Open Geospatial Consortium

Publication Date: 2015-08-19

Approval Date: 2015-06-05

Submission Date: 2015-05-06

External identifier of this OGC® document: http://www.opengis.net/doc/PER/t11-aviation-arch

Reference number of this document: OGC 15-025r2

Category: Public Engineering Report

Editor: Johannes Echterhoff

OGC® Testbed 11 Aviation - Architecture 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: OGC® Engineering Report
Document subtype: NA
Document stage: Approved for public release
Document language: English

i. Abstract

This OGC® document describes the architecture implemented in the OGC Testbed 11 Aviation thread.

ii. Keywords

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

ogcdoc, OGC document, aviation, architecture, testbed 11

Preface

This document is a deliverable of the OGC Web Services Testbed 11. This Engineering Report describes the architecture that was implemented in the Aviation thread. The document:

- Describes a high-level overview of the architecture, its components and their interactions.
- Contains a summary description of the various components within the architecture.
- Provides summaries for the various subject areas that the Aviation thread was concerned with and which are documented in detail in other Testbed 11 engineering reports.
- Documents lessons learned.
- Describes the scenario as well as use cases that have been designed for the demonstration of the Aviation thread developments.

Contents

Page

Figures

Page

OGC® Testbed 11 Aviation - Architecture Engineering Report

1 Introduction

1.1 Scope

This OGC® document describes the architecture implemented in the OGC Testbed 11 Aviation thread. The document:

- Describes a high-level overview of the architecture, its components and their interactions.
- Contains a summary description of the various components within the architecture.
- Provides summaries for the various subject areas that the Aviation thread was concerned with and which are documented in detail in other Testbed 11 engineering reports.
- Documents lessons learned.
- Describes the scenario as well as use cases that have been designed for the demonstration of the Aviation thread developments.

1.2 Document contributor contact points

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

Name	Organization
Aleksandar Balaban	m-click
Alexis James Brooker	Snowflake Software
Charles Chen	Skymantics, LLC
Daniel Balog	Luciad
Dean Hintz	Safe Software
Johannes Echterhoff (editor)	interactive instruments GmbH
Robin Houtmeyers	Luciad
Thibault Dacla	ATMOSPHERE
Thomas Forbes	Snowflake Software
Volker Grabsch	m-click

1.3 Future work

The topic specific engineering reports identify a range of items for consideration in future initiatives. For further details, see (OGC documents 15-024, 15-026, 15-027, and 15-028).

1.4 Foreword

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

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

2 References

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

OGC Testbed 11 Engineering Reports:

- [OGC 15-024] OGC Testbed 11 Aviation – Guidance on Using SBVR for Geometrical Constraints Engineering Report
- [OGC 15-026] OGC Testbed 11 Aviation Feature Schema Recommendations Engineering Report
- [OGC 15-027] OGC Testbed 11: Digital NOTAM Validation and Enrichment Service Engineering Report
- [OGC 15-028] OGC Testbed 11 Aviation – Data Broker Specifications Engineering Report

3 Terms and definitions

No specific terms and definitions apply for the purposes of this report.

4 Abbreviated terms

AFX	Aviation Feature Schema
AIXM	Aeronautical Information Exchange Model
AMXS	Aerodrome Mapping Exchange Schema
API	Application Programming Interface
BBOX	Bounding Box
CCI	Cross-Community Interoperability
COTS	Commercial Off the Shelf
CRUD	create, read, update, delete
DNOTAM	Digital NOTAM
EFB	Electronic Flight Bag
FIXM	Flight Information Exchange Model
FME	Feature Manipulation Engine
FNS	Federal NOTAM Service
FPS	Feature Portrayal Service
GEOINT	Geospatial intelligence
GML	Geography Markup Language
GSIP	GEOINT Structure Implementation profile
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
ICAO	International Civil Aviation Organization
ICU	Intensive Care Unit
ISO	International Organization for Standardization
JDK	Java Development Kit
NOTAM	Notice to Airmen
OCL	Object Constraint Language
OGC	Open Geospatial Consortium
RDF	Resource Description Framework
SBVR	Semantics of Business Vocabulary and Business Rules
SE	Symbology Encoding
SKOS	Simple Knowledge Organization System
SLD	Styled Layer Descriptor
SOS	Sensor Observation Service

SQL	Structured Query Language
SWIM	System Wide Information Management
UUID	Universally unique identifier
WCS	Web Coverage Service
WFS	Web Feature Service
WFS-T	WFS-Transactional
WMTS	Web Map Tile Service
WPS	Web Processing Service
WXXM	Weather Information Exchange Model
XMI	XML Metadata Interchange
XML	Extensible Markup Language
XSLT	Extensible Stylesheet Language Transformations

5 OGC Testbed 11 Aviation Architecture - Overview

The tasks for the Testbed 11 Aviation thread were to:

- Develop guidance on using geometrical constraints in Semantics of Business Vocabulary and Business Rules (SBVR)
- Advance a Digital Notice to Airmen (NOTAM) validation service
- Advance a Digital NOTAM enrichment service
- Advance use of Aviation Feature Schema (AFX)
- Advance the concept of data brokering within the Aviation Architecture

These tasks have been accomplished in Testbed 11. Chapters 7, 8, and 9 provide summaries with references to further details. The components that enable the functionality required by each task are described in chapter 6. A high-level overview of the Aviation thread architecture and its components is given in the following.

The Aviation thread architecture can be separated into three tiers (see Figure 1):

- The *Client Tier* contains the client applications.
- The *Business Process Tier* contains components that offer services on top of the Access Tier: validation, portrayal, model rules derivation, and data brokering.
- The *Access Tier* contains Web Feature Services serving aeronautical data compliant to AIXM 5.1, Digital NOTAM 2.0, and the Aviation Feature Schema (AFX). Feature enrichment functionality is also provided.

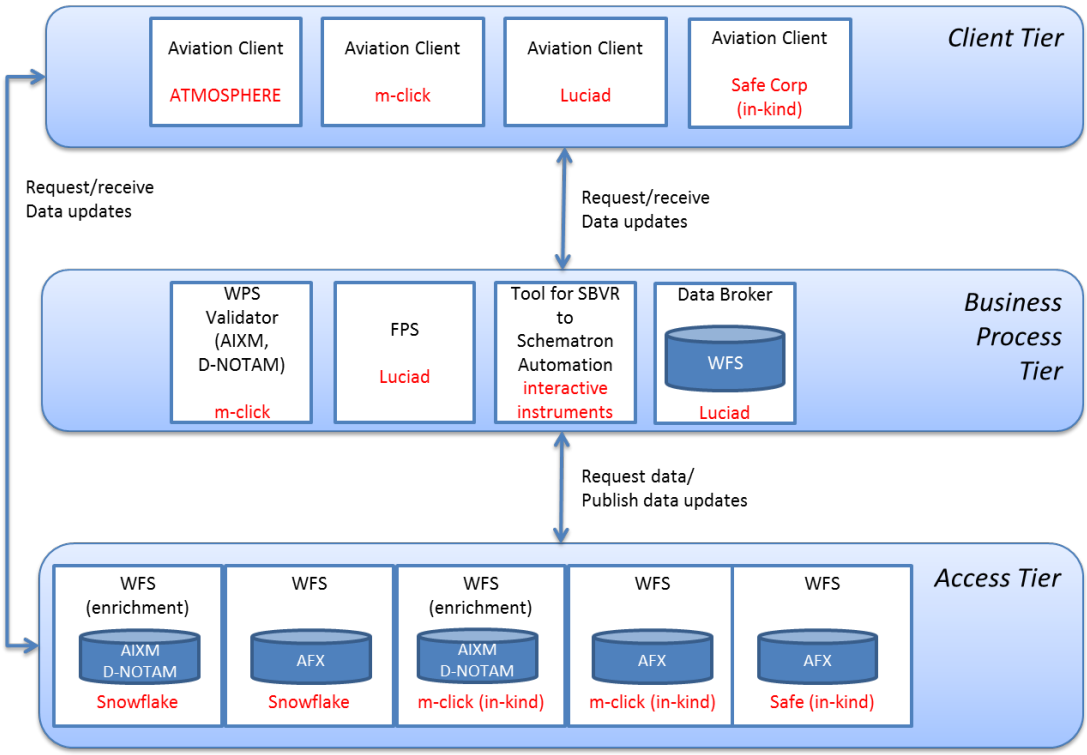


Figure 1 – Testbed 11 Aviation Architecture – High-Level Overview

Figure 1 shows the links between the tiers and the general functionality that is invoked. The Figure also shows which participants provided which components. A summary description of the components is provided in chapter 6.

A more detailed view of the interaction between the different components is shown in Figure 2. The “SPARQL for Symbology” as well as the “SLD/SE Producer” are components defined within the Cross-Community Interoperability (CCI) thread architecture.

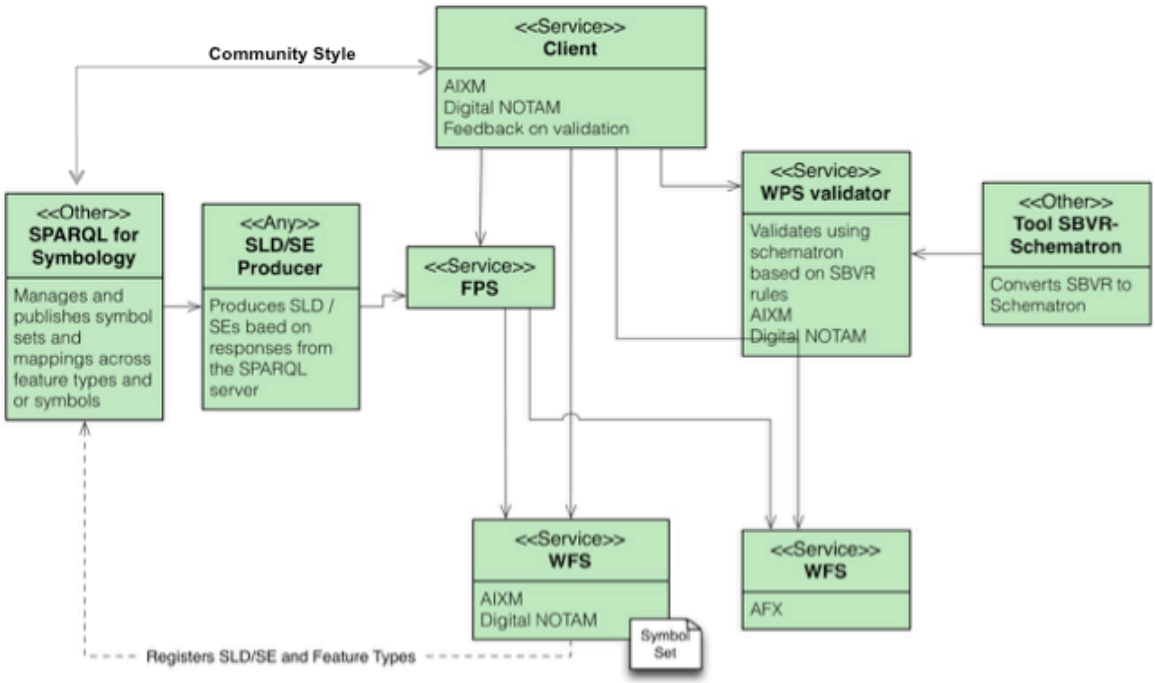


Figure 2 – Components and their interactions

6 Component Descriptions

6.1 Data Broker

6.1.1 Luciad

6.1.1.1 Introduction

The Data Broker is a web service component supporting OGC WFS and capable of redistributing data from other WFS servers.

To support this, Luciad provided a WFS Data Broker service component using its COTS software product LuciadLightspeed. LuciadLightspeed offers a rich set of standards-based software components, including an OGC Web Services Suite equipped with an OGC-compliant WFS service component. One of this component's benefits for the Data Broker task is its open data back-end API, allowing users to easily connect to any type of storage component – such as other OGC web services.

6.1.1.2 Functional overview

The Data Broker did not need any new functionality other than the capabilities already defined by OGC WFS / Filter.

The Data Broker provided by Luciad has the following functionality:

- OGC-compliant WFS 1.1.0 & 2.0.0 service interface with support for the following requests: GetCapabilities, DescribeFeatureType and GetFeature. Supported request encodings are HTTP GET and POST.
- Support for connecting with OGC WFS server components with support for version 1.1.0 or 2.0.0 and with support for AIXM 5.1 output.
- Support for OGC Filter 1.1.0 and 2.0.0.
- Support for various automatic & on-the-fly feature type operations useful for the Data Broker:
    - Combination of similar feature types from different OGC WFS data sources into one feature type.
    - Conflation of features between different OGC WFS data sources.
    - Enriching of features by adding ISO 19115-based lineage information.

6.1.1.3 Deployment characteristics

The Luciad Data Broker is based on Java Servlet technology. To run, it requires a Java servlet container or application server compatible with Java Servlet 2.5 or higher. Apache Tomcat 7 has been used by Luciad during Testbed 11. Other than being capable of running a Java Virtual Machine 1.7 (or higher) and an appropriate servlet container / application server, no requirements are posed on the underlying hardware or operating system.

6.1.1.4 Challenges

The Data Broker is a new concept introduced in Testbed 11 to support the setup of cascading WFS data sources, which brings a number of challenges - e.g., conflation of duplicate features, on-the-fly feature enrichment, etc. One key challenge is caching, i.e. the research for approaches to enable the Data Broker to efficiently cache data from its WFS data sources. Specific difficulties here are (1) the ability for the Data Broker to decide upon an optimal caching approach (2) the ability for the Data Broker to know when a cached feature has been updated, taking into account the possibilities within OGC WFS. A number of approaches have been identified and tested in practice; more information can be found in the Data Broker Specifications Engineering Report (OGC 15-028).

6.1.1.5 Accomplishments

The key accomplishments for Luciad's Data Broker service component in Testbed 11 include:

- Fast setup and deployment of Luciad's COTS-based Data Broker server, ready-to-use by other participants within a month after the start of the project.
- Successful demonstration of an OGC WFS-based cascading web services setup using the available WFS data sources in the project (see section 6.4).
- Implementation of a data processing pipeline capable of handling relevant data broker responsibilities:
    - feature enrichment: integration of ISO 19115 lineage information per feature to indicate the data source & broker processing step.
    - feature conflation: detection & resolution of duplicate features & identifiers.
- Implementation of caching to reduce the amount of queries needed to be sent by the broker to its WFS data sources.

6.2 Feature Portrayal Service

6.2.1 Luciad

6.2.1.1 Introduction

A Feature Portrayal Service (FPS) is an OGC service that enables the user to render maps based on feature data and styling information. This user input is defined in an OGC Styled Layer Descriptor (SLD) document, which gives access to the feature data, either embedded or as a link to an OGC Web Feature Service, and the styling information, encoded with OGC Symbology Encoding (SE).

Within Testbed 11 Aviation, an FPS is used to support the rendering of aeronautical data using the symbology of a specific community. To do so, it interfaces with an OGC WPS-based SLD / SE Producer component to get the right symbology. The necessary information to allow the FPS to request the symbology needs to be part of the original request sent by client to the FPS; the client itself has gathered this information from the

GeoSPARQL server. The response produced by the FPS is a bitmap image. One of the main requirements addressed by this architecture is to have the right symbology for each geographic territory – for instance, different symbols might be used in the U.S. and in Europe.

To support this, Luciad provided an FPS service component using its COTS software product LuciadLightspeed. LuciadLightspeed offers a rich set of standards-based software components, including an OGC Web Services Suite equipped with an OGC-compliant WMS 1.1.1 & 1.3.0 service component with support for the SLD / SE extension.

6.2.1.2 Functional overview

The FPS did not need any new functionality other than the capabilities already defined by OGC's WMS, SLD and SE standards. The custom business logic, i.e. the generation of the SE style information and the construction of an SLD for the FPS is respectively offloaded to the OGC WPS-based SE Producer and the GeoSPARQL service.

The FPS provided by Luciad has the following functionality:

- OGC-compliant WMS 1.1.1 & 1.3.0 service interface with support for the following requests: GetCapabilities, GetMap and GetFeatureInfo. Supported request encodings are HTTP GET and POST.
- Support for the SLD 1.0 / 1.1 profile (FPS), including support for user-defined styles and user-defined layers. User-defined layers can either embed the feature data or link to an OGC WFS.
- Support for SE 1.1 to define styling rules.
- Support to render any type of GML-based feature data (AIXM, WXXM …).
- Support to render AFX.

6.2.1.3 Deployment characteristics

The Luciad FPS is based on Java Servlet technology. To run, it requires a Java servlet container or application server compatible with Java Servlet 2.5 or higher. Apache Tomcat 7 has been used by Luciad during Testbed 11. Other than being capable of running a Java Virtual Machine 1.7 (or higher) and an appropriate servlet container / application server, no requirements are posed on the underlying hardware or operating system.

6.2.1.4 Challenges

One particular encountered challenge in Testbed 11 Aviation was the design & development of an architecture and workflow to support the common symbology requirements. Given a client, a GeoSPARQL server, an SLD/SE producer and an FPS, the goal was to find an optimal workflow to be able to automatically find the right symbology for a given feature data set – by reusing existing capabilities from standards and services as much as possible. This challenge was tackled in cooperation with the CCI

thread (which had a similar symbology requirement) and the relevant component producers.

6.2.1.5 Accomplishments

The key accomplishments for Luciad's FPS service component in Testbed 11 include:

- Fast setup and deployment of Luciad's COTS-based FPS server, ready-to-use by other participants right after the start of the project.
- Successful integration of the new AFX aeronautical format.
- Delivery of an FPS server capable of supporting the architecture developed within Testbed 11 to address the common symbology requirements.

6.3 SBVR to Schematron Automation Tool

6.3.1 interactive instruments

6.3.1.1 Overview

The SBVR-to-Schematron automation tool has been implemented as a ShapeChange extension. ShapeChange is an Open Source tool that takes application schemas constructed according to ISO 19109 from a UML model, transforms them as needed, and derives implementation representations.

The most commonly used target representation is XML Schema, with support for the following standard encoding rules:

- GML 3.2 encoding rule for GML application schemas
- GML 3.3 extensions
- ISO/TS 19139 encoding rule
- INSPIRE encoding rule

In addition to the generation of XML Schema documents, the generation of Schematron documents from Object Constraint Language (OCL) constraints in the UML model is supported.

Other target representations include feature catalogues in DOCX and HTML, RDF schemas, code list dictionaries in GML and SKOS, as well as SQL-DDL and ArcGIS workspace models[1].

ShapeChange directly accesses Enterprise Architect (EA) models via their Java API. ShapeChange can read XMI 1.0 and directly access GEOINT Structure Implementation Profile (GSIP) model databases.

[1] The SQL-DDL and ArcGIS workspace targets are both in beta stage. All targets and the associated encoding rules are documented on http://shapechange.net.

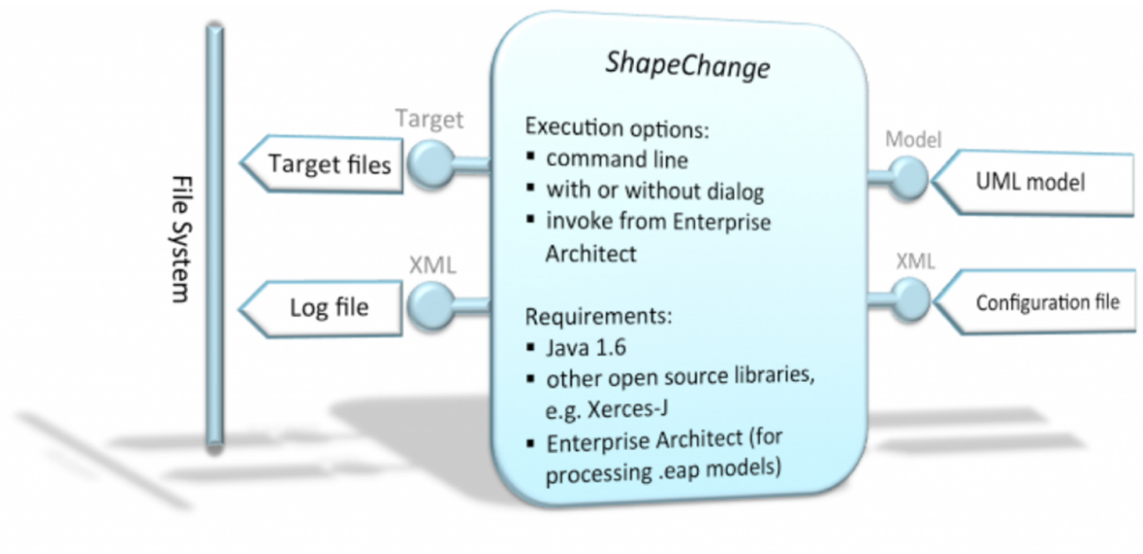


Figure 3 – Overview of ShapeChange, the basis for the SBVR to Schematron automation tool

In Testbed 11, new modules were added to ShapeChange. These modules support loading of model constraints expressed in SBVR, parsing SBVR constraints, and deriving Schematron code from them. Detailed documentation of this topic can be found in OGC document 15-024.

6.3.1.2 Deployment Characteristics

The basis for the SBVR-to-Schematron extension was ShapeChange version 2.0.0. ShapeChange is a java application that can be executed via the command line, with or without a graphical user interface.

6.3.1.3 Challenges

Defining a grammar that represents the (majority of the) AIXM business rules, and writing an according parser – all within less than four months – was a challenge in itself.

A major challenge, however, was to solve the issue that AIXM is not an ISO 19109 application schema and is not compliant with the GML application schema encoding rules. Unions, for example, are handled differently in AIXM.

Another challenge was that the AIXM conceptual schema does not fully represent the properties of AIXM features. Time slices are completely ignored, even though conceptually they belong to AIXM features and are highly important for implementing the AIXM Temporality Model.

Furthermore, the way AIXM extensions work is different to how conceptual schema are usually extended according to ISO 19109 or UML in general.

Eventually, all these issues could be solved. For further details, see (OGC document 15-024).

6.3.1.4 Accomplishments

Within Testbed 11:

- ShapeChange has been extended to read, load, and parse SBVR constraints. AIXM business rules were the main focus in this testbed, but SBVR constraints can be defined for any other application schema as well.
- ShapeChange has been extended to automatically derive Schematron code from SBVR constraints.
- A new model transformation has been implemented that allows AIXM extension schemas to be merged with the core schema. This is an important precondition for parsing AIXM business rules, which do not differentiate between concepts from the core schema and extensions.

6.4 Web Feature Services (WFS)

6.4.1 m-click

6.4.1.1 Introduction

The OGC compliant WFS-TE (Web Feature Service) is an aviation data repository responsible for AIXM 5.1 data management, both static documents and dynamic data such as Digital NOTAM messages. The WFS-TE service endpoint is an implementation of the OGC WFS-T 2.0 with additional functionalities for dealing with temporality aspects of aviation entities in accordance with the AIXM 5.1 temporality model (beyond the GML 3.2 temporality).

6.4.1.2 Implemented Standards

The validation component uses the following standardized data types, standards and service endpoints:

- OGC WFS-T 2.0
- WFS-TE Temporality Extension
- AIXM 5.1
- Digital NOTAM 2.0

The public service endpoint implementation is based on OGC WFS-T 2.0, AIXM 5.1 and the Digital NOTAM specification. Further the service end-point implements the WFS temporality extension (WFS-TE), which provides additional support for temporal based data querying.

6.4.1.3 Functions

The m-click WFS-T (transactional) component provides standard CRUD (create, read, update, delete) data management functions based on the WFS-T 2.0 service endpoint specification. The component supports all aspects and fulfills every requirement from the

official OGC WFS-T specification. Additionally, the repository also supports special temporality functions used for the AIXM 5.1 data management.

6.4.1.4 Components

The WFS data repository consists of subcomponents as depicted in the following figure:

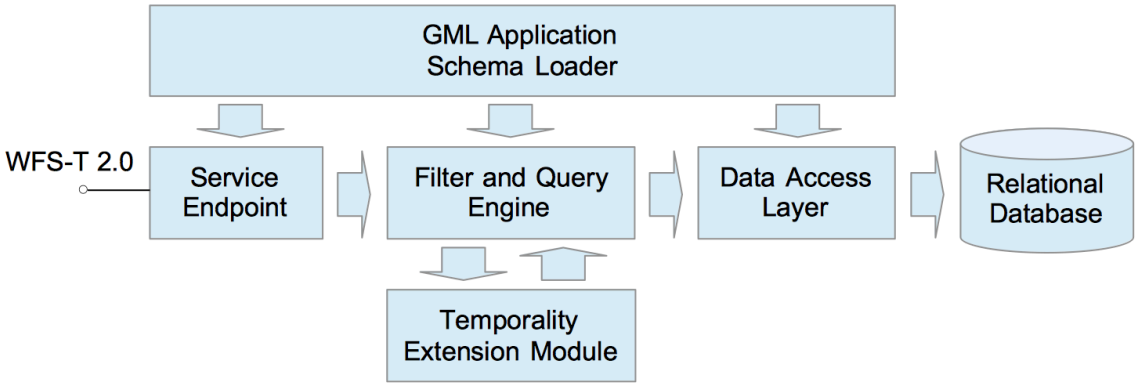


Figure 4 - WFS Component Diagram

6.4.1.5 Execution Process Flow

The WFS 2.0 implementation is based on the synchronous request-response message exchange pattern. Basically, the component is activated by an incoming data request call (query request). Further processing is performed in accordance with following activity diagram:

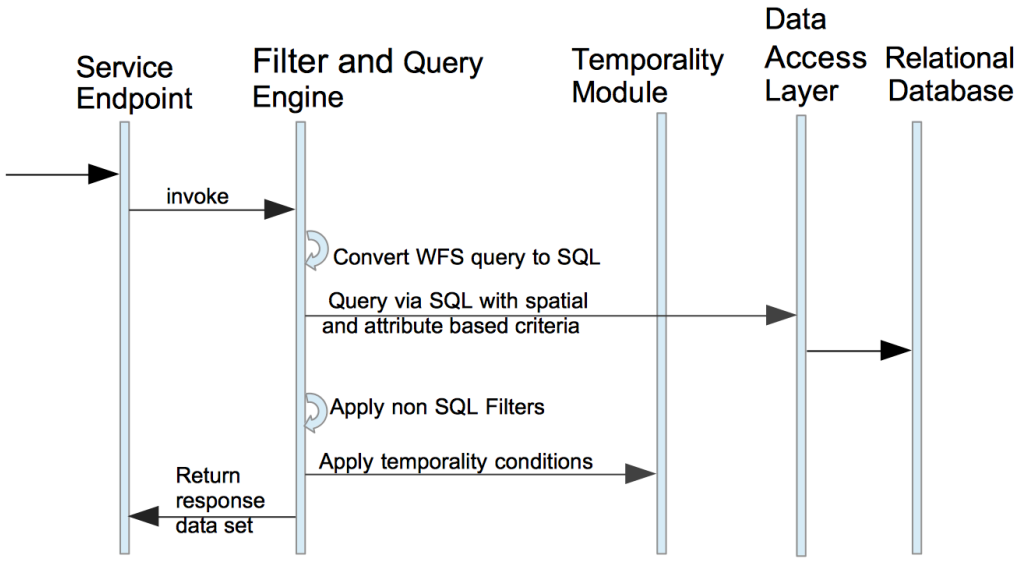


Figure 5 - Process flow diagram

6.4.1.6 Deployment

The m-click WFS is implemented based on an OGC compliant WFS data repository called *deegree*. The WFS runs as a Java EE web application that is bundled with a Java EE 7 compliant Web Container, which implements the Java Servlet specification. Additionally, for purposes of access control and security a dedicated HTTP server is placed in front of the Java web container as protective reverse proxy.

The *deegree* based solution stack is also augmented with an additional, unique m-click component, responsible for dealing with WFS queries based on the temporality extension for AIXM 5.1 (an OGC discussion paper). This extension helps WFS clients deal with the intricacies of the AIXM Temporality Model when requesting AIXM data.

6.4.1.7 Accomplishments

During the activities on conception, design and implementation of this component several technologies, data formats and COTS products suitable for validation service implementation were identified:

- WFS 2.0 implementation based on an open source solution stack can run reasonably fast and deliver quite good performances.
- The Temporality Extension was enabled for the testbed without affecting performance overhead.
- The ability of *deegree* to auto-adjust to any GML Application Schema proved to be useful, demonstrating the flexibility of transitioning from DNOTAM Event Specification schema version 5d to 5e during the testbed with minimal effort.
- Performed more schema validation and data cleaning on the “WFS EU/US” datasets from previous testbeds.

6.4.2 Safe Software

6.4.2.1 Introduction

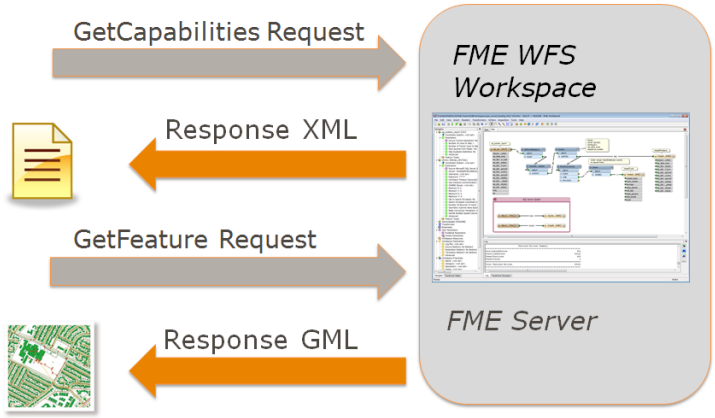
Safe Software provided a limited AFX WFS capability primarily for testing purposes. The goal was to observe what challenges might arise when deploying the new aviation AFX GML and schemas via WFS.

For the OGC Testbed 11, Safe Software’s components were deployed using Safe’s three primary data integration products: FME Desktop (Data Inspector and Workbench), FME Server and FME Cloud. Data Inspector is the viewing and inspection tool which is described in detail under Safe’s Aviation Client section (6.6.4). FME Workbench is the primary tool for authoring data conversions and transformations. FME Server is used to deploy data transformations in a services oriented environment. FME Cloud publishes Server solutions to the cloud (Amazon). Safe’s Aviation AFX WFS was authored in Workbench and then deployed on a FME Server running in FME Cloud (which runs in the Amazon cloud).

Workbench is normally used to author transformation scripts or workspaces to convert from one format, datastructure or dataset to another. FME OGC web services are hosted

by publishing a service broker workspace to the data streaming service on FME Server. Instead of a workspace that only handles data conversion, the service broker workspace handles the web message traffic – accepting requests and generating responses according to the chosen service standard. For WFS, this means the workspace accepts WFS GET (a URL) or POST (XML encoding) requests (GetCapabilities, DescribeFeatureType and GetFeature) and generates the appropriate responses as XML or GML data streams. This WFS workspace simply needs to be published to FME Server's data streaming service in order to function as a web service – no coding or scripting required. Also, the workspace is configured to support FME Data Inspector as a WFS client.

Any web service, such as WCS, WPS, WMTS or SOS could also be supported by this FME service broker workspace approach. All that is required is to understand the web service protocol client / server requirements and configure accordingly.



FME workspace with WFS interface

6.4.2.2 Functional Overview

Presently, the AFX WFS was provided with only two feature types, spatial extents queries and one XML filter operation, though these could be extended relatively easily. In conjunction with FME's WFS client – Data Inspector – this was sufficient to test basic functionality of the standard WFS request types: GetCapabilities, DescribeFeatureType and GetFeature.

6.4.2.3 Deployment Characteristics

Safe's Aviation AFX WFS was authored in FME Workbench and then deployed on a FME Server hosted on FME Cloud (which runs in the Amazon cloud). FME version 2015.0 was used throughout. Authoring was done with FME Workbench and Data Inspector provided the test client, both running on Windows. FME Server on FME Cloud runs in a linux environment.

6.4.2.4 Challenges

The principal challenge encountered was processing the custom geometries associated with AFX. The problem is that the custom geometries contained in AFX are not strictly speaking GML. GML provides a finite list of supported geometries. AFX defines custom geometries that go beyond this. Still, given appropriate configuration settings, FME is able to read AFX However, existing GML writing does not support custom geometry writing. While this functionality should become available in the coming months, for the Testbed this limited the degree of functionality that could be implemented for our AFX WFS. Specifically, this meant additional configuration had to be performed per feature type, which is why this test AFX WFS was limited to just a couple of feature types.

The number and nature of the AFX schemas made deploying a complete DescribeFeatureType response something of a challenge. In the end Safe Software was able to provide a complete response so no local schemas were required to render the AFX GML.

Given the above challenges and the more prominent role required of Data Inspector as a demo and testing client, time and resource limitations did not permit a full publication of additional datasets, feature types and query filters on the AFX WFS. This may be a worthwhile exercise for future OGC testbeds.

6.4.2.5 Accomplishments

The main accomplishments for Safe Software hosting this AFX WFS was to be able to demonstrate that FME with FME Server can be used to both host and consume WFS web services. This Testbed activity also showed that FME was able to consume and process data from the new AFX GML format. In addition, it provided a testing platform to explore FME's current custom geometry support and determine what is needed to extend that to more fully support AFX publication as well as consumption.

6.4.3 Snowflake Software

6.4.3.1 Overview

The Snowflake Software components contribute to the access tier of the overall Testbed 11 Aviation Architecture. Components use Snowflake Software's commercial off-the-shelf products, GO Publisher and GO Loader Aviation, which are comprised of a series of flexible, scalable components supporting the transformation and data exchange requirements of aeronautical information systems. Snowflake Software provided the following read-only Web Feature Service (WFS) v2.0:

- Digital NOTAM Enrichment Service
- Aviation Feature Schema (AFX) Service

6.4.3.2 Digital NOTAM Enrichment Service

Typically AIXM 5.1 baseline data would not be transmitted as part of a digital NOTAM message. However, delivery of AIXM baseline data alongside the digital NOTAM

message is essential for validation purposes. As a result Snowflake Software prototyped a digital NOTAM enrichment service; the enrichment of digital NOTAM messages includes the retrieval of relevant baseline data that corresponds to the affected feature.

AIXM 5.1 baseline data and digital NOTAM data were loaded using commercial off-the-shelf software into a PostgreSQL (PostGIS) database. The enrichment process is invoked on load of digital NOTAMs, using internal processing logic; the loaded digital NOTAMs are enriched by the appropriate AIXM baseline data. Once internal processing of AIXM baseline data is complete, digital NOTAMs accompanied with associated AIXM baseline data can be retrieved from the WFS service. The spatial closure of digital NOTAMs is also calculated for spatial querying purposes.

WFS stored queries with predefined filter parameters for Testbed 11 scenario use cases were created to enable the request of an enriched digital NOTAM. Figure 6 illustrates the digital NOTAM enrichment service process flow.

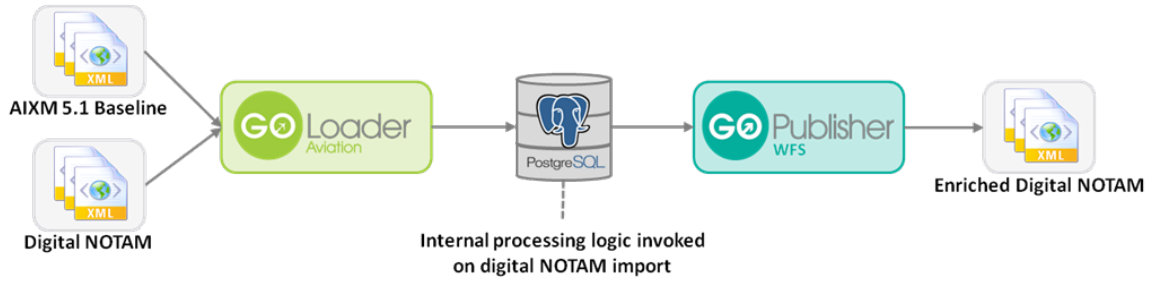


Figure 6 – Digital NOTAM enrichment process

6.4.3.3 Aviation Feature Schema (AFX) Service

The Aviation Feature Schema (AFX) format is designed primarily to aid reusability and therefore to be generic. The design is driven by the purpose of portrayal and mapping - not to replace existing aviation exchange standards such as WXXM, FIXM and AIXM. The AFX acts as a template for application schemas to implement by adding operational attributes; for example the Airport Mapping format can be implemented through extending the AFX schema.

End-user exposure to the complexities of aviation standards such as WXXM, FIXM and AIXM could be mitigated through the use of a simpler model focused primarily on portrayal (visualization and mapping).

Aeronautical data in the form of AIXM 5.1 from multiple sources were loaded to a consolidated database. The consolidated database schema was model-driven from the AIXM 5.1 schemas and AIXM 5.1 data loaded using the commercial off-the-shelf software, GO Loader Aviation.

A mapping from the AIXM 5.1 database schema to the Airport Mapping format (AMXS) implementing AFX was created before establishing a read-only WFS v2.0. This allows

aeronautical features to be requested in AMXS/AFX format. Figure 7 illustrates the AFX service process flow.

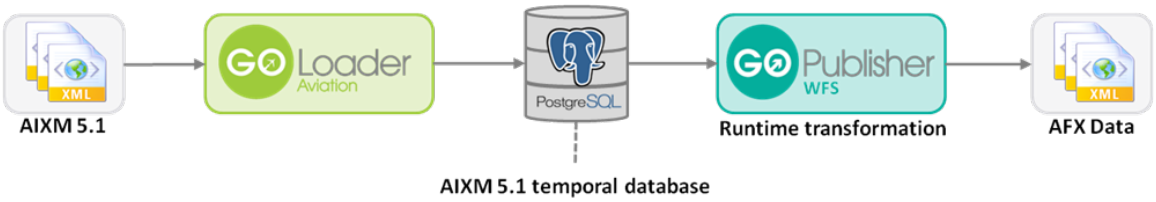


Figure 7 – AFX implementation process

6.4.3.4 Accomplishments

Accomplishments for Snowflake Software's access tier components within OGC Testbed 11 include:

- Successful development of digital NOTAM enrichment service prototype.
- Successful runtime WFS conversion from AIXM 5.1 to AMXS AFX implementation format.
- Demonstration of interaction with related business process and client tier components.

6.4.3.5 Challenges

Challenges encountered throughout Testbed 11 include:

- **Lack of bidirectional feature relations:** In order to successfully enrich digital NOTAMs with the relevant AIXM baseline data, multiple AIXM feature types must be returned. Inter-feature relationships within AIXM 5.1 are one-directional; use of the bi-directional relationships would enable simpler retrieval of relevant AIXM feature types.
- **Test data availability:** Test data for digital NOTAM enrichment required AIXM 5.1 baseline data and related digital NOTAMs. Provision of these data was not available and had to be mocked-up for demonstration purposes within the Aviation thread.

6.5 AIXM / DNOTAM Validator

6.5.1 m-click

6.5.1.1 Introduction

The Digital NOTAM Validation Service is responsible for semantic validation of AIXM 5.1 based payloads, both static documents and dynamic data. The Validation service endpoint is an implementation of the OGC Web Processing Service 1.0 (WPS). Though universally applicable, in Testbed-11 this component is primarily used for Digital NOTAM message validation.

6.5.1.2 Implemented Standards

The validation component uses the following standardized data types and service endpoints:

- OGC WPS 1.0
- AIXM 5.1
- Digital NOTAM 2.0 Event Specification
- Schematron rule-based validation (ISO/IEC 19757)
- ISO 19115 and 19139 (for Metadata)

The public service endpoint implementation is based on OGC WPS and AIXM 5.1, including the Digital NOTAM Event Specification 2.0.

6.5.1.3 Components

The D-NOTAM validator consists of the following subcomponents:

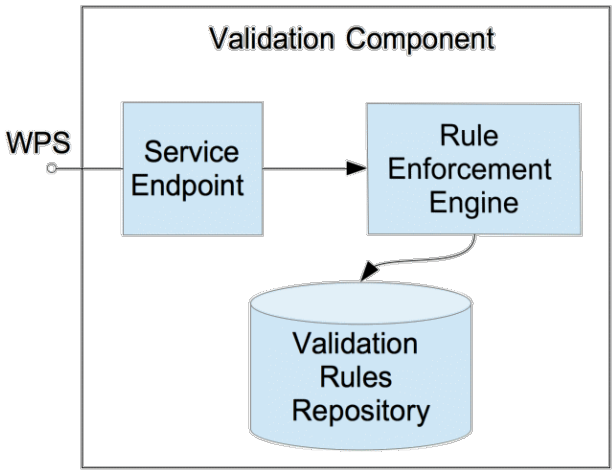


Figure 8 – Validator component diagram

6.5.1.4 Rule Validation

AIXM 5.1 and the Digital NOTAM Event Specification provide a large number of requirements, which specify declarative validation rules for data quality control. These rules are expressed using plain English, following an Aviation domain specific profile of the OMG SBVR standard. The SBVR rules have been translated into Schematron rules. The translation was supported by the SBVR to Schematron Automation Tool (6.3).

The validator executes the Schematron rules on any XML encoded AIXM/DNOTAM data it receives, and stores the validation result in metadata elements that are added to the AIXM/DNOTAM data.

The validation process is not fully independent. The availability of additional data collected and provided by external components - such as the enrichment service (for further details on enrichment, see chapter 8) or an AIXM data repository - is an important precondition for the validation of some rules.

6.5.1.5 Deployment

The validator is implemented as a Java EE web application and deployed in a Java EE 7 compliant Web Container which implements the Java Servlet specification. Additionally, for purposes of access control and security a dedicated HTTP server is placed in front of the Java web container as a protective reverse proxy.

The following solution stack and COTS products are being used:

- Java 7 (runtime environment)
- Saxon XSLT Processor (rule engine)
- ISO Schematron for XSLT2
- deegree - an OGC compliant WFS/WMS/WPS service implementation
- Apache Tomcat Web Container
- Nginx HTTP Server

6.5.1.6 Accomplishments

- Demonstrated that Schematron for XSLT2 can run reasonably fast and deliver good performance on a platform based on Java web container and standard, affordable server hardware.
- The Schematron/XSLT approach might not be able to support all kinds of validation rules, even in combination with an enrichment service. In some cases additional components have to be combined with the rules engine in order to implement certain rules. For example, if the domain of the validation function is a complete aviation static data repository (an airspace must not have geometry which intersects with other aerospace's geometries), it may not be possible to enrich a NOTAM at that extent, sufficient for this sort of validation.
- The OGC WPS is suitable for implementation of a validation service. The ability to load input data from an external source is advantageous. However, WPS is only a good match if the input data can be expected to be well formed XML.

- ISO 19115 – Geographic Information: Metadata – was chosen to encode the validation result in order to fulfill the “standard reusability policy”. We tried to avoid creating a special purpose data format. Unfortunately, ISO 19115 and its XML encoding (defined by ISO 19139) are very verbose and far beyond the needs of the validation service.

6.6 Aviation Client

6.6.1 Atmosphere

The ATMOSPHERE Aviation client component aims at demonstrating the easy and interoperable integration of a third party application within the OGC framework. The component has been derived from ATMOSPHERE’s existing solution PLANET, and updated to be able to communicate seamlessly with the proprietary ATMOSPHERE architecture and with all components of the OGC architecture (GeoSPARQL server, WFS, WMS, WPS, FPS …).

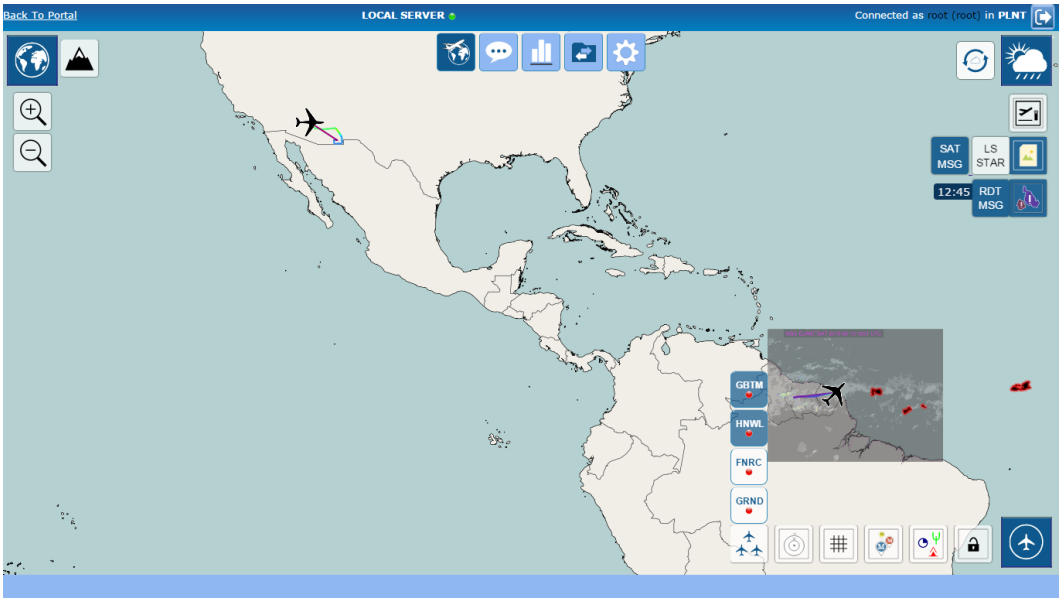


Figure 9 – PLANET collaborative solution for e-operations

The purpose of the component is to demonstrate that integrating an existing solution (namely the PLANET client) in the OGC framework, having it work in an interoperable way with other OGC-compliant components, can be achieved in little time and at limited technological cost.

In the frame of the Testbed, the PLANET client has been upgraded in order to comply with the following scenario: A flight leaves Charles de Gaulle to San Francisco, querying aeronautical information from various data providers, provided in various format, all in a seamless interoperable fashion from the end-user perspective.

To comply with this scenario, the client queries the Feature Portrayal Service (FPS) with a feature request destined for a Web Feature Service and an associated request for portrayal.

Prior to that, the client has connected to an ontology server, defining the community it belongs to, and inferred a request for styling that is also packaged within the FPS request.

In the Testbed, the PLANET-OGC client deals with various data formats, including plain AIXM (aeronautical features) and the associated portrayed images.

The main challenges faced during the Testbed were the following:

- Understanding of the full extent of the various standards to be used.
- Updating the client to create well-formed requests to the FPS, including links for data querying and portrayal styling

Thanks to the support of the architect and of all participants and component developers, those challenges were faced and addressed in due time with limited issues.

This Testbed, from a client developer perspective, clearly demonstrates that over the years, the technological cost for a third party client to integrate in the complex OGC framework decreases significantly.

6.6.2 Luciad

6.6.2.1 Introduction

The Luciad Aviation Client component focuses on the integration & testing of all service components developed within Testbed 11 Aviation and on the implementation of the envisioned use cases / scenarios based on those services. This component is based on Luciad's COTS product LuciadLightspeed, which offers numerous capabilities & benefits that are of direct use in the client: full support for AIXM (including its temporality and metadata models), FIXM, WXXM, connectors to OGC services such as WMS, WCS, WPS and WFS-T, flight simulation, 2D & 3D visualization. On top of this product, a thin layer has been developed to support custom functionality needed in Testbed 11 Aviation (e.g., AFX) and to provide the user with a dedicated user interface focusing on the envisioned use cases / scenarios. Figure 10 shows a screenshot of the Luciad client's user interface.

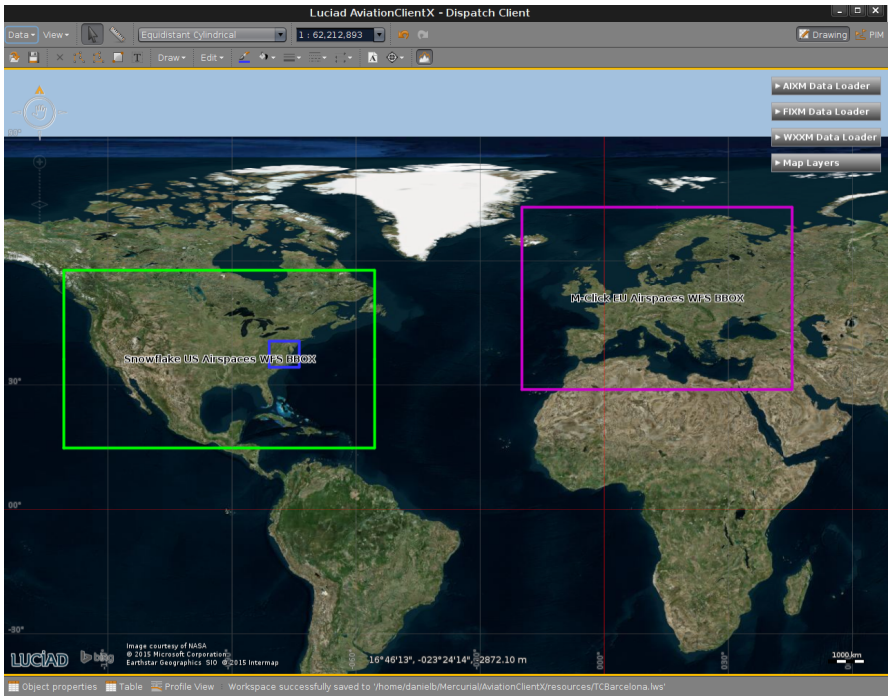


Figure 10 – The Luciad Aviation Client showing the geographic boundaries of a number of Aviation WFS data sources provided by the other participants.

6.6.2.2 Functional overview

The Luciad Aviation client provides the following functionality to support the Aviation thread in Testbed 11:

- Map-centric 2D & 3D display with intuitive user interface giving access to various actions, including:
  - Map controllers to manipulate the map (zoom & pan)
  - Map layer control with access to predefined background data layers
  - AFX, AIXM, FIXM and WXXM-based OGC web service connectors
  - Flight preview / simulation
  - Visual representation & browsing of feature properties inside a balloon
- Client interface to query data from an OGC Web Feature Service delivering AIXM and AFX data. Users can select the desired WFS, feature type and various filtering options (e.g., spatial filters).
- Client interface to connect with the OGC WPS-based Validation Service.
- Client interface to connect with the OGC FPS.
- Client interface to a WMS 1.1.1 & 1.3.0 service delivering background data. Users can query and retrieve layers from a WMS.

- Rendering engine with support for SLD / SE 1.0/1.1 to render vector feature and raster data. This engine integrates extensions to support ICAO Annex 4 rendering guidelines for aeronautical data.
- Support to represent & browse ISO 19115-compatible metadata and to encode / decode to / from an ISO 19139-compatible data source. The use of metadata extensions / profiles is supported.
- Wide range of data format support, including:
  - AIXM 3/4/5, FIXM 2/3 and WXXM 1.1, including schema extensions.
  - .Additional aeronautical and weather data formats like AIXM 3.3/4.0/4.5, ARINC 424, DAFIF, ASTERIX, ASDI and GRIB.
  - Raster format support (GeoTIFF, TIFF, JPEG, JPEG 2000, GMLJP2, JPIP, PNG, GIF, ECW, MrSID, CADRG/ADRG/USRP, DTED, USGS DEM, Oracle GeoRaster) for imagery and elevation background data.
  - Vector format support (ESRI Shape, MapInfo MIF/MAP, GML 2/3.1.1/3.2.1, SVG, DGN, DWG, Oracle/Informix/MSSQL databases) for vector-based background data.
  - Other formats: OBJ, OpenFlight, OGC KML 2.2 and GeoPDF.

6.6.2.3 Deployment characteristics

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

6.6.2.4 Challenges

One particular encountered challenge in Testbed 11 Aviation was the design & development of an architecture and workflow to support the common symbology requirements. Given a client, a GeoSPARQL server, an SLD/SE producer and an FPS, the goal was to find an optimal workflow to be able to automatically find the right symbology for a given feature data set – by reusing existing capabilities from standards and services as much as possible. This challenge was tackled in cooperation with the CCI thread (which had a similar symbology requirement) and the relevant component producers.

6.6.2.5 Accomplishments

The key accomplishments for Luciad's Aviation client component in Testbed 11 include:

- Successful integration of the new AFX aeronautical format.
- Demonstration of and interaction with all relevant service components provided within the Aviation thread.

- Contribution of a rich client equipped with lots of capabilities and features that help to demonstrate the Testbed 11 aviation use cases and to perform testing and integration with a wide variety of service components.
- Collaboration with Testbed 11 aviation service component providers on the developed functionality and provision of feedback from a client's perspective (e.g. SPARQL – FPS – WPS symbology architecture, and WPS DNOTAM Validator).

6.6.3 m-click

6.6.3.1 Introduction

The m-click WFS-TE Aviation Client is a Human Machine Interface (HMI) application that runs stand-alone in the web browser. The component provides GUI functions for WFS-TE query building, service invocation, AIXM entity visualization and map/terrain representation.

6.6.3.2 Functions

The component provides the following GUI functionalities:

- WFS query builder
- Service invocation
- Result set representation
- AIXM entity visualization
- Map visualization

6.6.3.3 Components

The WFS-TE Aviation Client has been built from the visual components as depicted on the following figure:

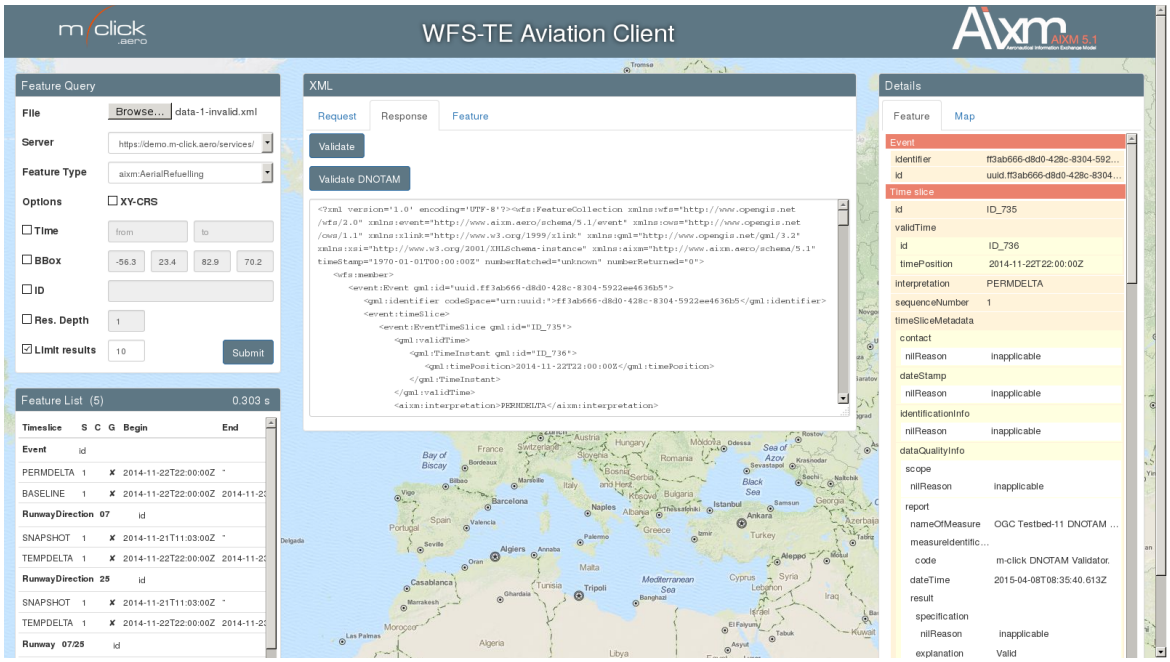


Figure 11 – Aviation Client Components

The "Feature Query" panel provides support for WFS-TE query design. The invisible WFS invoker component will push a query to a WFS server. Both query and result documents are displayed on the central panels labeled "XML". It also contains commands for document/NOTAM validation. The "Feature List" below the Feature Query provides entities returned as part of a result set formatted in a more convenient way. Features with proper geometry are depicted on the map component in the middle. On the right side the panel "Details" is used to display attributes of the selected aeronautical entity.

6.6.3.4 Execution Process Flow

The execution flow is based on the model view controller (MVC) paradigm.

6.6.3.5 Deployment

The m-click Aviation Client is a pure web application that runs stand-alone in the web browser (Client Component). It also contains server components that provide supportive functionalities. However, in principle, the client could run without that server component.

The application is hosted in a standard web server. It intensively uses java script on both client and server side und relies on Open Layers for cartographic material used for entity visualization.

6.6.3.6 Accomplishments

1. Integration of the (WPS) validation service into a WFS aviation client and into a Digital NOTAM validation workflow.
2. WFS query builder and native request and response document visualization.

3. Representation of AIXM 5.1 result sets, entities and attributes.

6.6.4 Safe Software

6.6.4.1 Introduction

The client component contributed by Safe Software for Testbed 11 is FME Data Inspector. FME represents Safe Software's suite of data conversion and transformation tools focused on managing the exchange of spatial and non-spatial data between systems with differing file formats and structures. FME is often described as an ETL tool for spatial data. FME also has additional capabilities to manage the complexities of spatial data's associated feature geometries, attribute tables, and coordinate systems. Safe provides data integration and consumption components based on FME to most leading GIS and CAD vendors.

FME Data Inspector is the primary FME tool for viewing and interrogating datasets and can read any FME supported format. This includes over 350 GIS, CAD, raster, point cloud, 3D, BIM, XML, JSON, web, database and tabular formats, along with comprehensive support for many OGC GML (GML, CityGML, AIXM, AFX), web services formats (WMS & WFS).

Data Inspector also allows the user to overlay disparate data sources so that multiple layers can be displayed together. In addition, a background map feature is provided so that web mapping services can be used to provide context. Full inspection capabilities allow any complex attribute and geometry model to be fully explored. 3D rendering and texturing are also supported.

6.6.4.2 Functional Overview

Safe's Data Inspector provides the following functionality to support the Testbed:

- Ability to read and display relevant OGC formats:
  - Virtually any GML, including AIXM 5.1, AFX
  - Other aviation formats such as AIXM 4.5, ARINC 424
  - OGC web services (WMS 1.1, 1.3, WFS 1.0, 1.1, 2.0) including support for WFS delivering complex schema GML.

- Ability to inspect the details of complex schemas
  - Complex attribute schemas:
    - Nested element structures
    - List or series elements
  - Complex geometries:
    - Multiple geometries
    - Nested and heterogeneous geometries, including 3D volumes such as air spaces
- Ability to overlay aviation data with basemap data from any of FME's 350 supported vector and raster formats or background map services. This provides context both for visualization as well as testing for positional accuracy.

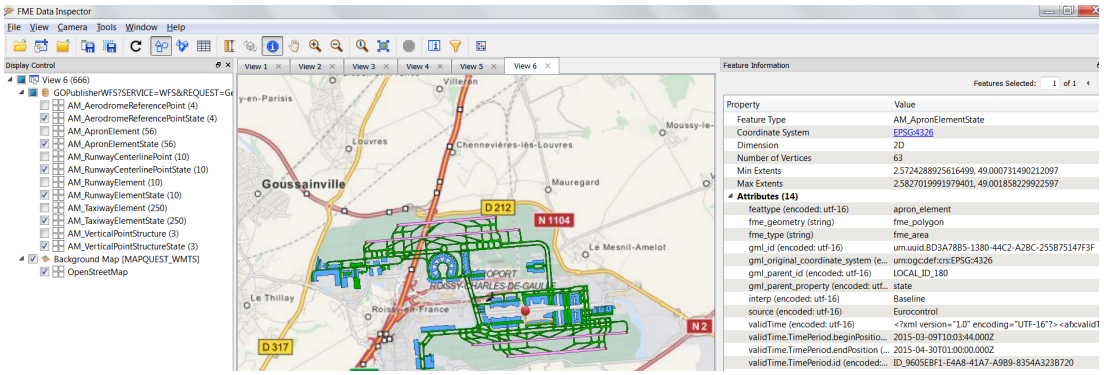


Figure 12 – FME Data Inspector viewing AFX WFS of Charles De Gaule airport

6.6.4.3 Deployment Characteristics

FME is offered across various platforms, and includes support for Windows, Mac and Linux. Self-contained installers make setup a largely automated process.

6.6.4.4 Challenges

The principal challenges encountered were primarily schema handling and web service interaction. Early on, the participants discovered that FME's AIXM support did not include support for the AIXM event extension. This was mitigated by simply including event schemas in the client's application schema path and then FME is able to read it.

Another problem was that while FME could read AIXM GML, the WFS client had some problems rendering GML geometries from AIXM WFS. Given the nested nature of AIXM, the FME client would read the parent objects but not all the associated child object geometries. This bug was rectified in FME early on in Testbed 11.

Another challenge was processing the custom geometries associated with AFX. The problem is that the custom geometries contained in AFX are not strictly speaking GML. GML provides a finite list of supported geometries. AFX defines custom geometries that go beyond this. Still, given appropriate configuration settings, FME is able to read it.

Perhaps the most significant challenge associated with supporting Testbed 11 with FME's Data Inspector as an Aviation client was interacting with the Testbed's various OGC web services. Variations in implementation by the service providers meant that significant time was spent diagnosing problems with web services queries and result rendering. Often, standard OGC request types were not fully supported. For example, in many cases, DescribeFeatureType did not initially return valid GML application schemas. Or if a schema was returned it was often incomplete or included other schemas that were not readily accessible. In the end, these deficiencies were communicated to the service providers and where needed we were able to configure the FME aviation client to mitigate these issues (e.g. provide local schemas as needed).

6.6.4.5 Accomplishments

The primary accomplishments that Safe Software made during Testbed 11 was to address the challenges encountered above, either in terms of workarounds, enhancements or bug fixes. We were able to verify that FME Inspector can read AIXM events when the schemas are included in the client's application schema path. These schemas will soon be incorporated into the standard FME installation. FME's WFS reader was extended to support nested objects so that any child geometries such as those encountered with AIXM are now read automatically.

Configuration options were refined so that FME's Inspector was able to consume both AIXM and the new AFX GML from all the WFS server components available within the Aviation thread of the Testbed. Finally, perhaps the most significant contribution for using FME as an aviation client for Testbed 11 is that this provided an opportunity to test a range of OGC services against a commonly used GIS data integration platform not specialized for use with aviation data. This provided mutual benefit opportunities for improvements to OGC support for all the components involved.

7 Data Brokering

The Data Broker concept developed in Testbed 11 enables the setup of cascading OGC WFS servers to form a data source chain, in which one service is capable of providing information coming from one or more other services.

Previous OGC testbeds have extensively trialed the use of the WFS for the provision of aeronautical data, in which the WFS was typically backed by a database. This resulted in a flat architecture, including one or more WFS data sources providing aeronautical data to applications.

In a real deployment, WFS solutions might be applied at more than one level in the data chain, and some data might not always be replicated at higher levels. For instance, WFS data sources could be set up at a national level, from where data might be further centralized at regional levels and made accessible to end-user systems. This is where the data broker concept comes into play; its overall goal is to see how WFS components can be cascaded and form a data source chain, as illustrated in Figure 13.

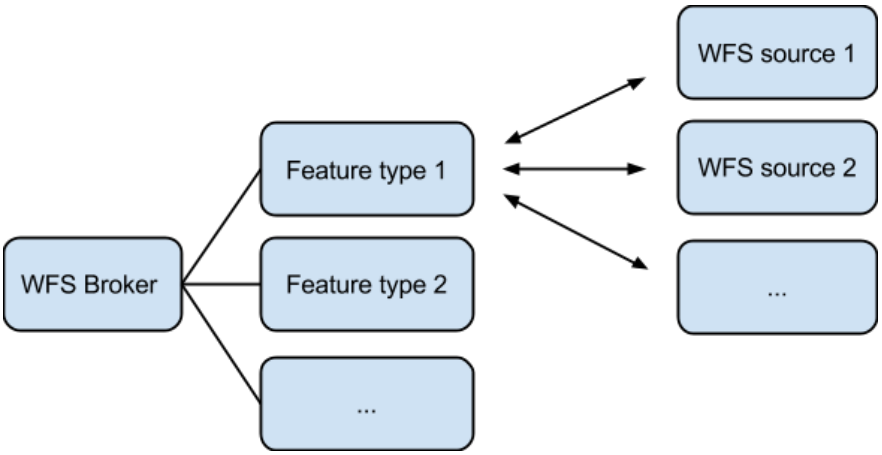


Figure 13 - High-level architecture of the WFS-based Data Broker

Within Testbed 11, an implementation was developed to review the overall feasibility of the concept and to investigate a number of specific broker responsibilities and use cases:

- Provenance and Lineage: to inform the user about the origin of the data, the Data Broker adds (1) origin information to its offered WFS feature types in the capabilities and (2) ISO 19115 Lineage metadata to each requested feature. This Lineage metadata includes information about the WFS data source and the aggregation step performed by the Broker.
- Conflation: the Data Broker performs a conflation step to merge data coming from multiple WFS data sources into a single coherent data source. This conflation step includes capabilities to merge similar feature types from different sources into one feature type and to avoid duplicate features in a query result.
- Caching: the Data Broker allows to cache data to reduce the amount of requests needed to be sent to its WFS data sources.

- Scalability: the Data Broker is implemented using a data processing pipeline that streams data feature-per-feature to reduce the time-to-response for the client and to avoid excessive memory usage.

For more information about the implementation and these responsibilities / use cases, please refer to the dedicated Engineering Report “OGC Testbed 11 Aviation – Data Broker Specifications ER” (OGC 15-028).

One of the main conclusions is that it is feasible to develop a Data Broker solely relying on the OGC WFS 2.0 standard. For an optimal setup, the ER does discuss a few additional recommendations & added features.

8 Digital NOTAM – Enrichment and Validation

Quality assurance and control is a primary concern in an aeronautical information system. The validation of static and dynamic aeronautical data – and thus also of Digital NOTAMs – is an important function. A service that can validate such data against established business rules is a key asset for data publishers and consumers.

In Testbed 11, a Digital NOTAM validation workflow has been implemented. This workflow had the following steps:

- Automatically derive Schematron rules from AIXM SBVR business rules.
- Load the Schematron rules into a validation service.
- Validate DNOTAM data at the validation service by:
  - enriching the data to be validated with static data as necessary
  - executing the Schematron rules on the data
  - storing the validation result using metadata fields of the data

The workflow was implemented using standards based software components. Brief descriptions for each of these components are available in chapter 6. Details about the individual steps and the workflow in general can be found in dedicated engineering reports. The first step in the workflow – the automated derivation of Schematron from business rules expressed using SBVR – is documented in detail in (OGC 15-024). The other steps – enrichment and actual validation – are documented in (OGC 15-027).

9 Aviation Feature Schema

9.1 Problem Statement

9.1.1 Overview

Developed by EUROCONTROL, the Aviation Feature Schema (AFX) is a template to implement application schemas by adding their operational attributes. For example, the Airport Mapping format can be implemented by extending AFX. The AFX defines concepts of geometry and temporality through predefined classes and properties, therefore these need not be redefined by application schemas. This means implementations of the AFX abide by the same structure, therefore aiding interoperability and allowing the rapid development of schemas. The AFX schema is designed to be generic and easily reusable, and it is not intended to replace the standard aviation models, for instance WXXM and AIXM.

The Aviation Feature Schema Recommendations Engineering Report (OGC document 15-026) assesses the suitability of the AFX as a template for lowering the GIS entry level for aviation data, providing recommendations of suitability and areas of improvement. The report is aimed at system and client developers that shall use AFX.

9.1.2 Business Value

The AFX model focuses on portrayal of aviation data, including visualization and mapping of aviation related features, with the aim of lowering the GIS-entry level in aviation.

Existing aviation data exchange formats, such as AIXM, WXXM and FIXM, focus on the exchange of aviation data from system-to-system. The exchange of information from system-to-system introduces complexities that are not required by the end-user wanting to visualize and map the data. AFX does not seek to replace existing aviation models, instead focuses on portrayal by promoting mapping and visualization of aviation related features that are important for improved situational awareness.

9.2 Work Conducted

The AFX was investigated through (1) AFX service provision, and (2) technical assessment of AFX suitability and quality written up as an Engineering Report (OGC document 15-026).

9.2.1 AFX Service Provision

Multiple instances of AFX implementations were created and made available via the WFS interface standard. The implementations extended AFX with the Aerodrome Mapping Exchange Schema (AMXS), therefore proving the use of AFX as a template for application schemas to extend from; details of this extension can be found in (OGC document 15-026).

Aeronautical Information Exchange Model (AIXM) data was converted to the AMXS AFX implementation using commercial off-the-shelf software. The AFX implementation

was then used to generate a WFS to illustrate that AFX implementations could be shipped as a standard web interface.

The WFS provided users with AFX data that could be loaded directly into simple commercial off-the-shelf GIS clients.

9.2.2 AFX Recommendations Engineering Report

The AFX Recommendations Engineering Report (OGC document 15-026) assesses the suitability of the AFX as a template for lowering the GIS entry level for aviation data.

The report is aimed at system and client developers that shall use the AFX by providing an assessment of AFX suitability as a template for application schemas and recommending areas of improvement. Recommendations and observations include the following:

- Assessment of GML profiling techniques to restrict AFX implementations to a simple subset of the GML specification (OGC® 05-033r9 – GML Simple Features Profile).
- Redefinition of AFX temporality concept.
- Assess impact of licensing terms on AFX uptake and implementation.

9.3 End-User Benefits

End-user exposure to the complexities of aviation exchange standards such as AIXM, WXXM and FIXM could be mitigated through the use of simpler models focused primarily on portrayal (visualization and mapping). The AFX's simplistic design focuses on portrayal by promoting mapping and visualization of aviation related features, which is important for improved situational awareness.

10 Lessons Learned

10.1 Data Brokering

The Data Broker concept can be implemented using standard OGC WFS functionality. However, for an optimal setup, several additional recommendations & added features have been put forward in the Data Broker Specifications ER. Highlights include:

- To optimize aggregation of similar AIXM feature types, it is recommended that a WFS serves a single AIXM feature type (e.g., AirportHeliport) per WFS feature type and with as name the AIXM feature type name.
- To enable proper conflation, the recommendation is to use a similar identification scheme for features across multiple WFS data sources. Applied to AIXM, this boils down to using a consistent UUID for the feature's gml:identifier, making sure that the same airport served by 2 WFS data sources have the same identifier value.
- To enable the detection of a changed WFS data source, the recommendation is for WFS servers to publish an UpdateSequence value in the capabilities (optional in the WFS standard).

It is important for the Data Broker implementation to stream data on a feature-by-feature basis, rather than on a query-by-query basis. This significantly increases time-to-response for the client, and it reduces overall memory requirements of the Data Broker. In combination with a feature processing pipeline, the Data Broker can implement all its responsibilities in a streaming way: conflation, caching and provenance/lineage enrichment.

Since a single query can contain data from multiple sources, it is important to store metadata on a per-feature basis. Not doing so would cause loss of information on the origin (lineage) of specific features.

11 Scenario

The demonstration scenario used for the OGC Testbed 11 Aviation thread is based on the same services architecture for both FAA and EUROCONTROL. Figure 14 illustrates the connectivity between the two SWIM regions.

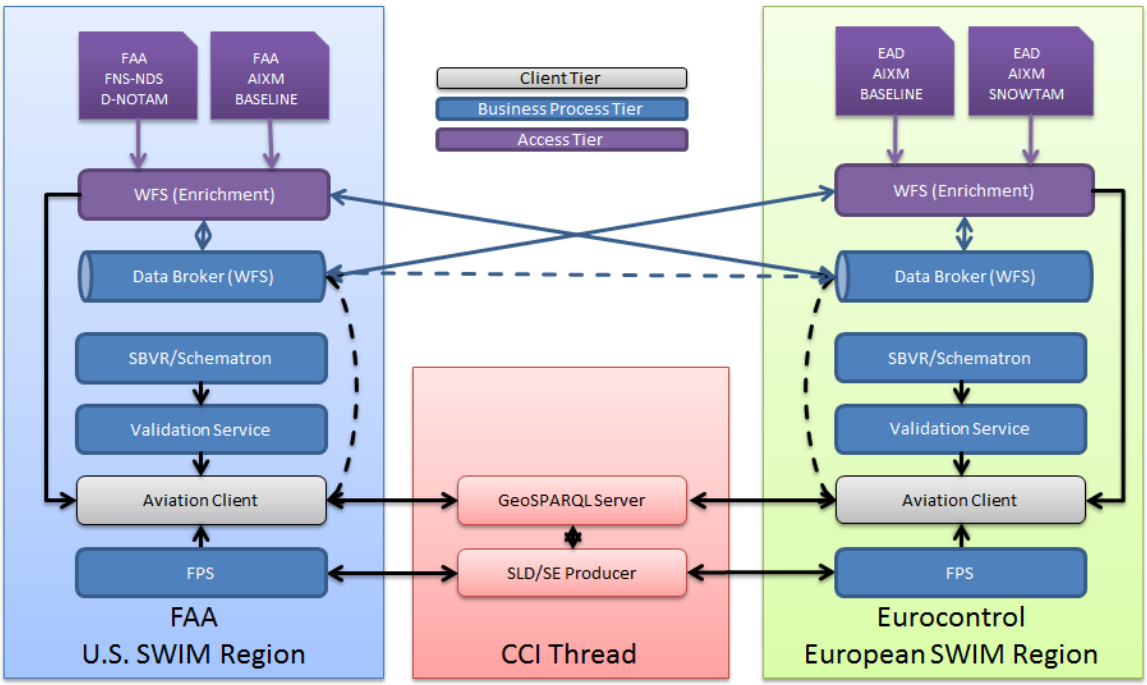


Figure 14 – Component relationships in the demonstration scenario

Three use cases have been defined for the demonstration. They are documented in the following sections.

11.1 Use Case 1 - Flooding causes call for Coast Guard/National Guard

In order to support the evacuation effort, drones are sent out to survey the land for dry areas to place temporary heliports, locations of refueling stations, and obstructions for air ambulances. The AFX schema is designed to remove the complexities of aviation formats, and can take advantage of styled data using SLD/SE. Using the AFX schema to simplify the complexity of aviation data, client applications can view this data easily and quickly to portray the data components around the SFO airport and around the hospital without need to manage the geometry and temporality of the data.

1. The search and rescue command is able to load and display data on their client application using the OGC AFX WFS. The operator can identify refueling stations around SFO airport and obstructions to avoid.
2. Next, they look over the area and find potential areas to place temporary heliports.
3. Now that an area has been identified, the search and rescue command can deploy drones to visualize the area to determine flood levels before finalizing on the heliport location.

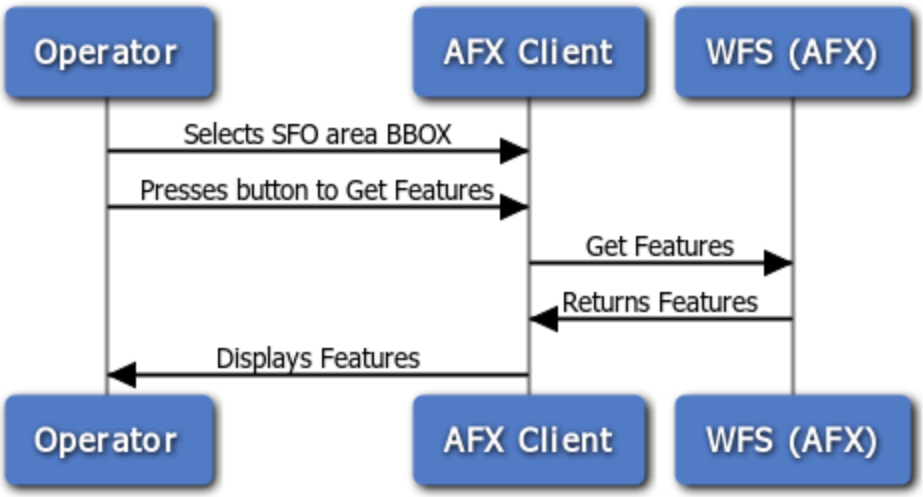


Figure 15 – Use case 1 – retrieving and displaying AFX data

Flooding causes an electrical outage at CPMC hospital in San Francisco. The National Guard and Coast Guard are being called in to assist the evacuation of the hospital's Intensive Care Unit (ICU) patients. Air ambulances are deployed to help evacuate the patients, and a temporary heliport is setup nearby. Once the locations of the temporary heliports are defined, the data can be updated in the FAA AIM WFS SWIM service and the aviation community can be notified of the update. The air ambulance pilots and navigators can then view the data on their electronic flight bags (EFB). The data element features are registered with the GeoSPARQL ontology service which enables the selection of preconfigured "Community Styles" which are presets for a symbology styling. Using community styles, EFBs can select a day (light) or night (dark) view for around-the-clock evacuation. Interoperability reduces the configuration time needed, which could be better used to focus on the evacuation operation at hand.

1. The temporary heliport is now confirmed and the data is uploaded to FAA's AIM database. Using OGC standards, search and rescue airmen can utilize their client application to query the GeoSPARQL ontology service for a predefined set of display styling called Community Styles. In this particular case, the community styles are shown for a daytime or nighttime view. This reduces the setup time required to operate a client application in a new region with new data services.
2. The operator then chooses the daytime style, called LIGHT style, and selects the San Francisco Bay area, and requests data from the Feature Portrayal Service (FPS) using the information gathered from the GeoSPARQL ontology. In this way, the preconfigured request will be forwarded by the FPS to a back-end Data Broker, which finds the data from various Web Feature Services and returns the features to the FPS. The FPS can then render the image for the client.
3. As the operator views the image, they can see the temporary heliport depicted nearby the CPMC hospital.
4. Pilots need to respond quickly, and need to know what obstructions there are, what buildings they can land on and setup temporary heliports, and where the refueling stations are. The operator can see necessary refueling stations at the San Francisco airport nearby.

5. This will be an around-the-clock operation requiring day/night visualization of the airspace. As the rescue evacuation operations linger on, the operator must switch their client displays to nighttime view. The operator can change their client community style to DARK, and perform the same query as before to the FPS.
6. The operator can now see the DARK nighttime view on the client application, and can also view the obstructions around the airport to be sure to avoid them.

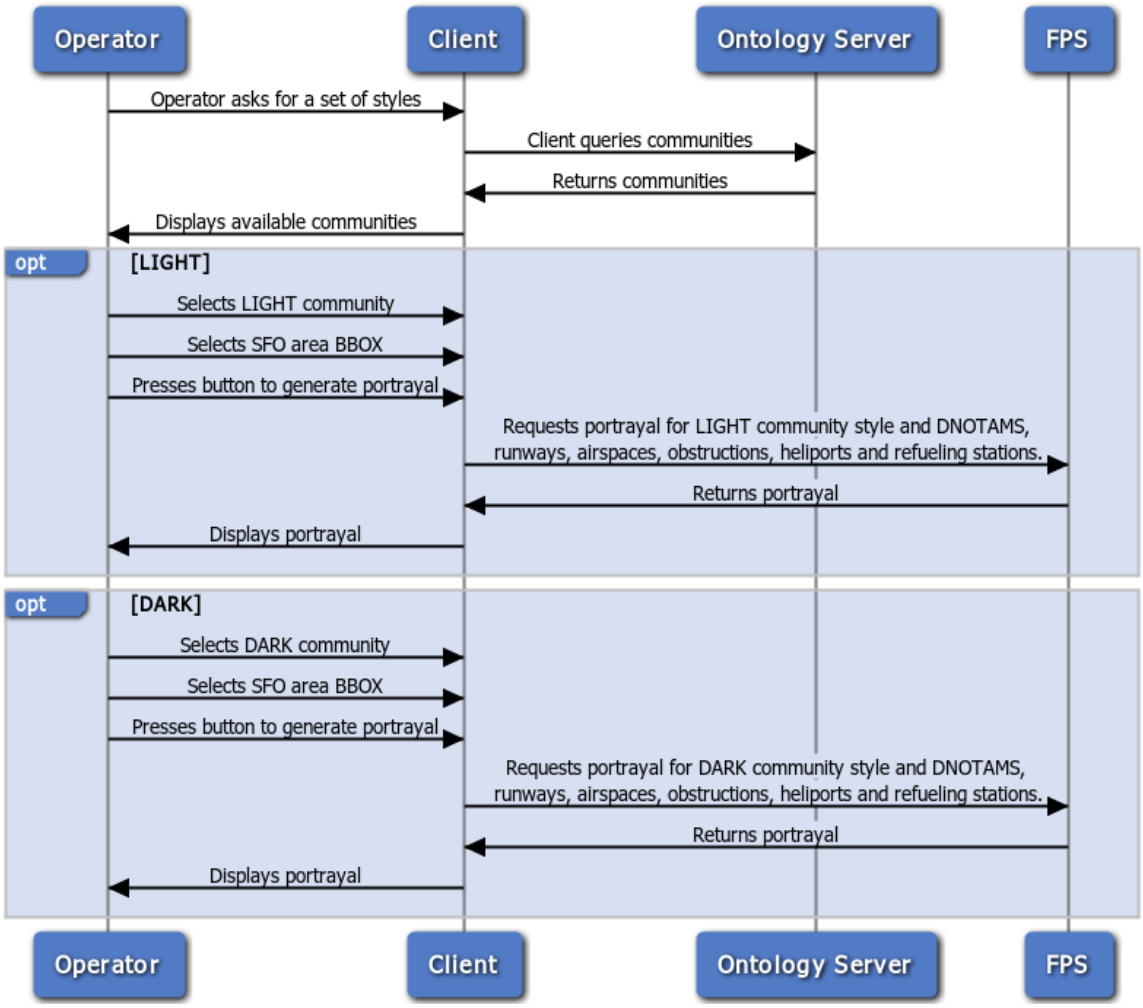


Figure 16 – Use case 1 – retrieving community styled maps of aeronautical data

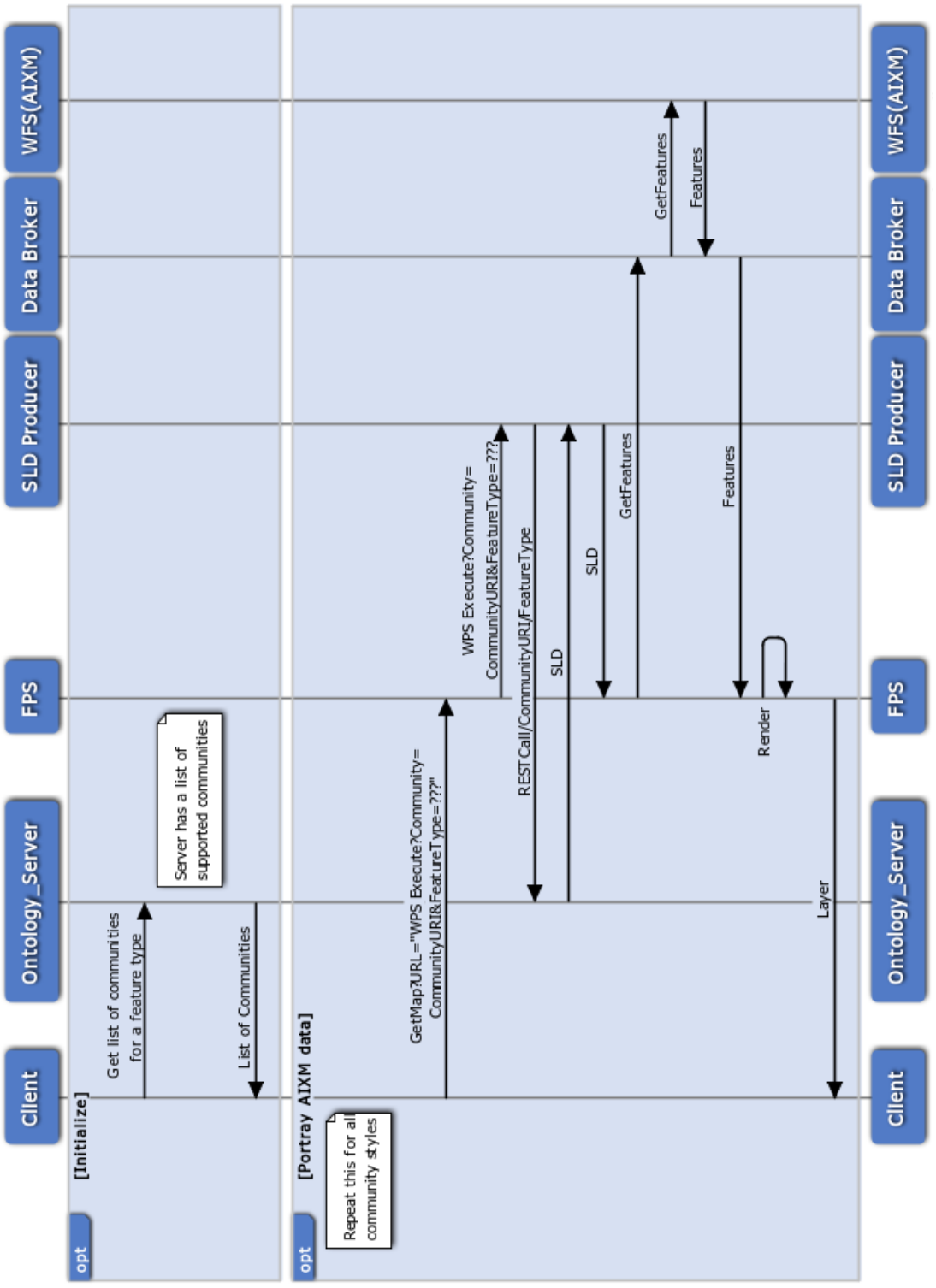


Figure 17 - Use case 1 – FPS backend interactions

11.2 Use Case 2 - Pre-flight briefing for a flight from CDG to SFO

United flight 991 prepares to take off from CDG (LFPG) airport in Paris, France. The flight is destined to arrive in SFO (KSFO) airport. A pilot can use a client to query a single service to portray both EU data and US data through the use of a Data Broker. Using Data Broker, client services do not need to know about all the data web services across regions. The Data Broker is aware of multiple WFS services and will query the correct WFS to simplify the process.

An airline dispatch operator receives a notice from the tower control and relays the information to the airline pilot. The pilot must change runways for departure and wishes to view the runway information for CDG prior to takeoff.

1. The operator executes a BBOX query on the client to portray the CDG area.
2. The client queries the FPS which queries the Data Broker, which requests the information from the EU AIXM WFS to provide airport visualisation for CDG.
3. The operator executes a BBOX query on the client to portray the SFO area.
4. The client queries the FPS which queries the Data Broker, which requests the information from the US AIXM WFS to provide airport visualisation for SFO.

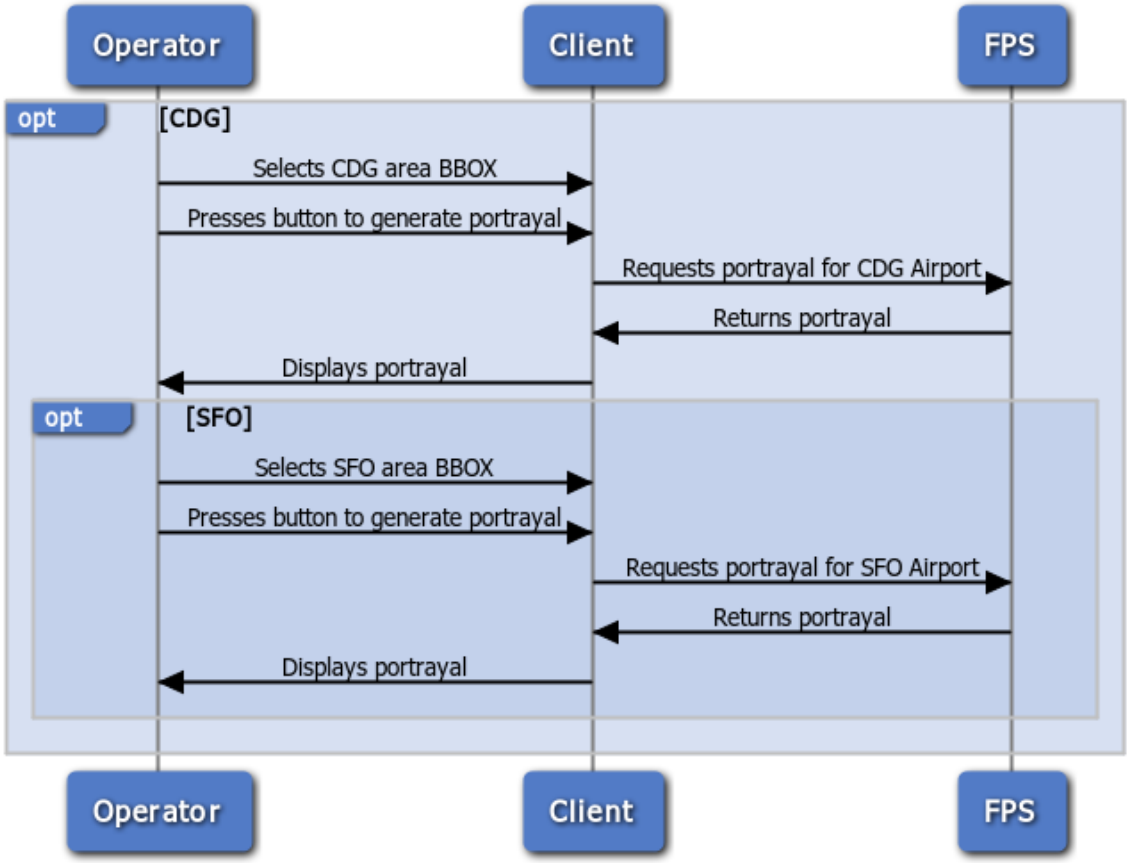


Figure 18 Use case 2 – client interactions

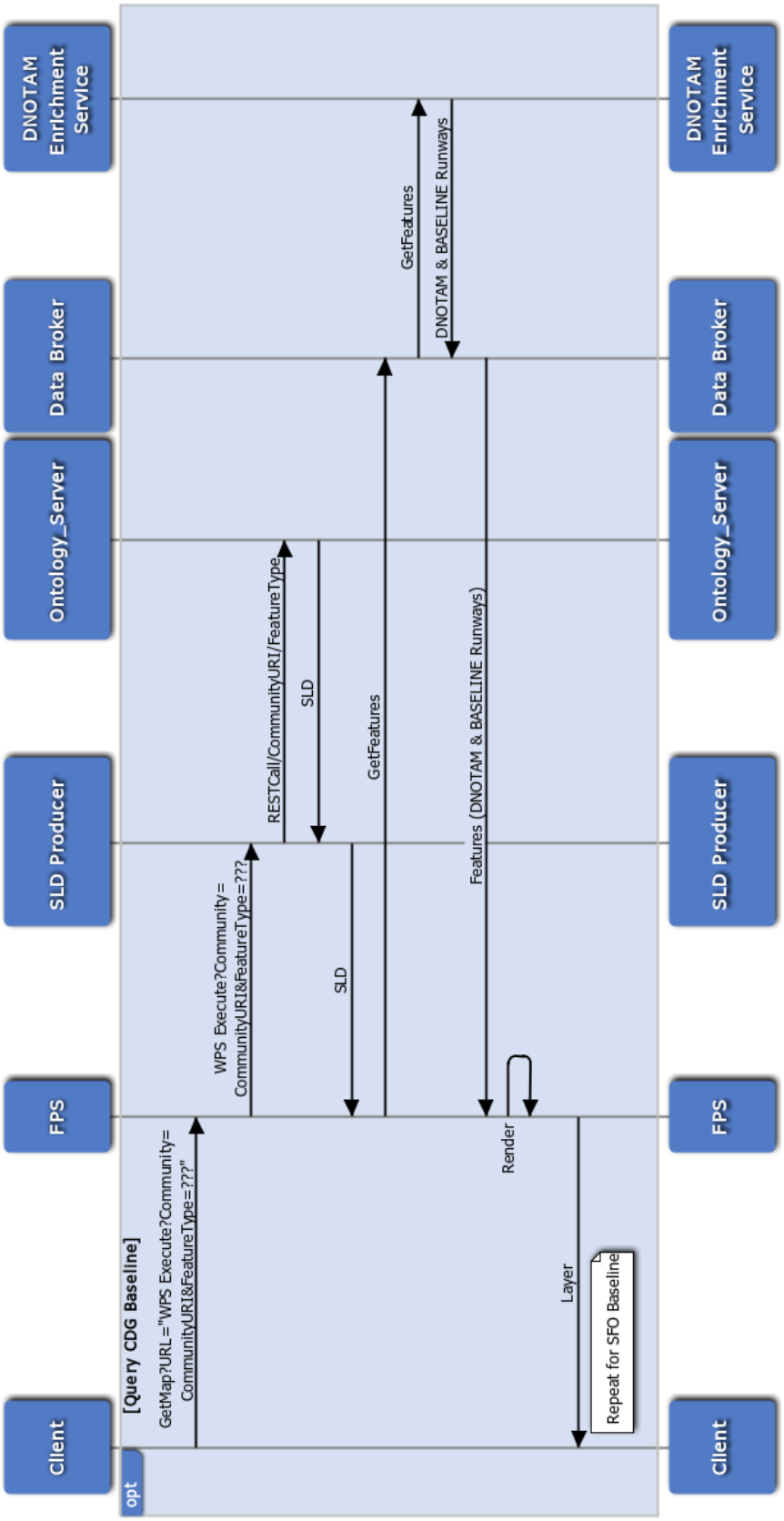


Figure 19 – Use case 2 – Backend interactions

11.3 Use Case 3 - Flooding affect airport runway in SFO

Flooding has worsened in SFO and it is now too dangerous to allow aircraft to land on major runways in SFO. A Digital Notice to Airmen (DNOTAM) is issued by the FAA via the Federal NOTAM Service (FNS).

An aviation client can view the DNOTAM in its textual format, but without BASELINE data, the feature, in this case a runway, cannot be portrayed without the BASELINE data to which the DNOTAM is correlated. Using a DNOTAM Enrichment service, the data is enriched with Baseline data and can be retrieved in a single DNOTAM query. The data can be validated by a DNOTAM validation service to check for errors against a set of business validation rules. These rules are provided in form of Schematron code that has automatically been derived from SBVR business rules for DNOTAMs using the SBVR-to-Schematron automation tool.

The DNOTAM alone cannot be portrayed due to lack of geometries and BASELINE data to which it references. Using a DNOTAM Enrichment Service, the DNOTAM can be "enriched" with BASELINE data elements from SFO airport. The enriched DNOTAM can be checked to ensure it follows the proper business rules and constraints to ensure proper portrayal. To do this, the client can query the DNOTAM Enrichment service for the enriched DNOTAM, then use the DNOTAM Validation service to validate the DNOTAM data. First, the DNOTAM data is validated against the EU rules. An error is detected. This error is logged, but nevertheless, it is a U.S.-based DNOTAM, so the operator reconfigures the client to validate against U.S. rules. Here, the DNOTAM validates correctly. This ensures that the issued DNOTAM can be processed properly with U.S. operational rules.

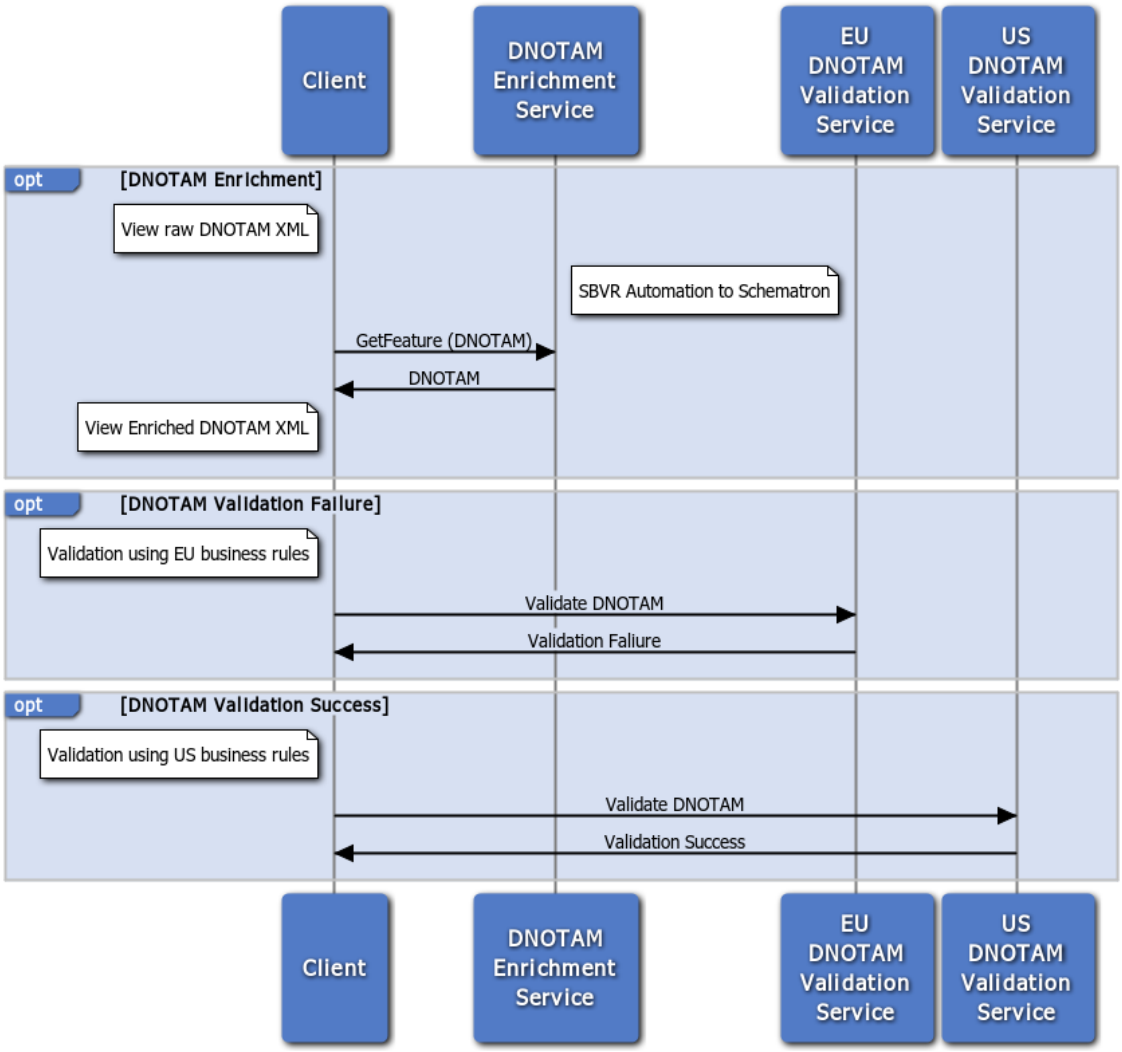


Figure 20 – Use case 2 – validating a Digital NOTAM

United Flight 991 departed from CDG and is crossing into the U.S. through Canada/Washington State FIR border. Upon FIR hand-off, United dispatch queries the FNS and discovers a DNOTAM for Flight 991's arrival airport describing a runway closure.

In order to portray the closed runway, the client will query the DNOTAM Enrichment service and display the runway closure. The DNOTAM Enrichment service responds to the query with the DNOTAM plus the enriched BASELINE data enabling the pilot to see the closed runway. The United Airline's dispatch operator can view the DNOTAM with BASELINE enrichment on the aviation client and contact the pilot in mid-flight with the updated information. Dispatch then proceeds with flight plan amendment for an airport diversion.

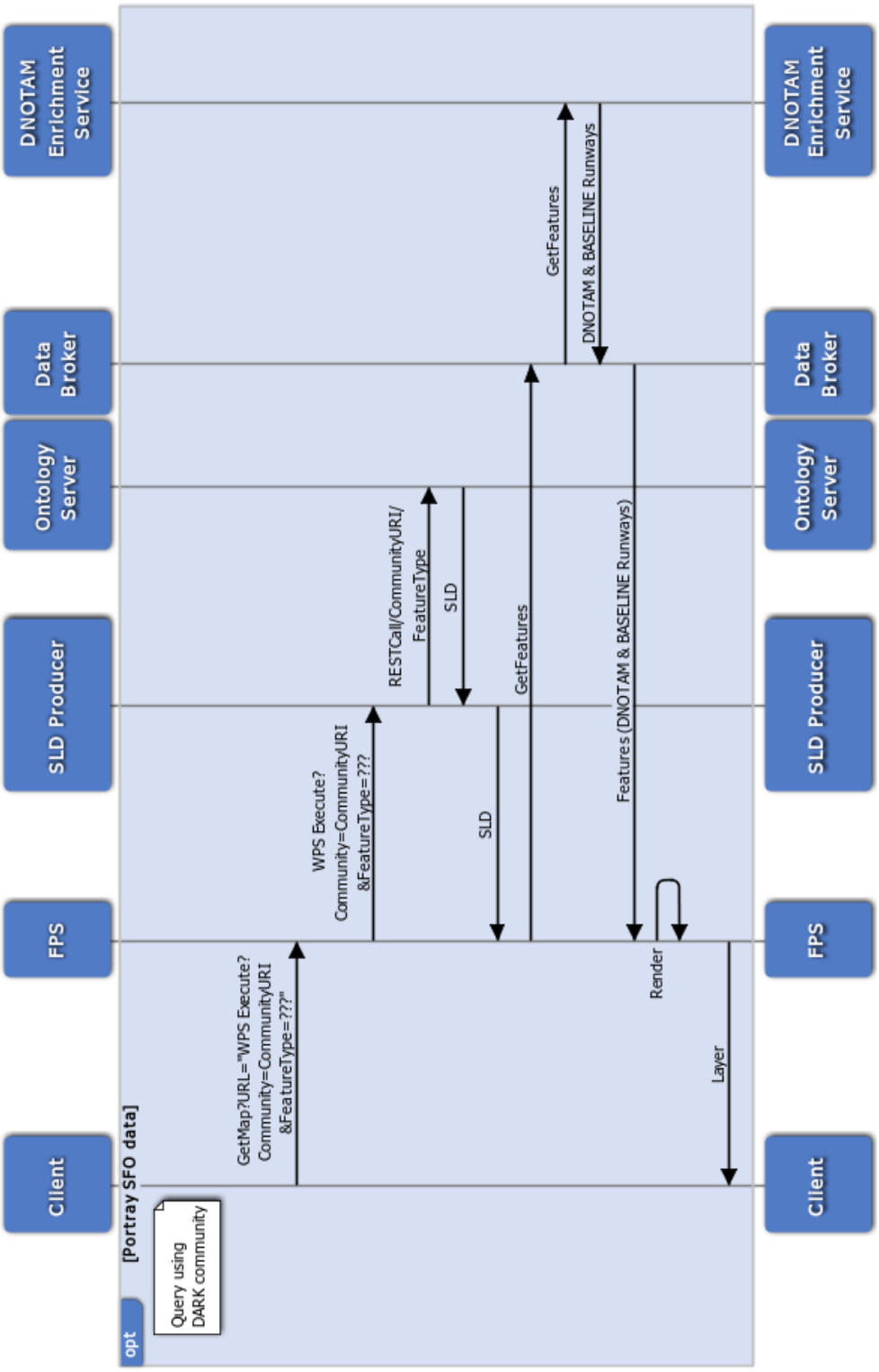


Figure 21 – Use case 3 – backend interactions

Annex A: Revision History

Date	Release	Editor	Primary clauses modified	Description
2015-05-13	1.0	Johannes Echterhoff	all	first complete version of the ER
2015-Jun-10	N.A.	Carl Reed	Various	Prepare for publication