Open Geospatial Consortium, Inc.

Date: 2009-07-27

Reference number of this document: OGC 09-050r1

**Version:** 0.3.0

Category: Public Engineering Report

Editor(s): Hans Schoebach

# OGC OWS-6-AIM Engineering Report



**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: | OpenGIS® Engineering Report |
| Document subtype: | NA |
| Document stage: | Approved for public release |
| Document language: | English |

## Preface

Suggested additions, changes, and comments on this draft report are welcome and encouraged. Such suggestions may be submitted by email message or by making suggested changes in an edited copy of this document.

The changes made in this document version, relative to the previous version, are tracked by Microsoft Word, and can be viewed if desired. If you choose to submit suggested changes by editing this document, please first accept all the current changes, and then make your suggested changes with change tracking on.

## Forward

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 Inc. 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

Page

# Figures

Page

# Code Listings

Page

# OGC OWS-6-AIM Engineering Report

## 1 Introduction

The OGC Web Services Aeronautical Information Management (AIM) subtask is a new OGC Interoperability thread focused on developing and demonstrating the use of the Aeronautical Information Exchange Model (AIXM) in an OGC Web Services environment. The AIM subtask focuses on evaluating and advancing various AIXM features in a realistic trans-Atlantic Aviation scenario setting by devising and prototyping a Web Services Architecture for providing valuable aeronautical and weather information directly to flight decks, Electronic Flight Bags (EFB) and hand-held devices (such as PDAs and Blackberries) while the airplane is at the gate or en-route to its destination (for the purposes of OWS-6, the aeronautical information in the latter case does not depend on the knowledge of the airplane's location). .

AIXM was developed by the Federal Aviation Administration (FAA) and Eurocontrol as a global standard for the representation and exchange of aeronautical information. It was designed as a basis for digital aeronautical information exchange and for enabling the transition to a net-centric, global aeronautical management capability. Both agencies seek to evaluate the potential of OGC Web Services and other information sharing technologies in conjunction with the net-centric System Wide Information Management (SWIM) concept by demonstrating and enhancing the use of models such as AIXM and WXXM in a Web Services Environment. It is envisioned that the core principles of OGC Web Services and standards may be included in the overall logic of future interoperable ATM information sharing on the local, regional and global levels.

### 1.1 Scope

This report establishes a baseline for the technical architecture, its alternatives and issues for implementing the use cases as specified in the OWS-6 AIM thread RFQ including the temporal WFS supporting the temporal FE 2.0 operators, the Event Service Notification architecture and the client EFBs.

The AIM thread focuses on providing up-to-date aeronautical and weather information to pilots and aircraft while at the airport gate or en-route to its destination by:

- Providing access to integrated, rich and distributed aeronautical and weather information in a standard way, both on demand and automatically,

- Maximizing amount of useful information delivered to aircraft avionics, EFB or hand-held electronic display devices while at the gate or en-route to destination,

- Filtering information using spatial and temporal criteria based on intended flight plan and current conditions.

To support these goals, the scope of the OWS-6 AIM thread is defined by the following areas of work:

- Use and enhancement of Web Feature Service and Filter Encoding specifications in support of AIXM features and 4-dimensional flight trajectory queries,

- Architecture of standards-based mechanism to alert/notify users of changes to user-selected aeronautical and weather information,

- Prototype of Aviation client for retrieval, integration and visualization of aeronautical, weather and other aviation-related data, emphasizing time and spatial filtering specified by the user. Automatic notification of changes to aeronautical information that may be relevant to the plane along its intended route of flight, as specified by the user. In-flight update notifications related to spatial-temporal events are sent as AIXM 5.0 based xNOTAM messages.

In general the architecture combines existing standards from OGC and other standards organizations like OASIS and W3C and implements new components as required. The system components are based on existing OGC and other standards for distributed processing environments and web services. This report describes the use of these standards and also notes alternatives that are available to address specific issues detected during the interoperability test experiments.

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 Inc. 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.

### 1.2 Document contributor contact points

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

| Name | Organization |
|---|---|
| Hans Schoebach (Editor) | Galdos Systems Inc. |
| Nadine Alameh | MobiLaps LLC/OGC |
| Robin Houtmeyers | Luciad |
| Ian Painter | Snowflake Software |
| Daniel Hardwick | Snowflake Software |
| Nuke Goldstein | The Carbon Project |

### 1.3 Revision history

| Date | Release | Editor | Primary clauses modified | Description |
|---|---|---|---|---|
| 2008-11-24 | 0.1 | Hans Schoebach | 6 | Added initial content |
| 2009-02-27 | 0.2 | Hans Schoebach | 7 | Added Comments and Content |
| 2009-03-31 | 0.3 | Hans Schoebach | 7 | Additional Comments and Content |
| 2009-04-15 | 0.4 | Hans Schoebach | all | Added Comments, Content and cleaning |
| 2009-07-13 | 0.3.0 | Carl Reed | Various | Prepare for public publication |

### 1.4 Future work

TBD

## 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, *OpenGIS® Web Services Common Specification*

OGC 05-078r4 *OpenGIS Styled Layer Descriptor Profile of the Web Map Service Implementation Specification*

OGC 08-133 *Sensor Event Service (SES) Specification (discussion paper)*

OASIS *WSN wsn-ws_base_notification-1.3-spec-os:*

*http://docs.oasis-open.org/wsn/wsn-ws_base_notification-1.3-spec-os.pdf*

WS-*Eventing:*

http://www.w3.org/Submission/2006/SUBM-WS-Eventing-20060315/

AIXM Schema can be found at this URL:

http://www.aixm.aero/public/standard_page/download.html

More Information on the Weather Information Exchange Model (WXXM) can be found here: http://www.eurocontrol.int/aim/public/standard_page/met_wie.html

## 3 Terms and definitions

### 3.1 Aeronautical terms

Additional to the given overview information of aeronautical things and terms that are used in this ER and the testbed are introduced.

- AIM

Aeronautical Information Management intends to use OGC web services to exchange AIXM encoded information between OGC WFS implementations and electronic flight bags (EFB), including Digital NOTAMs or xNOTAMs that are encoded as AIXM features and relate to a specific flight plan. Aeronautical Information can be augmented by weather data information such as Terminal Aerodrome Forecasts (TAF) that are delivered in WXXM encodings from a WFS.

- AIXM 5.0

Aeronautical Information Exchange Model 5.0 is a GML application schema used to encode aeronautical data such as airport runways, taxiways, obstacles, navaids etc. AIXM was developed by the Federal Aviation Administration (FAA) and Eurocontrol as a global standard for the representation and exchange of aeronautical information.

- WXXM

Weather Information Exchange Model is GML encoding of weather information based on Observations & Measurements Part 1 – Observation Schema specification. The information can be retrieved via a WFS request as O&M GML features.

- NOTAM

Notice To AirMen is a text based notification message to flight crews that inform them about changes to conditions or equipment relevant to flight operations. NOTAMs inform other people involved with flight such as controllers or dispatchers as well.

- Digital NOTAM (xNOTAM)

Digital NOTAM or xNOTAM messages are encoded as AIXM 5.0 features and adhere to GML schema definitions, which makes them machine readable and subject to XML schema validation. They may contain spatial and temporal properties that can be used by OGC filter expressions to retrieve only those xNOTAM updates that affect a specific flight.

- Navaid

Navaids or navigational aids are objects that facilitate navigation in aeronautics. Common examples are distance-measuring devices (DME, TACAN), devices that provide bearing information (VOR, TACAN, NDB) or guidance systems for runway approaches (ILS).

- Airspace

Airspaces are portions of the atmosphere that can be used for air traffic.

- Procedure

A flight procedure is the plan of operations that an aircraft has to follow while in the vicinity of an airport. A distinction is made between departure procedures (SID or

Standard Instrument Departure) and landing procedures (STAR or Standard Terminal Arrival Route; IAP or Instrument Approach Procedure).

- Route

Routes define the flight trajectory that an aircraft has to follow to reach its destination.

- AIXM temporality

AIXM 5.0 includes an exhaustive temporality model which enables a precise representation of the states and events of aeronautical features. A distinction is made between permanent changes and temporary status. A baseline defines the feature state as result of a permanent change (e.g., when a database was established and loaded). Future changes to a baseline are defined as deltas, and are either permanent or temporary. These deltas contain information on the changes, the feature ID and lifecycle metadata to define its validity. More information on the AIXM data model can be found here:

http://www.aixm.aero/public/subsite_homepage/homepage.html

### 3.2 OGC architecture

This section contains an overview regarding OGC or common architecture terms.

- HMI

Human machine Interface is an operating platform for human communication with machines. In IT this is often a user interface (UI) application to interact with an information system application

- SWE services like SOS, SAS, …

Sensor Web Enablement suite of service specification targeted at sensor networks for sensor data retrieval, storage and notification such as Sensor Observation Service (SOS) that delivers O&M encoded sensor data and SAS the Sensor Alert Service (SAS) that delivers notification of sensor events that can include the data and that are sent to subscribers.

- Other OGC services like WFS, WFS-T

The Web Feature Service delivers GML encoded features on request (GetFeature) the WFS-T supports transactions

- GML

Geography Markup Language is a standard OGC/ISO XML encoding of geographical and spatial related data

# 4 Conventions

## 4.1 Abbreviated terms

| | |
|---|---|
| AIM | Aeronautical Information Management |
| AIRMET | Airmen's Meteorological Information |
| AIS | Aeronautical Information Services |
| AIXM | Aeronautical Information Exchange Model |
| AIXM | Aeronautical Information Exchange Model |
| AJAX | Asynchronous JavaScript and XML |
| API | Application Programming Interface |
| ARP API | Asynchronous Request Processing API |
| ATC | Air Traffic Control |
| CDC | Change Data Capture |
| COM | Component Object Model |
| CORBA | Common Object Request Broker Architecture |
| COTS | Commercial Off The Shelf |
| CR | Change Request |
| DCE | Distributed Computing Environment |
| DCOM | Distributed Component Object Model |
| DCP | Distributed Computing Platform |
| DHCP | Dynamic Host Configuration Protocol |
| DSL | Domain Specific Language |
| EFB | Electronic Flight Bag |
| ES | Event Service |
| FAA | Federal Aviation Administration |
| FE | Filter Encoding |
| FES | Filter Encoding Specification |
| GeoXACML | Geospatial eXtensible Access Control Markup Language |
| GML | Geography Markup Language |
| HMI | Human Machine Interface |
| HTTP | Hypertext Transfer Protocol |
| HTTPS | Hypertext Transfer Protocol over Secure Socket Layer |
| IDL | Interface Definition Language |
| IP | Internet Protocol |
| ISO | International Standardization Organization |

| | |
|---|---|
| IT | Information Technology |
| KVP | Key Value Pair |
| MEP | Message Exchange Protocol |
| NOTAM | Notice to Airmen |
| O&M | Observations & Measurements |
| OGC IP | OGC Interoperability Program |
| OGC | Open Geospatial Consortium |
| OWS-6 RFQ | OWS-6 Request For Quotation |
| OWS-6 | Open Web Services, Phase 6 |
| PDP | Policy Decision Point |
| PEP | Policy Enforcement Point |
| PKI | Public Key Infrastructure |
| RIA | Rich Internet Applications |
| SAML | Security Assertion Markup Language |
| SAS | Sensor Alert Service |
| SES | Sensor Event Service |
| SIGMET | Significant Meteorological Information |
| SOS | Sensor Observation Service |
| SWE | Sensor Web Enablement |
| TAF | Terminal Aerodrome Forecast |
| METAR | Aviation Routine Weather Report |
| UI | User Interface |
| WFS | Web Feature Service |
| WFS-T | WFS-Transactional |
| WNS | Web Notification Service |
| WSN | Web Service Notification |
| WXXM | Weather Information Exchange Model |
| XACML | Extensible Access Control Markup Language |
| xNOTAM | Digital (XML based) notification to airmen |
| XML | eXtensible Markup Language |

### 4.2 UML notation

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

### 4.3 Used parts of other documents

N/A

### 4.4 Platform-neutral and platform-specific standards

N/A

### 4.5 Data dictionary tables

N/A

## 5 OWS-6 AIM Engineering Report overview

This Engineering Report (ER) for Aeronautical Information Management addresses, as specified, the test bed implementation of a distributed system supporting the pre- and in-flight information exchange and event driven processing of messages of concern to users i.e. pilots and crews in the context of a EFB flight plan information and its spatial – temporal dimensions, which means in a 3D geospatial + 1D temporal space.

The report will address the architecture of the Event Service (ES) as an Event Driven Architecture (EDA) and describe the potential and optional usage of different base standards like OASIS Web Services Notification (WSN), OGC Sensor Alert Service (SAS) or Sensor Event Service (SES) and OGC Web Notification Service (WNS) as well as evolving standards such as WS-Eventing (W3C Draft) and assess their capabilities in the context of the OWS-6 AIM use cases as described in Annex B of the RFQ. More information on the ES implementation core functionality can be found in the document OGC 08-133 OpenGIS Sensor Event Service (SES) Implementation Specification.

Further the report will address the implementation of a temporal WFS and note issues encountered during the implementation of a temporal WFS based on FES2.0. More detailed information on the temporal WFS implementation used in this thread can be found in section 7.16

The generation of update notifications in the form of xNOTAM (digital notification to airman) is described in section 7.2

xNOTAMs are pushed to subscribing clients by the Event Service. The clients receive and display the xNOTAM information in the context of a flight plan and its geospatial location and temporal context as a graphical representation on the user interface (UI) with the appropriate symbology. The client implementations details are further described in section 7.14

## 6 Test Bed Scenario Description

### 6.1 OWS-6 Scenario for Aviation Information Management (AIM) Thread

This scenario provides a fictitious, but realistic context for a demonstration of the functionality developed in the Aviation thread of OGC's Interoperability Program (IP)

initiative Open Web Services, Phase 6 (OWS-6), including interaction with other OWS components. The scenario is intended to prompt the exercising of interfaces and the use of encodings that will be developed, tested or enhanced within OWS-6.

One major objective is to demonstrate the ability of Web Feature Service (WFS) and the Filter Encoding Specification (FES) to provide access to aeronautical information in AIXM 5.0 format in response to direct user queries or in response to alerts generated when specific aeronautical information - as defined by that user- is updated. The scenario also includes a demonstration of retrieval of pertinent weather data and delivering it to the aircraft.

The participants in this scenario are the flight crew, Air Traffic Control (ATC), ground controllers, the custodians/providers of aeronautical information (and information updates), and the custodians/providers of the weather information.

**Error! Objects cannot be created from editing field codes.**

**Figure 1: OWS-6 AIM Scenario Diagram**

# 7 OWS-6-AIM Architecture and Technology Platform

The AIM thread of the OWS-6 initiative is intended to provide implementation best practice know how and feedback on viable standards and technologies used to implement a solution that meets the requirements and use cases of an automated near real-time information management for the aviation community. This will integrate AIXM encoded data for aviation relevant permanent and temporary information and other data sources such as weather data in a spatial – temporal context for pre-flight and in-flight information via web services.

The issues that arise during the implementation will be documented and if possible recommendations for there resolution will be provided.

## 7.1 Aviation Thread System Architecture Overview

The main Aviation Thread components that support the scenario are:

- Temporal WFS that serves the AIXM data mapped from a RDBMS and supports FES2.0 filter expressions to retrieve AIXM baseline data, AIXM deltas and xNOTAMs

- An Event Service that acts as an information broker between the data producers of xNOTAMs and subscribing clients (EFB) of the xNOTAMs. xNOTAM events are pushed to all clients that have provided a matching filter when subscribing

- Client EFB installed on mobile devices or as part of the Avionics system that query the temporal WFS to get the latest information from the AIXM WFS service related to their flight plan and then subscribe to the Event Service to receive related xNOTAMs while in the air

- Weather Service WFS's that provide weather information for weather stations (METARs) as GML features and forecasts (TAFs) as WXXM encoded features

The main components are shown in Figure 2 – Architecture Overview and Main Components

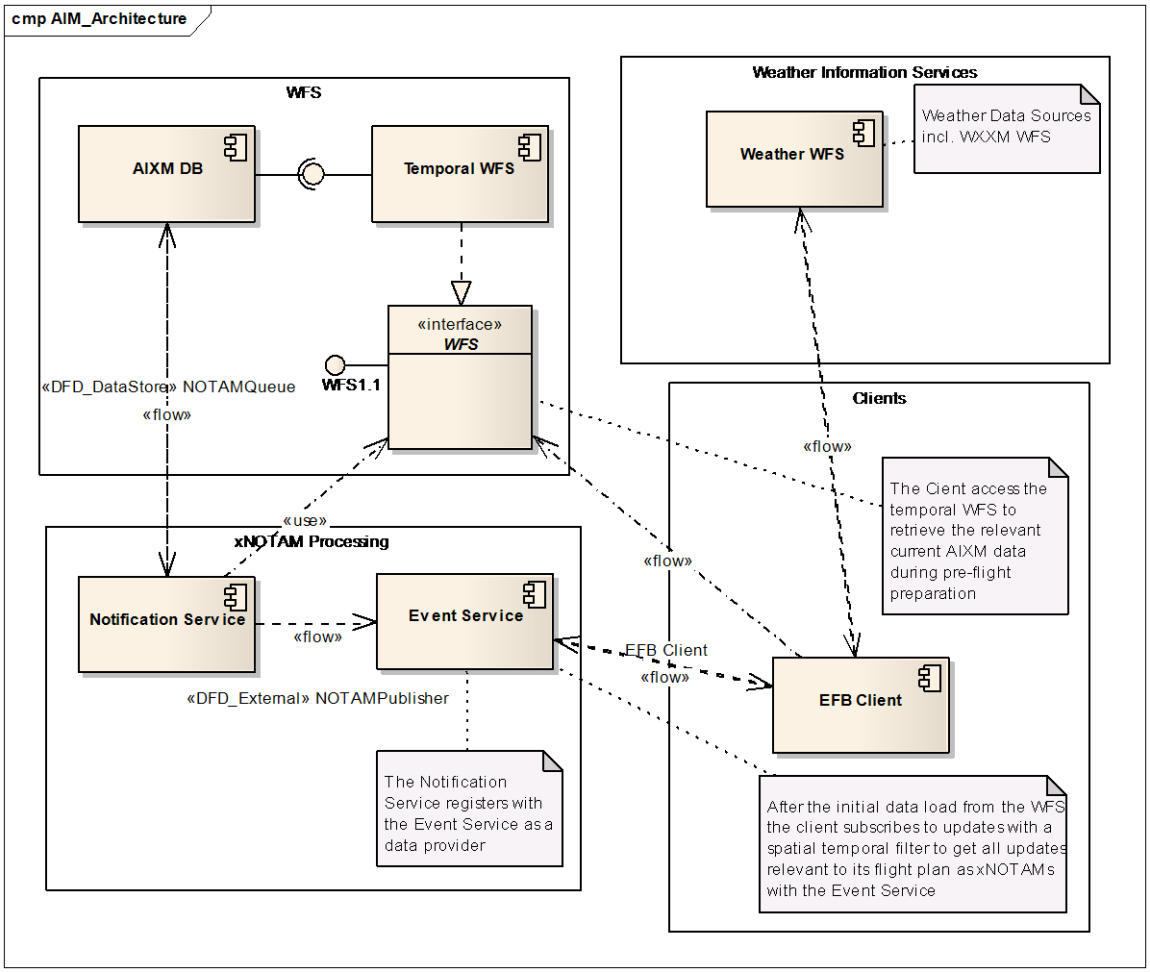
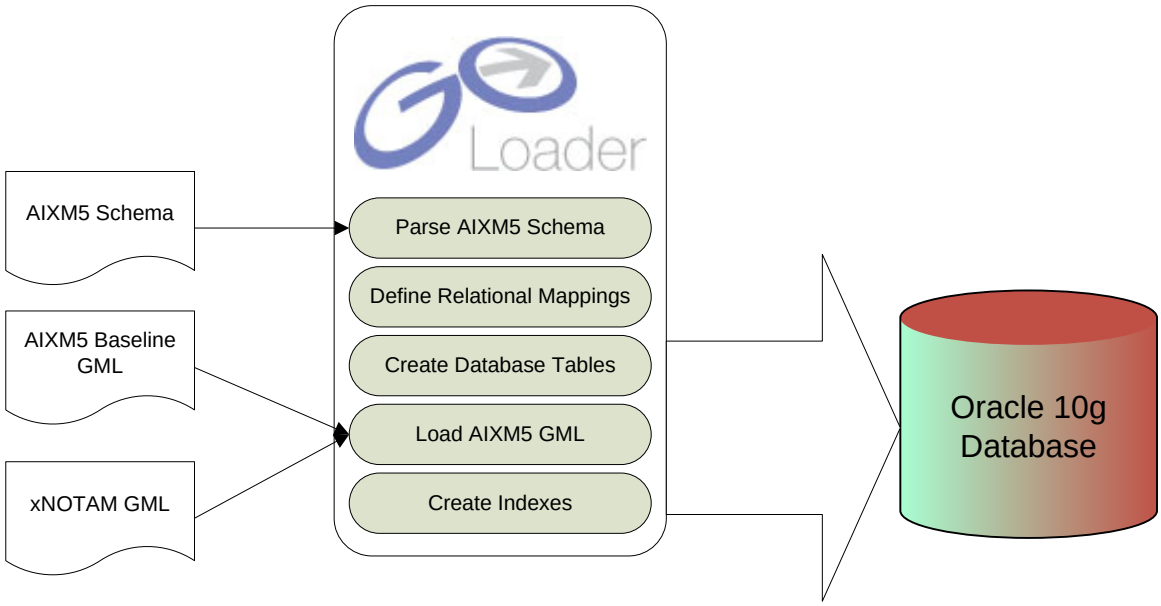



**Figure 2 – Architecture Overview and Main Components**

## 7.2 AIXM5 Data Loading and Management

### 7.2.1 Data Load Architecture

Data was supplied from Eurocontrol and FAA in a mix of ESRI Shapefile and AIXM5 GML. The majority of data was supplied in AIXM5 format. Snowflake utilized its GO Loader tool to perform the GML data loads. GO Loader is a generic GML loading tool capable of loading any GML2.2, GML 3.1.1 or GML 3.2.1 dataset. Through parsing the AIXM5 application schema, GO Loader was able to adapt itself to read and load AIXM5 without any bespoke coding.

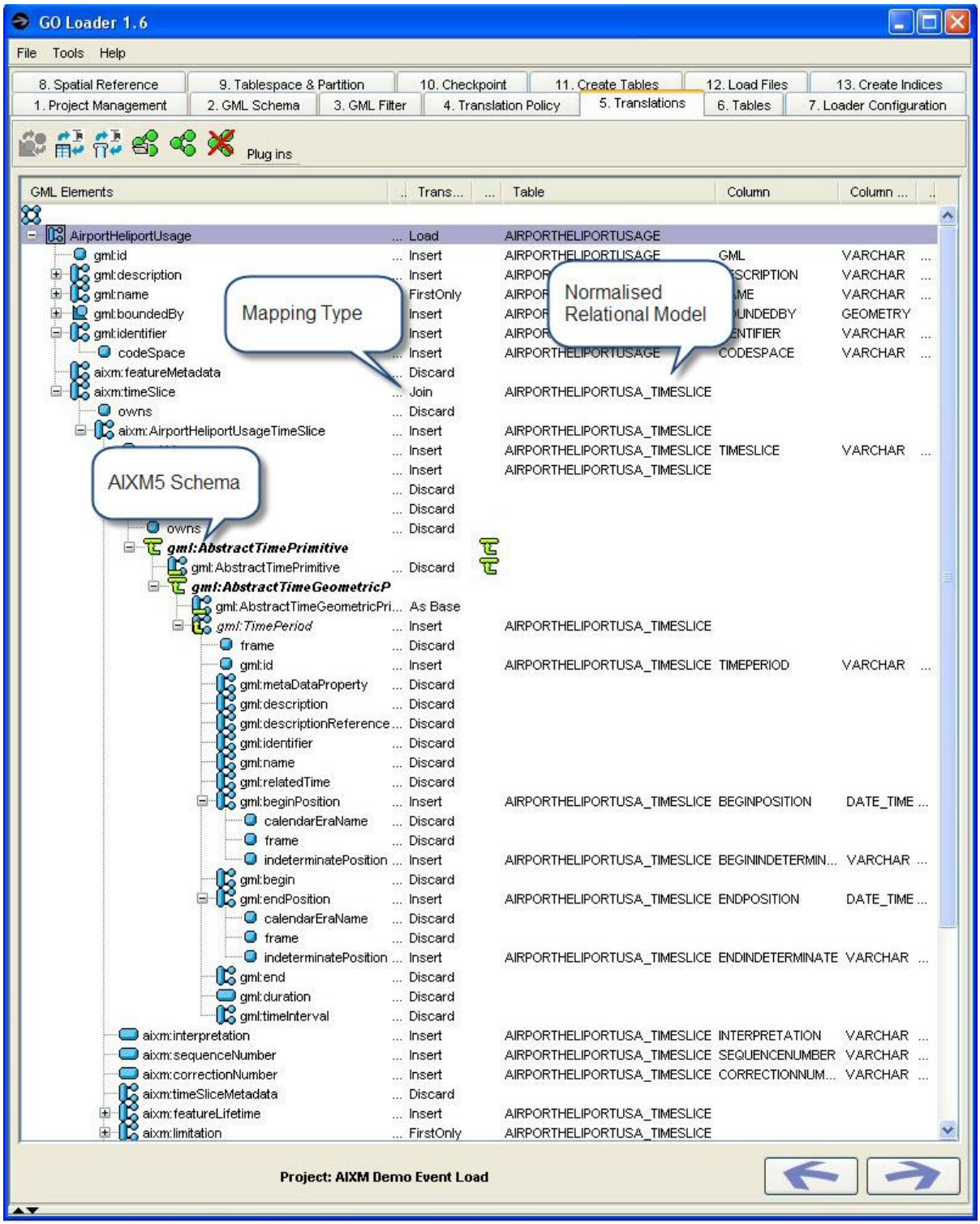



The steps for the data load were as follows:

1. Parse the AIXM5 Schema
2. Use GO Loader to define mappings from the AIXM5 Schema to the target relational model
3. Create the relational model from the mappings
4. Load the base line AIXM5 into the tables
5. Build all required indices (including spatial)

All the above stages were performed within the COTS product enabling a quick start in the early stages of the testbed - despite a lack of knowledge of the AIXM5 schema and data model.

The most user intensive part of the load process involved defining a mapping from the source schema to the target relational model. The mapping was performed in GO Loader Studio and involved a drag and drop process as show below.



## 7.3 Data Issues

### 7.3.1 Initial Data

For the initial WFS a large proportion of the early baseline data for the United States (US) was supplied as Shapefiles, the following issues were found with the data:

- The shapefiles were missing some required data, for example aixm:identifiers which had to be populated with new values.
- Geometries for each airport were supplied in separate shapefiles in a mix of WGS1984 and GCS_North_American_1983 projections. In some scenarios the projection system was identified within the shapefiles or the associated documentation in others the projection had to be presumed from their locations in coordinate space.
- The geometries in the shapefiles were not always valid, containing self intersections, duplicate vertices and other geometry errors – these were fixed using the Oracle geometry tools.
- Sections of Runway where runways crossed each other were often held in separate shapefiles from other runway sections. The shapefiles geometries were corrected, merged and reprojected to a common projection before being loaded into the Oracle database.
- Lengths and widths of the Runway features were not included and the data was reprojected to an appropriate UTM zone projection before being manually measured on screen.

The majority of European (EU) data was supplied in AIXM5 GML format. These AIXM datasets arrived later however they were by their nature more complete. Where both shapefile and AIXM5 were available, the AIXM data was used in preference to the shapefiles as they had populated aixm:identifier values. Where the shapefile and aixm datasets overlapped there was some effort required to match up the aixm:identifier references linking features from difference sources e.g. aixm:RunwayElement refenerces to aixm:Runway.

The shapefile and AIXM datasets did not always contain information on the same types of features. The (US) Shapefiles contained data to populate aixm:RunwayElements however the (EU) AIXM datasets contained aixm:RunwayDirections. The AIXM data supplied aixm:Runways but not the associated aixm:RunwayElements which contains the geometries making up a runway. As a result there was sometimes a need to mix and match data from the shapefile and aixm sources. Furthermore, sometimes the supplied AIXM NOTAMs contained xlink:href's to feature aixm:identifiers that did not exist in the supplied datasets.

### 7.3.2 Gml:id

The GMLID's supplied in AIXM base data and NOTAMs are unique within each file however they are not unique between files, resulting in gml:id's being duplicated and

therefore not valid when loaded into the common database. The advice was to remove the supplied GMLIDs and regenerate them in the database.

Eduard Porosnicu from Eurocontrol commented:

> *Concerning the gml:id, I understand that they play no role outside the GML file in which they appear. So, you are not supposed to store in your database the values that you get from our files. Just discard them and create new ones for the WFS output is perfectly OK. From the xNOTAM Trial, they are generated at the time when the file is output and probably using the same algorithm each time. No wonder that the same values appear in different files... we will try to do better in future.*

### 7.3.3 aixm:Identifier

The aixm:Identifier is the unique value for the feature. A real world object (e.g. airport) may be represented by multiple features (baselines and temp deltas) for different time periods but all contain the same aixm:identifier.

A lot of the initial (mainly USA) data was supplied as shapefile which did not have any information with which to populate the aixm:Identifier and these values had to be created. However later data (mainly EU) was supplied in AIXM format which did have aixm:identifier values. Where the shapefile and aixm datasets overlapped, there was considerable effort required to match up the two sets of data. The AIXM data was used in preference to the shapefiles where both were available as it contained populated properties and in particular aixm:Identifier's.

### 7.3.4 Example NOTAMs

In some scenarios the supplied AIXM xNOTAMs contained xlink:href's to feature aixm:identifiers that did not exist in the supplied datasets or that linked to features which then did not link to anything else. For the key event NOTAMS, e.g. closing the Swedish ESSA airport, a new NOTAM has been produced which is valid for the data in the WFS's database.

### 7.3.5 Time Indeterminate Value

The ISO 19108 – Temporal Schema indicates that returning data with a temporal query with an indeterminate date is invalid:

> *"This operation shall raise an exception if any input value of TM_TemporalPosition is indeterminate."*

To enable data which can be queried using the *current* FE2.0 Temporal specification all values with an indeterminate Position of unknown were set to a date well in the future e.g. in year 9999 which is effectively infinity.

Thus:

<gml:beginPosition>2009-02-127T07:00:00</gml:beginPosition>
<gml:endPosition indeterminatePosition="unknown"/>

Was changed to:

<gml:beginPosition>2009-02-127T07:00:00</gml:beginPosition>
<gml:endPosition>9999-12-31T23:59:59</gml:beginPosition>

All indeterminate time periods have been replaced with either BeginPosition '-4712-01-01 00:00:00Z' or EndPosition '9999-12-31 23:59:59Z'. A change request will be sent to improve this aspect of the draft FE2.0 spec

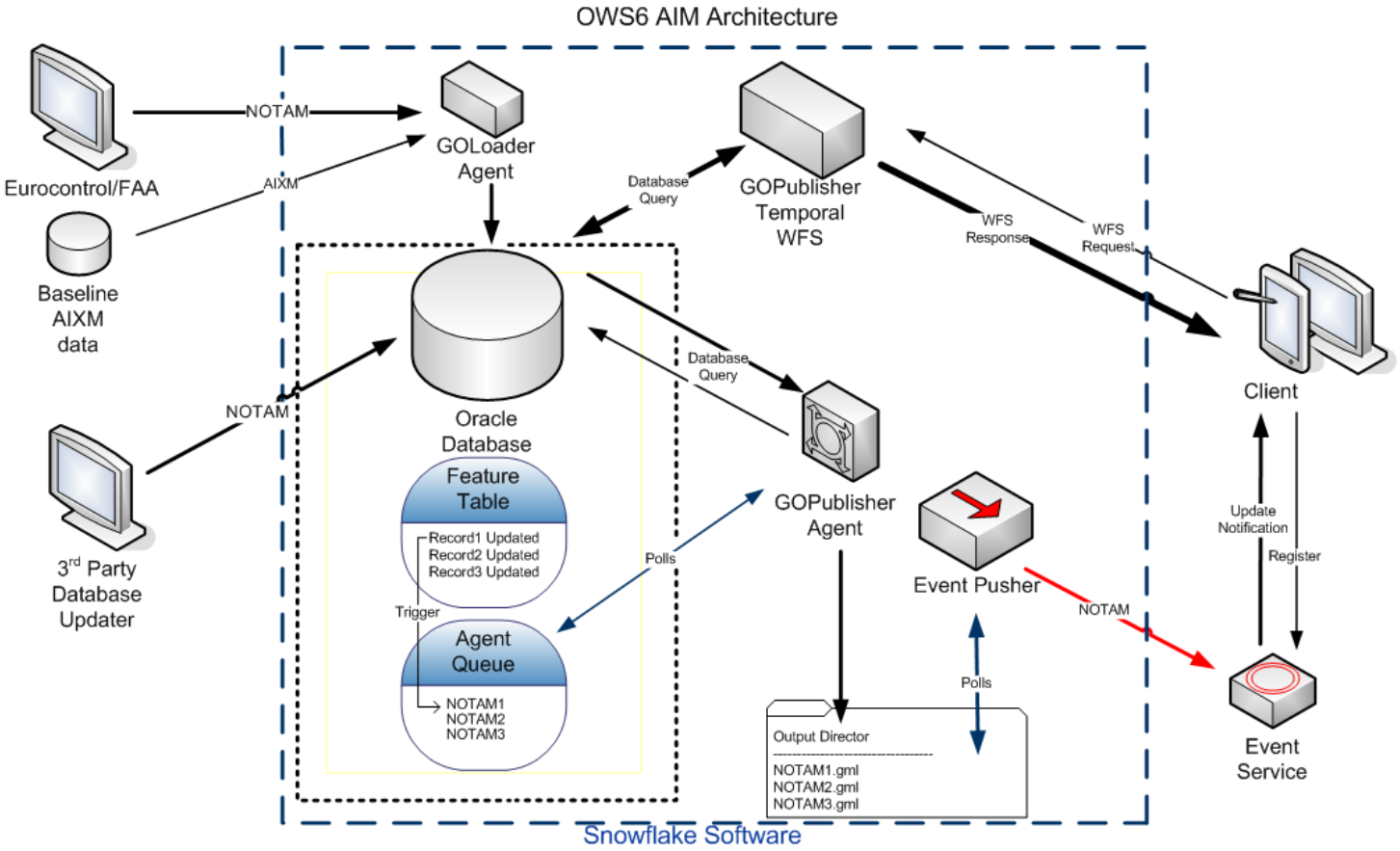
### 7.3.6 UCUM Unit of Measure compliance

Snowflake's WFS implementation utilized Oracle as underlying DBMS, as a result support from the UOM values in queries were passed directly to Oracle e.g. "NM" or "nautical mile" and "M" (meters). For full support of the UCUM codes, Snowflake needs to implement a lookup table to translate these codes for Oracle

Eduard Porosnicu from Eurocontrol commented:

> *There is a discussion outside the testbed on the update of AIXM 5.1 to be compliant either with UCUM or ISO 80000 series (or with both, we hope). Today AIXM 5.0 uses custom lists for units of measurement "M", "FT", "NM", etc. - typically the aviation domain abbreviations.*

## 7.4 OWS6 AIM WFS and Notification




### 7.4.1 Initial AIXM Load Steps

- AIXM5 schema is parsed by GO Loader and an Oracle11g relational model is created based on the schema
- Initial AIXM5 data is loaded into the relational database by GO Loader and indexes are built
- GO Publisher Desktop is used to configure a mapping from the relational model to the AIXM5 schema
- A WFS 1.1 with support for FE2.0 Temporal queries is hosted based on the mapping

### 7.4.2 NOTAM Processing Steps

NOTAM in AIXM5 received

AIXM5 loaded into Oracle 11g via GO Loader

GOPublisher Temporal WFS services stream data dynamically from the AIXM data

Triggers on the database tables populate a GOPublisher Agent queue database table

GOPublisher Agent reads the queue and constructs the NOTAM including a bounding box For NOTAM updates for non-geographic features GOPublisher Agent extracts the geometries from related features and constructs a bounding box.

NOTAM updates are pushed to the Event Service.

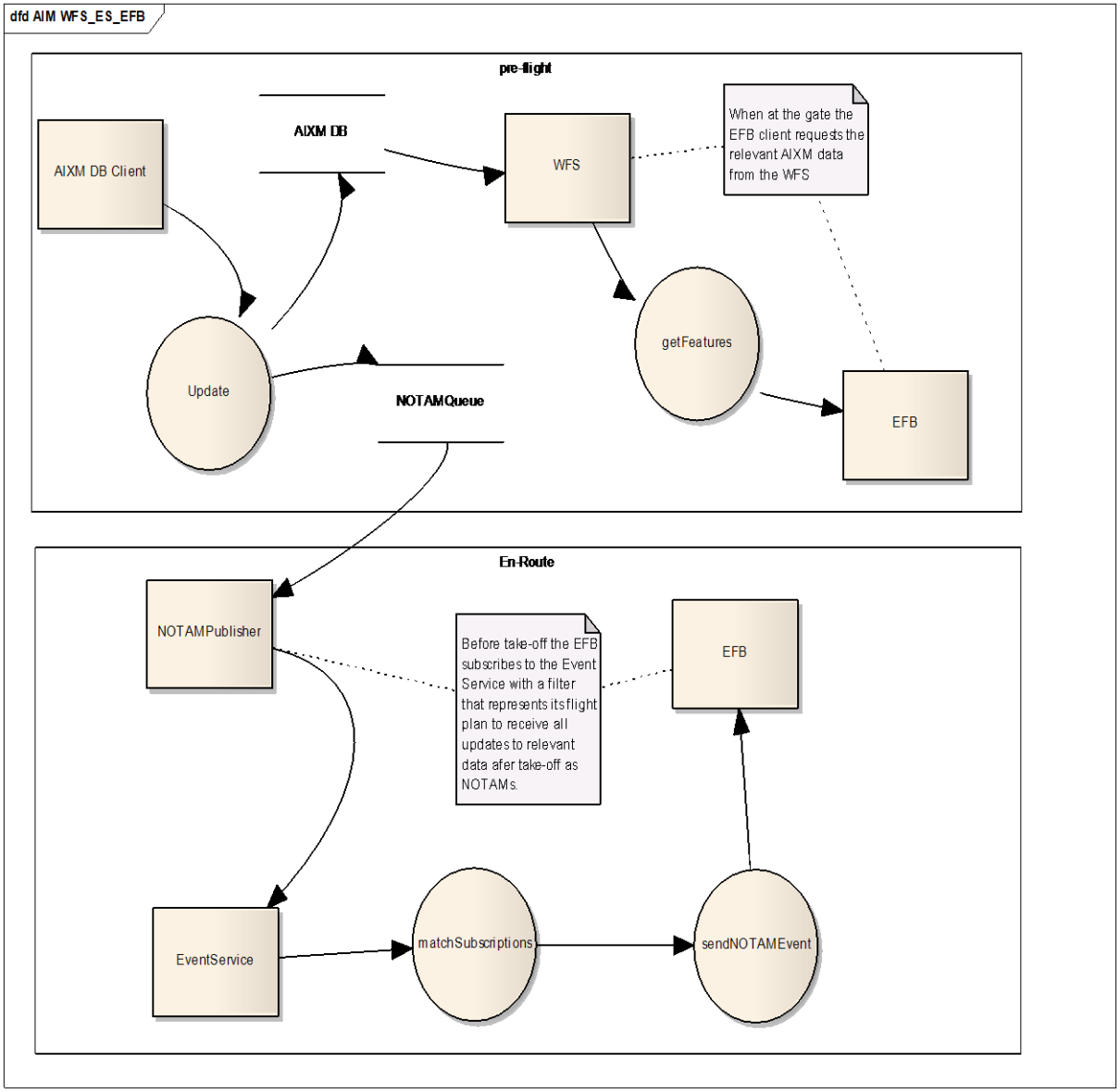



**Figure 3 – Pre-Flight Data Loading and In-Flight Notification**

The information flow diagram in Figure 3 shows the message exchange in the pre-flight and in-flight conditions and inter-operations between the Clients, WFS and the Event Service.

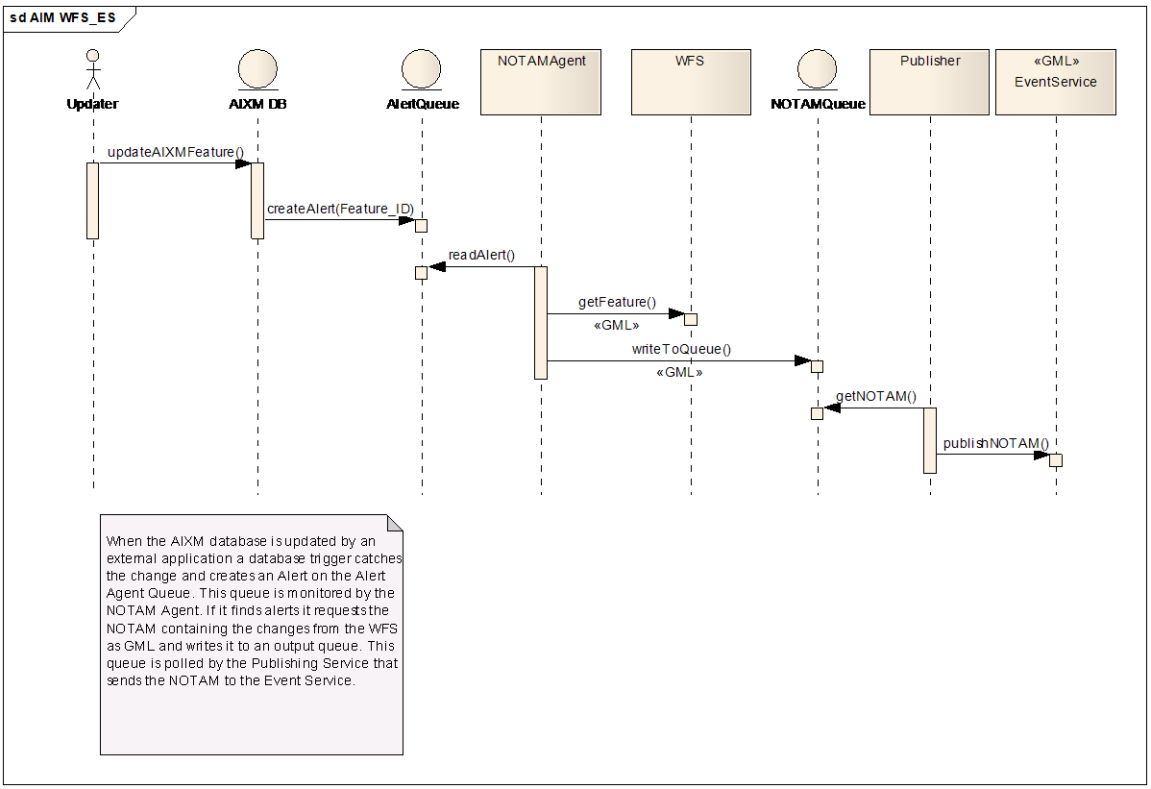



**Figure 4: Update Trigger and Publishing**

The updating of the AIXM database causes a trigger to create an update alert that is written to an Alert Queue. The alert queue is monitored by the NOTAM Agent and a NOTAM event is populated via a WFS request providing the feature ID of the changed AIXM feature and then sent to the NOTAM queue from where it is fetched by a Publisher service that pushes the NOTAM event to the Event Service for subscription matching and propagation to matching subscribers as is shown in Figure 4.

### 7.5 AIM System AIXM and Weather Data Integration

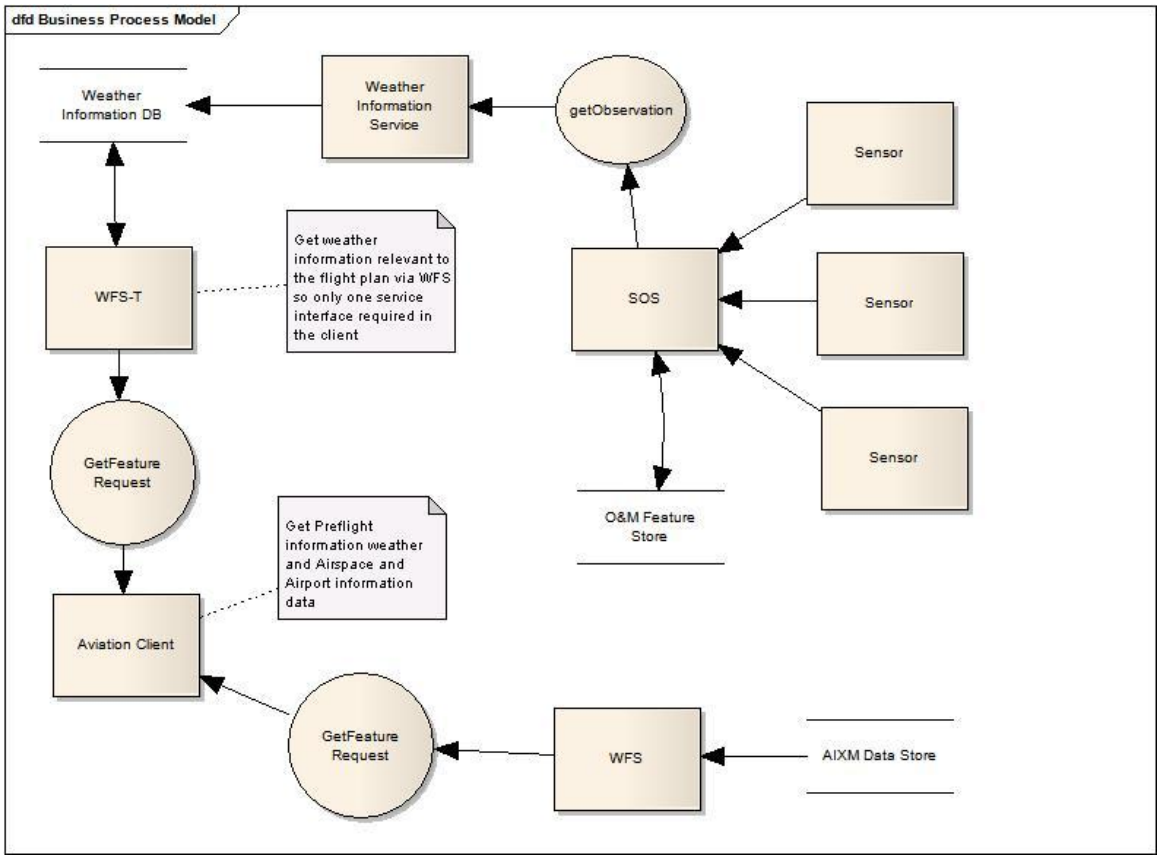



**Figure 5 – Dataflow Diagram of preflight Weather and AIXM information update for EFB**

The dataflow diagram in Figure 5 shows the information flow between the components to create a PIB (Pre-Flight Information Bulletin).

This involves WFS requests to the data sources that maintain the AIXM and Weather data bases. In providing WFS interfaces to these data sources the aviation clients only need to deal with GML encoded data formats (AIXM and WXXM) that can be validated against the schema to ensure data consistency before updating the local data store. In a possible scenario the Weather Information Service shown in the diagram will pull the updated weather information - generated by sensors from a SOS - and update the weather information database to keep the database current.

The portrayal of the weather information needs to be standardized. It has to be investigated if the Styled Layer Descriptor/ Symbology Encoding (SLD/SE) OGC standard can be used to provide a framework for this standardization.

The client will generate a WFS GetFeature request from a parameterized template to get the relevant data according to the flight plan. This means a GetFeature request with an OGC filter to specify the spatial and temporal constraints for the required data that relate to the flight plan.

Another WFS request template will drive a request to the endpoint of the AIXM data source to get the airspace, airport and Navaid information relevant to the flight plan. This does not

involve the Event Service as the data is pulled from the data sources via WFS requests from the client.

## 7.6 AIM xNOTAM Event Generation and Information Flow

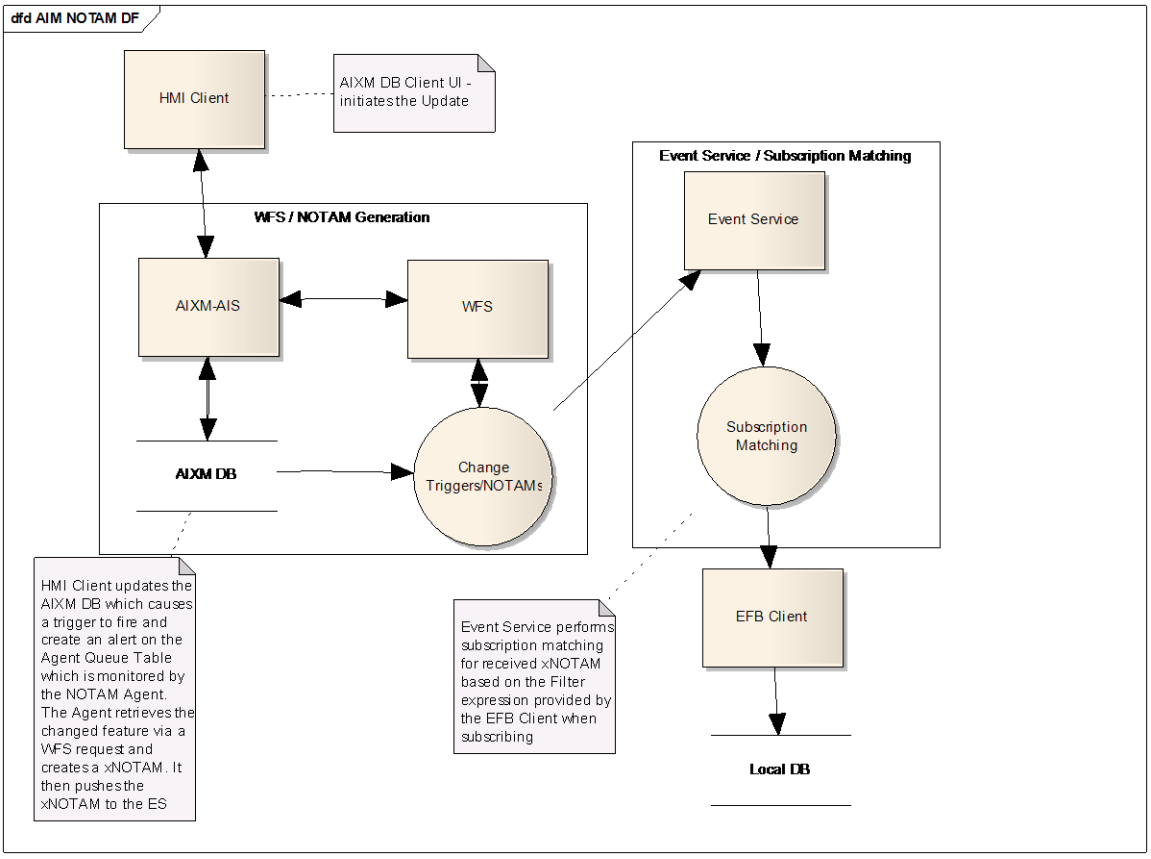



**Figure 6 — Information Flow of NOTAM Events**

The diagram in Figure 6 shows the delivery of xNOTAM messages to subscribers. The subscribers are mobile or built-in devices used on airplanes to inform the crew about relevant events while en route.

A typical Aeronautical Information Service (AIS) system that provides initial pre-flight data loading and subsequent updates via xNOTAM while on-route is shown in Figure 7 AIS System

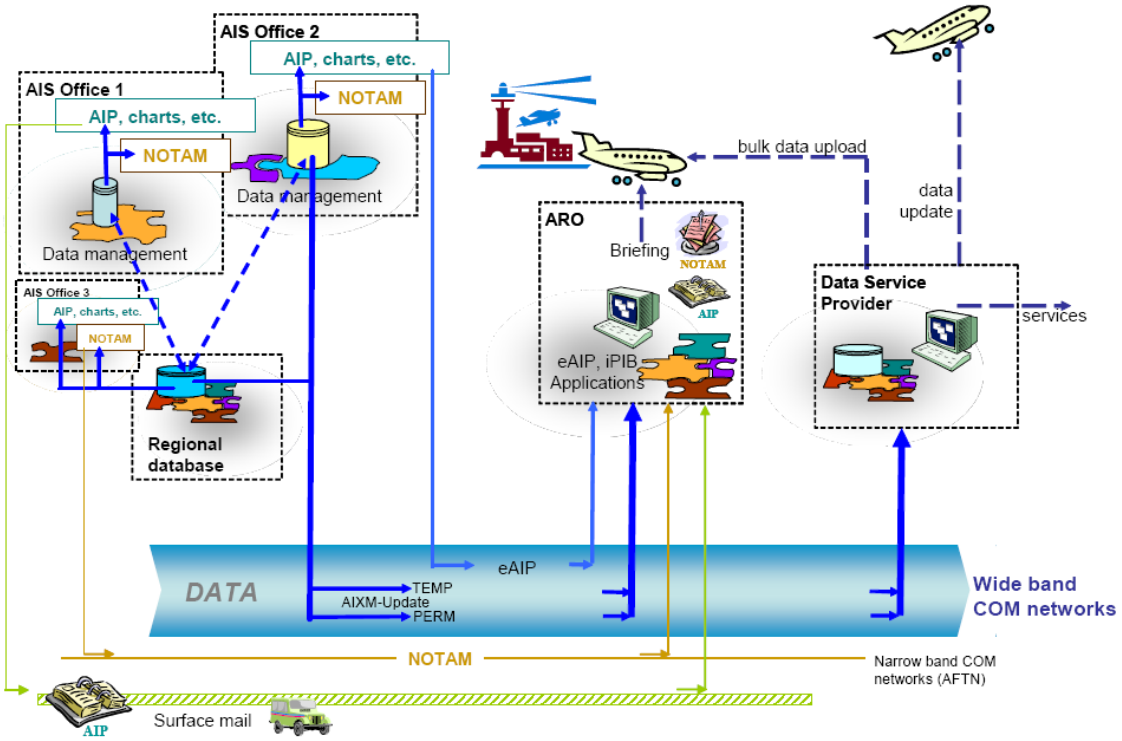



**Figure 7 AIS System**

In this version the NOTAM update is performed by an HMI client directly to the database of the AIS. This means that the change alert has to be generated on the database level.

This can be done via a Change Detection Module or Change Data Capture (CDC) module that sits on the database side and uses APIs or triggers depending on the database management product.

In case of Oracle a provided CDC API can be used. The current implementation in this AIM thread uses database triggers to alert an agent that creates the xNOTAMs via a WFS request. This is further described above in section 7.4.2, NOTAM Processing. The generated alert contains an identifier for the updated AIXM feature which can be used to retrieve the feature via a WFS GetFeature request. The xNOTAM agent wraps the NOTAM into an AIXM Event Message and pushes it to the Event Service which will match it against subscriptions. The following example shows a NOTAM event message.

```
<event:Event xmlns:event="http://www.aixm.aero/schema/5.0/event/0.1"
  xmlns:aixm="http://www.aixm.aero/schema/5.0"
  xmlns:fua="http://www.aixm.aero/schema/5.0/extensions/eur/fua"
  xmlns:gml="http://www.opengis.net/gml/3.2" xmlns:xlink="http://www.w3.org/1999/xlink"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.aixm.aero/schema/5.0/event/0.1
http://www.aixm.aero/schema/5.0/event/0.1/Event.xsd
http://www.aixm.aero/schema/5.0/extensions/eur/fua
http://www.aixm.aero/schema/5.0/extensions/eur/fua/FUA_Features.xsd"
  gml:id="Event-0003">
  <event:name>[A0493/08] DVOR HUL u/s</event:name>
  <event:description>(A0493/08 NOTAMN Q) EBBU/QNVAS/IV/BO /E /000/999/5045N00439E025 A) EBBU
B)
    0804100800 C) 0804101000 E) HULDENBERG DVOR HUL 117.550MHZ U/S DUE MAINT.)
</event:description>
  <event:type>TEMPORARY</event:type>
  <event:hasMember>
```

```
    <aixm:Navaid gml:id="VID54386">
      <gml:identifier codeSpace="http://www.eurocontrol.int/xnotam"
        >b603c765-6e43-4218-b95f-ba8cb17d2c16</gml:identifier>
      <aixm:timeSlice>
        <aixm:NavaidTimeSlice gml:id="VID924703">
          <gml:validTime>
            <gml:TimePeriod gml:id="VID000001">
              <gml:beginPosition>2008-04-10T08:00:00</gml:beginPosition>
              <gml:endPosition>2008-04-10T10:00:00</gml:endPosition>
            </gml:TimePeriod>
          </gml:validTime>
          <aixm:interpretation>TEMPDELTA</aixm:interpretation>
          <aixm:sequenceNumber>1</aixm:sequenceNumber>
          <aixm:correctionNumber>0</aixm:correctionNumber>
          <aixm:type>VOR</aixm:type>
          <aixm:operationalStatus>UNSERVICEABLE</aixm:operationalStatus>
        </aixm:NavaidTimeSlice>
      </aixm:timeSlice>
    </aixm:Navaid>
  </event:hasMember>
</event:Event>
```

**Listing 1: Example of an xNOTAM**

The above example shows an xNOTAM that informs subscribers about a temporary "Unserviceable" status of a Navaid on the 10th of April 2008 between 08:00 and 10:00 o' clock.

Note that there is no location information in that xNOTAM so its relevance to particular subscribers' flight plan cannot be determined. Subscription matching is limited to the feature type <Navaid/> its feature ID and the time slice. Location information will be extracted from a related feature via a subsequent WFS request and then added to the NOTAM if the feature itself does not have the location information. The location information is provided as a gml:boundedBy property as shown in the following example:

```
<event:name>ESSA runways closed</event:name>
<event:description>RWY 08/26, 01L/19R, 01R/19L closed temporarily due to snow contamination
and bad weather, starting 12 FEB 2009 at 07:00</event:description>
<event:type>TEMPORARY</event:type>
<event:hasMember>
  <aixm:AirportHeliportUsage gml:id="VID2678448">
    <gml:boundedBy>
      <gml:Envelope>
        <gml:lowerCorner>17.956,59.660</gml:lowerCorner>
        <gml:upperCorner>17.958,59.662</gml:upperCorner>
      </gml:Envelope>
    </gml:boundedBy>

  <aixm:timeSlice
  ...
```

**Listing 2: Example of use of the gml:boundedBy element**

In an operational system the feature ID (gml:id) cannot be used for subscription matching as it is not unique across multiple WFS servers. This can be solved by using a gml:identifier which is globally unique instead.

Subscription matching might get very complex depending on the use cases and it will be necessary to decouple the ES notification broker service from the subscription matching algorithm. This can be achieved for example by using specific plug-ins, which would implement a specific polymorph functionality behind a generic interface or API that will register itself if configured with the runtime environment. Another method would be that the

matching service in the ES is controlled by a content based routing dispatcher that contains rules defined in a Domain Specific Language (DSL).

Alternatively the Event Service will get an alert which contains only the necessary information for subscription matching and if there is a match it generates a WFS request applying the filter expression of the subscription to retrieve the actual xNOTAM. In this way the ES does not have to process the filter against the data in the xNOTAM itself and this should improve the performance of the event data propagation.

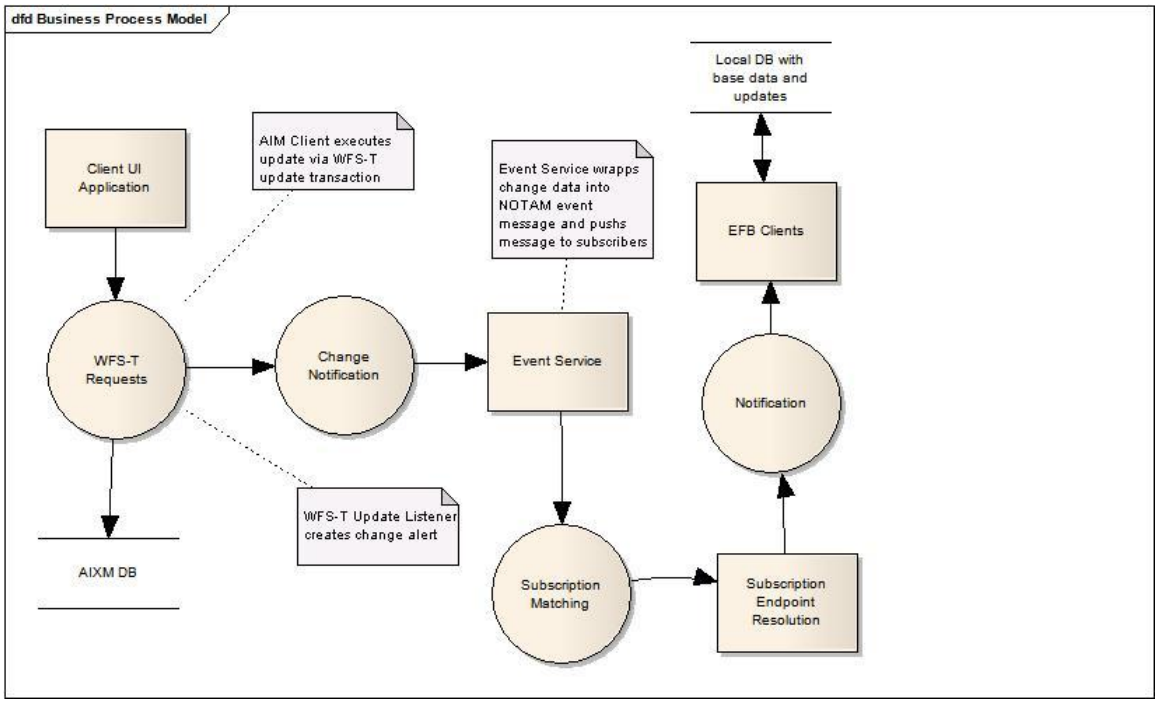



**Figure 8 – NOTAM Update via WFS-T Transaction**

This variant shown in Figure 8 creates NOTAM updates through the WFS-T interface to update the AIXM database. This requires a client that implements this interface.

The WFS-T implements a transaction interface that supports INSERT, DELETE and UPDATE operations of GML features. The client has to generate the appropriate XML messages that correspond to these operations and provide the GML encoded data to perform the update.

This solution is recommended in a production environment as it does not require database triggers or Oracle Change Data capture (CDC) API calls, which are specific to a particular RDBMS product and is therefore more generic. It requires the WFS-T to provide a post processing callback method to catch the changes upon commit from the persistence data store. The post processing method will store metadata describing the changes and assemble the xNOTAM message and event wrapper. The xNOTAM event is than pushed to the Event Service for subscription matching and dissemination to subscribers.

### 7.7 Event Notification Brokering

The data sources in AIM do not send notification messages directly to the subscribers but use an event broker service called Event Service (ES) to propagate messages to subscriber endpoints. This means that data sources do not have to know anything about subscribers and

subscribers don't know anything about the data sources. They just need to know what information is available that they can subscribe to. This is achieved via a Publication / Subscription methodology that is used in other notification services such as WS-Notification the OASIS standard specification, WS-Eventing, which is currently a W3C draft specification and the more specific OGC Sensor Event Service (SES). WS-N is based on SOAP and might have too much overhead for the aviation clients. WS-Eventing allows domain specific message formats and encodings to be defined and might be a potential choice for asynchronous message delivery to the subscribers without using SOAP. The ES implementation decisions will be documented to stste whether the brokering service will use a standards based implementation or define its own event brokering specification that is targeted to the AIM domain.

### 7.8 Event Source Registration (Advertising)

Data sources that want to publish messages have to register with the Event Service to announce the feature types they are able to provide. The Event Service needs to know which data types and formats it can advertise to potential subscribers. Subscriptions to features that are not registered with the ES should be rejected to avoid tangled subscriptions. These subscriptions would never get a notification message because the event type cannot be matched but which the subscriber would not be aware of. By default the Event Service should accept any subscriptions to feature types that the data producer supports.

The registration process could be providing the Capabilities document of the WFS that sits on the AIXM database. This will describe the feature types that the data source supports including the geographic area (BBox) it covers which can be used for subscription matching. The Capabilities can be used to validate the incoming events to avoid inconsistent data or formats.

As the capabilities document structure and content of a common Event Service that is applicable to AIM are not yet defined, the registration of xNOTAM generators at the Event Service was not implemented.

### 7.9 Event Notification Subscription

Mobile or built-in client devices need to express interest in relevant NOTAM messages for the context of the flight. This is done by sending a subscription to the ES. The subscription contains a generic filter expression e.g. an OGC filter encoding expression that can be used by the ES for subscription matching. Events that are pushed into the ES are matched against the existing subscriptions taking the filter expression into account. Only events that meet the matching requirements get pushed to the subscribers. This ensures that only events that are relevant to a specific flight context are propagated to that client. This also avoids filtering on the client side.

Subscriptions to xNOTAM messages should be constrained to only the relevant spatial – temporal space that the flight plan specifies to avoid receiving irrelevant messages that would use unnecessary communication bandwidth.

### 7.10 Event Notification Application Protocol

The communication between data providers or producers with the ES, and the ES with client consumers or subscribers is based on XML messages and will carry a GML encoded payload. The clients and the ES will need to agree on an application protocol and wrapper of the GML payload message and also for the subscription handling. As the clients are likely executing in the more constrained environment the protocol should be targeted at the clients' capabilities. The client and ES also need to define a Message Exchange Pattern (MEP) to define the processing and confirmation of delivered messages. The client subscription procedure with the ES should be as reliable as possible and secured to prevent message interception and false subscriptions. As this is only required at the beginning of the flight plan the burden on the client is minimal especially when it is done when still at the gate.

In future implementations the usage of the Common Alerting protocol (CAP) can also be considered as it can carry an xNOTAM as a payload in the message.

For notification delivery both the use of a leaner REST approach and maintaining a subscription channel sequence number to allow the client to check for lost messages or gaps should be considered. This could be done in two separate phases 1. a notification pushed with an ID and 2. a feed read from the client to get the payload data. This still incurs problems with firewalls and DHCP managed IP addresses. These problems are further described in the SWE Security ER.

To address the bandwidth constraint of the communication between ES and clients it is also possible to use an optimization protocol like MTOM (Message Transfer Optimization Protocol) and compressed XML.

Another alternative could be to use WS-Eventing that provides a standard for asynchronous notification of events between XML-based Web services. WS-Eventing is a set of protocols, message formats and interfaces that allow a Web service to subscribe or accept subscriptions for event notifications. The WS-Eventing specification defines a single delivery mode, Push Mode, which is simple asynchronous messaging and does not prescribe the usage of SOAP but does not exclude it either.

### 7.11 Event Notification Transport protocol

The communication protocol between the endpoints in the AIM system will use a standards based transport protocol for message delivery and security requirements. This might require transport level security like HTTPS or WS-Security which is based on SOAP messages if message level security using encryption and digital signatures are required.

As xNOTAMs are related to flight security and navigation it may be a general requirement in this domain to have security measures in place but it might be impossible to implement in the AIM test bed implementation. This will also again depend on the capabilities of the clients and the available communication links while the airplane is in the air.

### 7.12 Event Notification Data Integrity

Since AIXM is based on temporal deltas the integrity of the data depends on the full history of changes since the last baseline. In an environment with intermittent internet connection, such as a plane's flight deck, the issue of data integrity must be addressed. For example:

Event A – informed that a runway has visibility problems but is still open

Event B – informed that the runway is now closed

Event C – informs that the visibility is improving

If the client does not receive event B, the operational picture will show that the visibility issue came and went but the runway never closed and is still open (which is not true).

To address this issue a reliable messaging architecture is needed where either the Event Service manages and resolves any event reception interferences.

### 7.13 Event Notification Security

This subclause will deal with security issues of the event notification system and authorization and authentication procedures.

The implementation of security measures are out of scope in OWS-6, but it is recommended for an operational system. Since the AIXM data and NOTAMs are essential for flight security it has to be ensured that these data are not tampered with or are issued from unauthorized sources. It is therefore recommended that an operational system will implement available security technologies that cover authentication of providers and consumers of messages via digital certificates of security tokens based on standards such as SAML, authorization and access control policies via Policy Decision Point/Policy Enforcement Point (PDP/PEP) services using XACML or GeoXACML policies and a Public Key Infrastructure (PKI) for message encryption and digital signatures of the messages send and receive between the clients, the ES and the WFS.

## 7.14 Client Operations

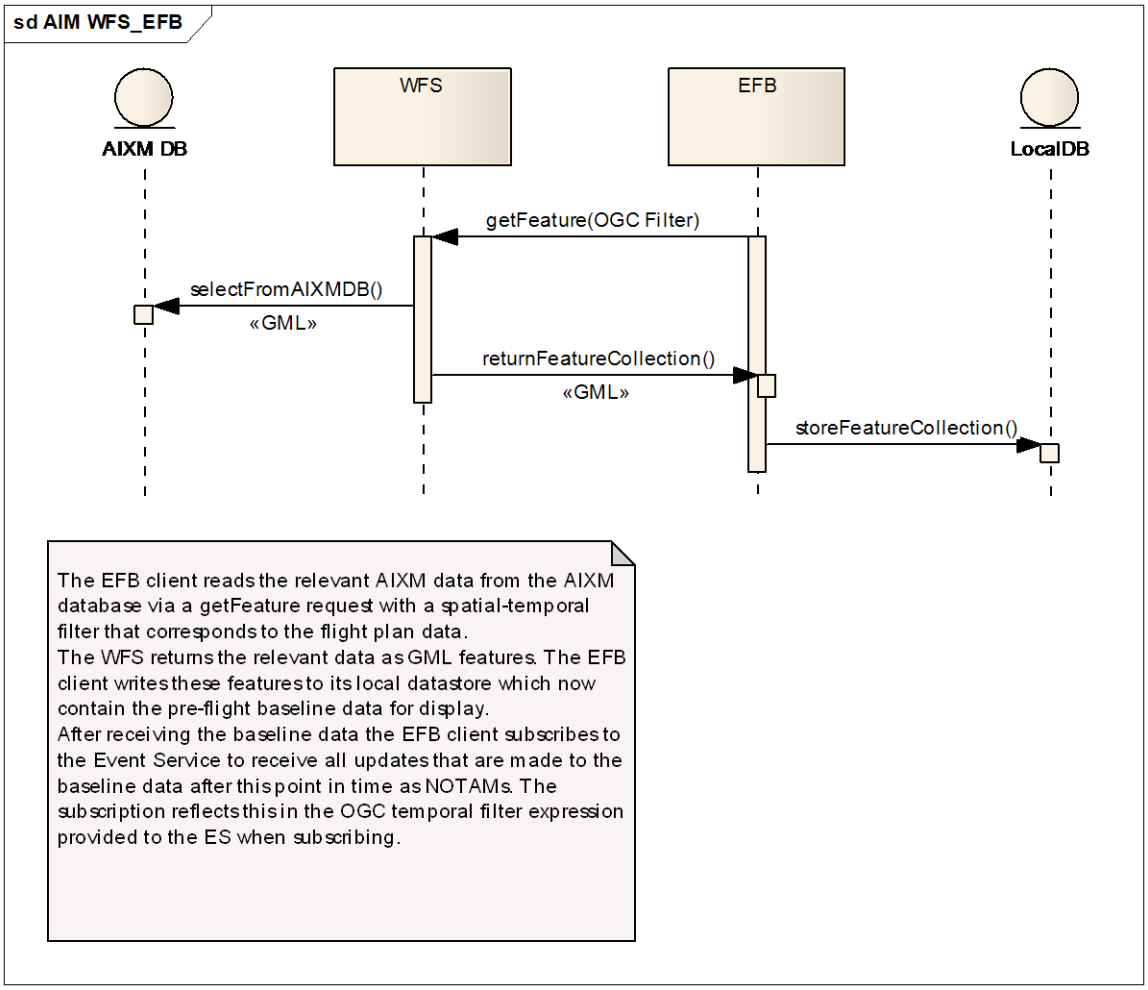



**Figure 9 – Retrive Updates via WFS Request**

### 7.14.1 Pre-Flight Initial Data Load

This will retrieve data from the AIXM database via a WFS *GetFeature* request to get all Baseline data and Deltas that are relevant to the flight plan before take off. In order to receive the relevant data the client has to generate a WFS request that contains an OGC filter encoding expression to constrain the data to the specific areas covering the flight routes.

Without an OGC filter, a request could result in a massive amount of data to be returned by the server, wasting both network bandwidth and processing time at the client for irrelevant data. Although the WFS specification allows the amount of requested features to be delimited through a MAXFEATURES parameter, but this isn't a viable option either because a client does not know beforehand the geographical location and content of the returned data.

A spatial OGC filter encoding expression allows the area of interest for data queries to be defined. This may require more than one request, in order to delimit the complexity of the request and the number of feature instances that are returned in the response. A simple but

effective spatial OGC filter uses a geographical bounding box to define the area of interest. Such a request has the following format, and may be parameterized to be replaced at runtime at the client:

```
<?xml version="1.0" encoding="UTF-8"?>
<wfs:GetFeature xmlns:wfs="http://www.opengis.net/wfs"
 xmlns:aixm="http://www.aixm.aero/schema/5.0"
 xmlns:ogc="http://www.opengis.net/ogc"
 xmlns:gml="http://www.opengis.net/gml"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <wfs:Query typeName="aixm:RunwayElement">
       <ogc:Filter>
    <ogc:BBOX>

<ogc:PropertyName>timeSlice/RunwayElementTimeSlice/extent/ElevatedSurface</ogc:PropertyName>
            <gml:Envelope>
            <gml:lowerCorner srsDimension="2">-97.08500339403274
32.87225782312555</gml:lowerCorner>
            <gml:upperCorner srsDimension="2">-96.99884156350542
32.91926992256661/gml:upperCorner>
          </gml:Envelope>
    </ogc:BBOX>
      </ogc:Filter>
   </wfs:Query>
 </wfs:GetFeature>
```

**Listing 3: Example of use of a BBOX spatial filter to retrieve RunwayElement data for the area of Dallas Forth Worth International Airport**

### 7.14.2 In-Flight diversion airport/runway request

During the flight, a pilot might need to search for aeronautical data within the vicinity of the flight plan. For example, the AIM scenario describes the query for alternate diversion airports due to bad weather conditions at the primary selected diversion airport (ILN).

To support spatial filtering for an area around a flight path, a DWithin filter can be used. This type of filter only returns data that is contained within a given distance from a path. For example, the following query searches for airports that lie within 100 nautical miles from the flight path:

```
<?xml version="1.0" encoding="UTF-8"?>
<wfs:GetFeature xmlns:wfs="http://www.opengis.net/wfs"
 xmlns:aixm="http://www.aixm.aero/schema/5.0"
 xmlns:ogc="http://www.opengis.net/ogc"
 xmlns:gml="http://www.opengis.net/gml"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <wfs:Query typeName="aixm:AirportHeliport">
       <ogc:Filter>
          <ogc:DWithin>
              <ogc:PropertyName>
timeSlice/AirportHeliportTimeSlice/ARP/ElevatedPoint</ogc:PropertyName>
              <gml:LineString srsName="urn:ogc:def:crs:OGC:1.3:CRS84">
                     <gml:coordinates xmlns:gml="http://www.opengis.net/gml" decimal="."
cs="," ts=" ">-97.03833333,32.89666667 …</gml:coordinates>
              </gml:LineString>
             <Distance units="nautical mile">100</Distance>
          </ogc:DWithin>
       </ogc:Filter>
   </wfs:Query>
 </wfs:GetFeature>
```

**Listing 4: Example of use of a DWitin spatial filter to retrieve airports within the vicinity of the flight path**

The request can be extended with other filtering properties to define further characteristics of desired airports. For example, a pilot might only require airports that offer runways with a minimum distance or certain surface characteristics. The following query searches for runways that pass the same DWithin spatial filter as in the previous query, but that also are at least 1500 meters long and do not have a grass surface.

```
<GetFeature xmlns="http://www.opengis.net/wfs"
xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:gml32="http://www.opengis.net/gml/3.2"
xmlns:gml="http://www.opengis.net/gml" xmlns:ogc="http://www.opengis.net/ogc"
xmlns:ows="http://www.opengis.net/ows"
xmlns:aixm="http://www.aixm.aero/schema/5.0"
xmlns:fes="http://www.opengis.net/fes/2.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation=""
service="WFS" version="1.1.0">
  <Query typeName="aixm:Runway">
    <ogc:Filter>
      <ogc:And>
        <ogc:DWithin>
          …
        </ogc:DWithin>
        <ogc:Not>
          <ogc:PropertyIsEqualTo>

<ogc:PropertyName>timeSlice/RunwayTimeSlice/surfaceProperties/SurfaceCharacteristics/compositi
on</ogc:PropertyName>
            <ogc:Literal>GRASS</ogc:Literal>
          </ogc:PropertyIsEqualTo>
        </ogc:Not>
        <ogc:PropertyIsGreaterThan>
          <ogc:PropertyName>timeSlice/RunwayTimeSlice/length</ogc:PropertyName>
          <ogc:Literal>1500</ogc:Literal>
        </ogc:PropertyIsGreaterThan>
        <ogc:PropertyIsEqualTo>
          <ogc:PropertyName>timeSlice/RunwayTimeSlice/length/@uom</ogc:PropertyName>
          <ogc:Literal>M</ogc:Literal>
        </ogc:PropertyIsEqualTo>
      </ogc:And>
    </ogc:Filter>
  </Query>
</GetFeature>
```

**Listing 5: Example of use of a composite filter to retrieve runways that comply to several spatial and non-spatial requirements**

This reveals a difficulty when it comes to comparing numbers that are expressed in a certain unit of measure. Because the unit of measure used for the runway length is defined in a separate element in AIXM 5, a client needs to make sure that this is taken into account when specifying a runway length.

The resulting airport or runway data can then be used by the clients to retrieve other related aeronautical data. For example, RunwayElement data might be queried to visualize the geometry of the runways at one or more airports. This can be done by using the aixm:identifier information available in the returned runway data, and turning them into a property-based OGC filter to delimit the data to the previously requested runways.

```
<GetFeature xmlns="http://www.opengis.net/wfs"
xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:gml32="http://www.opengis.net/gml/3.2"
xmlns:gml="http://www.opengis.net/gml" xmlns:ogc="http://www.opengis.net/ogc"
xmlns:ows="http://www.opengis.net/ows"
xmlns:aixm="http://www.aixm.aero/schema/5.0"
xmlns:fes="http://www.opengis.net/fes/2.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation=""
```

```
service="WFS" version="1.1.0" outputFormat="text/xml; subtype=gml/3.1.1">
  <Query typeName="aixm:RunwayElement">
    <ogc:Filter>
      <ogc:Or>
        <ogc:PropertyIsEqualTo>

<ogc:PropertyName>timeSlice/RunwayElementTimeSlice/associatedRunway/@href</ogc:PropertyName>
          <ogc:Literal>KBWI_R0421_10-28</ogc:Literal>
        </ogc:PropertyIsEqualTo>
        <ogc:PropertyIsEqualTo>

<ogc:PropertyName>timeSlice/RunwayElementTimeSlice/associatedRunway/@href</ogc:PropertyName>
          <ogc:Literal>KPHL_R0441_9L-27R</ogc:Literal>
        </ogc:PropertyIsEqualTo>
        <ogc:PropertyIsEqualTo>

<ogc:PropertyName>timeSlice/RunwayElementTimeSlice/associatedRunway/@href</ogc:PropertyName>
          <ogc:Literal>KPHL_R0443_17-35</ogc:Literal>
        </ogc:PropertyIsEqualTo>
        <ogc:PropertyIsEqualTo>

<ogc:PropertyName>timeSlice/RunwayElementTimeSlice/associatedRunway/@href</ogc:PropertyName>
          <ogc:Literal>KPHL_R0440_9R-27L</ogc:Literal>
        </ogc:PropertyIsEqualTo>
        <ogc:PropertyIsEqualTo>

<ogc:PropertyName>timeSlice/RunwayElementTimeSlice/associatedRunway/@href</ogc:PropertyName>
          <ogc:Literal>KEWR_R0463_4R-22L</ogc:Literal>
        </ogc:PropertyIsEqualTo>
        <ogc:PropertyIsEqualTo>

<ogc:PropertyName>timeSlice/RunwayElementTimeSlice/associatedRunway/@href</ogc:PropertyName>
          <ogc:Literal>KEWR_R0460_11-29</ogc:Literal>
        </ogc:PropertyIsEqualTo>
      </ogc:Or>
    </ogc:Filter>
  </Query>
</GetFeature>
```

**Listing 6: Example of use of a property-based filter to retrieve the geometry of some runways at various airports**

### 7.14.3 Compressing AIXM 5.0 data for efficient transmission

In an environment with limited bandwidth, measures can be taken to improve the efficiency of data transmission between the WFS server and its clients. Because AIXM is XML-based, an interesting approach is to compress the data that is being transmitted. XML is very verbose because of its textual heritage, and is therefore well suited for compression. Experiments show that AIXM 5.0 data can be reduced to up to 5% of its original size when compressed. One way to compress transmitted data is to use the content encodings defined in HTTP 1.1 (RFC 2616). It will have to be investigated if the decompressing processing time overhead is worth the transfer time improvement. These encodings include commonly used compressions formats like gzip, x-zip and deflate. Clients can use the HTTP header field ‘Accept-Encoding’ to notify the server that compressed data is supported in one of the specified encodings. A server that recognizes this header field can take it into account when sending data to the client. Through the HTTP header field ‘Content-Encoding’, the client can detect the encoding used by the server.

An alternative to traditional compression methods is the use of Binary XML. Next to the advantage of compression, it also offers the advantage of decreased parsing time or random access. However, there is currently no widely adopted standard for Binary XML, which might affect the interoperability.

### 7.14.4 Event Notification Subscription Message

To be able to receive updates at the client after the loading of the latest AIXM updates related to the flight plan as xNOTAMs, the client system has to subscribe to the Event Service by sending a subscription message.

This message will create a subscription at the ES. It contains an OGC filter to constrain the events to the flight scope and a delivery endpoint to specify the address to send the NOTAM message to.

A typical subscription request message looks like the following:

```
<?xml version="1.0" encoding="UTF-8"?>
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:wsa="http://www.w3.org/2005/08/addressing" xmlns:wsnt="http://docs.oasisopen.
org/wsn/b-2">
<soap:Header>
<wsa:To>http://v-tml.uni-muenster.de:8080/SESforAIXMv02/
services/SesPortType</wsa:To>
<wsa:Action>http://docs.oasis-open.org/wsn/bw-
2/NotificationProducer/SubscribeRequest</wsa:Action>
<wsa:MessageID>uuid:4e595160-185a-9b3c-3eb6-592c7c5b0c7a</wsa:MessageID>
<wsa:From>
<wsa:Address>http://www.w3.org/2005/08/addressing/role/anonymous</ws
a:Address>
</wsa:From>
</soap:Header>
<soap:Body>
<wsnt:Subscribe>
<wsnt:ConsumerReference>
<wsa:Address>http://v-tml.uni-muenster.de:8080/SESMuseConsumer</
wsa:Address>
</wsnt:ConsumerReference>
<wsnt:Filter>
<wsnt:MessageContent Dialect="http://www.w3.org/TR/1999/RECxpath-
19991116">*</wsnt:MessageContent>
</wsnt:Filter>
</wsnt:Subscribe>
</soap:Body>
</soap:Envelope>
```

**Listing 7: Example of a subscription request**

As can be seen this subscription message example is contained in a SOAP envelope and uses WS-Addressing elements to specify the target address of the receiving end of the xNOTAM target.

The example illustrates the subscription for all possible events, indicated by the '*' character in the Filter/MessageContent element. By replacing the value with an OGC Filter encoding expressions, a subscription can be made for a particular area and/or for a specified time range. For example, by inserting the following OGC Filter encoding expression, a client only subscribes for events that are located in the area around Dallas Forth Worth International Airport:

```
<fes:Filter
xmlns:fes="http://www.opengis.net/fes/2.0"
xmlns:gml32="http://www.opengis.net/gml/3.2"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.opengis.net/fes/2.0
http://schemas.opengis.net/filter/2.0.0/filterAll.xsd">
  <fes:BBOX>
```

```
    <gml32:Envelope srsName="urn:fes:def:crs:EPSG::4326">
      <gml32:lowerCorner> -97.08500339403274 32.87225782312555</gml32:lowerCorner>
      <gml32:upperCorner> -96.99884156350542 32.91926992256661</gml32:upperCorner>
    </gml32:Envelope>
  </fes:BBOX>
</fes:Filter>
```

**Listing 8: Example of a filter expression**

There might be a problem with the clients IP address if it is configured via DHCP as this might change the IP address after the registration has occurred. This will need some further investigation. This also relates to problems accessing clients that are firewall protected and cannot be accessed through a web service port. This topic is also discussed in the OWS-6 SWE Security Engineering Report.

One possible solution to this problem could be to use an Asynchronous Request processing (ARP) approach, where the client initiates a HTTP request which is held open for an undetermined time using an ARP API on the application server that hosts the event service. This is an evolving technology now frequently used in Rich Internet Applications (RIA) based on AJAX. This will require some further considerations for the client and server side implementations and platforms.

Another solution could be the use of an extra event consumer component that lies between an AIM client and the AIM Event Service, and that offers synchronous communication with the AIM client in order to resolve dynamic IP issues. The event consumer listener itself needs to be reachable over the Internet, to be able receive the events from the AIM Event Service asynchronously. As such, it is registered as a consumer at the AIM Event Service. Next, it offers requests for the AIM clients to check for the latest events in a synchronous way. The AIM clients can use these requests through polling to retrieve and/or check for any new events. Although polling might be unfeasible in a highly unreliable connectivity environment, it removes the requirement of the AIM client to be reachable over the Internet. This solution has been implemented by Luciad.

The Carbon Project implemented a similar mechanism using RESTful WCF (Windows Communications Foundation) to bypass firewall restrictions. The proxy ES provides an end-unit to the AIM ES while channeling the interaction from and to the client inside the firewalled domain.

Also the use of multiple XMPP servers (one in the internet, one in a private demilitarized zone) may be a way to solve this problem and will also be mentioned in the future work section.

### 7.15 Filtering

It should be further noted that the subscription message for the AIM Event Service will contain an OGC filter expression encoded in FES 2.0 to specify the spatial and temporal scope of this subscription as this is related to the flight plan which specifies the departure

airport, destination airport, route, area airports and alternate airports etc that relates to that particular flight.

The flight route is defined by line- strings or waypoints that are encoded in GML point geometries as defined by the following schema:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:ns0="http://www.luciad.com/dynamicNamespace/0"
xmlns:gml="http://www.opengis.net/gml" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xs="http://www.w3.org/2001/XMLSchema" xsi:schemaLocation="http://www.opengis.net/gml
net/opengis/gml/3.1.1/base/gml.xsd http://www.luciad.com/dynamicNamespace/0 flightplan.xsd "
targetNamespace="http://www.luciad.com/dynamicNamespace/0" elementFormDefault="qualified">
<xs:import namespace="http://www.opengis.net/gml"
schemaLocation="net/opengis/gml/3.1.1/base/gml.xsd"/>
 <xs:element name="CheckPoint" type="ns0:CheckPointType" substitutionGroup="gml:_Feature"/>
  <xs:complexType name="CheckPointType">
    <xs:complexContent>
      <xs:extension base="gml:AbstractFeatureType">
        <xs:sequence>
          <xs:element name="Name" type="xs:string" minOccurs="0"/>
          <xs:element name="geometryProperty" type="gml:GeometryPropertyType" minOccurs="0"/>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
</xs:schema>
```

**Listing 9: The XML Schema used for the flight path in the scenario**

By using point geometries instead of line-strings, additional properties can be stored for each point – e.g., the waypoint designator. This can be useful if a client wants to label each waypoint.

The waypoints in the flight plan are expanded by a buffer to specify the relevant spatial area of interest for the NOTAM event matching service and the requesting of baseline AIXM data updates via the AIXM WFS spatial filter operations.

The temporal operators that are defined in the FES 2.0 specification are:

```
<fes:Temporal_Capabilities>
    <fes:TemporalOperands>
           <fes:TemporalOperand name="TimePeriod"/>
    </fes:TemporalOperands>
    <fes:TemporalOperators>
         <fes:TemporalOperator name="After"/>
         <fes:TemporalOperator name="Before"/>
         <fes:TemporalOperator name="Begins"/>
         <fes:TemporalOperator name="BegunBy"/>
         <fes:TemporalOperator name="TContains"/>
         <fes:TemporalOperator name="During"/>
         <fes:TemporalOperator name="TEquals"/>
         <fes:TemporalOperator name="TOverlaps"/>
         <fes:TemporalOperator name="Meets"/>
         <fes:TemporalOperator name="OverlappedBy"/>
         <fes:TemporalOperator name="MetBy"/>
         <fes:TemporalOperator name="Ends"/>
         <fes:TemporalOperator name="EndedBy"/>
     </fes:TemporalOperators>
</fes:Temporal_Capabilities>
```

**Listing 10: FES2.0 Temporal Operators**

The semantics of the temporal operators are depicted and further described in section 7.15 Filtering

Please note that the TOverlaps operator is misspelled in the spec as “TOveralps” which has been addressed in a change request. The CR doc can be found here: http://portal.opengeospatial.org/index.php?m=projects&a=view&project_id=280&tab=2&artifact_id=32657 . Please also note that the TimePeriod operand can have a beginPosition and endPosition values that have an indeterminedPosition attribute is supposed to raise an exception at the application (WFS). This causes problems in AIM as it is often used to specify and “unknown” state of the present or absent timevalue provided to indicate that the value is not likely valid or completely unknown. This is addressed in a change request as well.

The change request will describe additional enumeration values for the “TimeIndeterminateValueType”

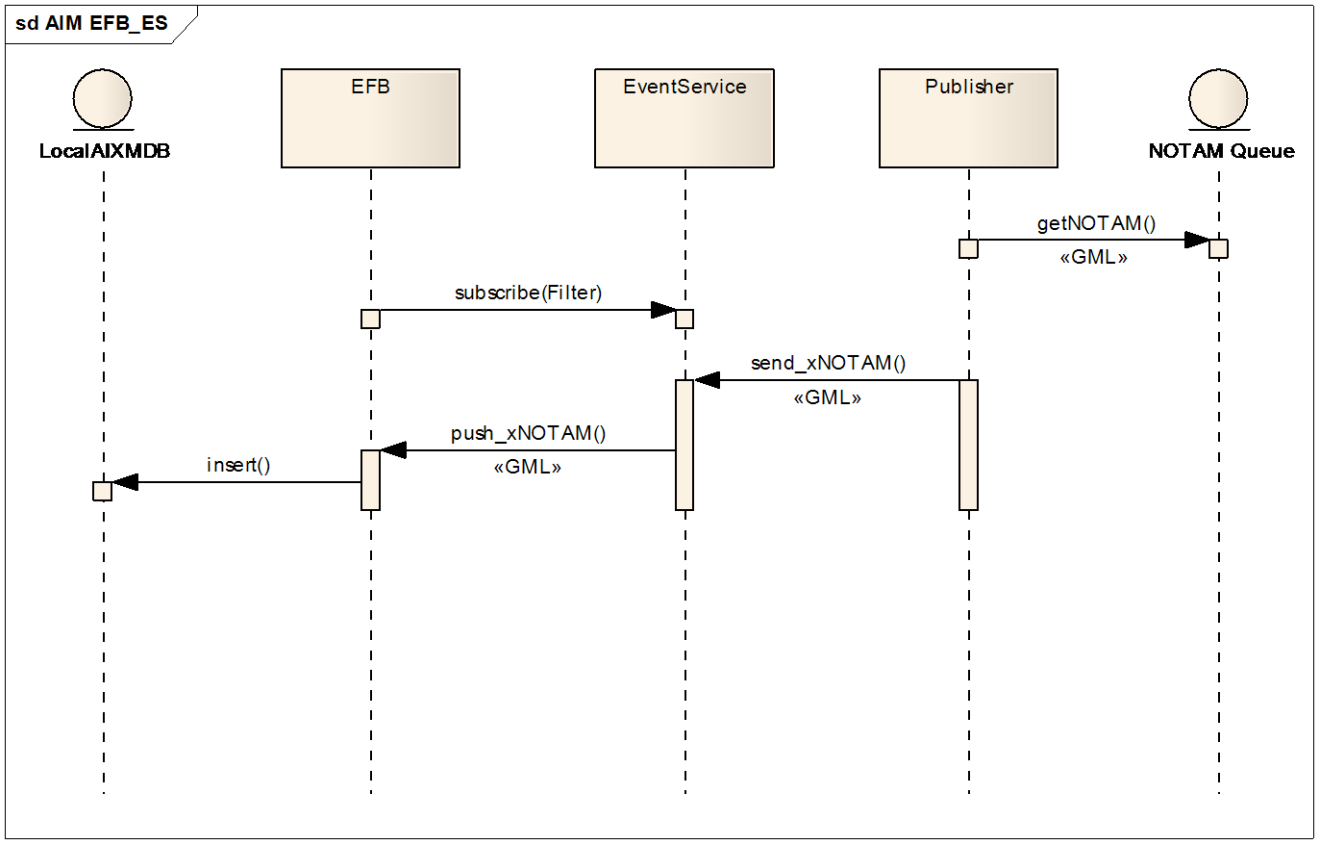



**Figure 10 – Publish & Subscribe Sequence**

The sequence of operations from subscribing to the ES to receiving of xNOTAMs is shown in the sequence diagram of Figure 10. The filter expression that is provided with the subscription message contains a timeslice value that defines the time window for which the client wants to receive NOTAMs that occur between the timestamp values.

An example of a timeslice filter expression is shown below:

```
<?xml version="1.0" encoding="UTF-8"?>
<wfs:GetFeature xmlns:wfs="http://www.opengis.net/wfs"
```

```
   xmlns:aixm="http://www.aixm.aero/schema/5.0" xmlns:ogc="http://www.opengis.net/ogc"
   xmlns:gml="http://www.opengis.net/gml/3.2" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"
   maxFeatures="100">
   <wfs:Query xmlns="http://www.aixm.aero/schema/5.0" typeName="AirspaceUsage">
      <ogc:Filter>
         <fes:Not xmlns:fes="http://www.opengis.net/fes/2.0"
            xmlns:aixm="http://www.aixm.aero/schema/5.0"
xmlns:gml="http://www.opengis.net/gml/3.2">
            <fes:Or>
               <fes:Before>
                  <fes:ValueReference>
                     aixm:timeSlice/aixm:AirspaceUsageTimeSlice/gml:validTime
                  </fes:ValueReference>
                  <fes:Literal>
                     <gml:TimePeriod gml:id="id1">
                        <gml:beginPosition>2008-06-10T10:30:00.000
                        </gml:beginPosition>
                        <gml:endPosition>2008-06-18T13:30:00.000
                        </gml:endPosition>
                     </gml:TimePeriod>
                  </fes:Literal>
               </fes:Before>
               <fes:After>
                  <fes:ValueReference>
                     aixm:timeSlice/aixm:AirspaceUsageTimeSlice/gml:validTime
                  </fes:ValueReference>
                  <fes:Literal>
                     <gml:TimePeriod gml:id="id2">
                        <gml:beginPosition>2008-06-10T10:30:00.000
                        </gml:beginPosition>
                        <gml:endPosition>2008-06-18T13:30:00.000
                        </gml:endPosition>
                     </gml:TimePeriod>
                  </fes:Literal>
               </fes:After>
               <fes:Meets>
                  <fes:ValueReference>
                     aixm:timeSlice/aixm:AirspaceUsageTimeSlice/gml:validTime
                  </fes:ValueReference>
                  <fes:Literal>
                     <gml:TimePeriod gml:id="id3">
                        <gml:beginPosition>2008-06-10T10:30:00.000
                        </gml:beginPosition>
                        <gml:endPosition>2008-06-18T13:30:00.000
                        </gml:endPosition>
                     </gml:TimePeriod>
                  </fes:Literal>
               </fes:Meets>
               <fes:MetBy>
                  <fes:ValueReference>
                     aixm:timeSlice/aixm:AirspaceUsageTimeSlice/gml:validTime
                  </fes:ValueReference>
                  <fes:Literal>
                     <gml:TimePeriod gml:id="id4">
                        <gml:beginPosition>2008-06-10T10:30:00.000
                        </gml:beginPosition>
                        <gml:endPosition>2008-06-18T13:30:00.000
                        </gml:endPosition>
                     </gml:TimePeriod>
                  </fes:Literal>
               </fes:MetBy>
            </fes:Or>
         </fes:Not>
      </ogc:Filter>
   </wfs:Query>
</wfs:GetFeature>
```

**Listing 11: Example of a timeslice filter expression**

NOTAM event messages may also contain bounding box information which is used by the event service matching algorithm to determine whether the event is relevant for the area of interest matching the flight plan route waypoints.

An example of a NOTAM message containing a BBOX envelope is shown below.

```
<aixm:AirportHeliportUsage gml:id="VID2678448">

<event:name>ESSA runways closed</event:name>

<event:description>RWY 08/26, 01L/19R, 01R/19L closed temporarily due to snow contamination
and bad weather, starting 12 FEB 2009 at 07:00</
<gml:boundedBy>
       <gml:Envelope>
               <gml:lowerCorner>17.956,59.660</gml:lowerCorner>
               <gml:upperCorner>17.958,59.662</gml:upperCorner>
         </gml:Envelope>
</gml:boundedBy>

……..
<gml:beginPosition>2009-02-127T07:00:00Z</gml:beginPosition>

<gml:endPosition indeterminatePosition="unknown"/>
```

**Listing 12: NOTAM with BBOX Information**

### 7.15.1 Event Notification Processing

This section describes the event handling of a notification message upon reception of a NOTAM message for a feature type, in a spatial and temporal scope that the client has previously subscribed to be notified of.

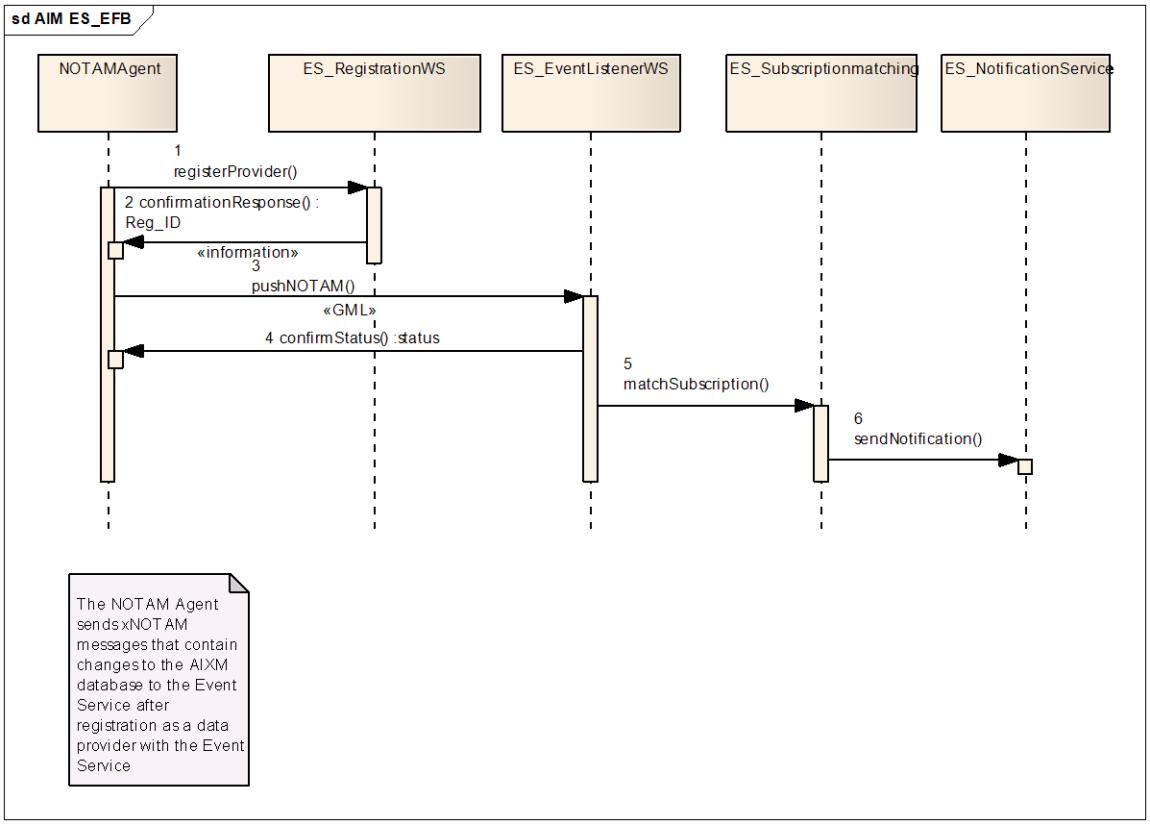



**Figure 11 – Event Processing Sequence Diagram**

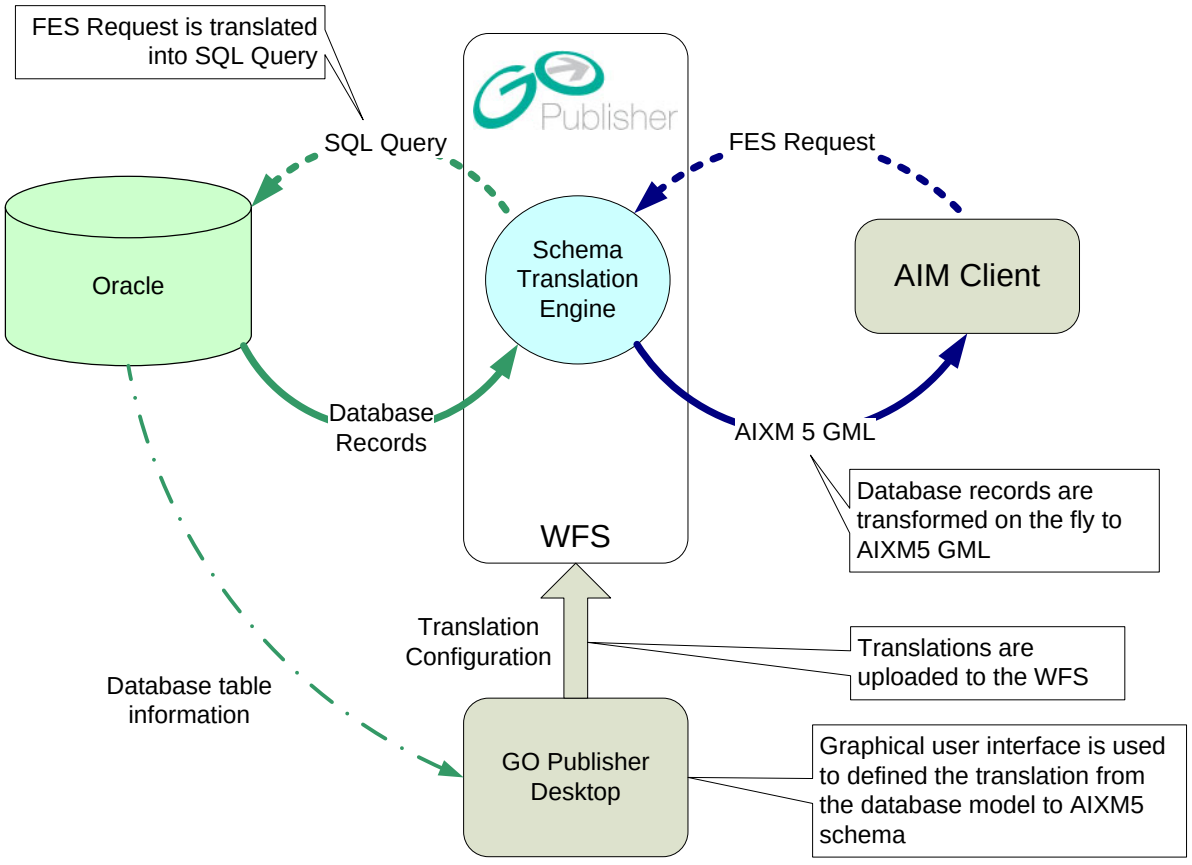
Note that the ES implementation in this OWS-6 AIM thread did not implement the registration phase of the temporal WFS as in the above diagram as only one event producer was used.

The Event Service acts as the information broker between the AIXM database change events and the clients (EFB) that want to get notifications of events that are relevant to their flight plan i.e. it spatial-temporal scope. The events that are created when a change to an AIXM feature occurs are propagated to the clients through the Event Service as GML encoded AIXM features. This is shown in the sequence diagram in Figure 11.

### 7.16 Temporal WFS with AIXM5.0 Schema mapping

WFS Server Side Architecture

The following diagram shows the server side components deployed in the WFS architecture hosted by Snowflake:




The WFS architecture comprises of:

1. Oracle 10gR2 for the underlying database
2. An AIXM5 relational database model automatically generated and loaded from the AIXM5 schema using Snowflake's GO Loader product.

3. GO Publisher Desktop to create valid AIXM5 GML from the relational model

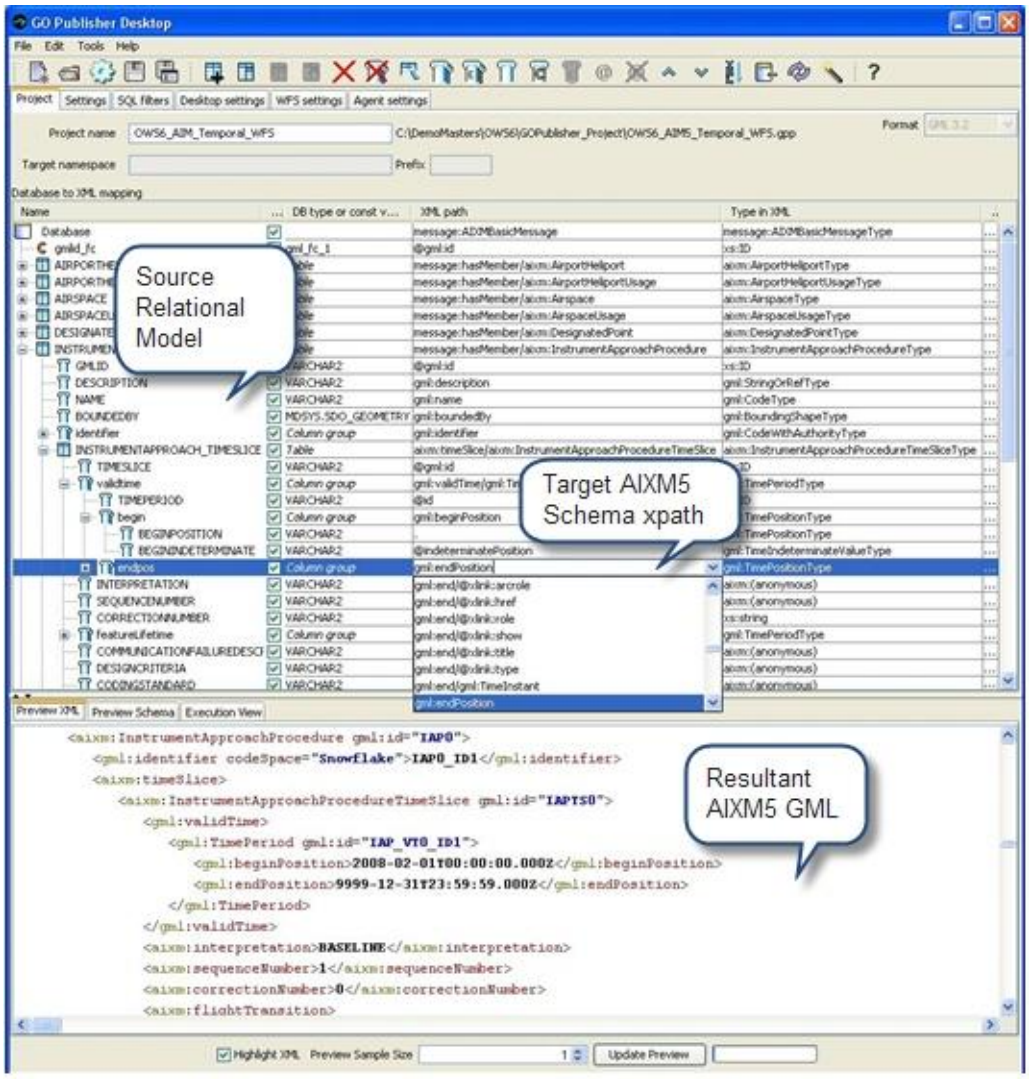
4. GO Publisher WFS deployed by GO Publisher Desktop to Apache Tomcat.

The architecture was deployed using the latest COTS software, with the exception of GO Publisher WFS which was extended to support the additional Filter Encoding Specification 2.0 Temporal Operators.

### 7.17 Schema Translating WFS

For the AIXM5 Web Feature Service Snowflake deployed its GO Publisher WFS. GO Publisher WFS is a schema translating WFS which enables it to solve a key use case of 'publishing data to a given GML application schema (AIXM5) from an existing relational database model'. In the OWS6-AIM testbed the relational database model was a fully normalized model created by Snowflake's GO Loader product directly from the AIXM5 schema. This model was used purely for productivity in the testbed and the capabilities of the WFS to serve AIXM5 are in no way tied to a specific relational model.

Setting up the WFS required a transform to be created from the relational model onto the AIXM5 application schema. To achieve this GO Publisher Desktop was used to 'map' the components of the relational model onto the AIXM5 schema elements.




Screenshot showing GO Publisher desktop mapping to AIXM5

Once this mapping was achieved, an instance document was created and validated inside GO Publisher Desktop before publishing as WFS. The mapping process and hosting of the AIXM5 WFS on the underlying database was achieved within the first week of the testbed.

### 7.18 Temporal Extensions ISO 19108 – Temporal Schema

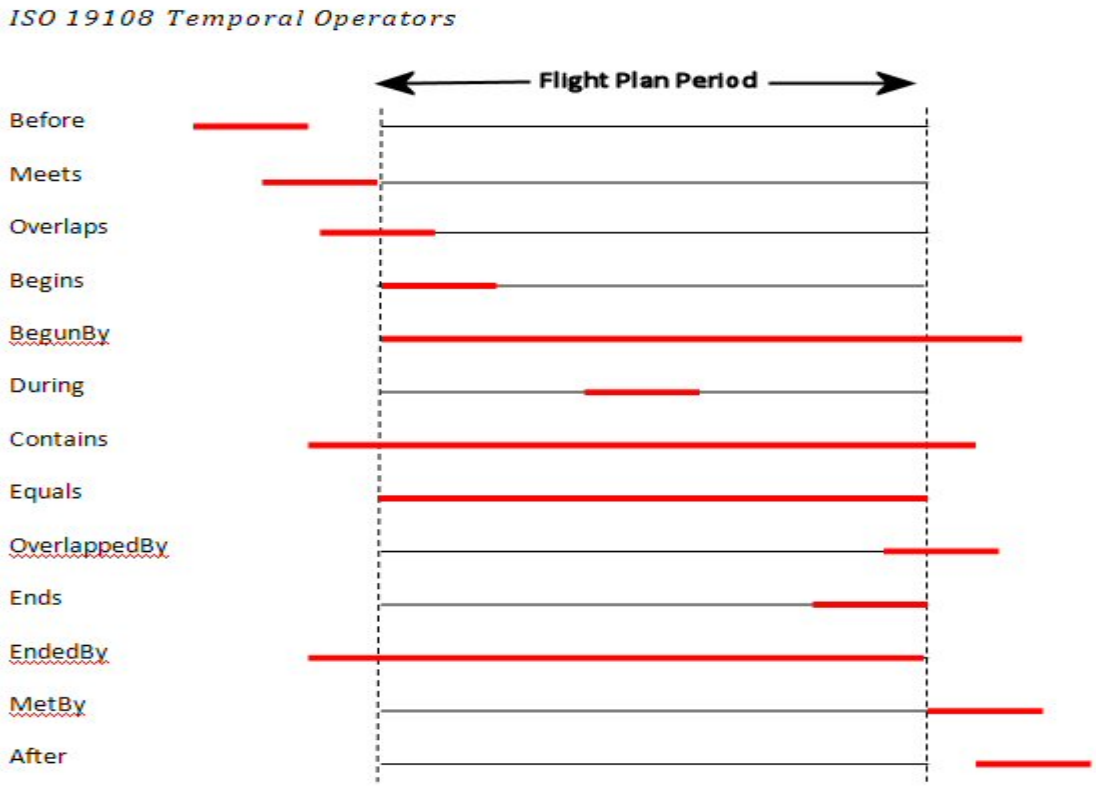
One of the major objectives of the AIM thread is to

"… demonstrate the use of the Web Feature Service and the Filter Encoding Specification to identify the necessary enhancements needed to provide 4D flight trajectory information filtering.",

and to:

"Exercise the temporal aspects of AIXM5 and provide temporally-based spatial queries using WFS/Filter Encoding"

To achieve these goals Snowflake extended its COTS product GO Publisher WFS to support the temporal elements defined in the draft Filter Encoding Specification (FES) 2.0. FES2.0 implements ISO19108 – Temporal Schema and covers the following temporal operations:



Rather than implement the full FES2.0 draft standard, Snowflake 'back ported' only the temporal elements of FES2.0. Adding the following temporal operators to its existing FES1.1 implementation:

```
<fes:Temporal_Capabilities>
    <fes:TemporalOperands>
           <fes:TemporalOperand name="TimePeriod"/>
    </fes:TemporalOperands>
    <fes:TemporalOperators>
         <fes:TemporalOperator name="After"/>
         <fes:TemporalOperator name="Before"/>
         <fes:TemporalOperator name="Begins"/>
         <fes:TemporalOperator name="BegunBy"/>
         <fes:TemporalOperator name="TContains"/>
         <fes:TemporalOperator name="During"/>
         <fes:TemporalOperator name="TEquals"/>
         <fes:TemporalOperator name="TOverlaps"/>
         <fes:TemporalOperator name="Meets"/>
         <fes:TemporalOperator name="OverlappedBy"/>
         <fes:TemporalOperator name="MetBy"/>
         <fes:TemporalOperator name="Ends"/>
         <fes:TemporalOperator name="EndedBy"/>
     </fes:TemporalOperators>
</fes:Temporal_Capabilities>
```

As this was implemented on a WFS1.1 release rather than a WFS2.0 future work should include the implementation of a WFS2.0 with full FES2.0 support that is also supported by the clients.

### 7.18.1 Change Request to support AnyInteracts query

A key use case for the testbed was to perform a query that would:

"Return Runway timeslices that fall within 100 nautical miles of a flight path (using the runway bounding box) during the period of the flight (Temporal queries) which are not covered in grass (PropertyIsEqualTo /Not .i.e. hard surface) and are over (PropertyIsGreaterThan ) 1500 Metres (5000 ft) long. Result includes an xlink to the Airports containing valid runways.

In order to support the 'during the period of the flight' aspect of the temporal query, the following query logic was created in the Filter:

NOT Before

```
Flightpath.end.position < AIXM.begin.position
```

NOT Meets

```
Flightpath.end.position = AIXM.begin.position
```

NOT After

```
Flightpath.begin.position > AIXM.end.position AND Flightpath.end.position >
AIXM.end.position
```

NOT MetBy

```
Flightpath.begin.position = AIXM.end.position
```

Whilst this was achieved using the FES2.0 operators, it was seen as being unnecessarily verbose and over complicated for the client implementation, for example:

```
<?xml version="1.0" encoding="UTF-8"?>
<wfs:GetFeature xmlns:wfs="http://www.opengis.net/wfs"
xmlns:aixm="http://www.aixm.aero/schema/5.0"
xmlns:ogc="http://www.opengis.net/ogc" xmlns:gml="http://www.opengis.net/gml/3.2"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" maxFeatures="100">
   <wfs:Query xmlns="http://www.aixm.aero/schema/5.0" typeName="Runway">
      <ogc:Filter>
         <ogc:And xmlns:fes="http://www.opengis.net/fes/2.0"
xmlns:ogc="http://www.opengis.net/ogc"
xmlns:aixm="http://www.aixm.aero/schema/5.0"
xmlns:gml="http://www.opengis.net/gml/3.2">
            <ogc:PropertyIsEqualTo>
               <ogc:PropertyName>
timeSlice/RunwayTimeSlice/type</ogc:PropertyName>
               <ogc:Literal>RWY</ogc:Literal>
            </ogc:PropertyIsEqualTo>
            <ogc:Not>
               <ogc:PropertyIsEqualTo>
                  <ogc:PropertyName>
timeSlice/RunwayTimeSlice/surfaceProperties/SurfaceCharacteristics/composition
</ogc:PropertyName>
                  <ogc:Literal>GRASS</ogc:Literal>
               </ogc:PropertyIsEqualTo>
            </ogc:Not>
            <ogc:And>
               <ogc:PropertyIsGreaterThan>

   <ogc:PropertyName>timeSlice/RunwayTimeSlice/length</ogc:PropertyName>
                  <ogc:Literal>1500</ogc:Literal>
               </ogc:PropertyIsGreaterThan>
               <ogc:PropertyIsEqualTo>

   <ogc:PropertyName>timeSlice/RunwayTimeSlice/length/@uom</ogc:PropertyName
>
                  <ogc:Literal>M</ogc:Literal>
               </ogc:PropertyIsEqualTo>
            </ogc:And>
            <fes:Not>
               <fes:Or>
                  <fes:Before>

   <fes:ValueReference>aixm:timeSlice/aixm:RunwayTimeSlice/gml:validTime</fes:Va
lueReference>
```

```
                        <fes:Literal>
                            <gml:TimePeriod gml:id="id1">
                                <gml:beginPosition>2008-06-
10T10:30:00.000Z</gml:beginPosition>
                                <gml:endPosition>2008-06-
18T13:30:00.000Z</gml:endPosition>
                            </gml:TimePeriod>
                        </fes:Literal>
                    </fes:Before>
                    <fes:After>

    <fes:ValueReference>aixm:timeSlice/aixm:RunwayTimeSlice/gml:validTime</fes:Va
lueReference>
                        <fes:Literal>
                            <gml:TimePeriod gml:id="id2">
                                <gml:beginPosition>2008-06-
10T10:30:00.000Z</gml:beginPosition>
                                <gml:endPosition>2008-06-
18T13:30:00.000Z</gml:endPosition>
                            </gml:TimePeriod>
                        </fes:Literal>
                    </fes:After>
                    <fes:Meets>

    <fes:ValueReference>aixm:timeSlice/aixm:RunwayTimeSlice/gml:validTime</fes:Va
lueReference>
                        <fes:Literal>
                            <gml:TimePeriod gml:id="id3">
                                <gml:beginPosition>2008-06-
10T10:30:00.000Z</gml:beginPosition>
                                <gml:endPosition>2008-06-
18T13:30:00.000Z</gml:endPosition>
                            </gml:TimePeriod>
                        </fes:Literal>
                    </fes:Meets>
                    <fes:MetBy>

    <fes:ValueReference>aixm:timeSlice/aixm:RunwayTimeSlice/gml:validTime</fes:Va
lueReference>
                        <fes:Literal>
                            <gml:TimePeriod gml:id="id4">
                                <gml:beginPosition>2008-06-
10T10:30:00.000Z</gml:beginPosition>
                                <gml:endPosition>2008-06-
18T13:30:00.000Z</gml:endPosition>
                            </gml:TimePeriod>
                        </fes:Literal>
                    </fes:MetBy>
```

```
                    </fes:Or>
                </fes:Not>
                <ogc:DWithin>
                    <ogc:PropertyName> gml:boundedBy</ogc:PropertyName>
                    <gml:LineString srsName="urn:ogc:def:crs:OGC:1.3:CRS84">
                        <gml:coordinates xmlns:gml="http://www.opengis.net/gml"
decimal="." cs="," ts=" ">-97.03833333,32.89666667 -96.995,32.78333333 -
96.94,32.765 -96.83166667,32.76833333 -96.33166667,32.97 -95.56333333,33.075 -
94.07333333,33.51333333 -92.64333333,34.06 -92.55,34.095 -89.98333333,35.015 -
85.18166667,36.61333333 -84.56666667,36.80166667 -84.05,36.95666667 -
83.03,37.255 -82.13666667,37.505 -81.80666667,37.59666667 -81.12333333,37.78 -
80.39,38.04333333 -79.73166667,38.27166667 -78.99833333,38.50666667 -
77.46666667,38.935 -76.97833333,39.495 -76.29166667,40.12 -
75.68333333,40.58166667 -75.455,40.72666667 -74.86833333,40.995 -
73.82166667,41.665 -72.71666667,42.16166667 -70.61333333,43.425 -
69.49166667,43.925 -67.15666667,44.90166667 -65.87166667,45.40666667 -
64.57166667,46.18833333 -61.77333333,47.43 -58.67,48.58333333 -55,49.71666667 -
52.06833333,50.50333333 -50,52 -40,56 -30,59 -20,60 -10,60 -9.5,59.965 -
8.568333333,60.02 -1.286666667,59.87833333 0.015,59.99166667
5.211666667,60.31166667 7.5,60.295 9.915,60.23666667 11.075,60.19166667
12.51333333,60.12833333 14.17,60.03 14.40833333,60.01833333 16.73833333,59.845
16.99,59.825 17.91833333,59.65166667</gml:coordinates>
                    </gml:LineString>
                    <Distance units="nautical mile">100</Distance>
                </ogc:DWithin>
            </ogc:And>
        </ogc:Filter>
    </wfs:Query>
</wfs:GetFeature>
```

To resolve this complexity, it was decided to raise a Change Request for the addition of a shortcut AnyInteracts temporal operator. Replacing the same query with an AnyInteracts spatial operator results in:

```
<?xml version="1.0" encoding="UTF-8"?>
<wfs:GetFeature xmlns:wfs="http://www.opengis.net/wfs"
xmlns:aixm="http://www.aixm.aero/schema/5.0"
xmlns:ogc="http://www.opengis.net/ogc" xmlns:gml="http://www.opengis.net/gml/3.2"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" maxFeatures="100">
   <wfs:Query xmlns="http://www.aixm.aero/schema/5.0" typeName="Runway">
      <ogc:Filter>
         <ogc:And xmlns:fes="http://www.opengis.net/fes/2.0"
xmlns:ogc="http://www.opengis.net/ogc"
xmlns:aixm="http://www.aixm.aero/schema/5.0"
xmlns:gml="http://www.opengis.net/gml/3.2">
            <ogc:PropertyIsEqualTo>
               <ogc:PropertyName>
timeSlice/RunwayTimeSlice/type</ogc:PropertyName>
               <ogc:Literal>RWY</ogc:Literal>
            </ogc:PropertyIsEqualTo>
            <ogc:Not>
               <ogc:PropertyIsEqualTo>
                  <ogc:PropertyName>
timeSlice/RunwayTimeSlice/surfaceProperties/SurfaceCharacteristics/composition
</ogc:PropertyName>
                  <ogc:Literal>GRASS</ogc:Literal>
               </ogc:PropertyIsEqualTo>
            </ogc:Not>
            <ogc:And>
               <ogc:PropertyIsGreaterThan>

   <ogc:PropertyName>timeSlice/RunwayTimeSlice/length</ogc:PropertyName>
                  <ogc:Literal>1500</ogc:Literal>
               </ogc:PropertyIsGreaterThan>
               <ogc:PropertyIsEqualTo>

   <ogc:PropertyName>timeSlice/RunwayTimeSlice/length/@uom</ogc:PropertyName
>
                  <ogc:Literal>M</ogc:Literal>
               </ogc:PropertyIsEqualTo>
            </ogc:And>
            <fes:Not>
               <fes:Or>
                  <fes:AnyInteracts>
```

```
    <fes:ValueReference>aixm:timeSlice/aixm:RunwayTimeSlice/gml:validTime</fes:Va
lueReference>
                        <fes:Literal>
                            <gml:TimePeriod gml:id="id1">
                                <gml:beginPosition>2008-06-
10T10:30:00.000Z</gml:beginPosition>
                                <gml:endPosition>2008-06-
18T13:30:00.000Z</gml:endPosition>
                            </gml:TimePeriod>
                        </fes:Literal>
                    </fes:AnyInteracts>
                </fes:Or>
            </fes:Not>
            <ogc:DWithin>
                <ogc:PropertyName> gml:boundedBy</ogc:PropertyName>
                <gml:LineString srsName="urn:ogc:def:crs:OGC:1.3:CRS84">
                    <gml:coordinates xmlns:gml="http://www.opengis.net/gml"
decimal="." cs="," ts=" ">-97.03833333,32.89666667 -96.995,32.78333333 -
96.94,32.765 -96.83166667,32.76833333 -96.33166667,32.97 -95.56333333,33.075 -
94.07333333,33.51333333 -92.64333333,34.06 -92.55,34.095 -89.98333333,35.015 -
85.18166667,36.61333333 -84.56666667,36.80166667 -84.05,36.95666667 -
83.03,37.255 -82.13666667,37.505 -81.80666667,37.59666667 -81.12333333,37.78 -
80.39,38.04333333 -79.73166667,38.27166667 -78.99833333,38.50666667 -
77.46666667,38.935 -76.97833333,39.495 -76.29166667,40.12 -
75.68333333,40.58166667 -75.455,40.72666667 -74.86833333,40.995 -
73.82166667,41.665 -72.71666667,42.16166667 -70.61333333,43.425 -
69.49166667,43.925 -67.15666667,44.90166667 -65.87166667,45.40666667 -
64.57166667,46.18833333 -61.77333333,47.43 -58.67,48.58333333 -55,49.71666667 -
52.06833333,50.50333333 -50,52 -40,56 -30,59 -20,60 -10,60 -9.5,59.965 -
8.568333333,60.02 -1.286666667,59.87833333 0.015,59.99166667
5.211666667,60.31166667 7.5,60.295 9.915,60.23666667 11.075,60.19166667
12.51333333,60.12833333 14.17,60.03 14.40833333,60.01833333 16.73833333,59.845
16.99,59.825 17.91833333,59.65166667</gml:coordinates>
                </gml:LineString>
                <Distance units="nautical mile">100</Distance>
            </ogc:DWithin>
        </ogc:And>
    </ogc:Filter>
</wfs:Query>
</wfs:GetFeature>
```

### 7.19 Other Filter Encoding Issues and Options

This section describes the implementation details of the WFS-T for processing updates of AIXM data, the event notification to the Event Service and the retrieval of AIXM features in particular xNOTAM in the spatial – temporal context of a flight plan. It describes alternatives and issues encountered during implementation and test.

As described in section 7.10 the current implementation relies on database triggers to create the NOTAM update events. This has some problems when these triggers have to be implemented into different database systems as they might not be compatible. A more generic way to do this would be through a WFS trigger that creates the NOTAM in the context of the WFS update transaction. This would be independent from the underlying database system and would also eliminate the additional WFS request to populate the xNOTAM event.

During the interoperability experiments and tests with xNOTAM event notification it was determined that there are inconsistencies between the GML 3.2 specification and the FES2.0 filter operations semantics for timeslice values. This particularly refers to the TimePeriod beginPosition and endPosition values with undetermined content. These will be addressed in a change request to the WFS/FES2.0 Draft Implementation Specification. The CR document can be found here:
http://portal.opengeospatial.org/files/?artifact_id=32663&version=1 doc. [***OGC Doc 09-023r1].*** This will affect the gml:TimeIndeterminateValueType by adding three new enumeration values (estimated, +infinity, -infinity)..

### 7.20 Data Encoding and Metadata issues with AIXM 5.0 and GML 3.2

In aviation applications metadata are very important to identify the originator, confidence level of the validity of the data, the update dates and lifecycle metadata. Metadata in GML are defined by the ISO 19139 schema which is very verbose. Only a small subset of the 19139 metadata are needed for the communication with the aviation clients but because of the nested tags it significantly increases the size of the messages to the clients which only need a limited amount of the values. This imposes validation and bandwidth issues and complicates the WFS queries.

In order to get only the required values is a problem because the returned response has to be schema valid.

The difficulty is a result of the AIXM schema defining multiple <timeSlice> properties and the WFS not providing any way to retrieve a specific one. There is no way to exclude a <timeSlice> property from the returned feature request result. The GetGMLObject request in WFS1.1 or the proposed GetPropertyValue in WFS2.0 also don't allow subsets of the property to be returned.

One possible solution to that problem is to use xlink:href instead of the inline values to store this data somewhere else.

The client would have to specify whether this xlink reference should be resolved and if so to provide the resolution depth.

The schema snippet for such a request is shown below:

```
<xsd:element name="GetPropertyValue" type="wfs:GetPropertyValueType">
   <xsd:complexType name="GetPropertyValueType">
      <xsd:complexContent>
         <xsd:extension base="wfs:BaseRequestType">
            <xsd:sequence>
               <xsd:element ref="ogc:Filter" minOccurs="0"/>
            </xsd:sequence>
            <xsd:attribute name="typeName"
                           type="xsd:QName" use="required"/>
            <xsd:attribute name="valueReference"
                           type="xsd:string" use="required"/>
            <xsd:attribute name="resultType"
                            type="wfs:ResultTypeType" default="results"/>
            <xsd:attribute name="maxValues" type="xsd:positiveInteger"/>
            <xsd:attribute name="resolve" type="wfs:ResolveValueType"
                           default="none"/>
            <xsd:attribute name="resolveDepth"
                           type="wfs:positiveIntegerWithStar" default="*"/>
            <xsd:attribute name="resolvePath" type="xsd:string"
                           default="none"/>
            <xsd:attribute name="resolveTimeout" type="xsd:positiveInteger"/>
          </xsd:extension>
       </xsd:complexContent>
   </xsd:complexType>
```

**Listing 13: Property request using XLink Resolution**

To reduce the verbosity of the returned metadata and reduce the complexity of the queries it is proposed for consideration for future work to provide a version i.e. a timestamp of the feature itself to determine if any changes have happened to the feature at all. In this way unnecessary complex queries from the clients can be reduced or avoided. In case subsequent queries are needed to find out the specific changed property metadata it can still be executed as described above.

It is expected that queries like the ones described above will not be very performant against large datasets. It is therefore proposed for consideration in future work like OWS-7 to elevate the most relevant metadata of the AIXM5.x features into a CSW-ebRIM registry service to improve the performance of spatial-temporal queries with access to the full dataset if required as a fetch to the repository item as managed by the registry or as external reference.

### 7.21 Issues with distributed features data with AIXM 5.0 and GML 3.2

The datasets used in the project contained features that use xlink to point to additional information, such as geometries related to the feature. A concern was raised about such use. This methodology creates an implementation hurdle for client developers. The WFS client will read features and then follow the xlinks to resolve crucial information, such as geometries. Furthermore, there is no limit or knowledge on how deep the nesting is, in other words the linked information may link to more information and so on. This creates complexity and performance issues.

Moreover, the linked model may fall apart in certain implementation scenarios. For example, if the feature to be resolved is fetched through a filter on a certain property the filter will be meaningless on the linked types. Although a query based on id is possible, if

there is a large number of features that need to be resolved this becomes an unpractical approach.

### 7.22 Weather Data Sources and Formats

The weather data used in the AIM scenario is provided by several data sources and in different output formats.

#### 7.22.1 NWS WFS service with WXXM-encoded TAFs

The National Weather Services' Meteorological Development Laboratory (MDL) offers a WFS 1.1 service that delivers WXXM-encoded TAFs for the airports of interest in the AIM scenario. The TAFs can be selected for a particular flight plan by specifying an OGC filter in the WFS GetFeature request. For the OWS-6 AIM thread scenario the filter example is shown below:

```
<ogc:Filter>
      <ogc:And>
        <ogc:PropertyIsEqualTo>
          <ogc:PropertyName>gml:identifier</ogc:PropertyName>
          <ogc:Literal>KDFW</ogc:Literal>
        </ogc:PropertyIsEqualTo>
        <ogc:PropertyIsEqualTo>
          <ogc:PropertyName>gml:identifier</ogc:PropertyName>
          <ogc:Literal>KATL</ogc:Literal>
        </ogc:PropertyIsEqualTo>
        <ogc:PropertyIsEqualTo>
          <ogc:PropertyName>gml:identifier</ogc:PropertyName>
          <ogc:Literal>KILN</ogc:Literal>
        </ogc:PropertyIsEqualTo>
        <ogc:PropertyIsEqualTo>
          <ogc:PropertyName>gml:identifier</ogc:PropertyName>
          <ogc:Literal>ESSA</ogc:Literal>
        </ogc:PropertyIsEqualTo>
        <ogc:PropertyIsEqualTo>
          <ogc:PropertyName>gml:identifier</ogc:PropertyName>
          <ogc:Literal>ESSP</ogc:Literal>
        </ogc:PropertyIsEqualTo>
      </ogc:And>
      <fes:Not xmlns:fes="http://www.opengis.net/fes/2.0"
xmlns:wx="http://www.eurocontrol.int/wx/1.1"
xmlns:avwx="http://www.eurocontrol.int/wxxs/1.1"
xmlns:gml="http://www.opengis.net/gml/3.2">
        <fes:Or>
          <fes:Before>

<fes:ValueReference>avwx:aerodromeWxForecast/wx:Forecast/wx:forecastTime/gml:TimePeriod</
fes:ValueReference>
            <fes:Literal>
              <gml:TimePeriod gml:id="id1">
                <gml:beginPosition>2009-02-12T00:00:00.000</gml:beginPosition>
                <gml:endPosition>2009-02-13T14:00:00.000</gml:endPosition>
              </gml:TimePeriod>
            </fes:Literal>
          </fes:Before>
          <fes:After>

<fes:ValueReference>avwx:aerodromeWxForecast/wx:Forecast/wx:forecastTime/gml:TimePeriod</
fes:ValueReference><fes:Literal>
         <gml:TimePeriod gml:id="id2"><gml:beginPosition>2009-02-
12T00:00:00.000</gml:beginPosition>
                <gml:endPosition>2009-02-13T14:00:00.000</gml:endPosition>
              </gml:TimePeriod></fes:Literal></fes:After>
          <fes:Meets>
<fes:ValueReference>avwx:aerodromeWxForecast/wx:Forecast/wx:forecastTime/gml:TimePeriod</
fes:ValueReference>
            <fes:Literal>
```

```
              <gml:TimePeriod gml:id="id3">
                <gml:beginPosition>2009-02-12T00:00:00.000</gml:beginPosition>
                <gml:endPosition>2009-02-13T14:00:00.000</gml:endPosition>
              </gml:TimePeriod>
            </fes:Literal>
          </fes:Meets>
          <fes:MetBy>

<fes:ValueReference>avwx:aerodromeWxForecast/wx:Forecast/wx:forecastTime/gml:TimePeriod</
fes:ValueReference>
            <fes:Literal>
              <gml:TimePeriod gml:id="id4">
                <gml:beginPosition>2009-02-12T00:00:00.000</gml:beginPosition>
                <gml:endPosition>2009-02-13T14:00:00.000</gml:endPosition>
              </gml:TimePeriod>
            </fes:Literal>
          </fes:MetBy>
        </fes:Or>
      </fes:Not>
    </ogc:Filter>
```

**Listing 14: Query to retrieve all TAFs relevant for the flight scenario**

### 7.22.2 NWS National Digital Forecast Database WFS service

The National Weather Service provides an experimental WFS service for the National Digital Forecast Database. Currently, it gives access to forecast data available as points that are encoded in GML 2 or 3.1.1

### 7.22.3 NNEW WFS service with SIGMET/TAF/METAR

A weather data WFS service from NNEW, containing AIRMET/SIGMETs, TAFs and METARs was provided for the AIM scenario. The data is available as polygons (SIGMET) and points (TAF, METAR), encoded in GML 2 or 3.1.1.

### 7.22.4 Conclusion

The retrieval of weather information as GML features via a standard WFS interface rather than text based messages opens up many opportunities for aviation clients. It is possible to combine aeronautical and weather information using a similar WFS feature request structure and OGC filters to retrieve only information about a particular flight and weather conditions as related to a particular flight plan. The ability to validate the WXXM messages against the application schema ensures the integrity of the data when received by the client. This is not possible using text based messages for TAFs and METARs. Using the same interface (WFS) to retrieve weather and aeronautical information i.e. WXXM and AIXM lowers the implementation barrier for EFB clients.

In the current OWS6 AIM thread three such aviation clients are under development and are demonstrating the visualization of weather and aviation information via the standard WFS interfaces of the AIXM and WXXM Servers.

In future versions WFS services providing weather data should also register with the Event Service in future implementations so that clients can subscribe to changes relevant to the flight plan to these data sources to receive notifications while en-route.

# 8 OWS-6 AIM Accomplishments / Lessons learned

- Demonstrated that a standards-based web-services architecture revolving around AIXM 5.0 can meet the Aviation requirements for timely on-demand data access, integration, notification and visualization

    - Except for minor changes/glitches, the underlying OGC and ISO standards (GML, WFS, FE) fully supported the Aviation requirements

    - By enabling access to subsets of information (rather than full downloads of data), the web-services architecture opens up issues related to data integrity and consistency at the client

    - AIS data bases using AIXM5 need data validation and QA at load time or update time to avoid data inconsistencies e.g. missing identifiers, tangling references etc.

    - The need for frequent client-server communications in this architecture (for updates and events) highlights issues related to reliability, security and bandwidth of underlying messaging channels

- Demonstrated feasibility of using WFS and FE to provide on-demand access to AIXM 5.0 baseline information as well as temporary and permanent deltas

    - Demonstrated practicality of starting with existing Shapefile source data and serving it as AIXM 5.0 via WFS

    - Successfully implemented accurate and timely retrieval of relevant subsets of AIXM 5.0 data based on spatio-temporal FE filters

        - Identified changes to GML and FE to accommodate for the uncertainty in the end time periods of aviation data changes

            - This has been addressed in a GML change request

        - Identified need to query data based on time of upload/update of data/changes

            - Achieved by adding appropriate metadata to features in the database (and the events) using ISO GML metadata

            - Increased the size of the data significantly making it harder for Aviation clients to efficiently parse and interpret the data

        - Uncovered limitations in AIXM 5.0 data model in supporting certain expected queries (such as returning airports with 3 or more runways)

        - Uncovered issues with time indeterminate position “Unknown”

- Demonstrated integration of weather data affecting route of flight within given interval of time
    - Demonstrated successful access and retrieval of relevant WXXM and other GML-based weather data via WFS
        - Using the same standard interface (WFS) and exchange format (GML) for both weather and aeronautical information lowers the learning curve and implementation barrier for Aviation clients
    - Uncovered issues related to textual nature of aeronautical weather data and consistent portrayal/symbology options
- Demonstrated ability of quick prototyping and implementation of Aviation client functionality based on standard web service interfaces and exchange formats for filtering, accessing, integrating and visualizing AIXM and weather data
    - Three different Aviation clients are under development
    - Client prototyping experiences highlight overhead associated with web-services architecture and AIXM 5.0 including
        - Inherent complexity of GML schemas and impacts on parsing and processing efficiencies given the hierarchical sub-classing from GML abstract data types and substitution groups
        - Difficulty in mapping of AIXM data model to existing (binary) data models in operational software (e.g. PCAvionics' MountainScope)
        - Critical responsibility of data integration at the client (baseline, deltas, events, etc) to continuously maintain an accurate and up-to-date representation of the data
        - Critical responsibility of resolving and maintaining feature and property inter-dependencies in AIXM 5.0
        - Reliable messaging and connection issues in support of push-based event alert mechanism
        - Issues of pushing notifications to clients that have dynamic IP addresses
        - Issues of pushing notifications to clients behind a firewall
- Demonstrated standards-based architecture for event alert notifications
    - Demonstrated the feasibility of subscription based data updates
    - Demonstrated feasibility of incorporating the OASIS WSN mechanism in the architecture

- Demonstrated feasibility and reliability of mechanism for matching subscription requests (based on time and space) to various events
- Experience to-date uncovered issues related to latency of events and reliability of messaging
- WSN SOAP-based approach creates a considerable overhead for Aviation clients (open connection issues, firewall issues, etc)

# 9 Next Steps- Issues that still need to be addressed/explored

- Further improving/adapting underlying standards, e.g.
  - Overcoming/addressing GML ISO metadata complexity and size issues
  - Optimizing WFS FE spatiotemporal filters
  - Simplifying/decoupling of AIXM schemas
  - Manage AIXM metadata and lifecycle via a registry service
- Understanding/improving metrics for system, e.g.
  - Performance of spatiotemporal queries
  - Trigger changes in the WFS-T rather than database
  - Simplify subscription matching
  - Support multiple data sources at the Event Service
  - Latency of events and updates
  - Data integrity strategies
- Investing in further client development, e.g.
  - Investing in reusable components (e.g. open source for parsing AIXM, WXXM, etc)
  - Exploring efficient ways for visualizing the data (e.g. 3D, Goggles, etc)
  - Capturing steps for easy transition/evolution of current EFB software providers
- Improving the Event Alert architecture, e.g.
  - Supporting weather events
  - Addressing intermittent access issues

  - Exploring other Event standards such as WS-Eventing (W3C Draft spec)
  - Explore usage of CAP (Common Alerting Protocol)
  - Investigate using a REST architecture style for xNOTAM delivery to remove the SOAP overhead from the clients
  - Implement Transport and Message Level Reliability and Security (TLS, MLS)
  - Persist Subscriptions and Registrations (Subscription and Registration Lifecycle management)
  - Explore Pluggable ES Architecture to use different (Domain Specific) filter encodings
  - Resolve Alerts in the Event Service via a WFS request with the subscription filter
  - firewall issues, …
  - Define an OGC standard for eventing / event services (even SAS is not an OGC standard)

- Building on the OWS-6 AIM architecture, e.g.
  - Updating of feature base via WFS-T for events
  - Implement WFS2.0 with full FES2.0 support
  - Addressing intermittent access issues
  - Incorporating elements of existing infrastructure (e.g. SWIM)

# Annex A

## Scenario Steps

**A** **OWS Flight 123** is scheduled to depart at 02:00 UTC on 12 FEB 2009 from Airport **DFW**[1] for Airport **ARN/ESSA**[2].

*A.1* *Pilot has flight plan **DFW** to **ARN/ESSA***

*A.2* *Assume baseline data is loaded into EFB for both airports along with alternate airports **CFE**[3], **NRK/ESSP**[4] and **ILN**[5])*

A.3 *AIM client shows graphical display of flight route as well as origin and destination airports*

**B** Due to some delays, the pilot enters the aircraft at 04:30 UTC on 12 FEB 2009 (three hours after his briefing) and requests an automatic feed to his EFB of any updates to aeronautical information for **ARN/ESSA** for **NRK/ESSP** (alternate airport in Sweden to be used in the event that a landing at **ARN/ESSA** cannot be made), and **ILN** (alternate airport close to the filed route of flight to be used in an emergency occurring during the initial segment of the flight)

*B.1* *AIM client makes request to WFS to retrieve any changes that might have occurred during the last three hours in the data for **DFW**, **ARN/ESSA**, **ILN**, and **NRK/ESSP;** Or AIM client makes request to WFS to retrieve new snapshot of those airports (assuming the snapshot contains information about temporal validity of features/attributes)*

*B.2* *AIM client updates graphical display of all four airports*

*B.3* *AIM client subscribes to alerts for all changes for the next three hours regarding **ARN/ESSA**, **ILN**, and **NRK/ESSP**.*

**C** Shortly after requesting the automatic feed, updated aeronautical information for **ILN** is displayed on the EFB indicating that **ILN** is closed (starting 04:45 UTC on 12 FEB 2009) due to bad weather.

---

[1] Dallas Fort Worth International Airport (US)

[2] Stockholm – Arlanda Airport (Sweden)

[3] Clermont-Ferrand Auvergne Airport (France); more complete dataset which supports nice 3D displays showing building; role in scenario still TBD.

[4] Norrkoping Airport (Sweden)

[5] Airborne Airpark (Wilmington, OH, US)

*C.1 An xNOTAM is received by the Database/WFS maintainer indicating that all **ILN** runways are closed until further notice. The changes are uploaded to the Database/WFS.*

*C.2 The Database maintainer wraps the xNOTAM with the AIXM Event Wrapper (also adding the geographic extent of the features affected in case that information is needed by the Event Service) and forwards it to the Event Service.*

*C.3 The Event Service forwards the messages AS IS to all AIM clients subscribed for changes to **ILN** if the events fall during the time period specified when they subscribed to the Event Service*

*C.4 The AIM client processes the xNOTAM and, if needed, makes a request to the WFS for retrieving the updates.*

C.5 *The update information is superimposed on the display of the route as a highlight at the affected location, which is expanded to show the update in either text or graphic format.*

**D** The pilot enters a request on the EFB for adverse weather conditions that will affect the planned route and the altitude of the flight in the next twelve hours (from 05:00 UTC to 17:00 UTC on 12 FEB 2009).

*D.1 AIM client makes a request to weather WFS to retrieve adverse weather conditions along the flight route and based on altitude of flight. Adverse weather conditions include icing, wind, convective weather, turbulence, lightning, etc. The specific ones to be used in the demonstration depend on the data served via the weather services contributed by NOAA and NNEW.*

*D.2 AIM client makes a request to weather WFS for forecast weather events (e.g. METAR, TAF)*

D.3 *The AIM client displays the weather information along with the forecast weather events. The display shows a strong weather system that will affect the flight route within the times specified.*

**E** The pilot is given an amended route to **ARN/ESSA** that will avoid the weather system.

*E.1 AIM client displays amended route of flight*

**F** The pilot enters a request on the EFB or hand-held electronic display device to retrieve and display information on airports within 100 nautical miles of the amended route of flight at which the aircraft will be able to land in the event of an emergency occurring while the aircraft is within 500 nautical miles from **DFW**.

*F.1 AIM client makes a request to WFS to return list of airports that*

- *Fall within 100 nautical miles of the amended route, and*

- *Are suitable for an emergency landing (hardsurface runway > 5000 ft; has precision approach; has jetA fueling capabilities)*[6]
- *Have at least three runways*
- *Are not military airports*

*F.2* ***ATL***[7] *meets the criteria and is returned.*

**G** The pilot requests more detailed information, including approaches, on **ATL**.

*G.1 If AIM client already had **ATL** baseline data loaded, then the AIM client issues a request to the WFS for the changes (including approaches) that have occurred since take-off. Otherwise, the AIM client asks for a snapshot (including approaches) for **ATL**.*

*G.2 The information is received and displayed on the AIM client, superimposed on a visualization of the specified part of the amended route of flight.*

**H** The pilot estimates the time of the closest approach to **ATL** and requests forecast weather information for **ATL** for a possible landing at that time

*H.1* AIM client *makes request for WFS weather service for forecast weather information (e.g. METAR, TAF, etc) for the **ATL** region*

*H.2 The AIM client displays the detailed aeronautical information for **ATL** superimposed on forecast weather event. The display shows no significant weather forecast for **ATL**.*

**I** The pilot requests an automatic feed to the EFB or hand-held electronic display device of any updates to aeronautical information for **ATL** and cancels the automatic feed for **ILN**.

*I.1AIM client subscribes to alerts related to **ATL** from current time until the estimated landing time.*

*I.2AIM client un-subscribes to alerts related to **ILN** (Shortly after this, **ILN** is re-opened and no alert is received by the AIM client)*

**J** The ground controller at **DFW** clears OWS Flight 123 to taxi for departure at 05:20 UTC on 12 FEB 2009. As the aircraft taxis, the EFB or hand-held electronic display device is disconnected from the Internet (to alleviate this disconnect, a WFS could possibly be stood up to provide the Taxi path to the pilot based on the D-Taxi Graph

---

[6] In the future, the selection of airports that are appropriate for emergency landing would be based on matching the airports characteristics (fueling, runways, precision approaches, etc) with both an aircraft profile (type of aircraft, etc) and a pilot profile (experienced vs. inexperienced, etc)

[7] Hartsfield-Jackson Atlanta International Airport (US)

standard which is currently under development by EUROCAE WG78 RCTA SC214 – Standards for air traffic data communication services).

*J.1 AIM client possibly can connect to the D-Taxi Graph WFS to access and display the Taxi path*

**K** After take-off, the pilot continues to receive updates to aeronautical information for **ATL**, **ARN/ESSA**, and **NRK/ESSP**. During the flight, whenever update information is available for these airports, an audible alert and a visual indication is given on the display of the EFB or hand-held electronic display device. The pilot also requests adverse weather conditions for the three airports.

*K.1 AIM client continues receiving and processing xNOTAMs related to the **ATL**, **ARN/ESSA**, and **NRK/ESSP**.*

K.2 *AIM client requests new adverse weather conditions for* **ATL**, **ARN/ESSA**, and **NRK/ESSP** *from WFS weather source*.

**L** After entering Stockholm FIR airspace at 13:10 UTC on 12 FEB 2009, an alert is received for **ARN/ESSA** indicating that all runways are closed (fires in close proximity to the airport are producing extensive smoke over the airport). Other possible events to consider include

- Temporary obstacles (since 05 JAN 2009)
- Restricted airspace around airport (one active 07:00-17:00 UTC, close but outside the flight trajectory; another ad-hoc one active 13:00 UTC till unknown because of the smoke/fire issues)
- Closed routes (the ones crossing the airspace that is active between 07:00-17:00)

*L.1 This information is super-imposed on the display of the route as a highlight at the affected location. The highlight is expandable to show the update as either text or graphic format.*

**M** The pilot notifies ATC of plans to divert to **NRK/ESSP** (during this time, no alerts on **NRK/ESSP** have been received). The pilot requests any weather advisories or METARs in effect for **NRK/ESSP**.

*M.1 The response to this request is displayed on the AIM client as the flight continues and indicates that there is no significant weather at **NRK/ESSP**.*

*M.2 A new event has occurred in the meantime at NRK/ESSP – a taxiway has been closed since 13:00 UTC.*

**OWS Flight 123** lands without incident at **NRK/ESSP**.

# Annex B

## XML Schema Documents

In addition to this document, this report includes several XML Schema Documents. These XML Schema Documents are bundled in a zip file with the present document.

The TBD abilities now specified in this document use TBD specified XML Schema Documents included in the zip file with this document. These XML Schema Documents combine the XML schema fragments listed in various subclauses of this document, eliminating duplications. These XML Schema Documents roughly match the TBD UML packages described in Annex B, and are named:

TBD.xsd

TBD.xsd

These XML Schema Documents use and build on the OWS common XML Schema Documents specified [OGC 06-121r3], named:

ows19115subset.xsd

owsCommon.xsd

owsDataIdentification.xsd

owsExceptionReport.xsd

owsGetCapabilities.xsd

owsOperationsMetadata.xsd

owsServiceIdentification.xsd

owsServiceProvider.xsd

All these XML Schema Documents contain documentation of the meaning of each element and attribute, and this documentation shall be considered normative as specified in Subclause 11.6.3 of [OGC 06-121r3].

# Annex C

# UML model

## C.1 Introduction

This annex provides a UML model of the AIXM temporal model.

Figure C.1 is a simple UML diagram summarizing the Timeslice Concept. In an UML diagram, the basic Time Slice concept is represented as below:

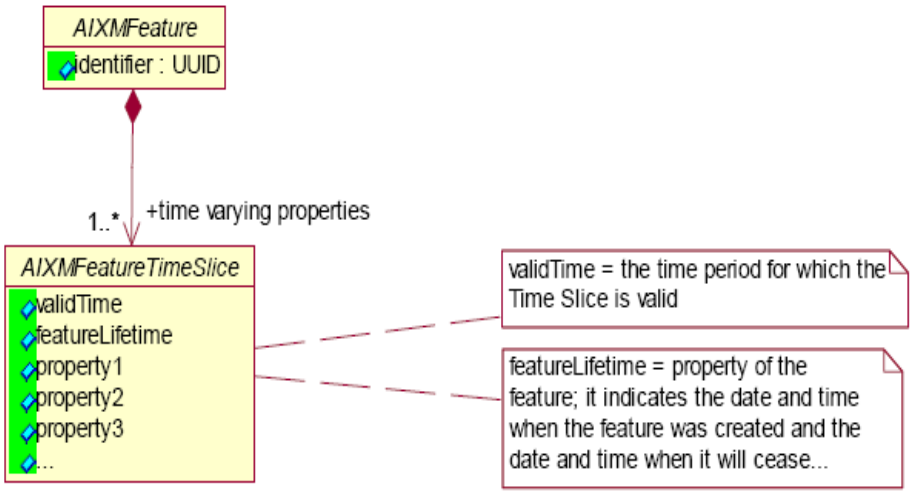



*Figure 3*

**Figure C.1 — Timeslice Concept UML diagram**

When applying the Time Slice concept, as described in the previous UML diagram, this triggers the split of every UML class that represents a feature into a main class and a "FeatureTimeSlice" class, as shown in the following diagram.

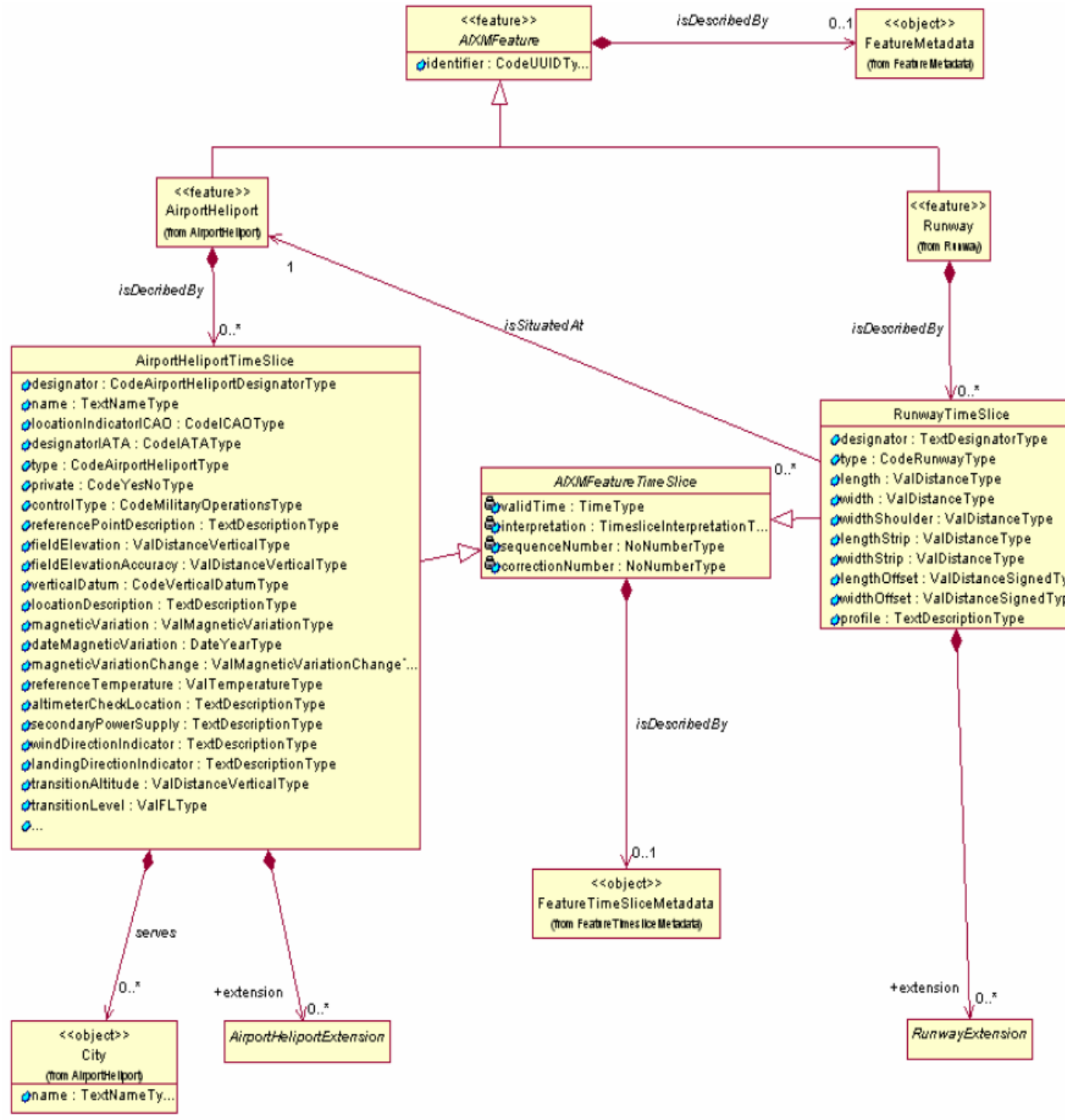



**Figure C.2 — Abstract Feature Class with Timeslice UML diagram**

The UML diagram shows how each and every <<feature>> inherits from the abstract AIXMFeature class. The concrete features are described by TimeSlices which are composed of properties. The TimeSlice inherits from the abstract

AIXMFeatureTimeSlice class. The diagram also shows that each AIXM Feature is described by FeatureMetadata and each TimeSlice is described by FeatureTimeSliceMetadate. Finally, each TimeSlice may contain an Extension. The

Extension mechanism allows each user of AIXM 5 to define and use his own specific attributes and classes, in addition to the core AIXM ones.

# Bibliography

[1] Guidelines for Successful OGC Interface Standards, OGC document 00-014r1