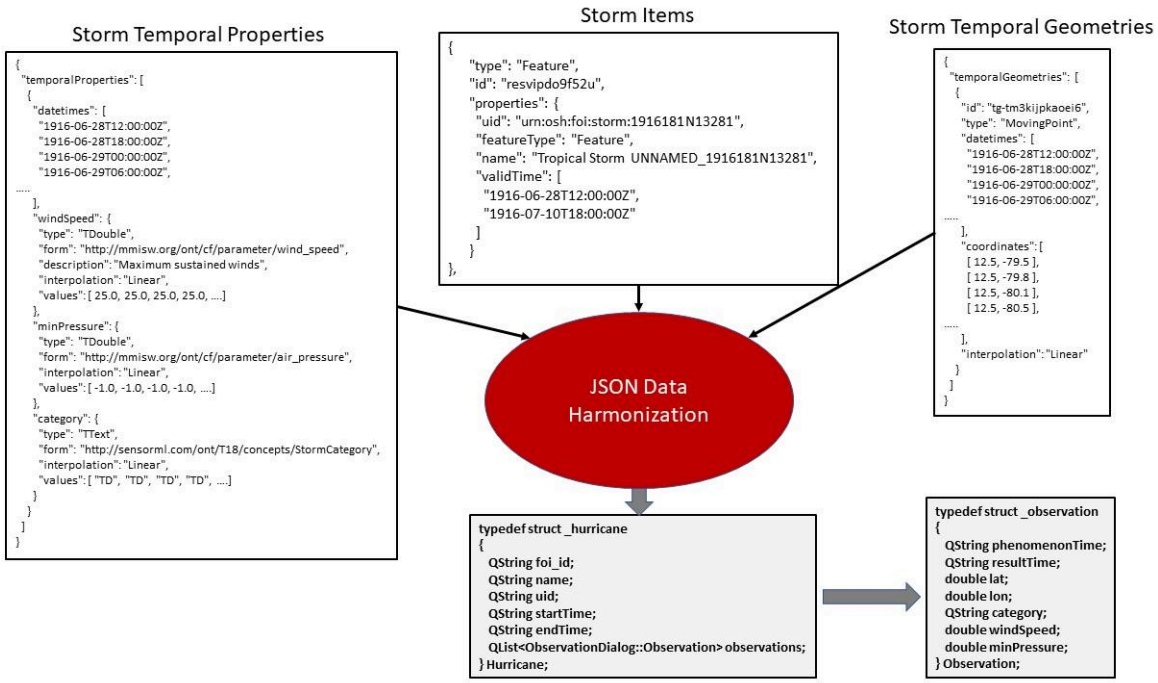


Figure 27 — JSON Data Harmonization

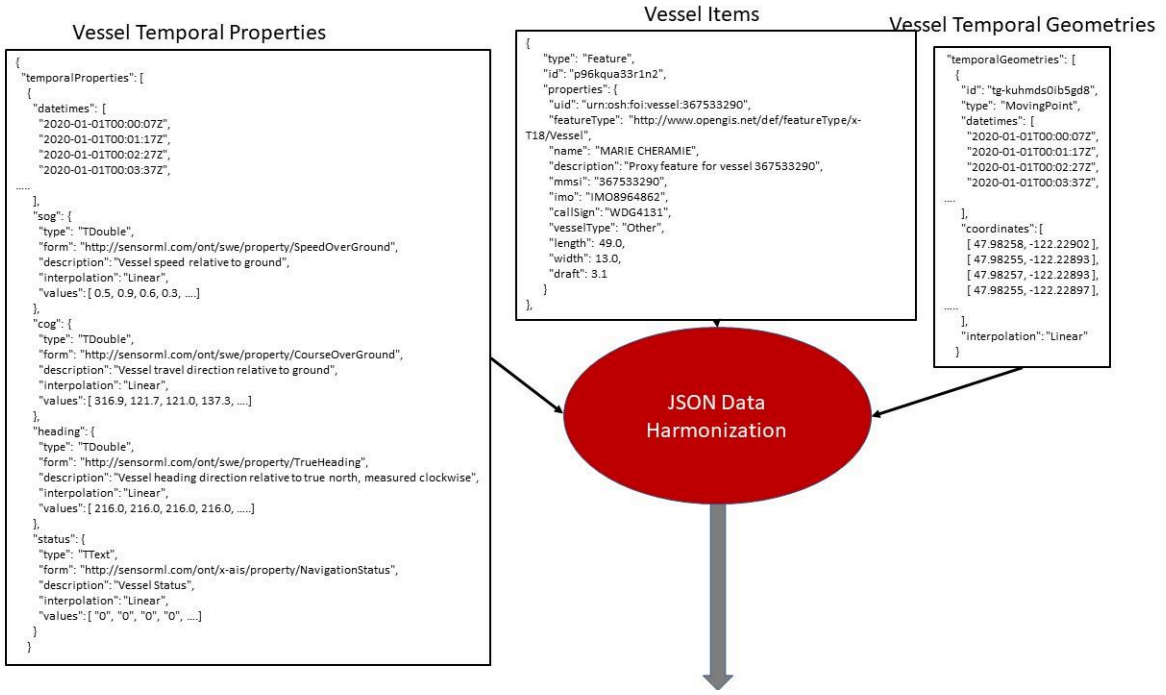


Figure 28 — JSON Data Harmonization Part 2

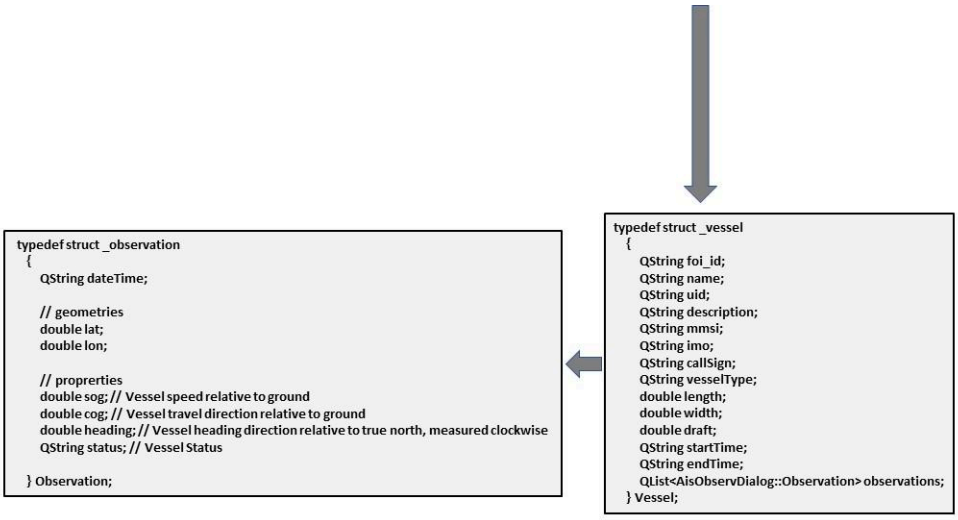


Figure 29 — JSON Data Harmonization Part 3

In the future, having the Sensorhub provide feature collections based on a search for AOI (Area Of Interest) and a datetime would be desirable. As an example as JSON:

```
{
  "type": "FeatureCollection",
  "timeStamp": "1916-06-28T12:00:00Z",
```

```
"numberMatched": 34,
"numberReturned": 10,
"features": [ {
  "type": "Storm",
  "id": "resvipdo9f52u",
  "geometry": {
    "type": "Point",
    "coordinates": [ … ]
  },
  "properties": {
      "uid": "urn:osh:foi:storm:1916181N13281",
      "featureType": "Feature",
      "name": "Tropical Storm UNNAMED_1916181N13281",
      "validTime": [
        "1916-06-28T12:00:00Z",
        "1916-07-10T18:00:00Z"
      ],
    "windSpeed": {
      "type": "TDouble",
      "form": "http://mmisw.org/ont/cf/parameter/wind_speed",
      "description": "Maximum sustained winds",
      "interpolation": "Linear",
      "values": [ 25.0 ]
    },
    "minPressure": {
      "type": "TDouble",
      "form": "http://mmisw.org/ont/cf/parameter/air_pressure",
      "interpolation": "Linear",
      "values": [ -1.0 ]
    },
    "category": {
      "type": "TText",
      "form": "http://sensorml.com/ont/T18/concepts/StormCategory",
      "interpolation": "Linear",
      "values": [ "TD" ]
    }
  }
}, { …
}, {
  "type": "Vessel",
  "id": " p96kqua33r1n2",
  "geometry": {
    "type": "Polygon",
    "coordinates": [ … ]
  },
  "properties": {
      "uid": "urn:osh:foi:vessel:367533290",
      "featureType": "http://www.opengis.net/def/featureType/x-T18/Vessel",
      "name": "MARIE CHERAMIE",
      "validTime": [
        "1916-06-28T12:00:00Z",
        "1916-07-10T18:00:00Z"
      ],
      "description": "Proxy feature for vessel 367533290",
      "mmsi": "367533290",
      "imo": "IMO8964862",
      "callSign": "WDG4131",
      "vesselType": "Other",
      "length": 49.0,
      "width": 13.0,
      "draft": 3.1,
    "sog": {
      "type": "TDouble",
```

```
        "form": "http://sensorml.com/ont/swe/property/SpeedOverGround",
        "description": "Vessel speed relative to ground",
        "interpolation": "Linear",
        "values": [ 0.5 ]
      },
      "cog": {
        "type": "TDouble",
        "form": "http://sensorml.com/ont/swe/property/CourseOverGround",
        "description": "Vessel travel direction relative to ground",
        "interpolation": "Linear",
        "values": [ 316.9 ]
      },
      "heading": {
        "type": "TDouble",
        "form": "http://sensorml.com/ont/swe/property/TrueHeading",
        "description": "Vessel heading direction relative to true north,
measured clockwise",
        "interpolation": "Linear",
        "values": [ 216.0 ]
      },
      "status": {
        "type": "TText",
        "form": "http://sensorml.com/ont/x-ais/property/NavigationStatus",
        "description": "Vessel Status",
        "interpolation": "Linear",
        "values": [ "0" ]
      }
    }
  } ]
}
```

In any case, the client implements a search tool for AOI and time interval in order to identify all the items (feature of interest) present in the scenario, and thus is able to establish the position of each item as a function of time.

The figure below shows the steps necessary to transform the satellite data into GeoTIFF format suitable for the visualization. In particular, the process beginning with layer extraction from an HDF file and conversion to GeoTIFF equalized in 8 bit per pixel, using GDAL tools.

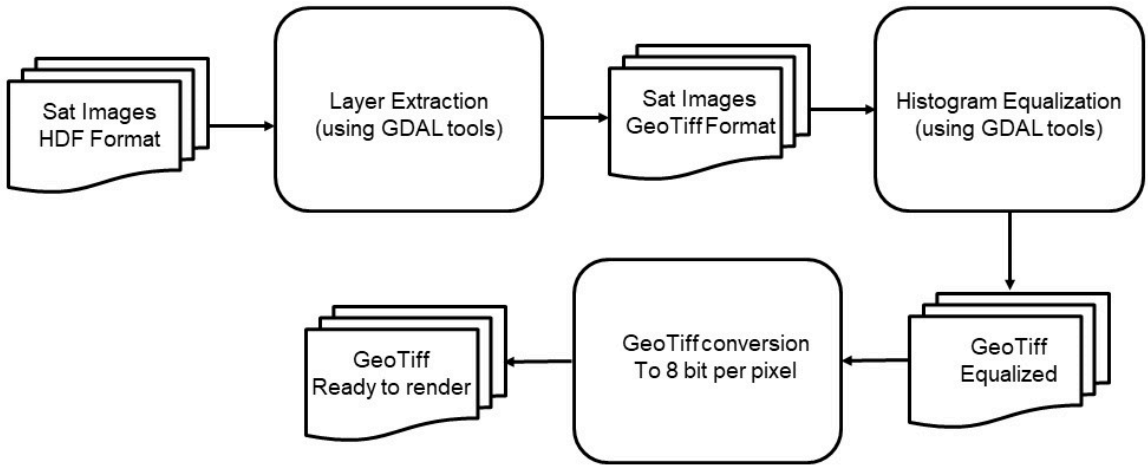


Figure 30 — Steps to Transform Satellite Data into GeoTIFF

9.6. Data fusion

The data fusion process focused on exporting images obtained from the merge of imagery data from different sources and enriching them with the data of one or more targets present in the geographical area defined by the same image saved in the form of JSON metadata. The targets present in a specific geographical area are the features of interest (vessel and storm items) obtained from the Sensor Hub. Example of the exported metadata in JSON:


```
{
  "type": "FeatureCollection",
  "timeStamp": "1916-06-28T12:00:00Z",
  "features": [
    {
 "type": "Feature",
 "geometry": {
  "type": "Point",
  "coordinates": [33.50873, 33.05445]
      },
 "properties": {
  "Vessels Name": "ODYSSEY OF THE SEAS",
  "Event": "Midnight position",
  "Event Content": "At N 33° 03' 16.03\" - E 033° 30' 31.43\"",
  "Area": "EMED",
  "Area Local": "East Mediterranean",
  "Speed": 20.600000381469727,
  "Course": 347,
  "Lat": 33.05445,
  "Lon": 33.50873
  }
    },
    {
 "type": "Feature",
 "geometry": {
  "type": "Point",
  "coordinates" : [ 34.64771, 31.83218 ]
  },
 "properties": {
  "Vessels Name": "LIBERTY",
  "Event": "Night position",
  "Event Content": "At N 31° 49' 55.84\" - E 034° 38' 51.76\"",
  "Area": "EMED",
  "Area Local": "East Mediterranean",
  "Speed": 0,
  "Course": 359,
  "Lat": 31.83218,
  "Lon": 34.64771
  }
    }
    ]
}
```


The temporal properties of the FOI will conform to the data dictionary extracted from a previous activity with EU Satcen, compatible with the “NATO geospatial feature concept” and EU Satcen data dictionary.

9.7. Data Output

The output of the above process was a sequence of orthorectified images (GeoTIFF format) with harmonized resolution and metadata as specified above.

The GUI consisted of two modules: A GIS 3D rendering window for viewing the tracks and a dialog box for searching and managing data.

Some client snapshots are shown below.

GIS 3D Rendering window

The rendering engine is based on osgEarth SDK.

Figure 31 — OSGEarth SDK

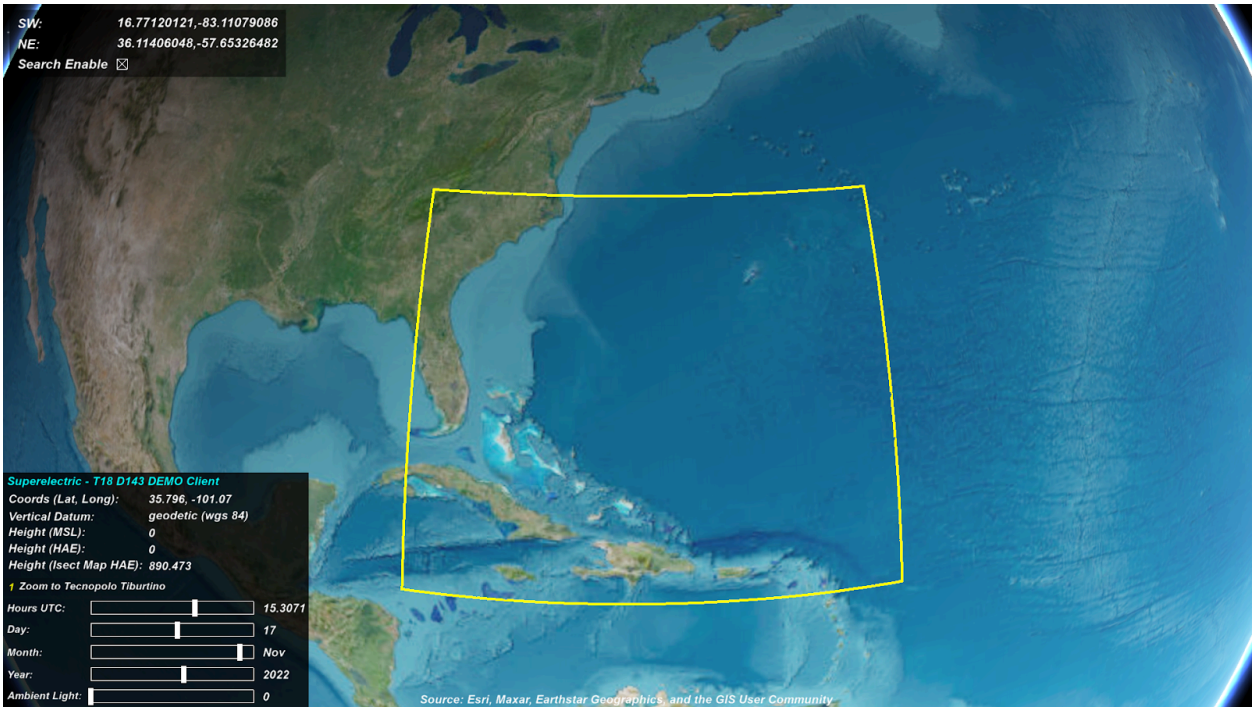


Figure 32 — SuperElectric Demo Client

Dialog Box

The dialog boxes are based on Qt sdk.

The main dialog box was composed of the following three sections.

- Hurricane
- AIS
- Saildrone

Each section provides the necessary actions (queries) to be sent to the server in order to obtain all the necessary data for each type of feature of interest.

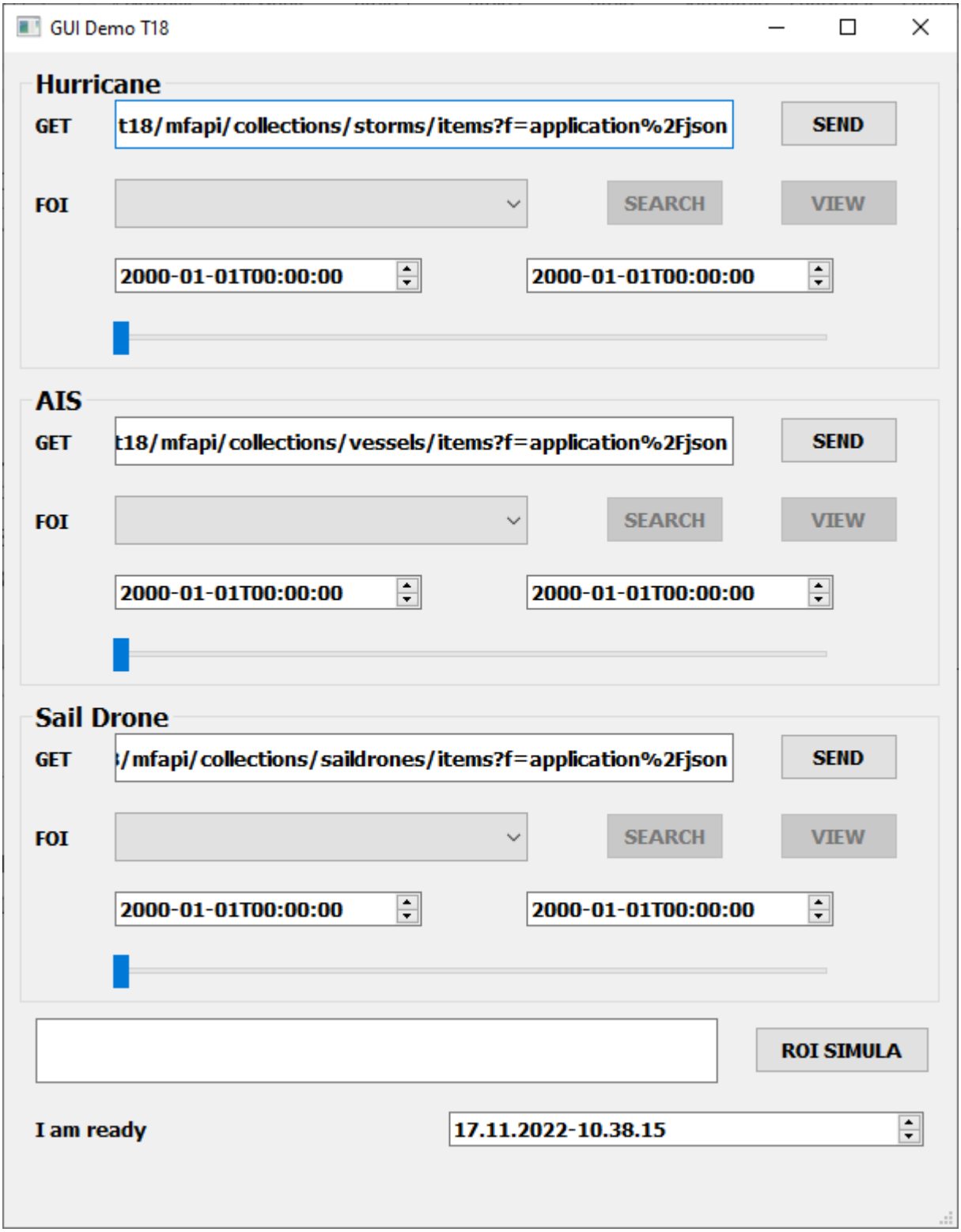


Figure 33 — Dialog Box

The following image shows the search for a particular hurricane and displays the observations data.

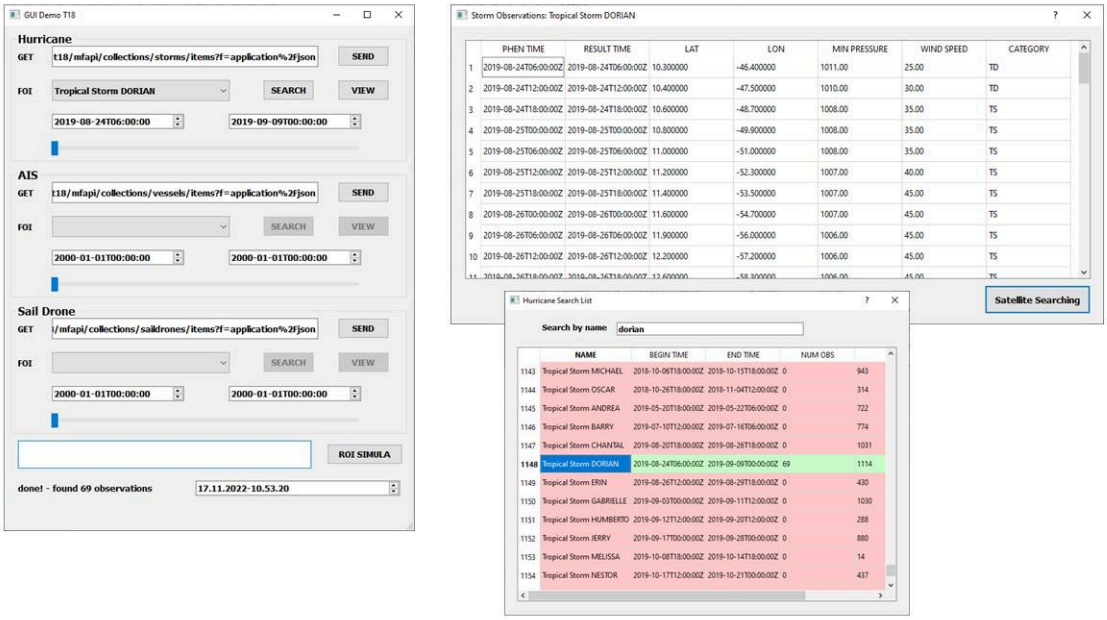


Figure 34 — Demo Client Search

Similarly, the following images represent searches for a particular vessel and saildrone.

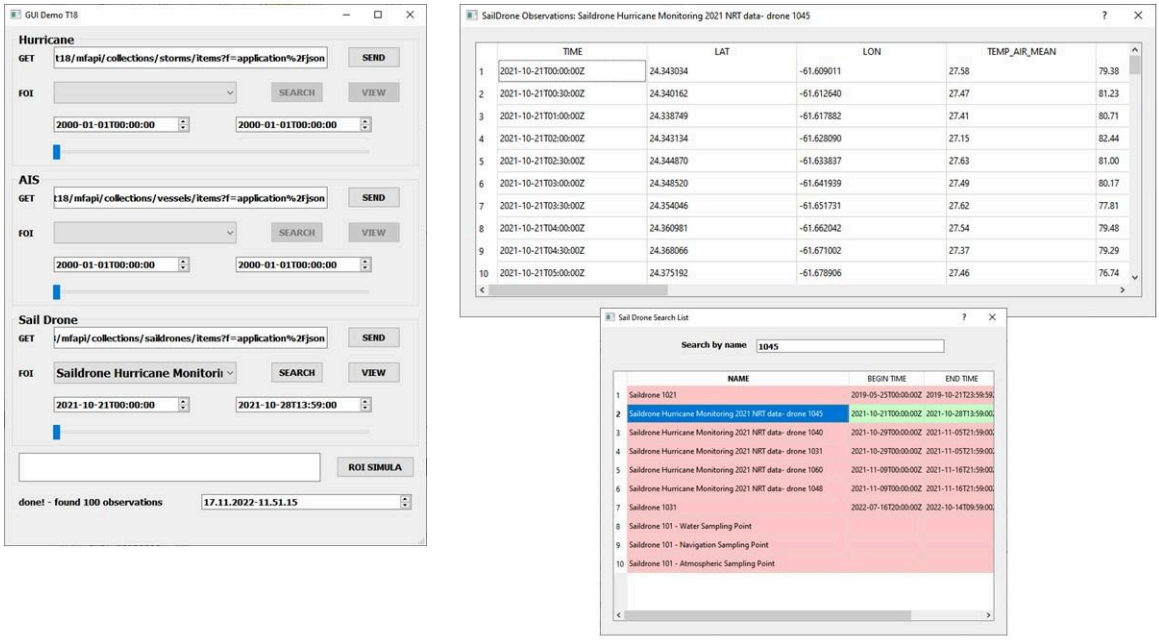


Figure 35 — Demo Client Search

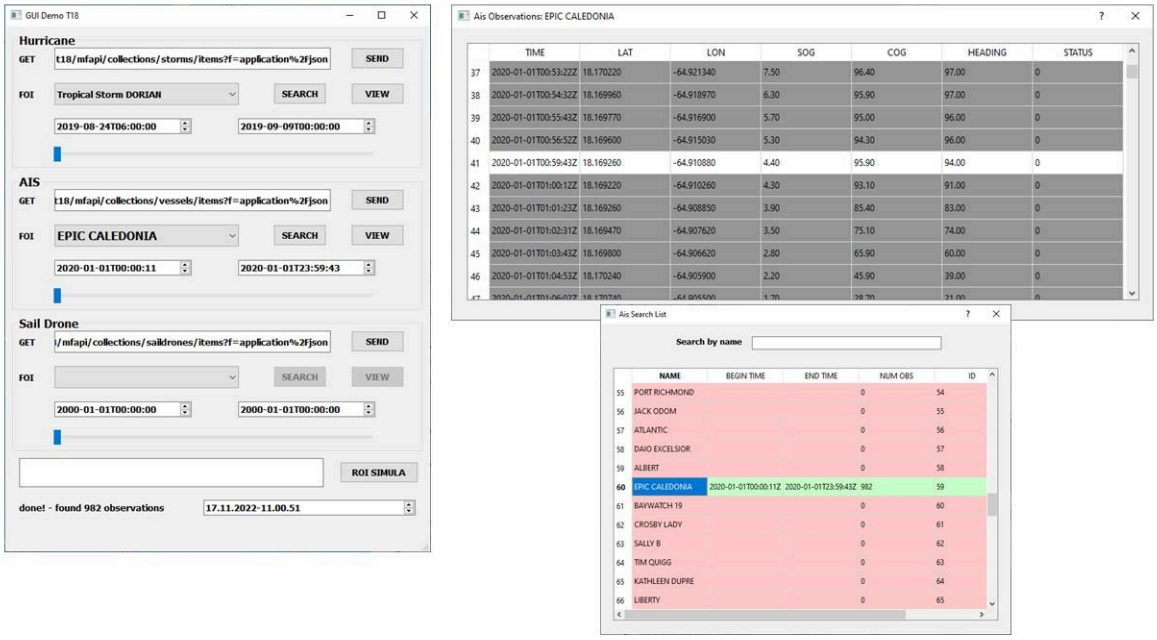


Figure 36 — Demo Client Search

In particular for AIS data, the client implemented a decimation to avoid managing a large number of observations.

Tracks Rendering

The following images represent some item tracks.

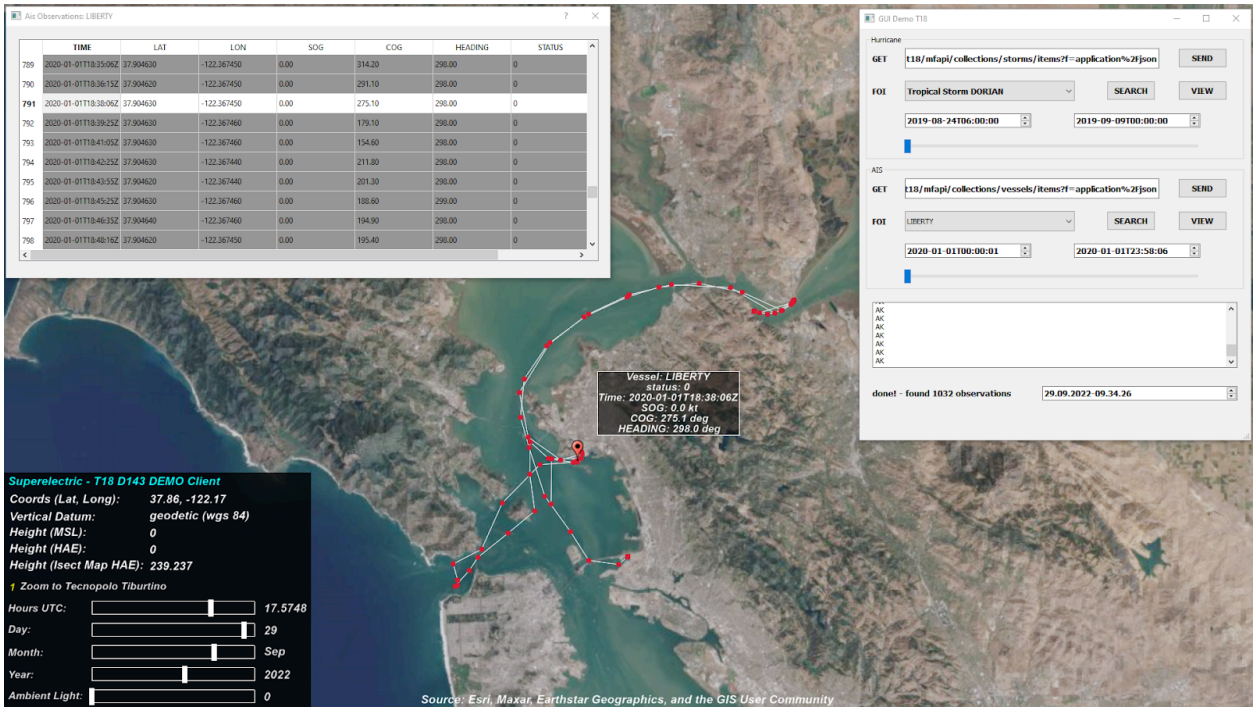


Figure 37 — Demo Client

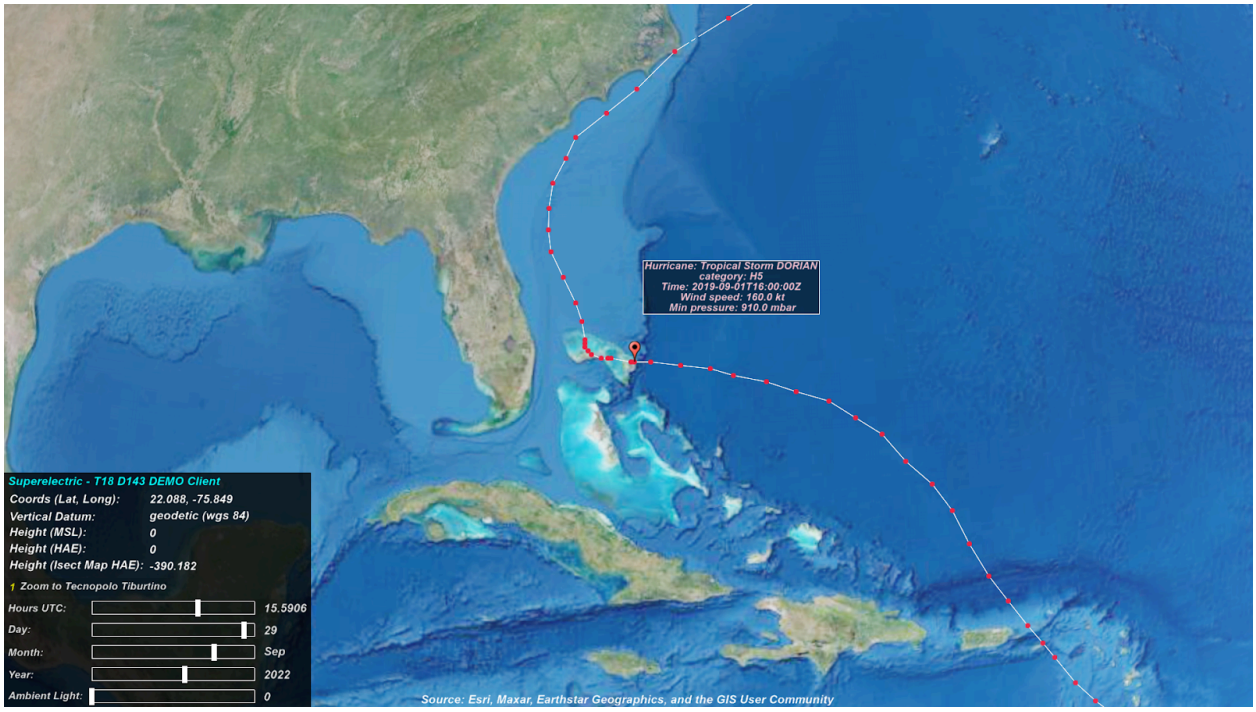


Figure 38 — Demo Client2

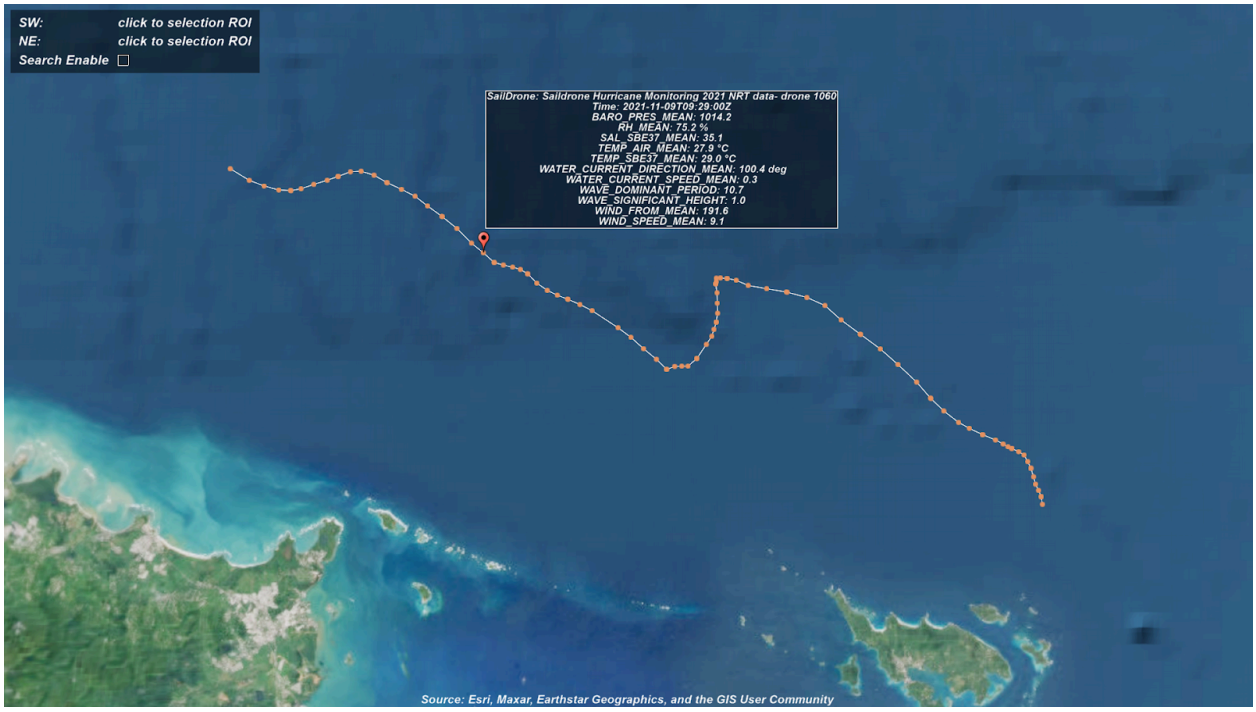


Figure 39 — Demo Client3

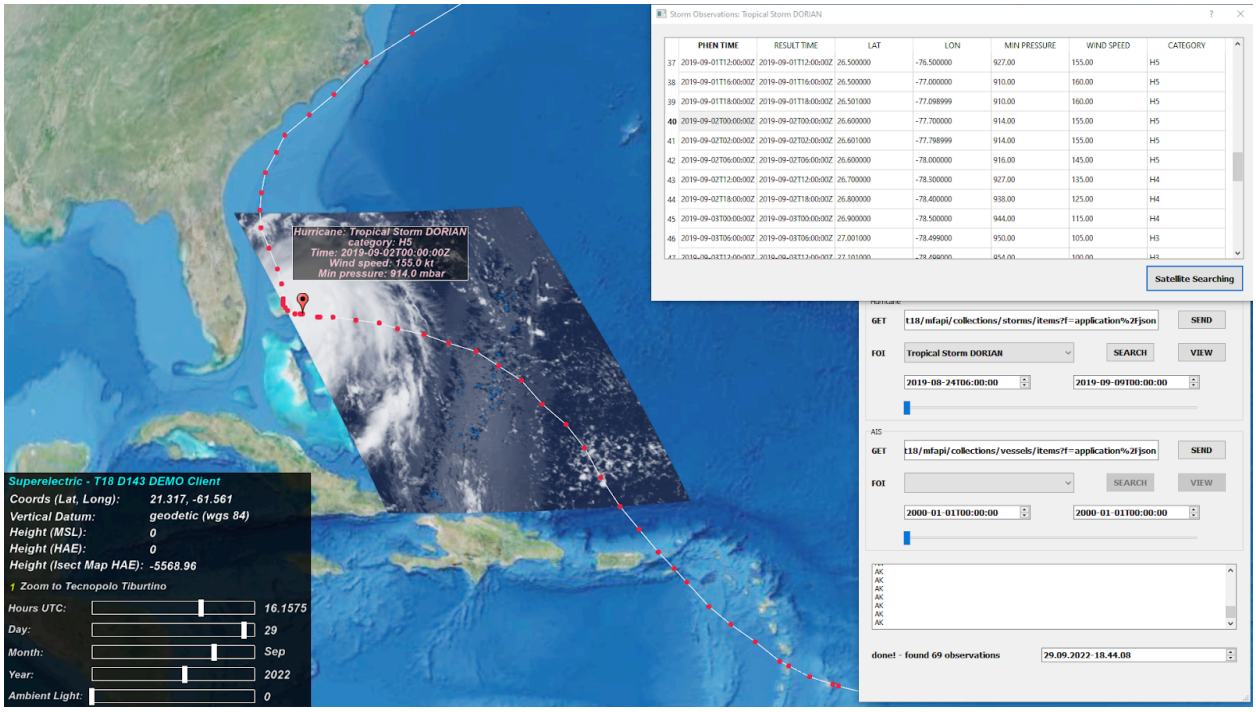


Figure 40 — Demo Client4

Area of Interest and Time Interval Searching Stage

The following image describes the searching stage in progress.

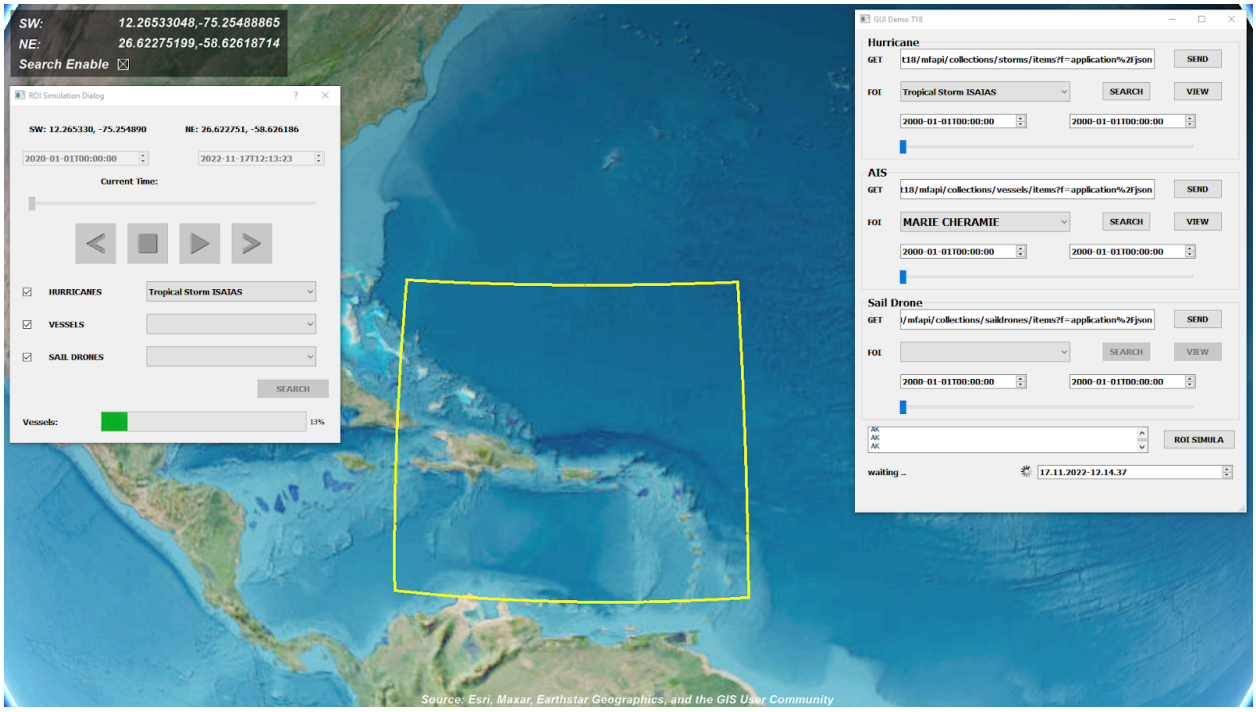


Figure 41 — Demo Client Searching

and this represent the completed stage:

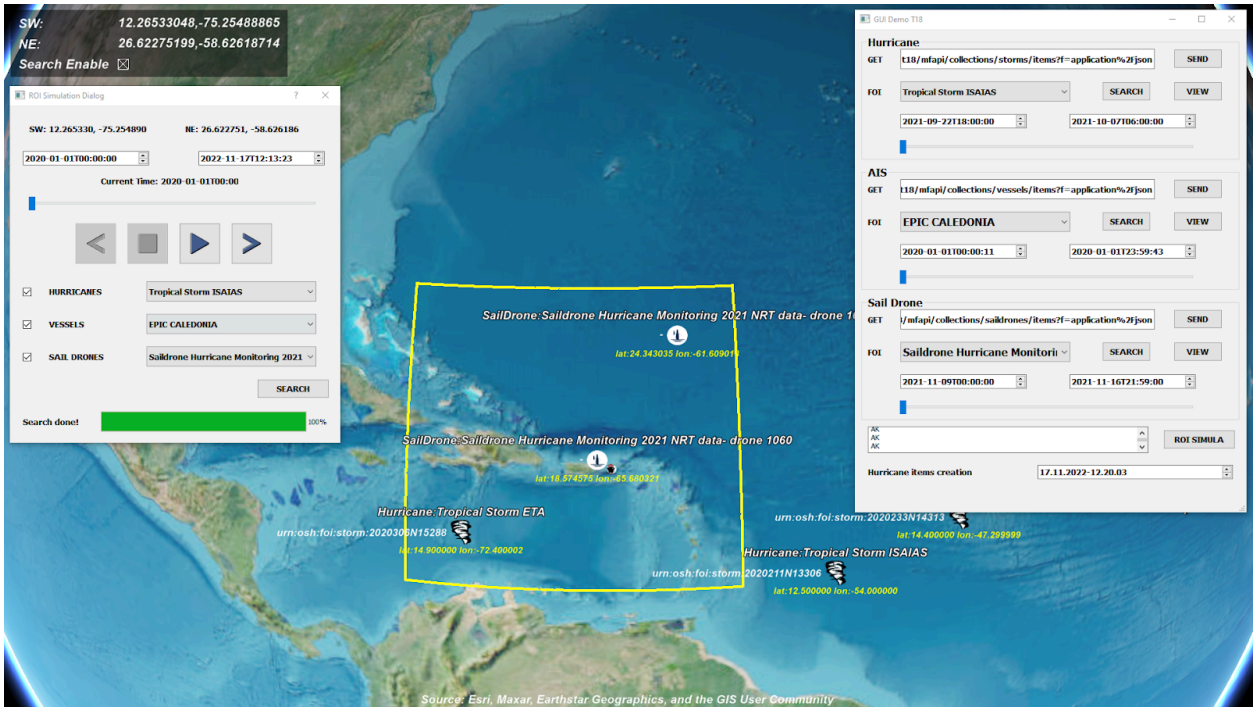


Figure 42 — Demo Client Complete Stage

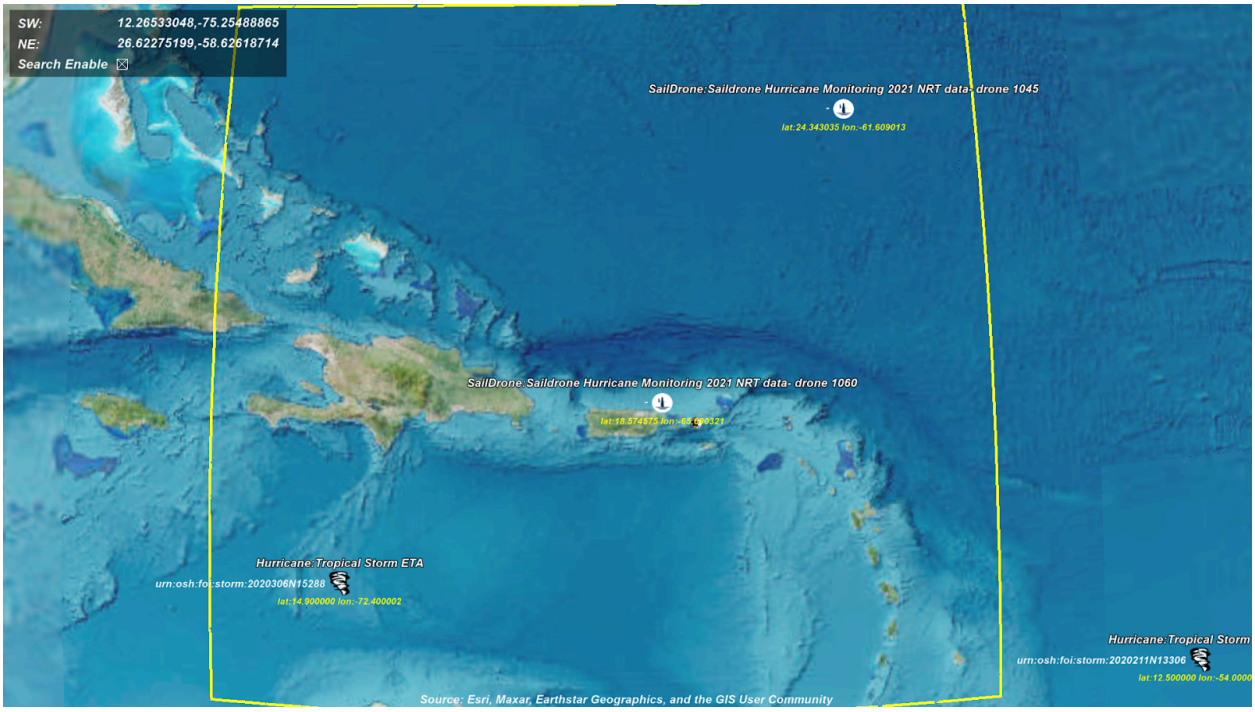


Figure 43 — Demo Client Complete Stage2

The simulation was expected to display all the items found by varying the time on the set interval. For each instant of time in the rendering window the positions of the items will be updated.

10

CLIENT: D143 PELAGIS

10 CLIENT: D143 PELAGIS

10.1. Description & Objective

The D143 Client produced by Pelagis was designed as a consumer of observation data based on the OGC Observations&Measurements draft standard. The client uses the D142 SensorHub as a container representing the observation systems for the North Atlantic Ocean Basin and leverages the Moving Features API and MF_JSON encoding to represent sampling features. Two key areas within scope are related to the observation of features in motion (AIS Vessel Traffic) and sampling features 'in motion' (Saildrones) that make observations of a feature of interest along its trajectory.

10.1.1. Background

This approach to address the needs for the shared use of ocean resources is to make Marine Spatial Planning a core foundation on which to build out vertical applications. The Pegasis platform is based on a federated information model represented as a unified social graph. This provides a decentralized approach towards designing various data streams, each represented by their well-known and/or standardized model. To date, service layers based on the OGC standards for Feature, Observations & Measurements, and Sensors APIs have been developed and extended for adoption within the marine domain. Previous work provides for data discovery and processing of features based on the IHO S-100 standard (Marine Protected Areas, Marine Traffic Management, ...) as well as connected observation systems as provided by NOAA's Integrated Ocean Observing System (IOOS) and its Canadian variant, CIOOS.

This project extends the core federated model to leverage the Moving Features specification to implement service endpoints for the NOAA open data pipelines for major weather events (Hurricane Tracking, Ocean Drifters, Saildrones, etc.) and AIS vessel traffic monitoring as provided through the Denmark Maritime Authority.

10.2. Architecture

The D143 Client is based on a federated design pattern abstracting the marine feature domain into a unified collection of service endpoints. This architecture effectively implements a federated marine spatial data infrastructure (FMSDI) applied as a generalized oceans observation system.

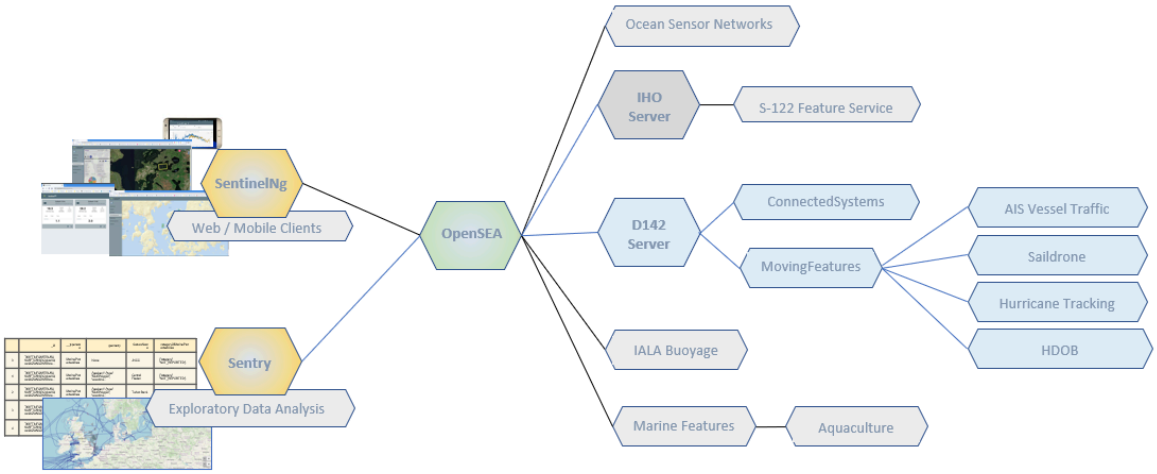


Figure 44 — D143 Client Architecture

Service endpoints for the D142 SensorHub support the specific application scenarios in scope for this project. The endpoints for the AIS Vessel Tracking (DNK) and each Saildrone monitoring platform provide the information services used to model observation systems relative to our 'ultimate' feature of interest — the North Atlantic Ocean Basin.

10.2.1. Feature model

10.2.1.1. Saildrones

A Saildrone is an Autonomous Surface Vessel that provides ocean-related observations through a set of sensing devices mounted to the host platform. Each Saildrone is equipped with a GPS to track the platforms location and various sensing devices to capture environmental measurements such as sea surface temperature, wind conditions, and wave heights.

10.2.1.2. Monitoring Station

A feature representing a real-world facility at which various types of observations are made.

10.2.1.3. AIS BaseStation

A specialized class of Monitoring Station responsible for receiving AIS message transmissions from vessels and navigation aids within its coverage area.

10.2.1.4. Vessel

An ocean going vessel (ship) responsible for the transport of goods and/or people between ports of call. For the purpose of this project, a vessel is limited to those types of ships that are required

to adhere to the AIS messaging requirements and transmit messages identifying the vessel, its location, and other properties of interest to verify safe passage.

10.2.1.5. AidsToNavigation (AtoN)

A Marine Aid to Navigation is defined to as a device, system, or service, external to vessels, designed and operated to enhance safe and efficient navigation of individual vessels and/or vessel traffic.

10.2.1.6. Ocean Buoy

A subclass of an Aid to Navigation, an Ocean Buoy is a marker at a fixed location used to identify a route or navigational channel for a vessel.

10.2.1.7. Monitoring Buoy

A specialized class of Ocean Buoy on which various sensors are placed to observe and collect marine observations such as sea surface temperatures.

10.2.1.8. Ocean Drifter

An ocean drifter is a specialized class of monitoring buoy in which the platform is free to collect ocean observations at various depths and locations.

10.2.1.9. Marine Protected Area (MPA)

This feature class models a marine protected area. A marine protected area is an area of concern regulated by a public agency with certain restrictions, regulations, and limitations as they relate to human activity within its boundary. Although a marine protected area *does not move* in terms of its location, it does have temporal properties associated with it that may be modeled in the context of a 'moving feature'. More specifically, the status of the marine protected area may be modeled as a temporal property and managed through the application of the OGC API-MovingFeatures.

10.3. Scenarios

The client application focuses on two primary scenarios. The first is a continuation of the OGC FMSDI program in which vessel traffic is monitored relative to Marine Protected Areas of the Baltic and North Seas. The second scenario focuses on modelling the NOAA Ocean Observation

System for the North Atlantic in which Saildrone ASV platforms were dispatched to gather environmental data during the 2021 Hurricane Season.

10.3.1. Baltic/North Sea Vessel Traffic

AIS Vessel traffic is of global concern. According to the World Bank, the volume of ship traffic on any day exceeds well over 100,000 active voyages and is a direct performance indicator of global economic trade. However, from the perspective of managing vessel traffic as it affects an area of interest, the responsibility falls to the regional maritime authority. In this scenario, we model the Danish Maritime Authority as the *Responsible Party* for vessel traffic within the Baltic/North Sea.

10.3.1.1. Workflow & Interfaces

Vessel traffic is provided by the D142 SensorHub representing the area of concern for the Denmark Maritime Authority. As described previously, vessel traffic is modeled as a *System* represented by a virtual AIS Base Station. The coverage area for the base station is equivalent to the Denmark EEZ so as to capture all vessel traffic within the DMA area of responsibility.

10.3.1.2. Model

The AIS vessel traffic is loaded into the D142 SensorHub through the D141 ingestion service. This service models AIS traffic using the OGC SSN ontology providing a physical view of the vessel tracking service. The D143 client is based on the OGC OMS specification and so the SSN model is mapped to the OGC OMS model when processing the result query. In this scenario, the vessel traffic is represented as a collection of features with temporal geometry and temporal properties observed through a virtual AIS base station.

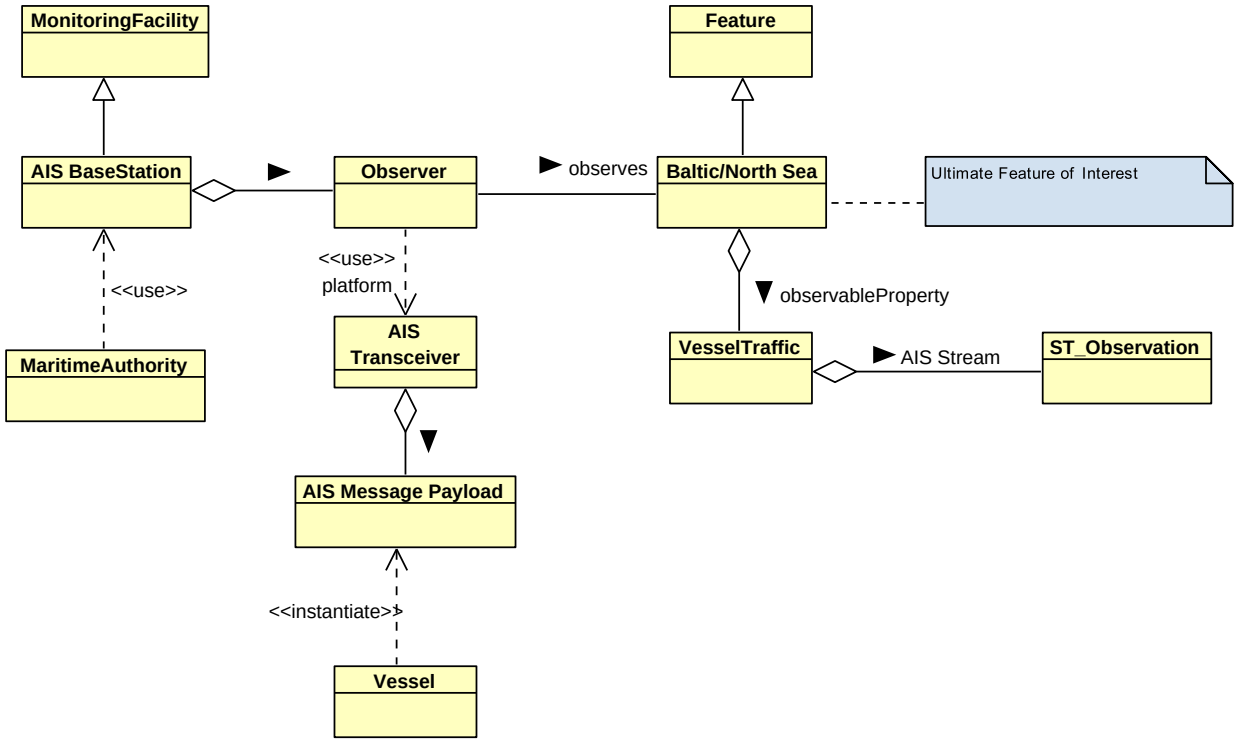


Figure 45 — D143 Client Model for AIS vessel traffic

10.3.1.3. Data Output

The following is a simplified example of querying for the specific voyage profile for an ocean vessel reported within the DMA's area of responsibility for March 14, 2022.

```
{'uid': 'urn:mrn:itu:mmsi:642122020',
 'featureType': 'http://www.opengis.net/def/featureType/T18/Vessel',
 'name': 'ALHANI',
 'description': '',
 'validTime': ['2022-05-14T00:00:00Z', '2022-05-14T23:59:59Z'],
 'mmsi': '642122020',
 'imo': '9331153',
 'callSign': '5AXF',
 'shipType': 'Tanker',
 'cargoType': 'No additional information',
 'length': 249.0,
 'width': 44.0,
 'draft': 7.7}
```

The vessel's trajectory and related temporal properties are retrieved through the JSON object's *links*.

```
link: temporalGeometry

[{'id': 'tg-44l1f3rcki3l6',
  'type': 'MovingPoint',
  'datetimes': ['2022-05-14T00:02:54Z',
   '2022-05-14T00:03:02Z',
   '2022-05-14T00:03:11Z',
   '2022-05-14T00:03:22Z',
```

```
    '2022-05-14T00:03:33Z',
    '2022-05-14T00:03:42Z',

    ...

    '2022-05-14T00:09:12Z',
    '2022-05-14T00:09:21Z',
    '2022-05-14T00:09:32Z',
    '2022-05-14T00:09:43Z'],
   'coordinates': [[56.120763, 12.514048],
    [56.120427, 12.514782],
    [56.120122, 12.515447],
    [56.11976, 12.516273],
    [56.11943, 12.517032],
    [56.119123, 12.517703],

    ...

    [56.106298, 12.544985],
    [56.10592, 12.545745],
    [56.105498, 12.546588],
    [56.105022, 12.5475]],
   'interpolation': 'Linear'}]
link: temporalPropertiesCollection
[{'datetimes': [
    '2022-05-14T00:02:54Z',
    '2022-05-14T00:03:02Z',
    '2022-05-14T00:03:11Z',
    '2022-05-14T00:03:22Z',
    '2022-05-14T00:03:33Z',
    ...
    '2022-05-14T00:09:12Z',
    '2022-05-14T00:09:21Z',
    '2022-05-14T00:09:32Z',
    '2022-05-14T00:09:43Z'],
   'sog': {'type': 'TDouble',
    'form': 'http://sensorml.com/ont/swe/property/SpeedOverGround',
    'description': 'Vessel speed relative to ground',
    'interpolation': 'Linear',
    'values': [
     11.4,
     11.5,
     11.5,
     11.6,
     11.7,
     ...
     13.7,
     13.7,
     13.7,
     13.7]},
   'cog': {'type': 'TDouble',
    'form': 'http://sensorml.com/ont/swe/property/CourseOverGround',
    'description': 'Vessel travel direction relative to ground',
    'interpolation': 'Linear',
    'values': [
     129.3,
     129.3,
     129.2,
     129.4,
     129.5,
     ...
     132.2,
```

```
    131.7,
    132.1,
    132.9]},
  'heading': {'type': 'TDouble',
   'form': 'http://sensorml.com/ont/swe/property/TrueHeading',
   'description': 'Vessel heading direction relative to true north, measured
clockwise',
   'interpolation': 'Linear',
   'values': [
    129.0,
    129.0,
    129.0,
    129.0,
    129.0,
    ...
    132.0,
    132.0,
    133.0,
    132.0]},
  'rot': {'type': 'TDouble',
   'form': 'http://sensorml.com/ont/swe/property/RateOfTurn',
   'description': 'The rate of turn (ROT) or angular velocity of the ship',
   'interpolation': 'Linear',
   'values': [
    0.4,
    0.0,
    0.0,
    0.4,
    0.0,
    ...
    -3.6,
    2.9,
    -0.7,
    -3.6]},
  'status': {'type': 'TText',
   'form': 'http://sensorml.com/ont/ais/property/NavigationStatus',
   'description': 'Vessel Status',
   'interpolation': 'Linear',
   'values': [
    'Under way using engine',
    'Under way using engine',
    'Under way using engine',
    'Under way using engine',
    'Under way using engine',
    ...
    'Under way using engine',
    'Under way using engine',
    'Under way using engine',
    'Under way using engine']}}]
```

10.3.1.4. Key Challenges & Observations

1. The volume of data related to ship traffic on a global scale quickly overloads the encoding scheme for moving features. However, from the perspective of managing vessel traffic as it affects a region of interest, the moving features encoding for the vessel trajectory and temporal properties was shown to be effective.

2. The moving features encoding provides a snapshot of a vessel's trajectory and temporal properties. Query capabilities extended to filter the area of concern within a spatial extent and time frame are effective at reducing the document size of the encoded scheme.

3. Modelling vessel traffic as a real-time stream of observations presents a challenge to the moving features encoding. The OGC Features API over a moving features collection provides for 'polling' of a vessel trajectory but further research is required to support real-time streaming of vessel traffic. For consideration is extending the MF_JSON encoding scheme to use links to a service endpoint providing real-time temporal properties and geometries to the client rather than an array of values.

4. The OGC SSN ontology defines a *System* as either a host of *Systems* or as a *Sensor* or *Actuator*. The approach of modelling an AIS base station as a *System of Systems* violates the dependency that the *System* also define the specific suite of sensors used to perform the duties of transmitting and receiving AIS messages across the network. As such, traceability requirements from the AIS system to the *procedures* required to authenticate the methods used to produce the positioning of ocean vessels cannot be maintained.

5. The North-Atlantic Ocean is the ultimate feature of interest for this project. This is subdivided into 'regions of interest' which are modelled as coverage areas. In the case of the AIS vessel traffic, the vessels are features of interest for the Danish Maritime Authority when observing individual features but it is the AIS Coverage Area within the Baltic/North Sea that is the ultimate feature of interest. The model identifies a 'virtual' AIS base station as a monitoring facility with an Observer for vessel traffic -→ the sampled feature is a collection of features (vessels) bound by the coverage area. Technically, the AIS base station 'observes' the ocean surface for observations of vessels. In this context, what are the sampled feature and observable properties since basically tracking space-filling events caused by the existence of an ocean vessel is being performed? Generally, all space-filling observations will fall into this design pattern.

10.3.2. NOAA Ocean Observation System — 2021 Hurricane Season

This scenario focuses on the NOAA observation system in place for the 2021 Hurricane Season. Saildrones were deployed to the North Atlantic to monitor ocean conditions relevant to major weather events. NOAA is the responsible party for managing the observation stream provisioned through each Saildrone platform.

10.3.2.1. Workflow & Interfaces

Saildrones play the role of both *observer* and *observed* features. As an *Observer*, each Saildrone platform hosts specialized equipment designed to measure specific phenomenon related to the ocean's environment. This, in the context of the North Atlantic Basin as the *ultimate* feature of

interest, provides a stream of related *Observations*. As the *Observed* feature, each Saildrone is a *Feature of Interest* to the management team responsible for the mission deployment.

In the latter case, the application scenario is very similar to the objectives of the AIS Vessel tracking. The Saildrone provides a continuous stream of positioning observations to the mission team. These observations are modeled as a *Datastream* in the SSN ontology that allows the D143 client to use the mfapi and MF_JSON encoding to report the Saildrone trajectory. Note that for this particular mission, no temporal properties related to the Saildrone platform were of interest.

As an observer, the Saildrone captures measurements against the properties of interest for the North Atlantic basin. As discussed previously, the challenge with this scenario is that the measurements are effectively a time-series of observations with no *locality*. As a result, the client application is required to have prior knowledge of the dependency between the Saildrone platform and the sensor equipment in order to instantiate a *Sampling Curve* of observations across the North Atlantic.

10.3.2.2. Model

As an Observer, the Saildrone platform represents a specialized type of MonitoringFacility equipped with sensing equipment to measure the ocean conditions along the trajectory of the Saildrone itself. As a Feature of Interest for Mission Control, a specialized Observer of the Saildrone emits observations on its geolocation. This specialized observation stream of geopositions is used to compose a SamplingCurve of Observations across the North Atlantic.

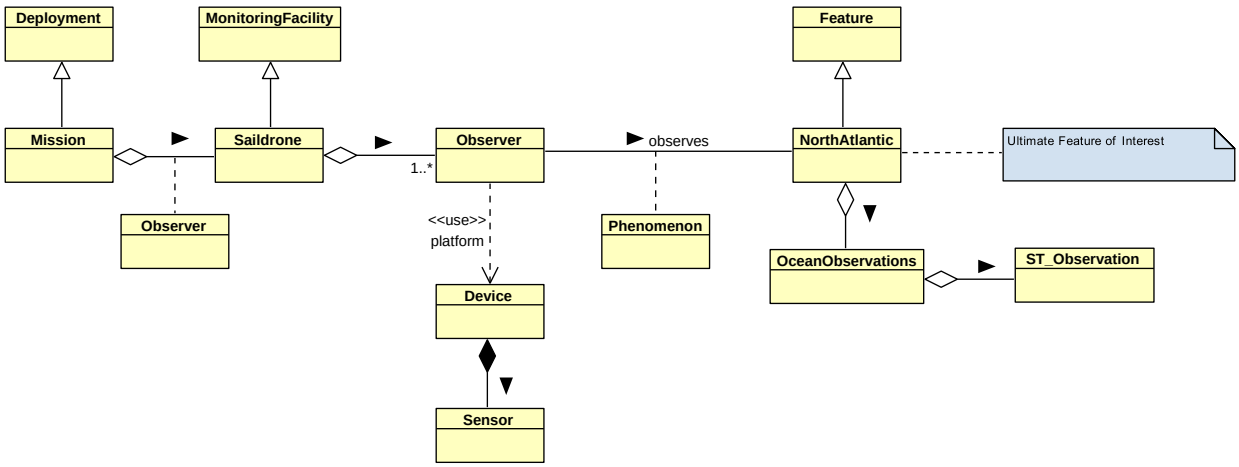


Figure 46 — D143 Saildrone Observation Model

10.3.2.3. Data Output

```
link: temporalGeometry
{'temporalGeometries':
  [{'id': 'tg-7ghcsqq1f1jfi',
   'type': 'MovingPoint',
   'datetimes': [
    '2021-10-29T00:00:00Z',
    '2021-10-29T00:30:00Z',
```

```
     '2021-10-29T01:00:00Z',
     '2021-10-29T01:24:00Z',
     '2021-10-29T01:54:00Z',
     ...
     '2021-10-30T22:00:00Z',
     '2021-10-30T22:30:00Z',
     '2021-10-30T23:00:00Z',
     '2021-10-30T23:24:00Z'],
    'coordinates': [
     [34.7411872, -76.0770688],
     [34.7448704, -76.0758656],
     [34.7483968, -76.0736768],
     [34.7510592, -76.0718848],
     [34.753872, -76.0688832],
     ...
     [34.7464, -76.1448448],
     [34.7394144, -76.14592],
     [34.7397824, -76.1432064],
     [34.751056, -76.1286272]],
    'interpolation': 'Linear'}]}
link: temporalPropertiesCollection
[{'datetimes': [
    '2021-10-29T00:00:00Z',
    '2021-10-29T00:30:00Z',
    '2021-10-29T01:00:00Z',
    '2021-10-29T01:24:00Z',
    '2021-10-29T01:54:00Z',
    ...
    '2021-10-30T22:00:00Z',
    '2021-10-30T22:30:00Z',
    '2021-10-30T23:00:00Z',
    '2021-10-30T23:24:00Z'],
   'TEMP_AIR_MEAN': {'type': 'TDouble',
    'form': '',
    'description': 'Mean Air Temperature',
    'interpolation': 'Linear',
    'values': [
     21.08,
     21.73,
     22.09,
     22.12,
     22.31,
     ...
     19.45,
     19.31,
     19.15,
     19.23]},
   'RH_MEAN': {'type': 'TDouble',
    'form': '',
    'description': 'Relative humidity',
    'interpolation': 'Linear',
    'values': [77.09,
     79.21,
     74.05,
     76.38,
     74.52,
     ...
     65.22,
     62.19,
     61.83,
     62.73]},
   'BARO_PRES_MEAN': {'type': 'TDouble',
```

```
    'form': '',
    'description': 'Air pressure',
    'interpolation': 'Linear',
    'values': [
     1009.33,
     1008.9,
     1008.46,
     1008.1,
     1007.9,
     ...
     1002.62,
     1002.76,
     1003.21,
     1003.33]},
   'TEMP_SBE37_MEAN': {'type': 'TDouble',
    'form': '',
    'description': 'Seawater temperature',
    'interpolation': 'Linear',
    'values': [
     23.5597,
     23.6702,
     23.8158,
     23.8885,
     23.9358,
     ...
     23.1876,
     23.0885,
     23.0781,
     23.2082]},
   'SAL_SBE37_MEAN': {'type': 'TDouble',
    'form': '',
    'description': 'Seawater salinity',
    'interpolation': 'Linear',
    'values': [
     34.8257,
     34.882,
     34.9626,
     34.992,
     35.0063,
     ...
     35.2432,
     35.2798,
     35.2796,
     35.2143]},
   'WIND_FROM_MEAN': {'type': 'TDouble',
    'form': 'http://mmisw.org/ont/cf/parameter/wind_from_direction',
    'description': 'Wind Direction',
    'interpolation': 'Linear',
    'values': [
     108.5,
     106.9,
     119.0,
     114.4,
     116.5,
     ...
     266.4,
     259.8,
     264.8,
     261.0]},
   'WIND_SPEED_MEAN': {'type': 'TDouble',
    'form': 'http://mmisw.org/ont/cf/parameter/wind_speed',
    'description': 'Wind speed nominal_sampling_schedule: 60s on, 240s off,
centered at :00',
```

```
 'interpolation': 'Linear',
 'values': [
  8.14,
  9.84,
  9.66,
  8.41,
  9.66,
  ...
  8.72,
  8.75,
  7.22,
  8.37]},
'WATER_CURRENT_SPEED_MEAN': {'type': 'TDouble',
 'form': '',
 'description': 'Speed of water current',
 'interpolation': 'Linear',
 'values': [
  0.313,
  0.32,
  0.298,
  0.336,
  0.289,
  ...
  0.115,
  0.132,
  0.145,
  0.138]},
'WATER_CURRENT_DIRECTION_MEAN': {'type': 'TDouble',
 'form': '',
 'description': 'Direction of water current',
 'interpolation': 'Linear',
 'values': [
  252.8,
  249.5,
  250.1,
  251.2,
  248.8,
  ...
  61.3,
  77.0,
  61.0,
  69.8]},
'WAVE_DOMINANT_PERIOD': {'type': 'TDouble',
 'form': '',
 'description': 'Dominant wave period',
 'interpolation': 'Linear',
 'values': [
  12.8,
  12.8,
  3.76,
  3.76,
  4.0,
  ...
  12.8,
  12.8,
  12.8,
  12.8]},
'WAVE_SIGNIFICANT_HEIGHT': {'type': 'TDouble',
 'form': '',
 'description': 'Significant wave height',
 'interpolation': 'Linear',
 'values': [0.896,
  0.878,
```

```
0.85,
0.85,
0.863,
0.953,
...
1.64,
1.514,
1.547,
1.547]}}]
```

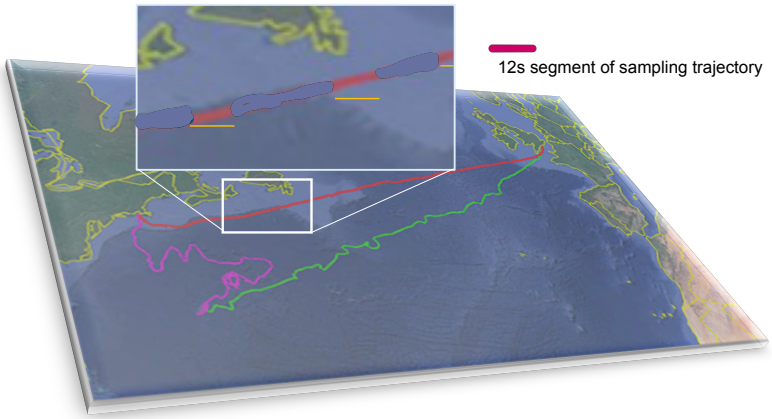
10.3.2.4. Key Challenges & Observations

1. Interpolation of Observation locations: As a sensing device moves from two separate locations along the path of its hosting platform it may produce temporal observations of a property of interest. To properly locate these observations the moving features API must provide the means to interpolate between two points along its path based on the time *t* of the observation. As per the OMS specification, the observation system should maintain both a *phenomenonTime* and a *resultTime* of an *Observation* to differentiate the actual time of the measurement as opposed to the time the measurement was made available. There is no equivalent concept when managing the location of an observation — i.e., the '*phenomenonLocation*' of the sensing device at the time of the measurement versus the '*resultLocation*' of the observation to indicate this is an interpolated location. In the case of the Saildrone observations the measurements were reported as a *resultTime*. The actual time of the observation — the *phenomenonTime* needs to be calculated based on the *sampling schedule* of each sensing device. Should an application dependent on the OMS model incorrectly request the interpolated position based on the resultTime vs the phenomenonTime, the location of the observation may be incorrect.

2. The MovingFeatures API provides a service to encode the movement of a Feature across a path. This concept, when applied to the field of observations systems, is used to denote a SamplingFeature moving along a curve which is best defined within the OMS specification as a *Sampling Curve*. This is an issue on the client side to ensure alignment with the OMS specification for the behavior and encoding of a moving sampling feature. Does SamplingCurve support a tight alignment of time and location similar to the goals of the Moving Features SWG? Is this an opportunity to define a spatio-temporal geometry (ST_Geometry) that encodes movement? How would this be extended for the orientation of a Feature independent of the Feature's geometry or shape?

3. In order to tightly couple an observation to its time and location of measurement, it would be necessary to extend the OMS Observation model to include

the concept of a *ST_Observation* — a specialized type of Observation with phenomenonTime, resultTime, **and** phenomenonLocation, resultLocation.



Of note is that the phenomenonLocation for Saildrones is actually a Curve — more specifically, an interval along the trajectory defined by the nominal sampling schedule. For example, the measurement of the sea surface temperature is the average temperature measured along a 12s segment of the trajectory reported with a result_time *t*.

4. The W3C Devices and Sensors Working Group defines a *Geolocation API* that encapsulates a Features location and orientation as its *GeolocationPosition*. This api targets sensing devices that provide global positioning of the host feature. The Moving Features API seems strictly focused on Geometry which in most use cases represents the Point location of the hosting feature. Orientation of the feature is maintained as a temporal property associated with the moving feature effectively decoupling the W3C concept from that of the OGC Moving Features. As more client applications are developed for the 3D+ space, the Moving Features workgroup — and OGC generally — may want to revisit this topic to ensure a tighter alignment between concepts.

5. The concept of a Moving Feature is affected when the Feature is *not moving*. In the Saildrone scenario, the initial set of observations were misleading as the sensing devices were activated prior to the physical deployment of the platform. In this case, observation measurements were collected while the platform was not in motion and needed to be pre-processed out of the final observation stream.

6. To determine the current location of a Feature in motion, the trajectory of the feature must be queried and assumed that the last observation is the most recent and reflects the current position. This can be an expensive operation on the client side and should be exposed as an extension of the moving features API. Since all features are capable of moving, whether they are in motion or not, it would be effective if the client applications could use the OGC Features API to query the feature.geometry to imply the current location and a separate property, such as trajectory, to get the moving feature's trajectory.

11

GEOPOSE VIDEO ANALYSIS

11 GEOPOSE VIDEO ANALYSIS

11.1. Scope

Building on the autonomous vehicle results from Testbed-17, Away Team identified road network use cases that could benefit from leveraging OGC GeoPose to aggregate geotagged video observations into Moving Features which enrich multi-sensor data.

Figure 47 — Tracking Moving Cyclist From Moving Vehicle In Testbed-17

11.2. Use Cases

Discussions with national road network authorities have highlighted the following key use cases.

- **Litter monitoring** to assess accretion rates of objects that can hinder traffic flows by harvesting geotagged dashcam footage from a fleet of vehicles which routinely traverse the road network for maintenance, delivery, or transportation purposes. Metrics can be used to plan and prioritize remedial action.
- **Wrong way traffic** to rapidly identify and locate vehicles traveling in the opposite direction to the correct traffic flow which is a safety-critical issue. Being able to interview these elusive drivers can also provide valuable feedback to help reduce incident recurrence with targeted improvements to road junction design.

The essential elements of these use cases can be reduced to the following concept demonstrations.

1. **Traffic cameras**: Associate observations of a moving vehicle seen by two static traffic cameras at different locations along a road.
2. **Fleet survey**: Associate observations of a static object seen by dashcams from two moving vehicles passing the same location at different times.
3. **Multi-view survey**: Associate observations of a static object seen from front- and rear-facing dashcams on a moving vehicle passing a location.

11.3. Data Analysis

The identified use cases require concurrent video streams from multiple cameras that include location and orientation data, which constitute OGC GeoPose, in order to correctly calculate the location of objects observed. Modern smartphones possess all the sensors needed to capture such data, including a digital camera, Global Navigation Satellite System (GNSS) receiver, and sensors such as a compass, accelerometer, and gyroscope to determine geo-aligned orientations.

11.3.1. Web Implementation

An initial demonstration was created to display a 3D compass in a web page that mimics a gimballed compass designed for use on ships and aircraft by compensating for their pitch and roll. This demo combines data streamed from multiple sensors on a mobile device to calculate orientation using W3C Geo-alignment features and displays the correct heading in a web browser regardless of the device's attitude.

This implementation works well when the device is either held flat in the horizontal plane or viewed vertically in portrait mode. However, the heading is incorrect when the device is viewed in landscape mode as the direction indicator remains aligned with the short edge of the screen which is pointing to the left (or right) of the observer in this case. Although technically still correct, this is misleading and unhelpful to the user.

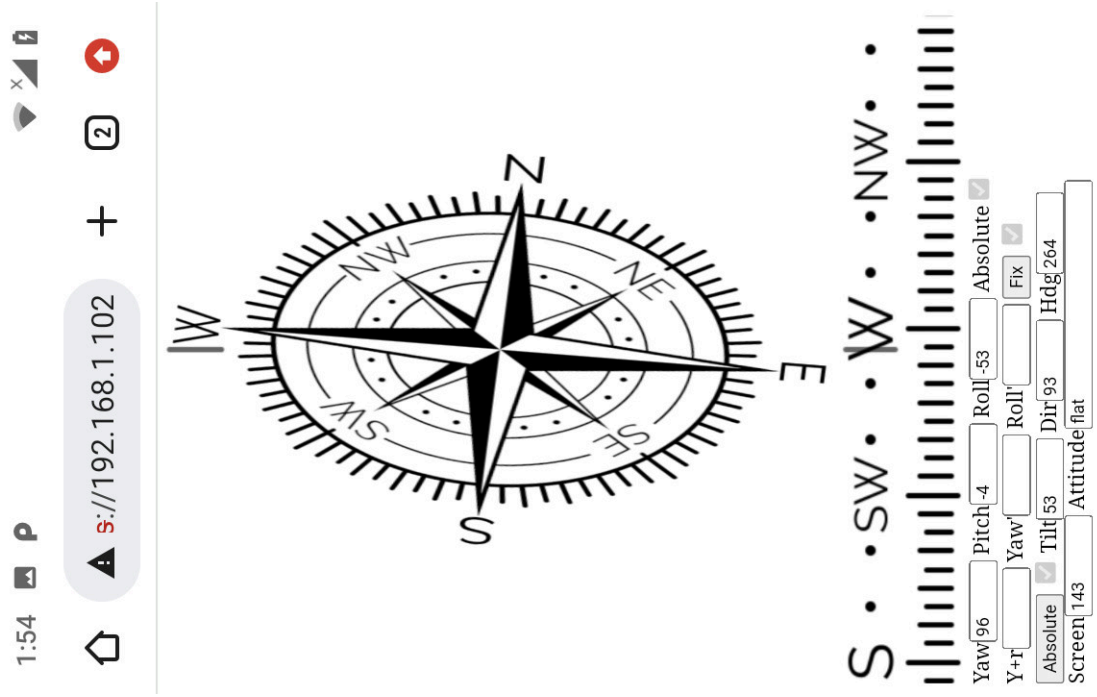


Figure 48 — 3D Compass Web Page With Fixed Heading Indicator

To address this issue, the direction indicator was redesigned to float upwards and remain aligned with the top of the screen regardless of the device's orientation. The resultant heading is correct in both portrait and landscape modes and transitions smoothly between these two states.

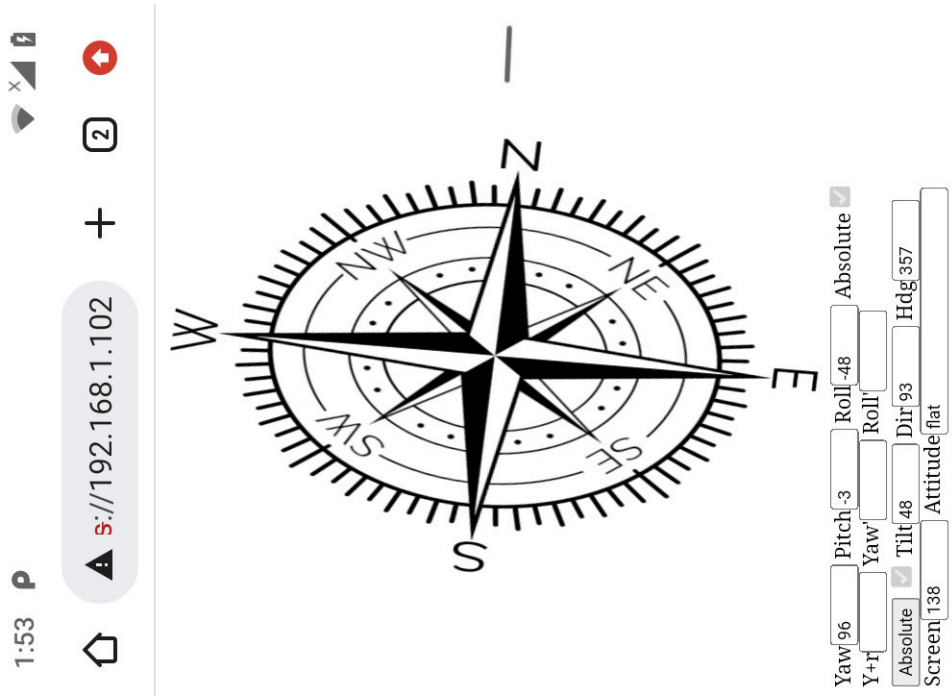


Figure 49 — 3D Compass Web Page With Floating Heading Indicator

The demo also highlighted that care is needed to correctly calculate heading when the device is inverted — such that the display is facing downwards with the user observing from below — since bearings are measured counterclockwise rather than in the customary clockwise direction.

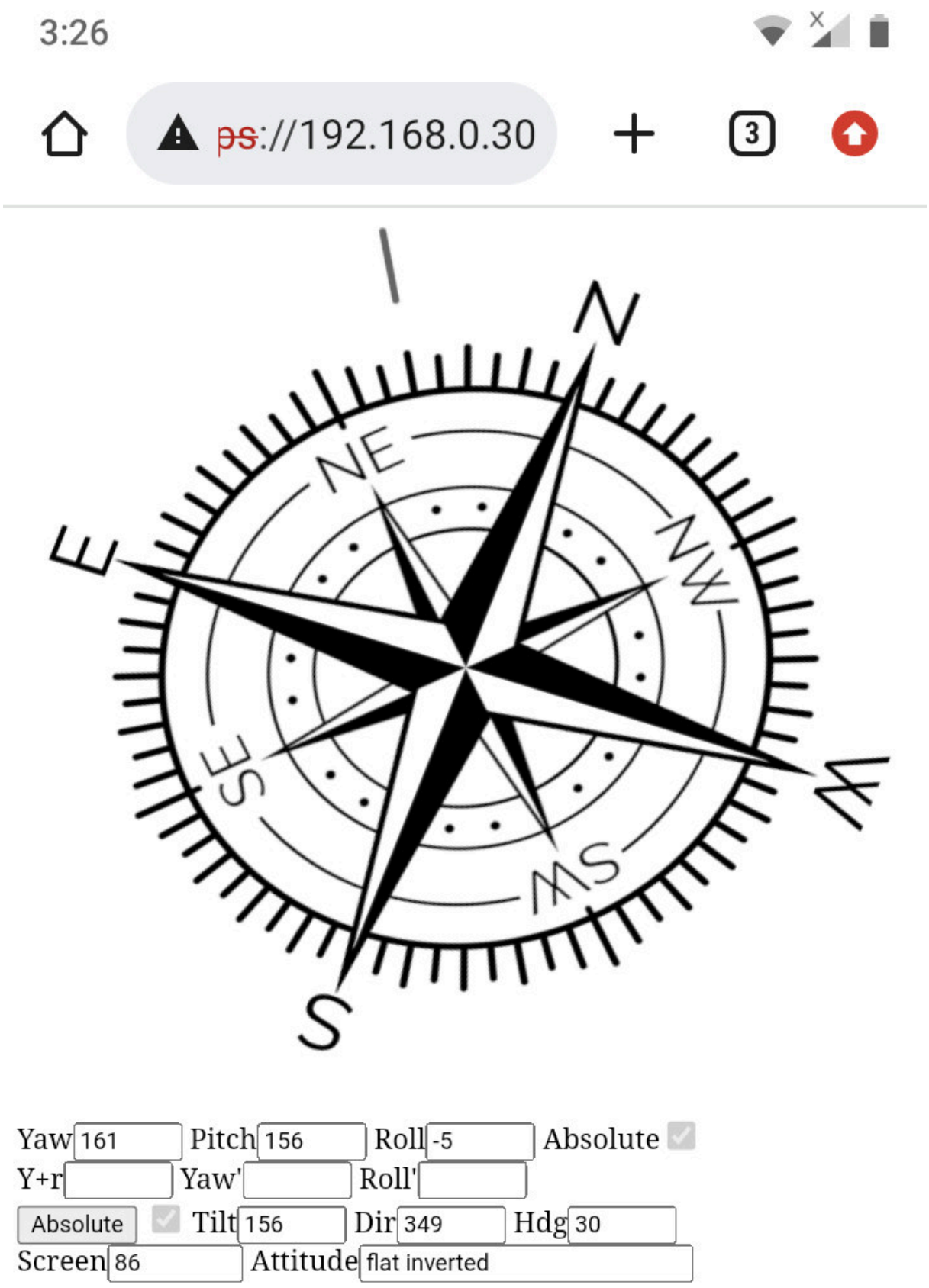


Figure 50 — 3D Compass Web Page Inverted Display

11.3.2. Mobile App Implementation

The 3D compass demo was ported to a native mobile app for Android devices in order to allow more autonomous operation by eliminating the requirement for web connectivity. Support is provided for both fixed and floating heading indicators which are better suited to horizontal and vertical device operation respectively.

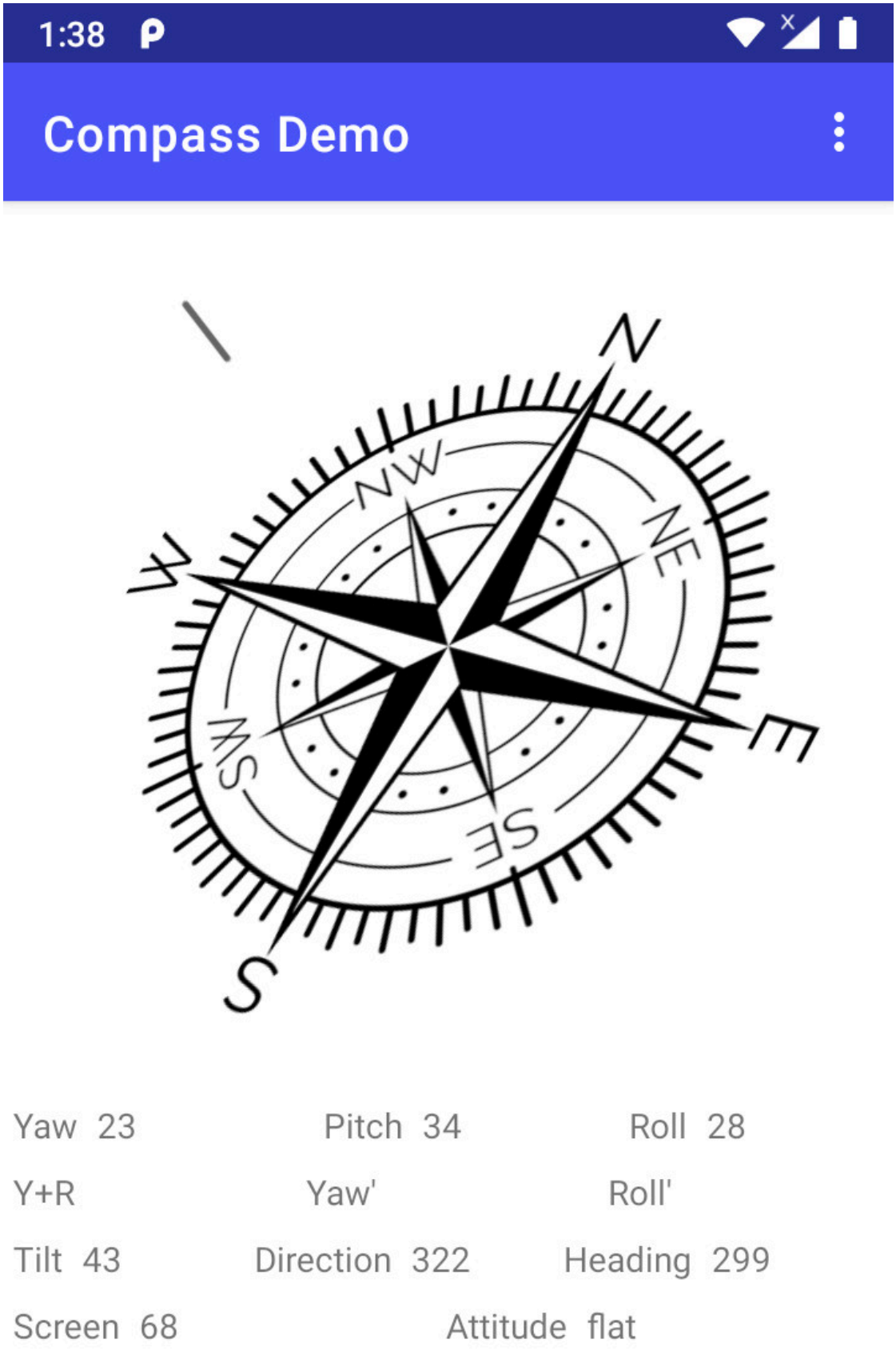


Figure 51 — 3D Compass Mobile App For Android

Development of the native app identified an anomaly whereby code written for the web API produced different results under the Android operating system on the same host device. This discrepancy was traced to a difference in the ranges for yaw, pitch, and roll angles under the two systems, which highlighted a wider issue.

The full range of orientations can be described using a only subset of the full range of angles. For instance:

- yaw can equate to compass headings in the range [0, 360);
- pitch can be limited to the subset between straight up and straight down in the range [-90, 90]; and
- roll can determine rotation from vertical, where zero represents upright, clockwise rotations extend to upside-down at 180 degrees, and anticlockwise rotations end similarly at -180 degrees, resulting in a full range of (-180, 180].

Choosing the above ranges produces a simple mapping in which heading equates to yaw angle and inversion is determined purely from roll angle when pitch angles are limited to half of their full 360 degree range. However, symmetry ensures that there are two alternative schemes where either roll or yaw angles are limited to half their full range and still allow the full range of orientations to be defined.

Table 16 — Limited Angle Ranges Permitting Full Orientation Range

Half Range	Yaw	Pitch	Roll
Pitch	[0, 360)	[-90, 90]	(-180, 180]
Roll	[0, 360)	(-180, 180]	[-90, 90]
Yaw	[0, 180)	(-180, 180]	(-180, 180]

Pitch angles in the ranges (-180, -90) and (90, 180] are valid, though these duplicate other orientations and are incompatible with the simple mapping described above. Hence the full ranges of yaw, pitch, and roll angles should be taken into account when calculating heading and inversion since failure to consider such cases can lead to incorrect results.

A snapshot feature was added to the app that assembles location and orientation data from the device's sensors into an instantaneous GeoPose JSON. This can be copied to the device's clipboard in several formats including Basic-Euler, Basic-Quaternion, and Advanced in order to share the captured GeoPose data with other applications.

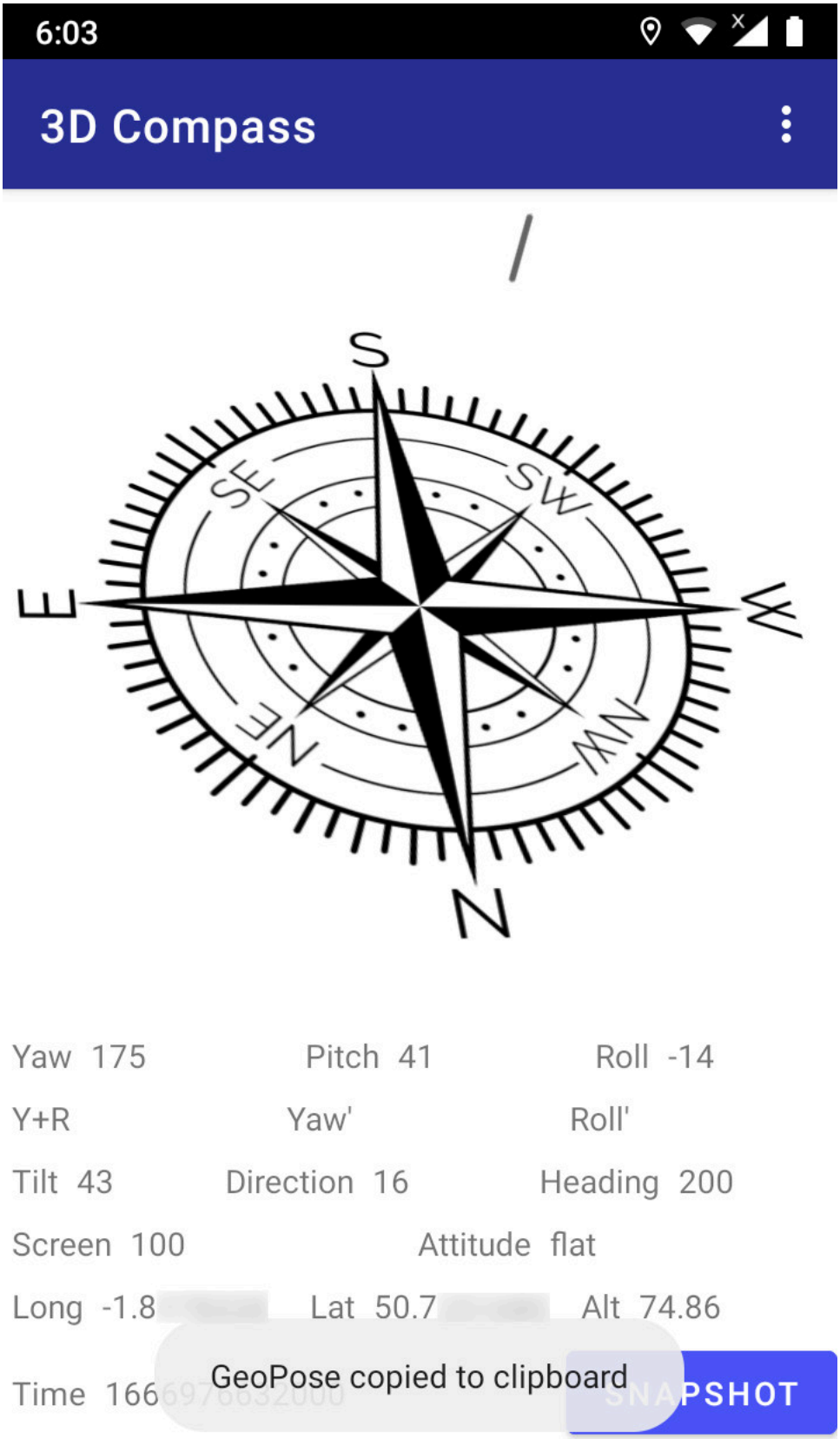


Figure 52 — 3D Compass Mobile App GeoPose Snapshot

GeoPose syntax compliance was verified using a prototype Swagger validator tool.

11.4. Results

The 3D compass web page has demonstrated the accuracy and responsiveness of W3C Geo-alignment features in modern mobile web browsers. Web and native mobile implementations have highlighted differences between their interfaces and helped to resolve the practical issues associated with capturing orientation on a smartphone.

Three key issues were identified when calculating heading from yaw, pitch and roll angles on a mobile device.

1. Heading calculations should remain accurate and representative for a device used in any attitude.
2. Device inversion should be correctly identified as heading is measured counterclockwise in this case.
3. The full range of yaw, pitch, and roll angles should be considered as these include many-to-one mappings to heading and may extend beyond the expected range.

11.5. Conclusions

Away Team has successfully identified real-world use cases that could benefit from aggregation of video data captured by multiple sensors with GeoPose. Significant progress has been made towards creating the tools required to capture such data in a sharable form using a low-cost mobile device. Practical demonstrations have shown how location and orientation data can be integrated with W3C interfaces in a web browser and with native APIs in a mobile app.

Several orientation issues have been resolved which enable data to be captured with GeoPose for the road network use cases described. Details of a planned collaboration with Ordnance Survey to capture these data are outlined in the Future Work section which may be suitable for inclusion and analysis in Testbed-19.

The 3D compass demo illustrates the concepts explored in an intuitive way and the mobile app allows GeoPose to be captured and shared with other applications in a format standardized by OGC. An Android implementation of this has been published free of charge by Away Team and is available from the Google Play store.

12

MFSI SUMMARY TABLES

12 MFSI SUMMARY TABLES

This chapter provides the summary tables for the moving features and sensor integration task.

12.1. Test Cases Definitions

The following tables describe the test case definitions.

Table 17 — Connected Systems API — Ingestion

TEST CASE	DESCRIPTION	SUCCESS CRITERIA
Ingest new system description (Geo JSON)	Create a new System resource by doing a POST request with a Geo JSON payload	System is created successfully with the provided information
Ingest new datastream (SensorML)	Create a new System resource by doing a POST request with a Sensor ML-JSON payload	System is created successfully with the provided information
Ingest new datastream	Create a new datastream resource by doing a POST request with the required JSON payload and SWE Common schema	Datastream is created successfully with the provided information and schema
Ingest new feature of interest (single)	Create a new FeatureOfInterest resource by doing a POST request with a GeoJSON payload	FOI is created successfully with the provided information
Ingest new feature of interest (batch)	Create several FeatureOfInterest resources by doing a POST request with an array of GeoJSON objects	All features are created successfully with the provided information

Table 18 — Connected Systems API — Retrieval

TEST CASE	DESCRIPTION	SUCCESS CRITERIA
Browse connected systems	Browse and page through the list of all systems exposed by the API (using offset and limit query parameters)	Server returns the collection of all systems by pages in GeoJSON or HTML format
Retrieve system by ID	Retrieve a specific system description knowing its server assigned ID	Server returns the description of the system with the given ID in GeoJSON or HTML format

TEST CASE	DESCRIPTION	SUCCESS CRITERIA
Filter connected systems (various options)	Filter the collection of systems using various query parameters: *Unique ID *Keywords *ValidTime *Location (bbox or geometry)	Server returns a collection containing only systems matching the filter in GeoJSON or HTML format
Browse features of interest	Browse and page through the list of all features of interest exposed by the API (using offset and limit query parameters)	Server returns the collection of all features of interest by pages in Geo JSON or HTML format
Retrieve feature of interest by ID	Retrieve a specific feature of interest description knowing its server assigned ID	Server returns the description of the feature with the given ID in Geo JSON or HTML format
Filter features of interest (various options)	Filter the collection of features of interest using various query parameters: *Unique ID *Keywords *ValidTime *Location (bbox or geometry) *Property values	Server returns a collection containing only features matching the filter in GeoJSON or HTML format
Retrieve observations by time range	Filter the collection of observations using the phenomenonTime query parameter	Server returns a collection of observations whose phenomenon Time is within the selected time range, in O&M JSON format

Table 19 — Moving Features API — Retrieval

TEST CASE	DESCRIPTION	SUCCESS CRITERIA
Browse list of Moving Features collections	View the list of all feature collections available from the server	Server returns the list of all configured feature collections
Browse Storms as Moving Features	Browse and page through the list of all features in the 'storms' collection (using offset and limit query parameters)	Server returns the collection of all 'storm' moving features by pages in GeoJSON or HTML format
Browse AIS Vessels as Moving Features	Browse and page through the list of all features in the 'vessels' collection (using offset and limit query parameters)	Server returns the collection of all 'vessel' moving features by pages in GeoJSON or HTML format
Browse Saildrone platforms as Moving Features	Browse and page through the list of all features in the 'saildrones' collection (using offset and limit query parameters)	Server returns the collection of all 'saildrone' moving features by pages in GeoJSON or HTML format
Filter Moving Features (various options)	Filter the a collection of moving features using various query parameters: *Unique ID *Keywords	Server returns a collection containing only features matching the filter in GeoJSON or HTML format

TEST CASE	DESCRIPTION	SUCCESS CRITERIA
	*ValidTime *Location (bbox or geometry) *Property values	
Retrieve Moving Features temporal geometries	Retrieve temporal geometries of a given feature, optionally filtering by time range	Server returns the values of the feature temporal geometries for the selected time range, in MF-JSON format
Retrieve Moving Features temporal properties	Retrieve temporal properties of a given feature, optionally filtering by time range	Server returns the values of the feature temporal properties for the selected time range, in MF-JSON format

Table 20 — Connected Systems API Ingestion Tests Matrix

TEST CASE	INGESTION SERVICE (PELAGIS)	INGESTION SERVICE (ASU)	WIKI EXAMPLES
Ingest new system description (GeoJSON)	pass	not tested	pass
Ingest new system description (SensorML)	not tested	not tested	pass
Ingest new datastream	pass	not tested	pass
Ingest new feature of interest (single)	pass	pass	pass
Ingest new feature of interest (batch)	not tested	not tested	pass
Ingest observations (single)	pass	pass	pass
Ingest observations (batch)	not tested	not tested	pass

Table 21 — Connected Systems API Retrieval Tests Matrix

TEST CASE	VISUALIZATION CLIENT (BOTTS-INC.)	WIKI EXAMPLES	PELAGIS CLIENT
Browse connected Systems	pass	pass	pass
Retrieve system by ID	pass	pass	pass
Filter connected systems (various options)	not tested	pass	pass

TEST CASE	VISUALIZATION CLIENT (BOTTS-INC.)	WIKI EXAMPLES	PELAGIS CLIENT
Browse features of Features Of Interest	not tested	pass	pass
Retrieve Features of Interest by ID	not tested	pass	pass
Filter Features of Interest (various options)	not tested	pass	pass
Retrieve observations by time range	pass	not tested	pass

Table 22 — Moving Features API Retrieval Tests Matrix

TEST CASE	VISUALIZATION CLIENT (SUPERELECTRIC)	VISUALIZATION CLIENT (PELAGIS)
Browse Hurricane as Moving Features	pass	pass
Browse AIS Vessels as Moving Features	pass	pass
Browse saildrone platforms as Moving Features	pass	pass
Filter Moving Features (various options)	not tested	not tested
Retrieve Moving Features temporal geometries	pass	pass
Retrieve Moving Features Temporal Geometries	pass	pass

13

SUMMARY

13 SUMMARY

This section summarizes lessons learned from the Testbed-18 MFSI activity and proposes future activities to build on this work and broader OGC activities.

13.1. Lessons Learned

Below are lessons learned for each deliverable.

13.1.1. Ingestion Service

The moving feature collection system for Hurricane tracking use case integrates moving features with static features including time, wind speed, air pressure, and category so that each static feature is attached with the "moving attribute." It supports the interoperability between multiple sensors for identification of different static features at different locations. It also lays a solid foundation for visualizing enriched information associated with the moving feature. The integrated features which are pushed into the Sensor Hub can be retrieved through a publish-subscribe message service via HTTP. It also enhances the reproduction of multi-sensor integration and information retrieval. Furthermore, applying the integration to the hurricane tracking use case improves timely disaster response and provides comprehensive information for disaster mitigation.

13.1.2. Sensorhub

A better alignment of the SensorWeb API with "OGC API — Features" made sense since observation systems and features of interest are also regular features and can have a (summary) GeoJSON and/or JSON-FG representations. Hence the idea of the new "OGC API — Connected Systems," that will be much more well integrated with the rest of the OGC API ecosystem.

It is possible to implement conversion from SWE/O&M data models to Moving Feature data models on the fly, while maintaining semantic information about the temporal variables.

It is also possible to generate a snapshot of time varying features of interest at a point in time using dynamic information from observations.

Limitations of MF-API / MF-JSON syntax occurred when several feature properties were observed at different times (such as by different sensors, at different sampling rates, etc.) and so forth.

13.1.3. Client

The client and other deliverables work well together. However, there is an improvement that can be made in Sensorhub. The only observations are in terms of data management relating to the same type of feature of interest such as the Saildrones.
Because of having different types of sensors, the Saildrone 1021 has different temporal properties than other saildrones. The D143 Superelectric Client has been implemented in order to manage these different types of observational data. Another useful observation is on the right data fusion for the storms where the HDOB type data has been inserted as an additional element of the “temporalGeometries” array, differentiating them with the “id”.

Per the temporal geometries:


```
{
  "temporalGeometries": [
    {
      "id": "tg-7k7p48g374pbu",
      "type": "MovingPoint",
      "datetimes": [
        "2022-09-15T13:20:00Z",
        "2022-09-15T13:20:30Z",
        "2022-09-15T13:21:00Z",
        "2022-09-15T13:21:30Z",
 ….
      ],
      "coordinates": [
        [12.2, -69.0],
        [12.2, -69.0],
        [12.2, -69.0],
        [12.2, -69.0],
 ….
      ],
      "interpolation": "Linear"
    },
    {
      "id": "tg-tm3kijpkaoei6",
      "type": "MovingPoint",
      "datetimes": [
        "2022-09-14T12:00:00Z",
        "2022-09-14T18:00:00Z",
        "2022-09-15T00:00:00Z",
        "2022-09-15T06:00:00Z",
        "2022-09-15T12:00:00Z"
      ],
      "coordinates": [
        [16.4, -49.1],
        [16.6, -50.1],
        [16.6, -51.3],
        [16.5, -52.4],
        [16.4, -53.6]
      ],
      "interpolation": "Linear"
    }
  ],
  "links": [
    {
      "rel": "next",
```

```
      "href": "https://api.georobotix.io/ogc/t18/mfapi/collections/storms/
items/sfmhi32ege8fm/tgeometries?offset=100",
      "type": "auto"
    }
  ]
}
```


Figure 53

Which unfortunately did not happen with the “temporalProperties”, where the “id” is not reported:


```
{
  "temporalProperties": [
    {
      "datetimes": [
        "2022-09-15T13:20:00Z",
        "2022-09-15T13:20:30Z",
        "2022-09-15T13:21:00Z",
        "2022-09-15T13:21:30Z",
 ….
      ],
      "aircraft_number": {
        "type": "TText",
        "form": "https://dbpedia.org/page/Aircraft_registration",
        "interpolation": "Linear",
        "values": ["AF303", "AF303", "AF303", "AF303", ….]
      },
      "geopotential_height": {
        "type": "TDouble",
        "form": "http://mmisw.org/ont/cf/parameter/geopotential_height",
        "interpolation": "Linear",
        "values": [NaN, NaN, NaN, NaN,….]
      },
….
….
      "rain_rate": {
        "type": "TDouble",
        "form": "http://mmisw.org/ont/cf/parameter/rainfall_rate",
        "interpolation": "Linear",
        "values": [NaN, NaN, NaN, NaN,]
      }
    },
    {
      "datetimes": [
        "2022-09-14T12:00:00Z",
        "2022-09-14T18:00:00Z",
        "2022-09-15T00:00:00Z",
        "2022-09-15T06:00:00Z",
        "2022-09-15T12:00:00Z"
      ],
      "windSpeed": {
        "type": "TDouble",
        "form": "http://mmisw.org/ont/cf/parameter/wind_speed",
        "description": "Maximum sustained winds",
        "interpolation": "Linear",
        "values": [30.0, 35.0, 45.0, 45.0, 45.0]
      },
      "minPressure": {
        "type": "TDouble",
        "form": "http://mmisw.org/ont/cf/parameter/air_pressure",
        "interpolation": "Linear",
        "values": [1009.0, 1006.0, 1002.0, 1002.0, 1002.0]
```

```
      },
      "category": {
        "type": "TText",
        "form": "http://sensorml.com/ont/T18/concepts/StormCategory",
        "interpolation": "Linear",
        "values": ["TD", "TS", "TS", "TS", "TS"]
      }
    }
  ],
  "links": [
    {
      "rel": "next",
      "href": "https://api.georobotix.io/ogc/t18/mfapi/collections/storms/
items/sfmhi32ege8fm/tproperties?offset=100",
      "type": "auto"
    }
  ]
}
```


Figure 54

This issue has created a problem in the client by proceeding with a work-around to make up for this lack. From the point of view of the client, it would be appropriate to homogenize the data as was done for the hurricane HDOBs data, with the difference of also adding the “id” for the temporal properties. It would be appropriate that the same thing could be implemented for the Saildrones as well.

13.2. Future Work

This section describes future work for some deliverables of the TB-18 MFSI task.

13.2.1. Ingestion service

The following future work is recommended.

1. Integration of moving features with static features for more sensors. Enriched information should be organized in a unified, lightweight, and compact data file. The integration of real-time heterogeneous data along with the moving features also needs to receive more attention. Sometimes, real-time information is more valuable than historical information for prompt decision making, especially in disaster events. For example, when a disaster happens, real-time information integration enables decision-making departments to comprehensively understand the situation that occurred in the current location so that timely response and actions will be taken to mitigate the damage.

13.2.2. Sensorhub

A SWG has been formed to work on a draft version of the Connected Systems API at OGC to better document and provide implementation guidelines for features of interest (sampling features) attached to moving platforms.

13.2.3. GeoPose: Multi-Camera Video Analysis

Collaborative plans have been made to record geotagged video data using a StreetDrone Twizy autonomous vehicle operated by the Ordnance Survey at their headquarters in Southampton, UK. Similar data were used to track a cyclist from a moving vehicle by calculating GeoPose in Testbed-17 Moving Features.

Figure 55 — StreetDrone Vehicle Operated By Ordnance Survey

The aim is to capture video from front- and rear-facing cameras aboard the vehicle concurrently with footage from multiple static locations. Code from Testbed-18 3D compass demos can be integrated with Away Team's TrkdCam app to record geotagged videos with camera orientations using smartphones as roadside cameras. The resultant data set should be suitable for demonstrating the essential elements of the road network use cases previously described, and could be extended to include streaming data for real-time use cases.

13.2.4. Client

Possible future developments could concern the final use by identifying as stack holders those users and/or security departments for monitoring of critical events such as the occurrence

of hurricanes, with the possibility of adding trajectory prediction and perhaps being able to immediately warn all subjects in danger, such as ships and aircraft. Naturally this implies the management of real-time dynamic data flows coming from different source “ingesters.”

A

ANNEX A (INFORMATIVE) REVISION HISTORY

A ANNEX A (INFORMATIVE) REVISION HISTORY

DATE	RELEASE	AUTHOR	PRIMARY CLAUSES MODIFIED	DESCRIPTION
2016-04-28	0.1	G. Editor	all	initial version
2022-11-25	0.2	B. Editor	all	version for review
2022-12-22	0.2	B. Editor	all	Feedback from reviewers — approval process
2023-01-15	0.2	B. Editor	bibliography, document number	Additional Feedback from reviewers — approval process
2023-01-25	0.3	B. Editor	bibliography, document number	Additional Feedback from reviewers — approval process

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] Ben Domenico: OGC 10-092r3, *NetCDF Binary Encoding Extension Standard: NetCDF Classic and 64-bit Offset Format*. Open Geospatial Consortium (2011). https://portal.ogc.org/files/?artifact id=43734.

[2] Akinori Asahara, Ryosuke Shibasaki, Nobuhiro Ishimaru, David Burggraf: OGC 14-084r2, *OGC® Moving Features Encoding Extension: Simple Comma Separated Values (CSV)*. Open Geospatial Consortium (2015). https://docs.ogc.org/is/14-084r2/14-084r2.html.

[3] Akinori Asahara, Ryosuke Shibasaki, Nobuhiro Ishimaru, David Burggraf: OGC 14-083r2, *OGC® Moving Features Encoding Part I: XML Core*. Open Geospatial Consortium (2015). https://docs.ogc.org/is/14-083r2/14-083r2.html.

[4] OGC API — Connected Systems (draft), 2022 https://github.com/opengeospatial/connected-systems.

[5] OGC: OGC 11-165r2: CF-netCDF3 Data Model Extension standard, 2012

[6] Lawrence Livermore National Laboratory: NetCDF CF Metadata Conventions – http://cfconventions.org/

[7] ESIP: Attribute Convention for Data Discovery (ACDD) – http://wiki.esipfed.org/index.php/