Open Geospatial Consortium

Publication Date: 2014-07-15

Approval Date: 2014-06-14

Posted Date: 2014-05-16

Reference number of this document: OGC 14-008

Reference URL for this document: http://www.opengeospatial.net/doc/PER/testbed10/aviation-architecture

Category: Public Engineering Report

Editor: Matthes Rieke

OGC® Testbed 10 Report on Aviation Architecture

Warning

This document is not an OGC Standard. This document is an OGC Public Engineering Report created as a deliverable in an OGC Interoperability Initiative and is not an official position of the OGC membership. It is distributed for review and comment. It is subject to change without notice and may not be referred to as an OGC Standard. Further, any OGC Engineering Report should not be referenced as required or mandatory technology in procurements.

Document type:	OGC® Engineering Report
Document subtype:	NA
Document stage:	Approved for public release
Document language:	English

Abstract

This document is a deliverable of the OGC Testbed 10 (Testbed-10). This document describes the architecture that was implemented in the Testbed-10 Aviation thread. Additionally, it provides descriptions of all software components involved in the Aviation architecture as well as a dedicated chapter focusing on the evaluation and design of FIXM 2.0. Here, a special focus lies on the integration into the data provisioning components, namely the Web Feature and Event Services.

Keywords

ogcdoc, ogc document, Testbed-10, aviation, architecture, fixm

OGC Web Services – Phase 10 Overview

The OGC test bed 10 builds on the outcomes of prior OGC initiatives (http://www.opengeospatial.org/resource/demos) and is organized around the following threads:

- Cross-Community Interoperability (CCI): Increase Geospatial community interoperability by building on CCI OWS-9 work in semantic mediation, volunteer geographic information (VGI), provenance and data quality, and Global Gazetteer. Explore the potential of interoperability in the hydrology domain and utilizing ontologies to more easily share and visualize geospatial data.

- Open Mobility: Explore the geospatial standards requirements needed to support the growing emerging mobile environment where client applications are mobile, information services are mobile, and increasingly distributed across cloud infrastructures. The Open Mobility thread will address these requirements while leveraging on the work achieved in the OWS-9 Testbed in the areas of Geopackages and Geopackaging services and new OWS Context encodings.

- Aviation: Develop and demonstrate the use of the Aeronautical Information Exchange Model (AIXM) and the Flight Information Exchange Model (FIXM), building on the work accomplished in prior testbeds to advance the applications of OGC Web Services standards in next generation air traffic management systems to support European and US aviation modernization programs.^

The Testbed-10 sponsors are:

- AGC (Army Geospatial Center, US Army Corps of Engineers)

- ESA (European Space Agency)

- EUROCONTROL

- (US Federal Aviation Administration)
- GeoConnections - Natural Resources Canada
- Corporation
- Martin Corporation
- NGA (US National Geospatial-Intelligence Agency)
- NOAA National Weather Service
- USGS (US Geological Survey)
- DSTL (UK Defence Science and Technology Lab)

License Agreement

Contents

Page

Figures

Page

Tables

Page

Listings

Page

OGC® Testbed 10 Report on Aviation Architecture

1 Introduction

1.1 Scope

This OGC® document describes the architecture implemented in the OGC Test Bed 10 (Testbed-10) Aviation thread, including:

- A description of the architecture used for the implementation of the Testbed-10 Aviation Use Cases
- A detailed description of the scenarios developed within the test bed
- An overview of the implemented components and workflows followed by a short description of each component
- The design decisions and discussions carried out in the field of FIXM 2.0 integration into the Aviation service architecture
- A brief overview of related Engineering Reports developed during the test bed (including the dissemination of weather data, the exchange of terrain data and the evaluation of FIXM 2.0 as part of the Aviation architecture
- Documentation of the issues and lessons learned as well as accomplishments and scenarios that were of general interest in the Aviation thread.

1.2 Document contributor contact points

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

Name	Organization
Matthes Rieke (MRI), editor (m.rieke<at>52north.org)	52°North GmbH
Matthias Pohl (pohl<at>m-click.aero)	m-click aero
Thomas Forbes (thomas.forbes<at>snowflakesoftware.com)	Snowflake Software
Johannes Echterhoff (echterhoff<at>interactive-	Interactive Instruments GmbH

instruments.de)	
Daniel Balog (daniel.balog@luciad.com)	Luciad NV
Ronald Berger (ronald.berger@frequentis.com)	Frequentis Vienna
Yuanzheng Shao (yshao3@gmu.edu)	George Mason University
Richard Martell (rmartell@galdosinc.com)	Galdos Systems Inc.

1.3 Revision history

Date	Release	Editor	Primary clauses modified	Description
2014-01-31	0.1.0	MRI	all	Initial draft report
2014-04-16	0.9.0	MRI	all	Pre-final version of the report
2014-05-16	1.0.0	MRI	1.4, 6	Edits for the final version of the report

1.4 Future work

Improvements in this document and the related work areas are desirable to the following topics:

- **Deeper embedding of the Web Coverage Service into the service architecture** – as many of the occurring events and updates on ATM data and features are related to changes in the weather conditions a profound integration of the WCS into the overall service architecture could be taken into consideration.

- **GML based encoding for FIXM 2.0** – In order to mature FIXM and to make it a well-established, interoperable data format new version of FIXM could take the recommendations of the [OGC 14-037] into consideration. This document provides detailed topics and aspect that could be covered with such work.

- **Extend the integration of FIXM features** – in Testbed-10 the main focus on FIXM data lied on using the FlightObject feature. In future test beds additional FIXM feature types could be integrated into the design process of scenarios.

1.5 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.

2 References

The following documents are referenced in this document. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply. For undated references, the latest edition of the normative document referred to applies.

OGC 06-121r3, OGC® Web Services Common Standard

In addition to this document, this report includes several XML Document files as specified in Annex A.

3 Conventions

3.1 Abbreviated terms

AIP	Aeronautical Information Publication
AIM	Aeronautical Information Manual
AIXM	Aeronautical Information Exchange Model
AMDB	Airport Mapping Database
AOC	Airline Operational Communication
ATM	Air Traffic Management
CCI	Cross Community Interoperability
CDMS	Client DMS module
COTS	Commercial Off The Shelf
CSW	OGC Catalogue Service for the Web
DEM	Digital Elevation Model
DMS	Data Management Service
DNOTAM	Digital NOTAM
DS-client	Dispatch synchronization client
EAD	European Organisation for the Safety of Air Navigation
EFB	Electronic Flight Bag
ePIB	Electronic Pre-flight Information Briefing
ES	Event Service
ESA	European Space Agency
FAA	Federal Aviation Administration
FES	Filter Encoding Specification
FPS	Feature Portrayal Service
GML	Geography Markup Language
HTTP	HyperText Transfer Protocol
IP	Internet Protocol
ISO	International Standards Organization
METAR	A format for reporting weather information
NOTAM	Notice To Airmen
OASIS	Organization for the Advancement of Structured Information Standards
OGC	Open Geospatial Consortium

OWS	OGC Web Service
OWS-10	OGC Web Services (OWS), phase 10 (Renamed Testbed-10)
QoS	Quality of Service
QName	Qualified Name
SAA	Special Activity Airspace
SESAR	Single European Sky ATM Research Programme
SLD	Styled Layer Descriptor
SOAP	Simple Object Access Protocol
SPS	SWIM Product Standardization
SWIM	System-Wide Information Management
TAF	Terminal Area Forecast
TCP	Transmission Control Protocol
URL	Uniform Resource Locator
URN	Uniform Resource Name
WCS	Web Coverage Service
WFS	Web Feature Service
WFS-T	WFS Transactional
WMS	Web Map Service
WPS	Web Processing Service
WSDL	Web Services Description Language
WS-N	OASIS Web Services Notification
WXXM	Weather Information Exchange Model
XML	Extensible Markup Language

3.2 UML notation

Most diagrams that appear in this report are presented using the Unified Modeling Language (UML) static structure diagram or sequence diagram, as described in Subclause 5.2 of [OGC 06-121r3].

4 Testbed 10 Report on Aviation Architecture overview

The Aviation service architecture of the Testbed-10 Aviation thread builds upon the work carried out in the previous test beds. The components are separated into three layers of functionality:

- Data Access Layer
- Business Processing Layer
- Client Layer.

Figure 1 illustrates the general architecture. Besides these three layers, additional components that are not included into the common workflows have been designed and implemented within Testbed-10. In particular, work on conformance testing of AIXM and GML encoding and on the automatic generation of data bindings for programming languages has been carried out.

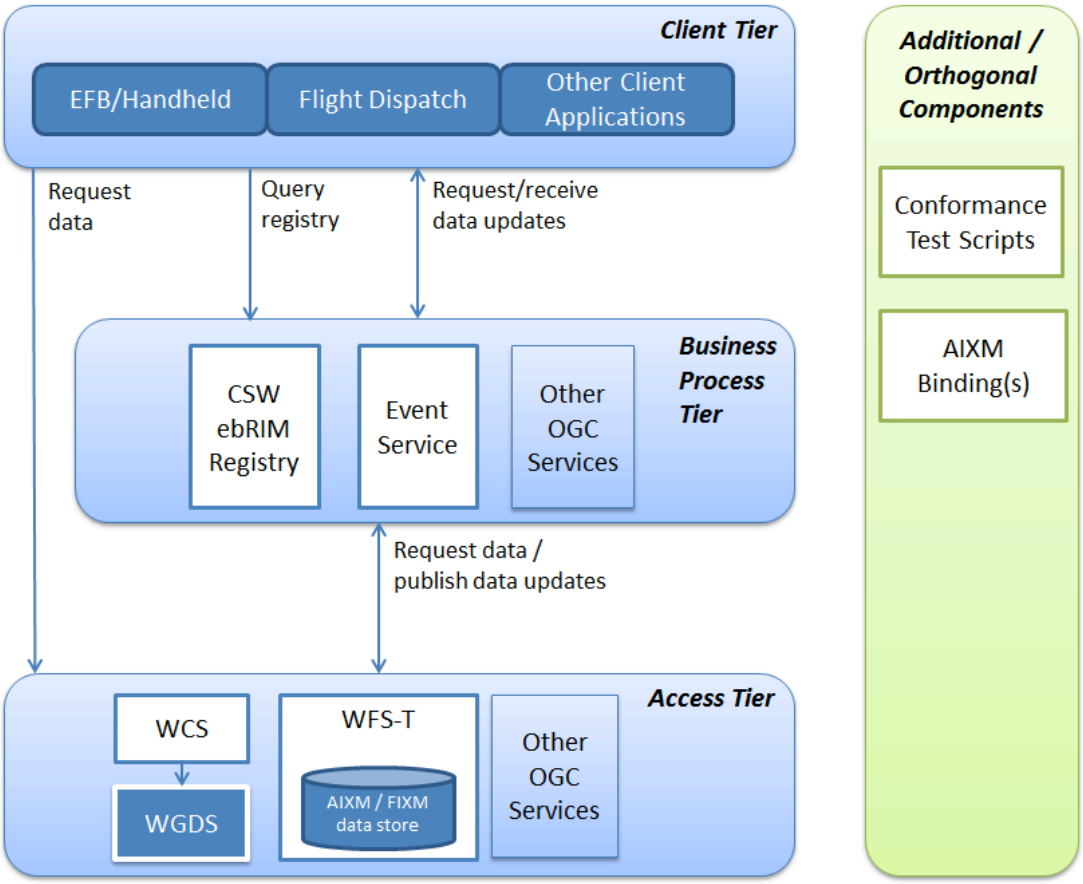


Figure 1: Aviation Service Architecture.

The remaining parts of this report provide insights on the architecture developed during the test bed. Section 5 illustrates service workflows and application patterns. In section 6 a brief summary on every developed component is provided, followed by a description of the scenarios in section 7 that have been developed for demonstrating the service architecture. Subsequently, related fields of work are presented in section 8. Finally, the lessons learned, accomplishments and recommendations are summarized in section 0, 10 and 0.

5 Common Workflows and Application Patterns

This section provides an overview of the software component interactions. Most interactions are illustrated using UML sequence diagrams.

5.1 Feature Data Access

The feature data building the foundation of the Aviation architecture is provided via OGC Web Feature Service (WFS) 2.0 instances. In order to demonstrate the interoperability two distinct implementations serving the same data sets have been developed. Figure 2 illustrates the basic request/response pattern for accessing feature data.

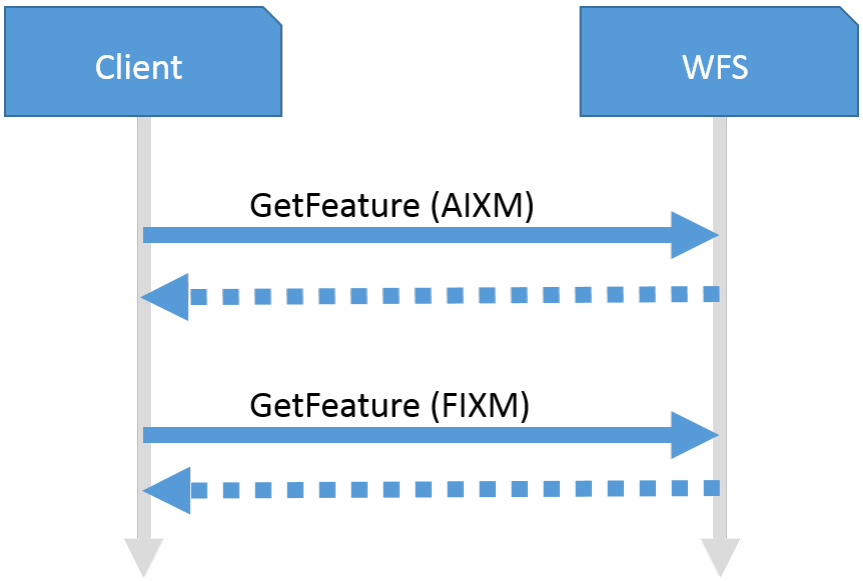


Figure 2: Client Interaction with WFS.

The client retrieves feature data from the WFS via the GetFeature operation. General information can be requested, for example, on airspaces and airports. Using specific filter criteria, it can also query the WFS to identify suitable alternate/diversion airports (e.g. via a spatial bounding box filter). The WFS instances provide both AIXM data as well as FIXM data. The FIXM data is used by the client to visualize information on Flights such

as the Route, diversions to this as well as metadata for a given Flight (e.g. dangerous goods cargo).

5.2 Weather Data Access

For accessing weather data an OGC Web Coverage Service (WCS) 2.0 instance has been developed. It is backed by the Web Gridded Document Service (WGDS) developed by NOAA's National Weather Service. The workflow is similar to accessing AIXM features via the WFS. Figure 3 shows the interaction. The client requests a coverage document via the GetCoverage operation. Using a set of parameters (bounding box, temporal filters) the resulting coverage collection can be specified. The WCS returns a response in the WXXM data format. Details on the weather data dissemination are provided in section 0.

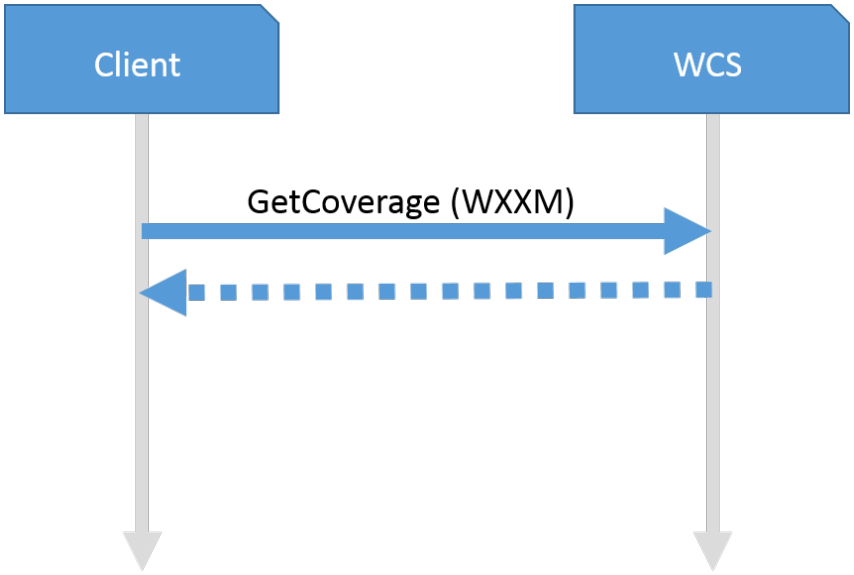


Figure 3: Client Interaction with WCS.

5.3 Publish/Subscribe Dissemination

The Aviation service architecture makes use of the publish/subscribe paradigm for providing updates on features in a push based fashion. The different components of the architecture can be assigned to different roles as used in a publish/subscribe environment.

5.3.1 Component Roles

- **Notification consumer** – a consumer is the component that defines an interest in a specific subset of the feature updates (e.g. FlightObject with a given ID) via a Subscribe request. It is capable of receiving feature updates that match the given subscription. In the Aviation architecture, the **Aviation Client** implements the Notification consumer.

- **Notification Producer** – a producer is responsible for providing updates on features once they are available. The **WFS 2.0 instances** take up this role. Once a transactional operation (e.g. WFS-T Insert) took place, the affecting feature is forwarded to the Notification Broker.

- **Notification Broker** – this component is responsible for managing the subscriptions as well as matching updates on features against these. Once a feature update matches a subscription, the update is pushed to the corresponding Notification consumer. In the Aviation architecture the **Event Service** implements this role.

5.3.2 Workflow

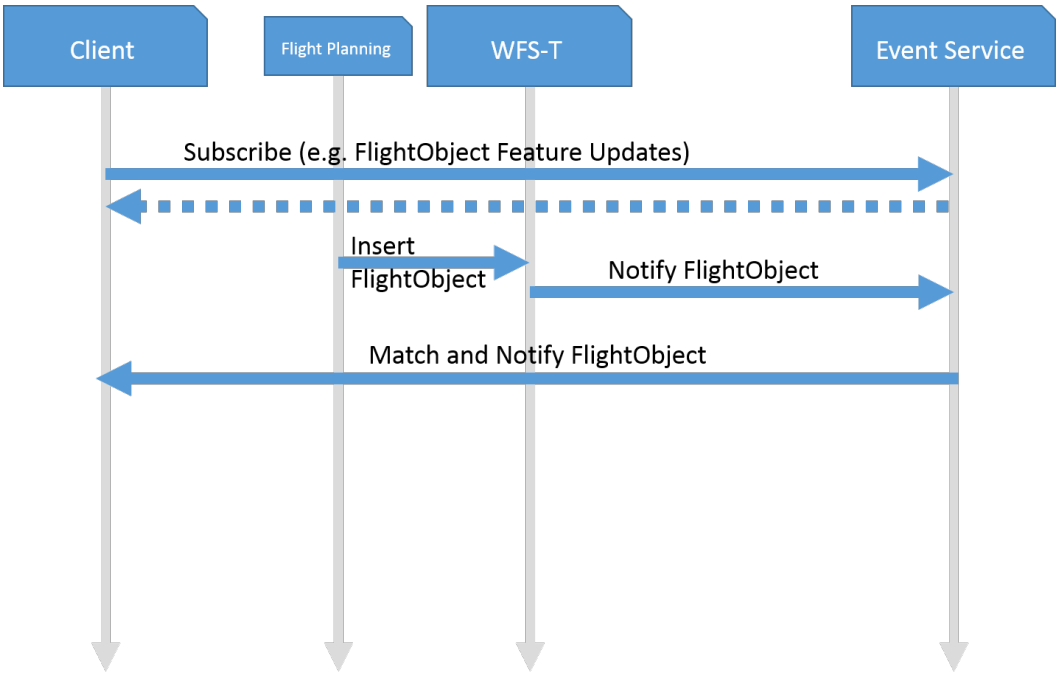


Figure 4: Publish/Subscribe Interaction.

Figure 4 illustrates the general publish/subscribe workflow. The Aviation client acting as a consumer subscribes at the Event Service for a subset of feature updates (here: updates on FIXM FlightObject features). Once a new FlightObject is inserted or updated at the WFS-T a corresponding Notification is sent to the Event Service. The Event Service internally evaluates the Notification against the registered subscriptions. A matching Notification is then forwarded to the Client.

5.3.3 Event Encoding for AIXM

Updates on AIXM features are encoded as a Digital NOTAM. Listing 1 provides an example of such an encoding. A Digital NOTAM is formed using an `AIXMBasicMessage` document. Two members are contained within it: the `Event`,

providing metadata on the creation of the feature update, and the updated feature itself. Here, the updated feature is a `Navaid` with a change in its `operationalStatus`. A Digital NOTAM makes use of the temporality concept of AIXM as defined in [DNOTAM ES].

Listing 1: Example Digital NOTAM Notification.

```
<wsnt:Notify xmlns:wsnt="http://docs.oasis-open.org/wsn/b-2">
  <wsnt:NotificationMessage>
    <wsnt:Message>
      <msg:AIXMBasicMessage gml:id="gmlID14726"
      xmlns:msg="http://www.aixm.aero/schema/5.1/message"
xmlns:gml="http://www.opengis.net/gml/3.2" xmlns:aixm="http://www.aixm.aero/schema/5.1"
xmlns:dnotam="http://www.aixm.aero/schema/5.1/event">
        <msg:hasMember>
          <dnotam:Event>
            <gml:identifier codeSpace="urn:uuid:">BCB4F904-9AEA-4F2E-97AE-
2D4A4344BD6C</gml:identifier>
            <dnotam:timeSlice>
            ...
            </dnotam:timeSlice>
          </dnotam:Event>
        </msg:hasMember>
        <msg:hasMember>
          <aixm:Navaid gml:id="gmlID14730">
            <gml:identifier codeSpace="urn:uuid:">BCB4F904-9AEA-4F2E-97AE-
2D4A4344BD6D</gml:identifier>
            <aixm:timeSlice>
              <aixm:NavaidTimeSlice gml:id="gmlID14731">
              ...
              <aixm:interpretation>TEMPDELTA</aixm:interpretation>
              <aixm:sequenceNumber>1</aixm:sequenceNumber>
              <aixm:correctionNumber>5000</aixm:correctionNumber>
              ...
              <aixm:availability>
              <aixm:NavaidOperationalStatus gml:id="gmlID14734">
              <aixm:operationalStatus>UNSERVICEABLE</aixm:operationalStatus>
              </aixm:NavaidOperationalStatus>
              </aixm:availability>
                <aixm:extension>
                  <dnotam:NavaidExtension gml:id="gmlID14735">
                    <dnotam:theEvent xlink:href="urn:uuid:BCB4F904-9AEA-4F2E-97AE-2D4A4344BD6C"/>
                  </dnotam:NavaidExtension>
                </aixm:extension>
              </aixm:NavaidTimeSlice>
            </aixm:timeSlice>
          </aixm:Navaid>
        </msg:hasMember>
      </msg:AIXMBasicMessage>
    </wsnt:Message>
  </wsnt:NotificationMessage>
</wsnt:Notify>
```

5.3.4 Event Encoding for FIXM

As FIXM 2.0 does not provide a temporality concept such as AIXM 5.1, the encoding of updates on FIXM features differs. For Testbed-10 it has been decided to provide the full FIXM feature to the Event Service, even if just one property has changed. Listing 2 shows the realization using the example of a FlightObject. The GML based encoding of FIXM 2.0 is described in [OGC 14-037].

Listing 2: Example FIXM Event Encoding.

```
<wsnt:Notify xmlns:wsnt="http://docs.oasis-open.org/wsn/b-2">
  <wsnt:NotificationMessage>
    <wsnt:Message>
        <ffo:Flight xmlns:wfs="http://www.opengis.net/wfs/2.0"
            xmlns:ffo="http://www.opengis.net/ows10/fixm/FlightObject"
            gml:id="ows10.20140416T091315.OWS10A">
            <ffo:flightStatus>
                <ffo:FlightStatus>
                    <ffo:filed>FILED</ffo:filed>
                </ffo:FlightStatus>
            </ffo:flightStatus>
            <ffo:flightIdentification>
                <ffo:FlightIdentification>
                    <ffo:aircraftIdentification>OWS10A</ffo:aircraftIdentification>
                </ffo:FlightIdentification>
            </ffo:flightIdentification>
            <ffo:flightType>SCHEDULED</ffo:flightType>
            <ffo:gufi>ows10.20140416T091315.OWS10A</ffo:gufi>
        </ffo:Flight>
    </wsnt:Message>
  </wsnt:NotificationMessage>
</wsnt:Notify>
```

6 Component Descriptions

6.1 Galdos GML for Aviation Compliance Test Suite

The test development work focused on validating geometry representations commonly appearing in AIXM 5.1 data sets that adhere to the AIXM-GML profile [OGC 12-028]. In particular, tests were developed for the following GML and AIXM geometry elements:

- gml:Point (aixm:Point, aixm:ElevatedPoint)
- gml:Curve (aixm:Curve, aixm:ElevatedCurve)
- gml:OrientableCurve
- gml:CompositeCurve
- gml:Surface (aixm:Surface, aixm:ElevatedSurface)

NOTE The following curve segments are included in the AIXM-GML profile: gml:ArcByCenterPoint, gml:CircleByCenterPoint, gml:Arc, gml;Circle, gml:GeodesicString, gml:Geodesic, gml: LineStringSegment.

The geometry elements covered by the tests are shown in Figure 5. It is important to note that the tests are **not** tied to any particular application domain. The validator will check all instances of the standard GML geometry elements *plus* any application-defined geometries that can substitute for them; the application schemas are inspected to discover these additional geometry elements.

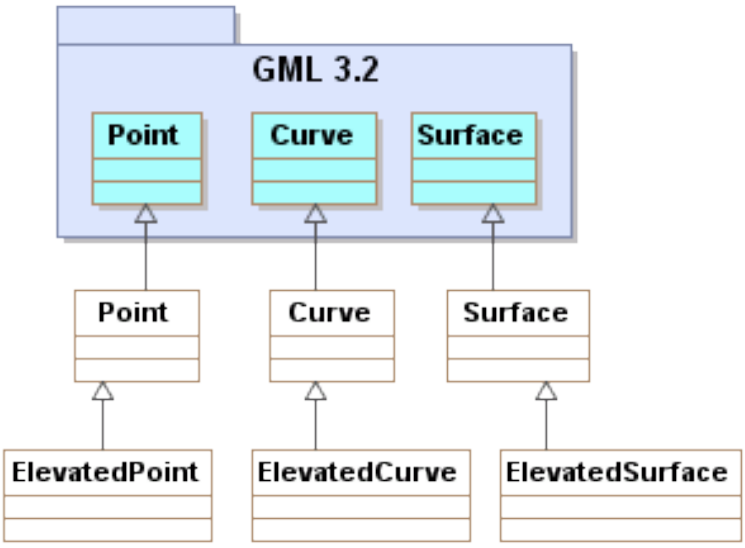


Figure 5: AIXM-GML geometry type coverage.

The geometry tests were implemented as extensions of the GML 3.2.1 conformance test suite. The suite may be run using the official OGC test harness, TEAM-Engine 4.0. The OGC maintains a testing facility that provides access to the latest test suites at this location: <http://cite.opengeospatial.org/te2/>.

6.1.1 Test execution

The interface provided by the TEAM-Engine web application presents a form wherein the location of the instance document to be validated is specified by an absolute URI value. Alternatively, the GML document may be uploaded for validation (Figure 6).

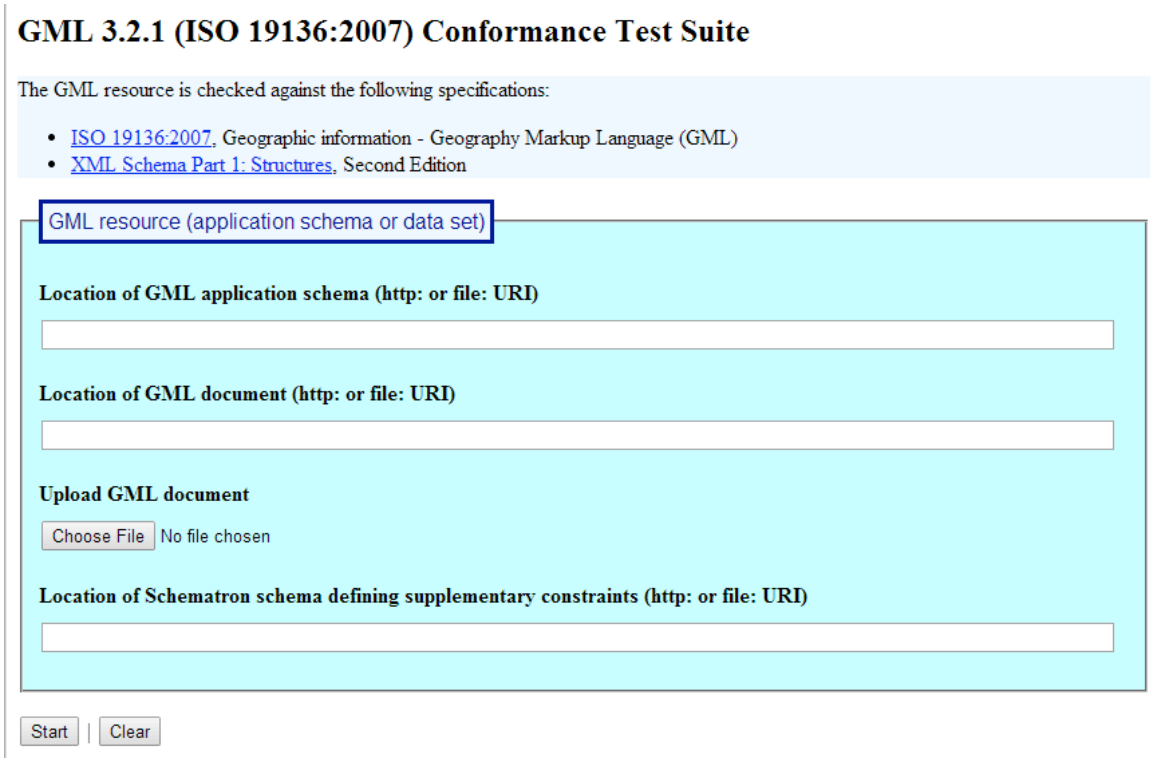


Figure 6: Test execution (web interface).

A REST API provides an alternative method of executing the test suite by submitting a GET request to the test controller endpoint. The test run arguments are summarized in Table 1. The *Obligation* descriptor can have the following values: M (mandatory), O (optional), or C (conditional).

Table 1: Test run arguments

Name	Type	Obligation	Description
xsd	URI	C (required if 'gml' is not supplied)	Refers to a GML application schema[a]
gml	URI	C (required if 'xsd' is not supplied)	Refers to a representation of a GML data instance[a]
sch	URI	O	Refers to a Schematron schema[b]

a. Ampersand ('&') characters appearing within a query parameter value must be percent-encoded as %26.
b. A Schematron schema that defines supplementary data constraints. See ISO 19757-3:2006.

The request URI has the following form (substitute the appropriate host name and port number for the actual test harness):

```
http://localhost:8080/teamengine/rest/suites/gml/3.2.1-
r15/run?gml=uri
```

where *uri* denotes an absolute URI that refers to the instance document to be validated; the 'http' and 'file' URI schemes are acceptable. The result entity is an XML document with <testng-results> as the document element.

6.2 GMU WCS WGDS Adapter

The National Weather Service's Meteorological Development Laboratory developed and has maintained a web service that supplies National Digital Forecast Database (NDFD) forecasts encoded in a local dialect of XML named Digital Weather Markup Language (DWML). This service and this dialect of XML are NOT OGC Compliant.

GMU WCS WGDS Adapter provides an OGC-compliant Web Coverage Service (WCS) HTTP GET method-based interface based on the WGDS SOAP API. OGC WCS defines three basic operations: GetCapabilities, DescribeCoverage and GetCoverage, which had been implemented via GMU's contribution. The following figure illustrates the workflow of WCS adapter for WGDS.

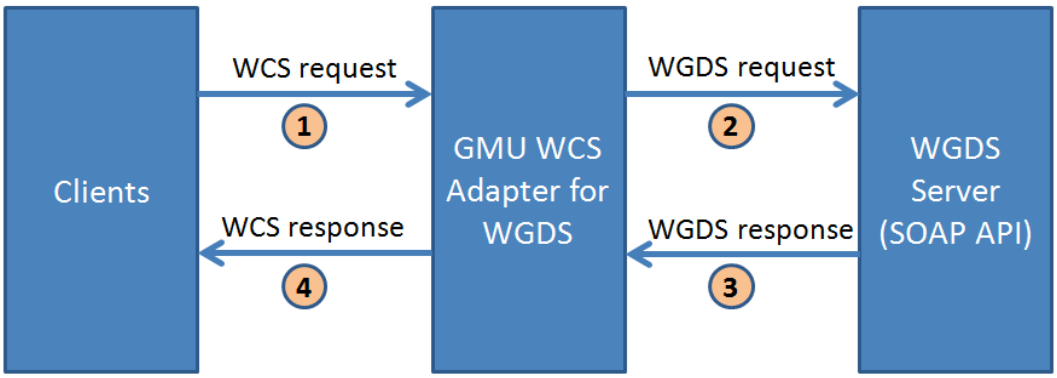


Figure 7: Workflow of GMU WCS adapter for WGDS.

6.2.1 Purpose in Testbed-10

The GMU team was assigned the task to implement an adapter which converts a standardized WCS request to a WGDS request, and convert WGDS response to WCS response. The client, such as m-click can interact with WCS adapter to query the data of interest, and integrate the data into various applications through standardized interface.

6.2.2 Interfaces

The WCS adapter was deployed to GMU server. The following URL describes the request for Capabilities document of WCS adapter:

http://129.174.131.8:9003/ows10/adapter.do?service=WCS&version=2.0&request=GetCapabilities

There are four coverage are supported in GMU WCS adapter for WGDS, including time-series_meteogram, glance_meteogram, 12_hourly_summary and 24_hourly_summary.

6.2.3 Accomplishments

The key accomplishments for GMU WCS Adapter for WGDS in Testbed-10 include:

1) provides OGC-compliant WCS interface to serve NDFD data
2) supports standardized WXXM output format
3) reuses “subset” parameter to specify the point (latitude/longitude) in WCS request

6.2.4 Parameters mapping

The parameters to native WGDS operation are inconsistent with the WCS parameters.

The inputs to native WGDS SOAP API request include:

1. One of the 4 types of product identifier: time-series meteogram, glance meteogram, 12 hourly summary, 24 hourly summary.
2. A list of latitude/longitude point(s) user requests data for.
3. Time Info in dateTime form (2013-08-01T08:00:00).
4. XMLDocType: DWML or WXXM
5. Unit:The Unit defaults to English ("e") U.S. standard units if the XMLDocType chosen is "WXXM"

It’s a major issue to expose these parameters in WCS GetCoverage request without breaking the WCS specification. The following table describes the mapping relation between the native WGDS API and the WCS API.

WGDS SOAP API	**WCS Adapter API**
list of coordinates	using “subset” parameter, for example: *subset=point,EPSG:4326(lat,long)*
weather elements	using “rangeType” parameter, for example: *rangeSubset=maxt;mint*

date and time	using “subset” parameter, for example: *subset=phenomenonTime("2006-08-01T00:00:00","2014-08-22T00:00:00")*
XMLDocType	using “format” parameter, for example: *format=WXXM*
Unit	N/A

6.2.5 Challenges and solutions

The challenges of implementing WCS adapter for WGDS and corresponding solutions are described in [OGC 14-038].

6.3 Luciad Aviation Client

To support the integration & testing of the various service components developed within Testbed-10 Aviation, Luciad contributed an Aviation Client component. This component is based on Luciad’s COTS product LuciadLightspeed, which offers numerous capabilities & benefits that are of direct use in the client: full support for AIXM (including its temporality and metadata models), FIXM, WXXM, connectors to OGC services such as WMS, WCS and WFS-T, flight simulation, 2D & 3D visualization, … On top of this product, a thin layer has been developed to support custom functionality needed in Testbed-10 and to provide the user with a dedicated user interface focusing on the Testbed-10 tasks. Figure 8 shows a screenshot of the Luciad client’s user interface.

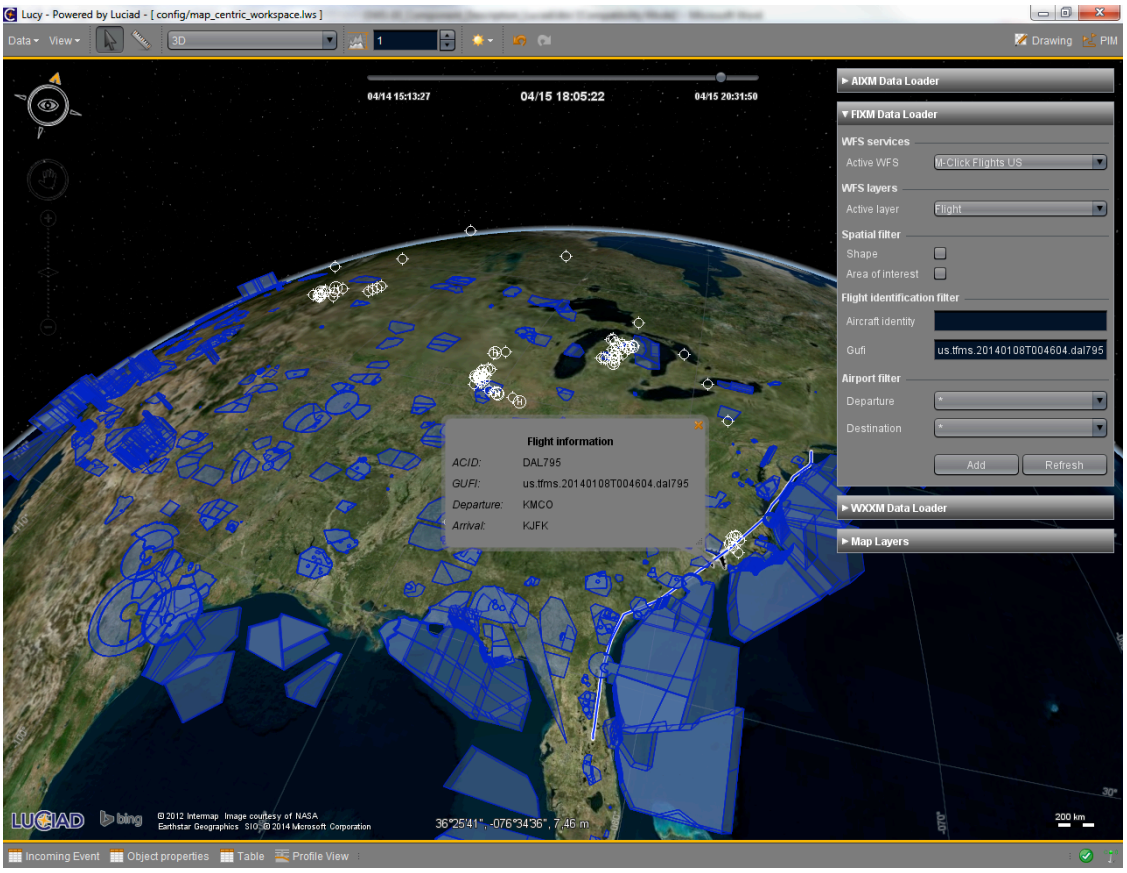


Figure 8: The Luciad Aviation Client in action.

6.3.1 Functional overview

The Luciad Aviation client provides the following functionality to support Testbed-10 Aviation:

- Map-centric display with intuitive user interface giving access to various actions, including:
  - Map controllers to manipulate the map (zoom, pan, …)
  - Map layer control with access to predefined background data layers
  - AIXM, FIXM and WXXM-based OGC web service connectors
  - Flight preview / simulation
  - Visualization in 2D & 3D
  - Visual representation & browsing of feature properties inside a balloon
- Client interface to query data from an OGC Web Feature Service delivering AIXM data. Users can select the desired WFS, AIXM feature type and various filtering options (e.g., spatial filters).

- Client interface to query data from an OGC Web Feature Service delivering FIXM data. Users can select the desired WFS and various filtering options, including spatial filters and property-based filters such as the gufi, the departure, the destination and the aircraft id.
- Client interface to file flight plans and publish simulated aircraft positions in the FIXM format, via an OGC Web Feature Transactional Service.
- Client interface to query data from an OGC Web Coverage Service delivering WXXM data. Users can select the desired WCS, coverage and various filtering options (e.g., spatial filters).
- Client interface to a WMS 1.1.1 & 1.3.0 service delivering background data. Users can query and retrieve layers from a WMS.
- Client interface to the Event Service. Users can subscribe / unsubscribe to events (also using OGC Filters) and consume incoming events (e.g., visualization of an incoming Digital NOTAM event).
- Rendering engine with support for SLD / SE 1.0/1.1 to render vector feature and raster data. This engine integrates extensions to support ICAO Annex 4 rendering guidelines for aeronautical data.
- Support to represent & browse ISO 19115-compatible metadata and to encode / decode to / from an ISO 19139-compatible data source. The use of metadata extensions / profiles is supported.
- Wide range of data format support, including:
  - AIXM 5.0 & 5.1 for aeronautical data (including extensions).
  - FIXM 2.0 (Testbed-10 version).
  - WXXM 1.1 for weather data (including extensions).
  - Additional aeronautical and weather data formats like AIXM 3.3/4.0/4.5, ARINC 424, DAFIF, ASTERIX, ASDI and GRIB.
  - Raster format support (GeoTIFF, TIFF, JPEG, JPEG 2000, GMLJP2, JPIP, PNG, GIF, ECW, MrSID, CADRG/ADRG/USRP, DTED, USGS DEM, Oracle GeoRaster …) for satellite imagery and elevation background data.
  - Vector format support (ESRI Shape, MapInfo MIF/MAP, GML 2/3.1.1/3.2.1, SVG, DGN, DWG, Oracle/Informix/MSSQL databases …) for vector-based background data.
  - Other formats: OBJ, OpenFlight, OGC KML 2.2 and GeoPDF.

6.3.2 Deployment characteristics

All development is done in Java, using the Java Development Kit (JDK) 1.7. The client application is developed on top of Luciad's COTS product LuciadLightspeed. The software runs on any operating system for which a Java Virtual Machine 1.7 or higher exists. For the 3D visualization, a graphics card with support for OpenGL 1.2 or higher is required.

6.3.3 Challenges

A reoccurring challenge when working on the client is obtaining proper interoperability with all services and overcoming implementation differences. Adherence to the specification of data formats and OGC services is only one step, as there is typically always room for implementation differences that impede interoperability in practice. We therefore want to emphasize the usefulness of official compliancy tests such as the ones developed by OGC (e.g., for the WMS, WFS and WCS), as they can help to increase overall interoperability. It might be an idea for the future to develop official compliancy tests to verify the guidelines and best practices have been developed the past years with respect to using OGC services in the Aviation domain.

6.3.4 Accomplishments

The key accomplishments for Luciad's Aviation client component in Testbed-10 include:

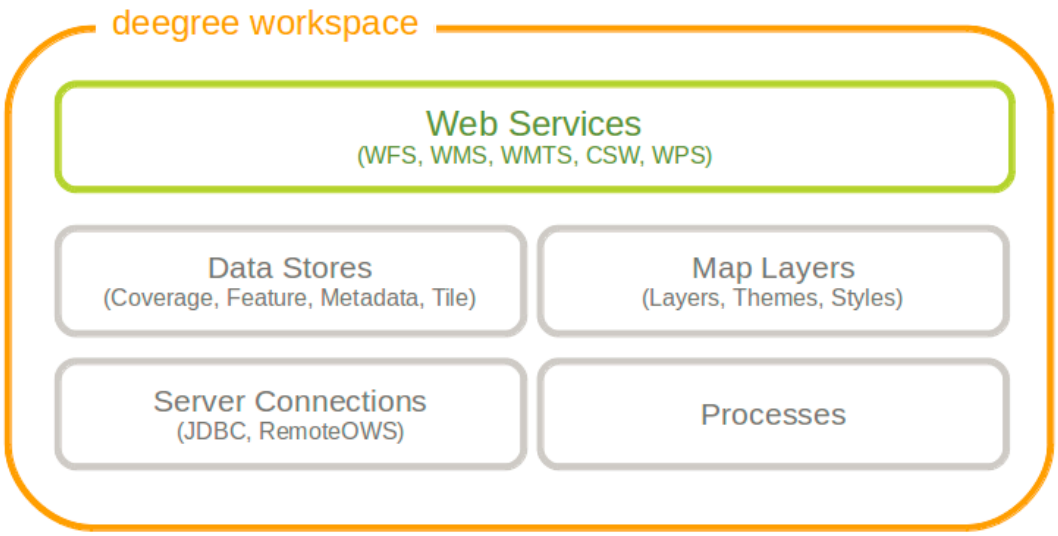
- Demonstration interaction with all relevant service components provided within the Aviation thread.
- Successfully demonstrated decoding, visualization and analysis of combined AIXM, FIXM en WXXM data in a single client, with all data obtained through OGC compliant services.
- Demonstrated the use of WFS-T to create, modify and delete FIXM 2.0 data on a service.
- Demonstrated the use of an Event Service to keep track of updates to FIXM 2.0 Flight Objects.
- Implemented and tested the use of WCS as a provider for WXXM data.
- Added support for Digital NOTAM 2.0 with visualization according to SAE-G10 guidelines.
- Tested the use of TIXM to represent elevation data.
- Contribution of a rich client equipped with lots of capabilities and features that help to demonstrate the Testbed-10 Aviation scenario and to perform testing and integration with a wide variety of service components

- Collaboration with Testbed-10 service component providers on the developed functionality

6.4 m-click Web Feature Service – Transactional (WFS-T)

The m-click Web Feature Service – Transactional (WFS-T) 2.0 is based on the deegree GML and web services framework, an established Open-Source, Java-based framework for advanced geospatial applications. Complete support for AIXM was added by m-click to meet the requirements of different trials (e.g. NM B2B Trial in 2012) in the context of SESAR.

deegree now serves as the base technology for building state-of-the-art AIXM solutions and web services. It offers a suite of state-of-the-art OGC web service implementations with a high level of standards-compliance and scalability. Here's a rough overview on deegree's components:



6.4.1 Purpose in Testbed-10

In Testbed-10 the m-click WFS-T was used as FIXM (GML) and AIXM data store to support WFS-T functionality according the demo scenario. It also serves as a data source for the Event Service, that is triggered on every data store update.

FIXM data from different sources was transformed to FIXM GML and loaded into the m-click WFS-T. Additional FIXM GML updates were inserted directly into the WFS-T and retrieved by the client using complex filters with logical and spatial operators.

6.4.2 Implementation

The deegree WFS is an implementation of the OGC Web Feature Service specification, which covers:

- WFS standard 1.0.0, 1.1.0 and 2.0.0 (upcoming OGC reference implementation)
- Support of WFS-T
- KVP, XML and SOAP requests
- GML 2/3.0/3.1/3.2 output/input
- Support for GetGmlObject requests and XLinks
- High performance and excellent scalability
- On-the-fly coordinate transformation
- Mapping of GML application schemas to relational models
- ISO 19107-compliant geometry model: Complex geometries (e.g. non-linear curves)
- Advanced filter expression support based on XPath 1.0
- Passes OGC WFS CITE test suites (including all optional tests)

On top of this, some specific new functionality has been implemented for Testbed-10.

- Support for FIXM GML Application Schema, developed within Testbed-10
- Support for the Testbed-10 Event Service that is triggered on updates of the FIXM data store to publish the information in real time.
- Support for the replace feature in combination with FIXM Event Service.
- Data loader XSLT scripts to perform complex processing and correction tasks on the data from different sources.

6.4.3 Data loading and service deployment

Data loading and processing was performed to provide the required AIXM and FIXM data for the demo scenarios.

- AIXM data sets have been extracted from previous OWS-9 services, validated, corrected and loaded into the m-click WFS-T services for European data and United States data.
    - EU-WFS-T containing European data
    - US-WFS-T containing North American data
- FIXM 2.0 Flight Object data has been received from Harris FOXS and converted into FIXM GML according the latest FIXM GML Schema.

- FIXM-WFS-T containing converted data from FOXS and directly inserted data from Luciad client

The service is deployed on the m-click Testbed-10 servers and accessible through the published endpoints. Access is restricted with login to the Testbed-10 participants that have signed the data agreements.

For database storage, we chose the so-called 'blob' mode offered by deegree's GML data store implementation. In contrast to a full relational decomposition (which is also offered by deegree), this approach works without subsequent SQL selects and allows very high GML download rates.

6.4.4 Challenges

6.4.4.1 Download and cleanup OWS-9 AIXM data for use in WFS-T

Downloading the OWS-9 AIXM data from the Snowflake WFS services took a few hours and was interrupted by network connection issues on our side.

After downloading, we were unable to insert this data directly, because the data does not validate against the AIXM 5.1 schema, which is a requirement for inserting it into the Deegree WFS.

We created an XSLT script that fixes all issues and converts the original data to valid AIXM 5.1 data. Finally, we were able to insert that modified AIXM data into the Deegree WFS and to make it available to others.

6.4.4.2 Retrieve Harris FIXM 2.0 data from FOXS

Retrieving the FIXM 2.0 flight datasets from FOXS raised a few issues for us.

- At the beginning we were unable to connect to the Cisco VPN of Harris, through which we had to connect to the FOXS service. We had to invest some effort to figure out what the reason was. Eventually we switched
  from Shrew Soft IKE client to the plain standard VPNC client for Linux (Debian package vpnc) that worked well and was easily configured.

- The first attempts to connect to FOXS resulted in exceptions and error messages, which were fixed by Harris as soon as we reported them. Also, there were some downtimes of the FOXS service, which were mostly announced beforehand and haven't been a blocker for us.

- We regularly hit the maximum number of replies per request. We worked around it by using many different attribute filters. The origin and destination attributes were perfect for that. In the first step, we compiled an almost complete list of airports, in the second step we requested the flights for all possible origin/destination combinations of airports, because geographic filters are not implemented yet in FOXS.

6.4.4.3 Transform and cleanup FIXM 2.0 data for use in WFS-T

The FOXS service supports the FIXM formats 1.0, 1.1 and 2.0. We requested FOXS datasets in FIXM 2.0 format as this was most closely to the final FIXM GML format developed during this Testbed-10.

- The FOXS service returns the FIXM 2.0 data embedded in JSON. So the first task was to write a small tool that removes the additional JSON encoding and groups the flights in a custom container XML element. That way, we had an intermediate result which we would process by further XML tools.

- For inserting the data into our WFS, we decided to operate on XML level and to write an XSLT script. This XSLT script is not a complete implementation that transforms all aspects of FIXM 2.0 to FIXM GML. There is a default rule that ensures that any unknown element or attribute is transformed to a custom error XML element. The script is well tested with almost 120,000 flights retrieved from Harris FOXS. All FOXS results which are valid FIXM 2.0 were converted to valid FIXM GML data.

- FIXM GML requires lots of gml:id values for elements which had no ID in FIXM 2.0. These are generated via XSLT's standard function "generate-id()" whose implementation is up to the XSLT processor. A more evolved solution would be to generating those IDs in a reproducible fashion, e.g. by combining natural keys and/or document positions. However, this is a tricky task and may generate duplicate IDs if not executed carefully enough. For the Testbed-10 test bed, we used generate-id(), even though this means that the same input FIXM 2.0 file may lead to slightly different output FIXM GML documents.

- The XSLT script's output is not just a GML feature collection, but a valid WFS-T Insert statement. That way, the result can be piped directly into any WFS-T.

6.4.5 Testbed-10 specific accomplishments

6.4.5.1 Adjust the GetCapabilities result for use with Reverse Proxies

Since the WFS server runs behind an HTTP reverse proxy, the GetCapabilities request showed the internal URLs instead of the external URLs. Before Testbed-10, this minor drawback was hardly worth mentioning, because the "meat" of GetCapabilities was perfectly correct, and because we were usually connecting to the WFS only via very specialized clients.

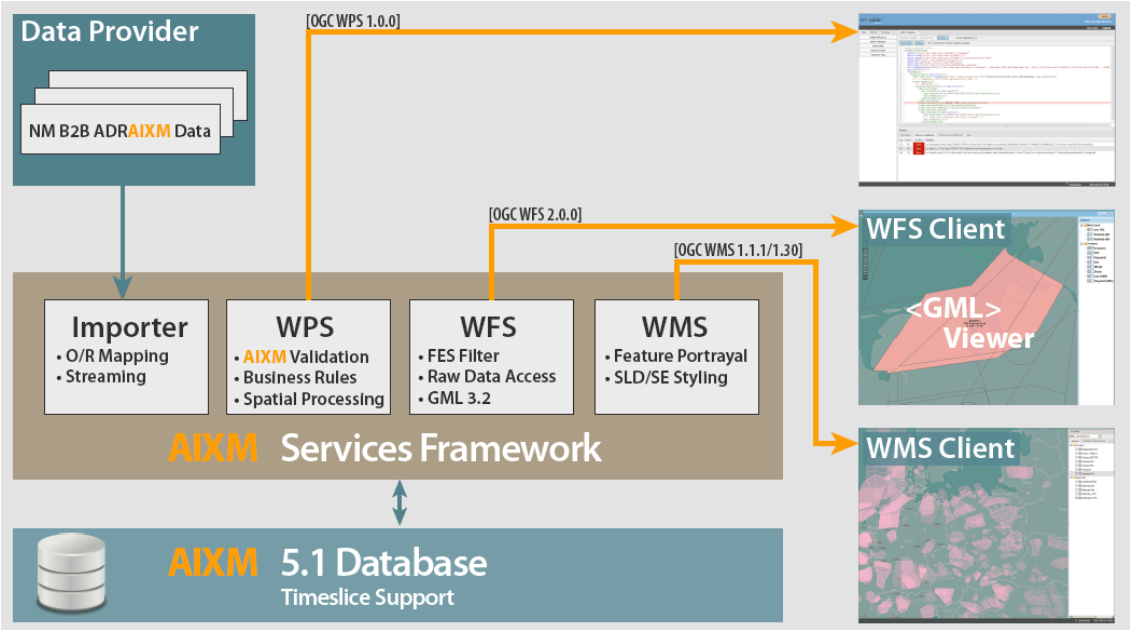
In Testbed-10, we noticed that this is a major issue for the Luciad client, whose fully generic protocol negotiation mechanisms depends on exact, working URLs returned by the GetCapabilities request. Fixing these URLs was harder than expected: The web server's reverse proxy did send the correct HTTP headers for determining the real request origin. The Deegree web service also had ready-to-use mechanisms to find the real request origin automatically. However, somewhere in the software stack between those, this information got lost. Now this minor issue is fixed.

6.4.5.2 Analyze problems in WFS client/server communication

During testing with Luciad's WFS client, some requests could not be processed and also no reasonable error message was given by the server. The problem with the error message required a fix to the WFS setup. The actual problem in the requests were minor issues in the WFS client. These issues have been identified succesfully and reported to Luciad.

6.5 m-click AIXM Validation Service

The AIXM Validator Service is a web service for generating comprehensive validity reports for XML-encoded AIXM datasets. It has been implemented as an OGC-compliant Web Processing Service (WPS) that can be accessed using any standard-compliant WPS client. Additionally, an easy-to-use web interface is available that allows to load and edit AIXM data in a web browser and display the validation results in a user-friendly way. The validator is part of the m-click AIXM services and has been implemented using the deegree framework.



6.5.1 Purpose in Testbed-10

Generating perfectly valid AIXM datasets seems to be quite a challenge. At the moment, it's rather common to find errors in AIXM datasets which render them unprocessable. The purpose of the validator service is to provide an easily usable tool for all Testbed-10 participants to perform comprehensive validity checks of any AIXM dataset.

XML Schema validation is probably a common task for AIXM application developers. Using the online AIXM Validator it is easy to detect and correct XML Schema violations without any additional software.

Besides schema-validity, the validator checks for business rules validity. Business rules are complex dependencies and domain specific requirements that must be satisfied by

AIXM datasets. These rules cannot be expressed using XML Schema (XSD), but are encoded using the Schematron language and custom XPath function. The validator uses Schematron rules based on the rule set from Eurocontrol's AIXM Rule Checker (ARC-Web) and the ongoing efforts in the AIXM-BR community project.

6.5.2 WPS compliance

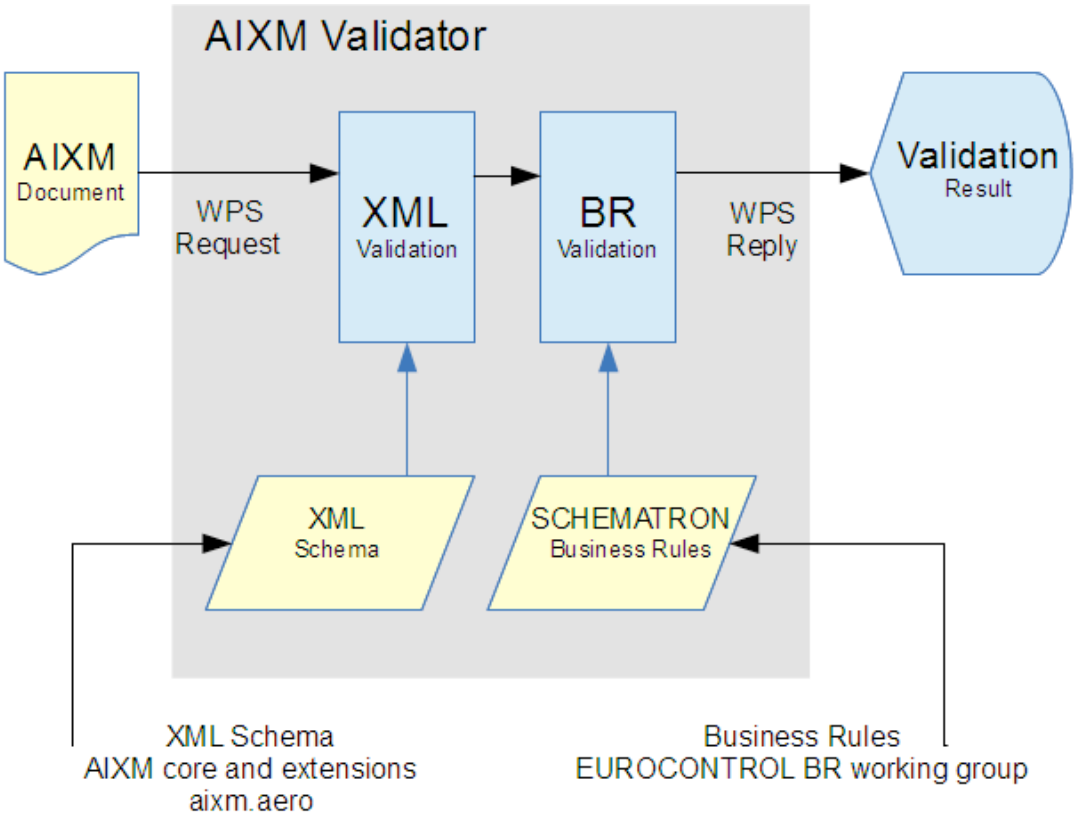
The validator has been implemented as a custom process for the deegree WPS. Therefore, it is fully compliant to the OGC WPS 1.0.0 specification and features the following possibilities:

- Processing of KVP, XML and SOAP requests
- Streaming access for complex input/output parameters
- Processing of huge amounts of data with minimal memory footprint
- Support of storing of response documents/output parameters
- Support of input parameters given inline and by reference
- Support of asynchronous execution (with polling of process status)

6.5.3 AIXM Validation in two phases

The validation process performs two phases:

- Schema (XSD) validation
- Business Rules validation



The validation report is returned to the WPS client or displayed in the web interface.

6.5.3.1 XML Schema Validation

The AIXM Schema is a GML application schema that encodes the AIXM objects and properties. XML Schema Validation has been implemented using the Xerces2J parser and the latest published AIXM core and extension schemas. It maintains a local copy of the official AIXM schemas with support for different root elements for specialized extensions (e.g. eASM). Schema location (xsi) information will be suppressed to apply always the maintained copy of the official AIXM schemas.

All Schema validation failures are returned in the WPS response using a simple XML structure that contains precise information (e.g. column and line where the error occurred) to easily pinpoint and fix the error.

6.5.3.2 Schematron Business Rules Validation

The AIXM Schema file defines the XML structure (allowed elements and attributes, ordering). However, not every constraint can be expressed using XML schema. These constraints (e.g. dependencies between the values of different attributes) are called "business rules" and have been encoded using the Schematron language. After the XML schema validation the Schematron business rule validation is performed. Each Schematron rule consist of two XPath expressions:

- Context - set of elements to be checked using this rule

- Assertion / Test - boolean expression that must evaluate to true for all elements in the context

The validator uses the ISO implementation of Schematron with some custom XPath functions. This means:

- The Schematron file is transformed to an XSLT script, using multiple ISO Schematron XSLT scripts which are executed in cascade.
- The resulting XSLT script is applied to the input data.
- Custom XPath functions are used for some checks that are hard to express using the standard function set (e.g. topological operators)

The Schematron result is an XML document containing an exact list of all assertion failures, as well as some general information about the validation. This document is transformed to a more user-friendly XML structure (similar to the Schema validation result) and also included in the response that is returned to the WPS client.

6.5.4 AIXM validations performed in Testbed-10

m-click used the validation service to check the AIXM datasets from OWS-9 for validity. Several problems were found and have been reported to the Testbed-10 participants. Additionally, the same report has been used to fix the issues, so they could be imported successfully into the m-click WFS-T component.

6.5.5 Setup

The validator component has been installed on m-click's servers. It is accessible via the Internet, but has been access protected, to ensure that access is restricted to Testbed-10 participants.

6.5.6 Challenges

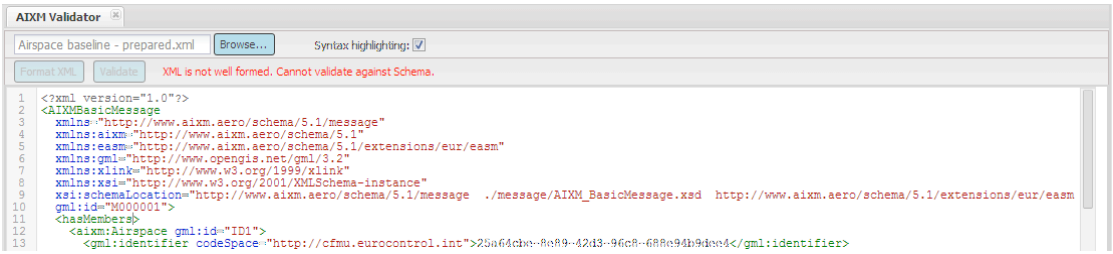
A key challenge was to find a suitable and robust way to encode the AIXM dataset in the WPS request. Originally, we used a so-called complex parameter for that. However, this approach showed two shortcomings:

- Non-well-formed AIXM datasets lead to WPS service exceptions, because the full XML request became non-well-formed. In principle, this is reasonable behavior, but not very convenient for users of the service. It is preferable to express this error in the validity report.
- Line and column information are relative to the full Execute request and do not relate directly to the AIXM dataset.

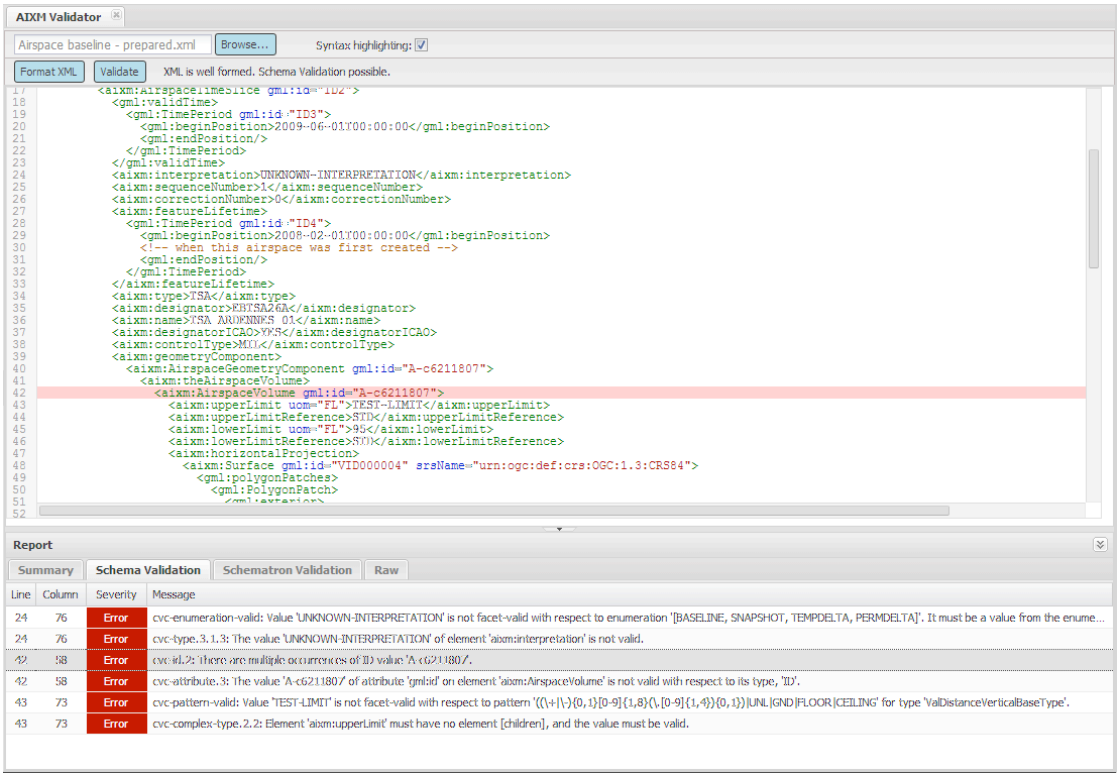
Both shortcomings have been solved by applying entity-encoding to the input AIXM dataset and using a simple string parameter in the WPS process.

6.5.7 m-click AIXM Validator web interface

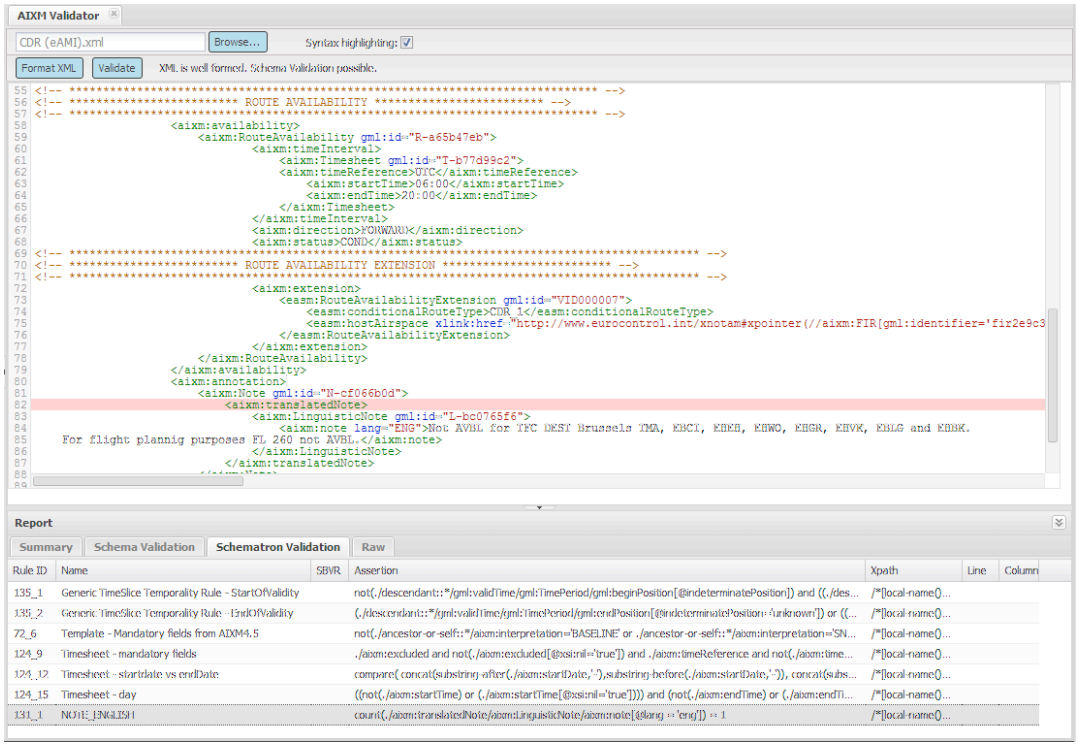
As a ready-to-use alternative to sending raw WPS requests, the web interface provides a web-based XML editor for loading and editing smaller AIXM datasets. As a first step, the XML document is checked for well-formedness:



If the AIXM document passes this check, the validation process can be executed. The document is sent to the AIXM Validation WPS and the result set will be displayed in the summary tab below the AIXM editor.



If XML Schema issues are detected, a list of all Errors and Warnings will be displayed together with the row and column number. A click on a single error in the table scrolls the AIXM document to the appropriate position where the error occurs.



code position where the business rule was not satisfied.

If Schematron Business Rules issues are detected, a list of all affected rules together with the XPath expression that describes the location in the AXIM dataset will be displayed. A click on a single issue in the table scrolls the AIXM document to the code position where the business rule was not satisfied.

6.6 Snowflake Software Web Feature Service – Transactional (WFS-T)

For the Aviation Thread of Testbed-10 Snowflake Software used its commercial off-the-shelf (COTS) products GO Publisher and GO Loader which are comprised of a series of flexible, scalable components supporting the transformation and data exchange requirements of aeronautical information systems as illustrated in Figure 9.

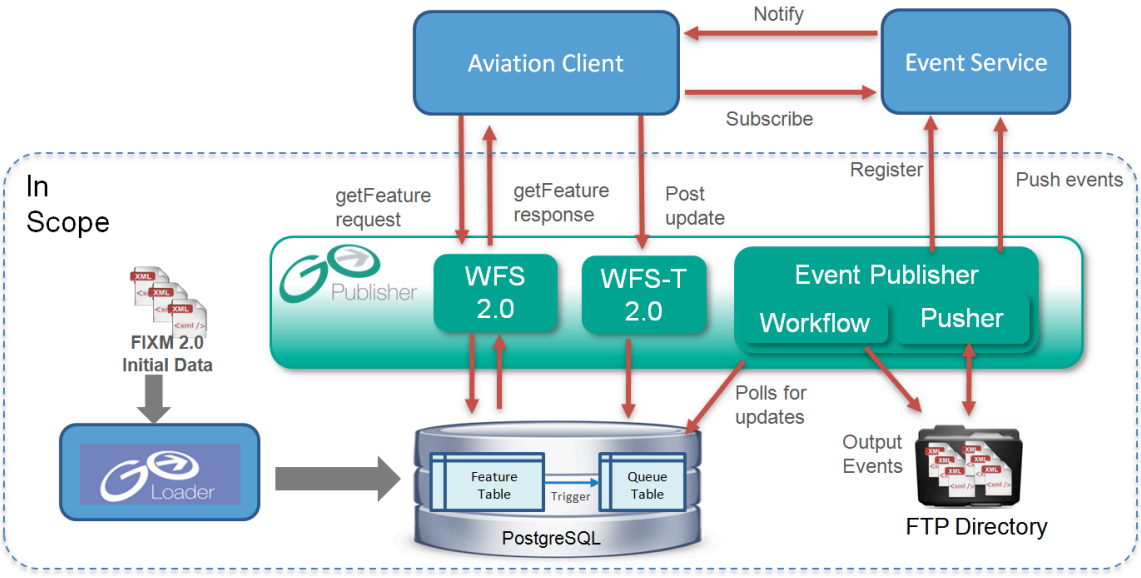


Figure 9: Overview of the Snowflake Aviation Component Architecture

Flight Information Exchange Model (FIXM) 2.0 data were received from Harris Flight Object Exchange Services (FOXS) and integrated into a single FIXM 2.0 database. The Snowflake Software component architecture consisted of three components:

- Web Feature Service (WFS) 2.0
- Transactional WFS (WFS-T) 2.0
- Event Publisher

6.6.1 WFS 2.0 (read-only)

An instance of the read-only WFS 2.0 was established to represent the FIXM data source. The WFS was configured so that FIXM 2.0 data were published as ISO 19136 (Geographic Information – GML) compliant data using the FIXM 2.0 GML schema developed as part of the Testbed-10 Aviation Thread. The WFS is capable of querying by Globally Unique Flight Identifier (GUFI) using a stored query.

6.6.2 WFS-T 2.0

A separate transactional Web Feature Service (WFS-T) 2.0 instance was established to support the insertion of FIXM data. The WFS-T was integrated into the data maintenance architecture for the consolidated database. On insertion of a Feature, a series of processes were then triggered to auto-generate additional information to publish the data via the European and United States WFS and Event Publisher in real-time.

6.6.3 Event Publisher

The Event Publisher is composed of two components:

- **GO Publisher Agent:** a server-side bulk data publishing system that generates event messages.
- **Event Pusher:** registers with one or more event services as an event source and pushes the messages generated to the event service brokers.

Publication of Events is triggered by the insertion of a feature into the database via a WFS-T insert transaction. The GO Publisher Agent generates an Event and transfers it to an Event Pusher which forwards the message to the Event Services.

6.6.4 Component Functionality

No new component functionality was developed for either GO Publisher WFS or the Event Publisher during Testbed-10. GO Publisher WFS currently implements a large proportion of the OGC WFS 2.0 specification and the Event Publisher continues to have sufficient functionality to connect to the Event Service.

The WFS-T provides capabilities to consume FIXM 2.0 data directly from existing FIXM data providers without prior modification or conversion therefore permitting simpler system integration.

6.6.5 Data available via the Components

The Testbed-10 consolidated database consisted of FIXM 2.0 data from the following sources:

- Harris Flight Object Exchange Service (FOXS).

Data also contained test events created to support the various scenarios that were inserted into the consolidated database via the WFS-T.

6.6.6 Accomplishments

Several key accomplishments were developed for Testbed-10 within the maintenance and publication architecture developed by Snowflake:

- The WFS-T consumes FIXM 2.0 data whilst a WFS query returns FIXM 2.0 GML data.
- The WFS-T was configured to allow events to be inserted into the database. On insertion additional processes for publication were activated via Event Publisher and WFS.

6.6.7 Challenges

FIXM specific challenges are described in the [OGC 14-037].

6.7 52°North Aviation Event Service

The Aviation Event Service implementation is provided by 52°North. It is based on the OGC Sensor Event Service discussion paper [OGC 08-133] and the corresponding Open Source project[1]. Following the Publish/Subscribe paradigm of OASIS' Web Service Notification (WS-N) family of standards it acts as a notification broker. The 52°North ES has been used in collaboration with the Institute for Geoinformatics in aviation threads of former OGC test beds starting with OWS-6, including the FAA SAA Dissemination Pilot. Figure 10 shows the general workflow as being applied in the Aviation thread of Testbed-10.

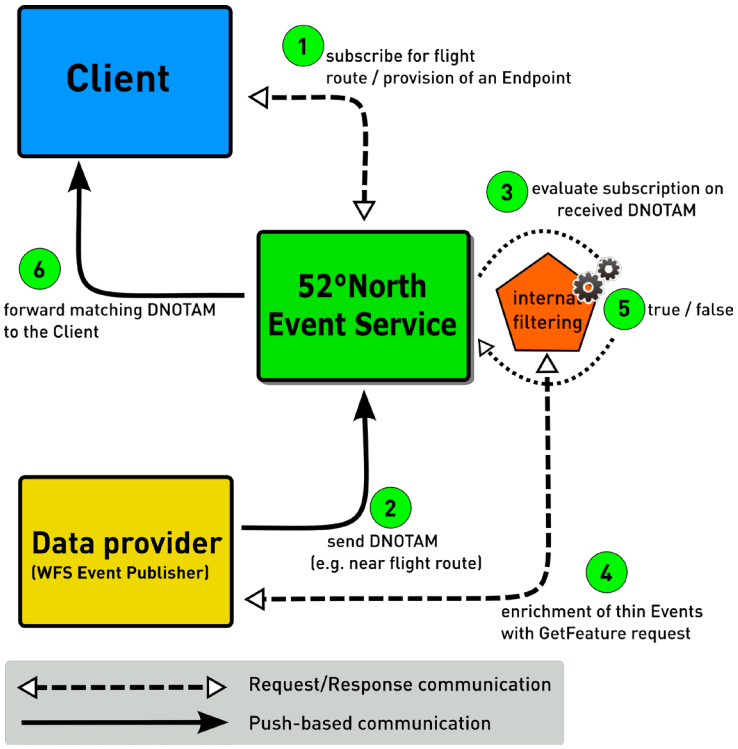


Figure 10: 52°North Event Service Workflow.

6.7.1 Interfaces

The ES consists of three endpoints. The PublisherRegistrationManager can be used to register data providers at the ES instance but is not used in this testbed and hence not described.

Endpoint	*Available Methods*	*URL*

[1] https://github.com/52North/SES.

Broker	GetCapabilities, Notify, Subscribe	http://ows.dev.52north.org:8080/AviationEventService/services/Broker
SubscriptionManager	Unsubscribe, PauseSubscription, RenewSubscription	http://ows.dev.52north.org:8080/AviationEventService/services/SubscriptionManager

To enable SOAP bootstrapping both endpoints provide a WSDL using HTTP Get method (<endpoint-url>?wsdl).

Besides the GetCapabilities method, all available methods are defined by OASIS WS-N. The methods used in this test bed are described in the following table.

Method	*Description*	*involved Testbed-10 Component*
Notify	Used to push a NotificationMessage to the ES. The contents are Digital NOTAM Events or position update messages of an aircraft (sent by clients for demo purposes) and separated into different topics (NOTAM-Topic, FIXM-Topic, Weather-Topic). This method does not use request-response communication, as notifications are pushed to the ES.	WFS providers, Clients (for demo purposes)
Subscribe	Used to define a subset of all incoming notifications. A subset can be defined using Topic filters, XPath Expressions and FES 2.0 filters. A SubscriptionReference is returned to the client, enabling the management of it. A Subscription can have an InitialTerminationTime (encoded as a period or a date time) to define its lifetime.	Clients
Unsubscribe	Used to cancel a Subscription (e.g. when the flight has arrived).	Clients

6.7.2 Testbed-10 Specific Changes to the Event Service Implementation

Compared to the ES used in the previous OGC test beds some major improvements have been made during this test bed. The 52°North Event Service implements all features created within the previous OWS-9 test bed, summarized as the “Advanced filtering functionality”. Besides these sophisticated methods on the filtering and dissemination of events, the FIXM 2.0 GML encoding developed during Testbed-10 has been integrated as a supported data source. This enabled clients to define filters on FIXM features using the same concepts as when filtering on AIXM features.

The work carried out within the Testbed-10 Report on Aviation Binding AIXM to Development Tools [OGC 14-007] has been practically applied within the service implementation. In particular, the generated JAXB data bindings for AIXM have been used to process updates on AIXM features.

6.8 52°North AIXM Binding Generation Tools

In the scope of the Testbed-10 Report on Aviation Binding AIXM to Development Tools [OGC 14-007] 52°North has a developed an Open Source project aiming and the simplification of integrating AIXM data into existing applications. The foundation for the project is the creation of automatically generated data bindings for both Java (using JAXB) and C# (using .NET frameworks xsd.exe). To ease the processing of AIXM data, a toolset of convenience methods has been developed on top of the generated bindings.

This toolset encapsulates the (de)serialization efforts from the actual feature objects. Therefore, a developed using the toolset does not have to take care on the JAXB specifics and concentrate on the actual access to the features and their properties. Listing 3 illustrates one of the convenience patterns.

Listing 3: AIXM Binding Tool Example.

```
public void processAIXMData(InputStream dataStream) throws
JAXBException {
    AIXMUnmarshallerMarshaller gmu = new AIXMUnmarshallerMarshaller();
    JAXBElement<?> je = gmu.unmarshal(dataStream);

    Object aixmType = je.getValue();

    if (aixmType instanceof NavaidType) {
        processNavaid((NavaidType) aixmType);
    }
    //other data types
}
```

6.8.1 Source Code

The source code for the Java toolset based on JAXB is available at:

https://github.com/52North/aixm-bindings/

The source code for the C# toolset based on .NET xsd.exe is available at:

https://github.com/52North/aixm-bindings-sharp

7 Scenarios

This section provides an overview on the scenarios developed for demonstrating the newly developed functionality during the Testbed-10.

7.1 Main Scenario

The main scenario applied in the Testbed-10 test bed is based on the one developed in OWS-9. An intercontinental flight departs from an Airport in Europe (see Figure 11). Its destination is the Chicago O'Hare Airport, alternative destination is the General Mitchell International Airport of Milwaukee. The focus of the main scenario is on the integration of FIXM 2.0 data into the general service workflow.

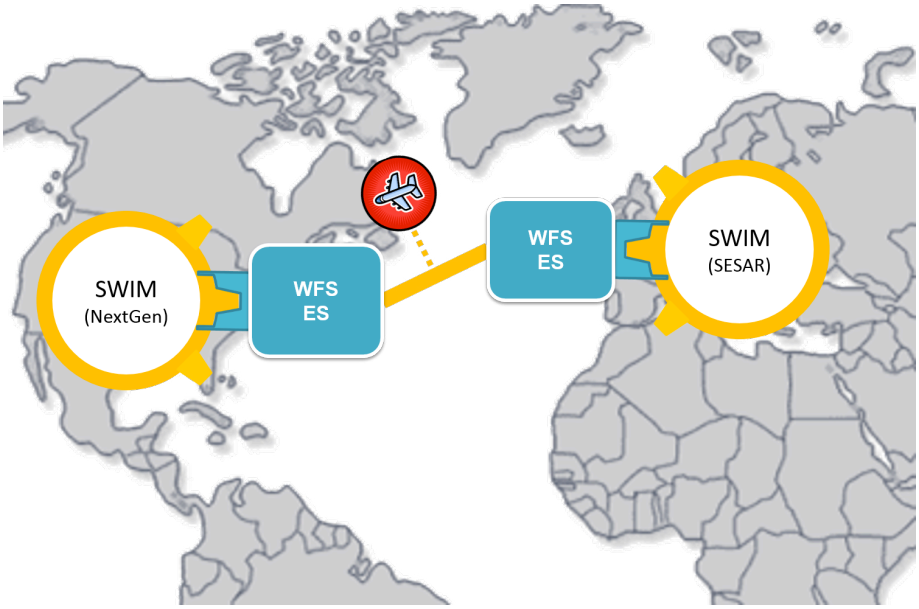


Figure 11: Main Scenario Overview.

The following table provides an overview on the single steps carried in the main scenario. The steps are grouped into flight phases:

- FP = Flight Planning
- IF = In-flight
- PF = Post-flight

Table 2: Steps of the Main Scenario.

Step	Phase	Description	Type	Involved Components	Sender	Receiver
0a	FP	Dispatcher retrieves aeronautical information	Client visualization, WFS (bbox)	Client (dispatch),		

			query	WFS		
0b	FP	Dispatcher reviews airspaces and routes for intended route, to see if they are active / closed	Client visualization	Client (dispatch)		
1	FP	A Flight Plan (encoded as a FIXM FlightObject) is inserted into the WFS - ICAO2012 conform	Narrative, WFS-T Insert	Client (dispatch), WFS	Client (dispatch)	WFS
2a	FP	An Event about the filling of the flight plan is created and pushed to the ES as as result of Step 1	ES Notify	WFS, ES	WFS	ES
2b	FP	Client receives the notification	ES Notify	ES, Client (in-flight)	ES	Client (in-flight)
3	FP	Additional information from related data sources are added to the FlightObject - Cargo loading manifest	WFS-T Replace	WFS	Client (dispatch)	WFS
4a	FP	An Event about the additional information is created and pushed to the ES as as result of Step 3	ES Notify	WFS, ES	WFS	ES
4b	FP	in-flight client receives the notification	ES Notify	ES, Client (in-flight)	ES	Client (in-flight)
5	FP	The client reviews the FlightObject before take-off	Client Visualization	Client (in-flight)		
6		The aircraft receives clearance for takeoff	Narrative			
7a	IF	Aircraft positional updates are added during the flight	WFS-T Replace (entire route + enroute position	client (dispatch), WFS	Client (dispatch)	WFS

			property)			
7b	IF	corresponding Event from 7a	ES Notify	WFS, ES	WFS	ES
7c	IF	in-flight client receives the notification	ES Notify	ES, Client (in-flight)	ES	Client (in-flight)
8a	IF	Dispatch client queries WCS to see weather condition for destination aerodrome. -> bad weather detected	Narrative, Client Visualization	Client (dispatch)	Client (dispatch)	WCS
8b	IF	An aerodrome closure event (due to the bad weather) for the destination airport is published.	Client visualization /editing	Client (dispatch), ES	Client (dispatch)	ES
9	IF	The client receives a DNOTAM about the aerodrome closure	Client Visualization, ES Notify	Client (in-flight), ES Notify	ES	Client
10	IF	As a result of step 9, an en-route diversion is added to the FlightObject - destination airport is updated - route is updated (route points already passed are kept, the rest will be updated with the new route)	WFS-T replace (by dispatch client)	Client (dispatch), WFS	Client (dispatch)	WFS
11	IF	The updated flight object is published		WFS, ES	WFS	ES
11b	IF	The in-flight client receives the update on the FlightObject's route	Client Visualization, ES Notify	Client (in-flight), ES	ES	client (in-flight)
12		The aircraft arrives at the alternate destination	Narrative			
13	PF	The FlightObject's status is updated after the flight has been completed - setting flight status	WFS-T update (via replace), Client Visualization	WFS, Client (dispatch)	Client (dispatch)	WFS

		to 'completed'				

8 Related Fields of Work

A set of related Engineering Reports describing concepts that have a bearing on the Aviation architecture have been developed during the Testbed-10. This section provides a short overview on these concepts.

8.1 FIXM 2.0 – GML Design and Evaluation

Built on a foundation of OGC and ISO standards, AIXM and WXXM have demonstrated that significant barriers to adoption can be mitigated through engaging with the wider open standards community and reusing best practice patterns for application schema design and service delivery.

At the data level AIXM and WXXM share common types and patterns from ISO 19136 Geography Markup Language (GML) making them interoperable between each other as well as saving cost and time through the reuse of globally implemented types such as feature models, geometry, temporality, unit of measure and coordinate reference systems. Being built on such a globally adopted standard as GML, AIXM and WXXM are compatible with hundreds of existing tools as well as being understandable by communities across the globe.

The Testbed-10 Report on FIXM GML Schema [OGC 14-037] demonstrates that the benefits of interoperability and open standards architectures are agnostic of the exchange model and that FIXM can be implemented following the same best practice as AIXM and WXXM and therefore realizing the same operational benefits.

The objectives of the report are to:

1. Demonstrate ISO 19136 compliance within FIXM 2.0 and document these in UML.

2. Regenerate elements of FIXM 2.0 application schema in accordance with ISO 19109 Rules for Application Schema where appropriate.

3. Propose recommendations for the future development of FIXM.

8.2 Exchange of Terrain Data

The subject of Exchange of Terrain Data has been left untouched in previous OGC® test beds. Terrain Data is an important component for the future electronic Terrain and

Obstacle Data (eTOD). Obstacle data is often modeled with a dependency on its underlying Terrain Data. For instance: Vertical structures that have a height, defined as "meters above terrain". When looking at air traffic routes for aerodromes in the vicinity of mountainous areas, it is essential that Terrain Data can be retrieved for analysis.

While classical Terrain Data formats (such as DTED) can be used in existing OGC® Services (such as OGC® WCS), it is found that they are often limiting in their capabilities, especially when it comes to representing metadata. The requirement for metadata is becoming increasingly important, and we will stress this point throughout the report.

The Testbed-10 Recommendations for Exchange of Terrain Data Engineering Report [OGC 14-006] focusses on identifying and analyzing possible Data Formats for the exchange of Terrain Data. For each of the identified Data Formats, an analysis in the context of a set of requirements and considerations is conducted.

8.3 Dissemination of Weather Data

NOAA's National Weather Service (NWS) created its Digital Services Program to meet their customers' increasing need for digital weather, water, and climate services. The foundation of this program is the National Digital Forecast Database (NDFD). Unfortunately, it is not OGC compliant. None of the established service models appears to fully support the functional requirements. The closest match seems to be Web Coverage Service (WCS). In response to this gap, the NWS has developed the Web Gridded Document Service (WGDS).

The Testbed-10 Engineering Report on Aviation Dissemination of Weather Data [OGC 14-038] describes how to provide an OGC-compliant Web Coverage Service (WCS) HTTP GET method-based interface based on the Webgridded Document Service (WGDS) SOAP API. OGC WCS defines three basic operations: GetCapabilities, DescribeCoverage and GetCoverage. The work within Testbed-10 focussed on designing and implementing an adapter which converts a standardized WCS request/response to a WGDS request/response.

9 Lessons Learned

The following aspects have been identified as issues the participants have come across when realizing the Aviation service architecture.

- **Relation of weather data to ATM features** – The Webgridded Document Service of NOAA provides weather forecasts on a national grid. Those gridded forecasts are not directly coupled to ATM features such as aerodromes. The workflow for retrieving weather data for a specific Airport region was achieved through spatial filtering. If the weather service architecture remains this way a common workflow for retrieving data related to ATM features might help.

10 Accomplishments

During the work carried out by all participants of the Aviation thread of Testbed-10 several accomplishments have been reached:

- **Design and implementation of a GML 3.2 based encoding for FIXM 2.0** – this effort was fundamental as a WFS 2.0 compliant implementation requires to serve data as GML 3.2 or a corresponding Application Schema.

- **Develop application patterns for disseminating updates FIXM 2.0 features –** as FIXM 2.0 does not provide a temporal concept equivalent to the one defined for AIXM 5.1, an alternative approach for the publish/subscribe based dissemination of FIXM features had to be developed in order to enable the integration of FIXM data into the Aviation service architecture.

- **Integration of Weather Data –** the access to weather data using a Web Coverage Service has been integrated into the Aviation scenarios for the first time within an OWS test bed. Backed by NOAA's Web-gridded Document Service and encoded as WXXM Forecasts the weather data brought in new aspects while developing interesting use cases and scenarios.

- **Terrain Data Exchange –** in a dedicated Engineering Report, a set of recommendations on exchange formats and protocols for terrain data has been developed.

11 Recommendations

During the work on this report and the overall work of all participants, several aspects have been collected that should be taken into account while maturing the applied concepts:

- **Deeper embedding of the Web Coverage Service into the service architecture** – as many of the occurring events and updates on ATM data and features are related to changes in the weather conditions a profound integration of the WCS into the overall service architecture should be taken into consideration. This provides advanced opportunities in order to develop more realistic and interesting use cases. An integration with the Event Service and the general concept of publish/subscribe based dissemination might also be of relevance.

- **GML based encoding for FIXM 2.0** – In order to mature FIXM and to make it a well-established, interoperable data format new version of FIXM should take the recommendations of the [OGC 14-037] into consideration. In addition, a similar

concept as the Digital NOTAM encoding for AIXM could be developed in order to simplify the provision of updates on certain AIXM features.

Bibliography

[1] [OGC 14-006] Testbed-10 Recommendations for Exchange of Terrain Data, OGC document 14-006

[2] [OGC 14-007] Testbed-10 Report on Aviation Binding AIXM to Development Tools, OGC document 14-007

[3] [OGC 14-037] OGC® Testbed-10 FIXM GML Schema, OGC document 14-037

[4] [OGC 14-038] Aviation Dissemination of Weather Data Engineering Report, OGC document 14-038

[5] [OGC 08-133] Sensor Event Service Interface Specification Discussion Paper, OGC document 08-133

[6] [DNOTAM ES] Digital NOTAM Event Specification – Version 1.0, EUROCONTROL and FAA, 2011

[7] [OGC 12-028] Guidance and Profile of GML for use with Aviation Data

