Open Geospatial Consortium

Date: 2011-11-23

Reference number of this document: OGC 11-055

Category: Public Engineering Report

Editor: Steve Miller

OGC® Special Activity Airspace (SAA Pilot Study

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. This document 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 :	Public Engineering Report
Document subtype:	NA
Document stage:	Approved for public release
Document language:	English

Preface

This public Engineering Report (ER) is a deliverable of the Open Geospatial Consortium (OGC) Special Activity Airspace (SAA) Pilot study, an Interoperability Program proof-of-concept implementation supported by the U.S. Government Federal Aviation Administration (FAA) Aeronautical Information Management (AIM) organization. This document reflects contributions from all members of the SAA Pilot study. The document describes the technical architecture and components that were defined and implemented in the study, and the findings and lessons learned by the Pilot study participants. It also includes discussion of the business value and utility of adopting OGC Web Services and other open standards for SAA data dissemination.

Comments and questions on the contents of this report can be addressed to aviation-info@opengeospatial.org.

License Agreement

Contents

Page

Tables

Page

Figures

Page

1 Introduction

1.1 Scope

This OGC® document describes the architecture used for the implementation of the SAA Dissemination Pilot Study demonstrations. This includes an overview of the implemented components and workflows, and discussions of lessons learned.

1.2 Document contributor contact points

The following individuals contributed to this document:

Name	Organization
Alameh, Nadine	Open Geospatial Consortium, Inc.
Brackin, Roger	Envitia Ltd.
Burggraf, David	Galdos Systems Inc.
Dries, Jeroen	Luciad NV
Echterhoff, Johannes	International Geospatial Services Institute (iGSI) GmbH
Everding, Thomas	University of Műnster – Institute for Geoinformatics
Grothe, Christian	Lufthansa Systems FlightNav AG
Landry, Glen	Center for Naval Analysis (CNA)
Lew, Kevin	Center for Naval Analysis (CNA)
Miller, Steve	ConceptSolutions, LLC
Painter, Ian	Snowflake Software Ltd
Perkins, Jim	FAA System Operations Airspace and AIM Office
Wilson, Debbie	Snowflake Software Ltd

Table 1-1. Engineering Report Contributors

1.3 Revision history

Date	Release	Editor	Clauses modified	Description
2011-03-11	0.1	Steve Miller		Initial draft version.
2011-03-30	0.2	Steve Miller		Restructured and added content from Wiki.
2011-05-06	0.3	Steve Miller		Restructured and added content from Wiki.
2011-05-10	0.4	Steve Miller	5.2, 5.3, 7.2, 8.11	Updates from Thomas Everding
2011-05-16	0.5	Steve Miller	1.4, 7.1	Further updates from Thomas Everding
2011-06-03	0.6	Steve Miller		Final draft incorporating contributions and edits received from Pilot Study participants
2011-06-15		Steve Miller		Edits from final draft review.
2011-11-27		Carl Reed		Final edits prior to publication

Table 1-2. Document Revision History

1.4 Future work

The following future work items for the use of OGC-based components for SAA data dissemination have been identified in the conduct of this study:

1. **OSGi bundling of OGC service components.** The relevant service components need to actually be deployed on the same FUSE Enterprise Service Bus (ESB). This may require the re-engineering of existing component software. Section 8.12 provides more details on the Open Services Gateway initiative (OSGi) and FUSE ESB integration results as well as issues identified by the Pilot Study participants.

2. **OGC component integration with SAA SWIM services**. Several potential approaches to OGC service interoperability with SAA SWIM services could be explored to facilitate transition of SWIM clients to the use of OGC Web Services. Existing service adapters could be modified to exchange messages with the SWIM services, and stored queries in the WFS could be used to present an OGC-standard facade for SWIM service operations,

3. **Use of OGC components to directly access FAA data stores.** Rather than creating local copies of SAA data, the WFS and Event Service components could be adapted to use data from existing FAA repositories

4. **FAA standards for SAA data portrayal**. Depending on the user role, such as military versus civilian flight planner, the client software could use Style Registry metadata to present customized formats to facilitate visual recognition of the feature properties and status.

5. **Use of Style Registry to customize cross-section display content.** The WPS used for the Envitia client currently does not support parsing an SLD to govern how the cross-section image is styled.

6. **Automation of business rules for SAA dissemination in AIXM 5.1**. Schematron expressions could be used to guard the integrity of AIXM-format messages that use SAA schema extensions, and to enforce constraints and conventions for timesheet usage. This would entail business rule development, test and integration with the OGC services.

7. **Event Service enhancements.** A dedicated getServerStatus operation, rather than broadcasting heartbeat messages, could be specified and realized (see section 8.8). The result could be shared with the OGC Architecture DWG and the OWS Common SWG, and used with other web services.

8. **FAA source data standardization.** The current encoding of airspace activations in the SUA Gateway feed contains the upper and lower altitude limits in ‘flight level’, whereas the baseline airspace definition altitudes are usually expressed in ‘feet’. For this pilot study, harmonization of these and other differences were not fully addressed. SAA source data needs to be standardized to meet the level of data consistency required for productive use.

1.5 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 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.

1. AIXM 5.0 and AIXM 5.1 (http://www.aixm.aero)
2. ISO 19117:2005 – *Geographic Information- Portrayal*
3. OASIS *OASIS/ebXML Registry Information Model v3.0*
4. OGC 05-78r4, *Styled Layer Descriptor Profile of WMS (SLD-WMS)*
5. OGC 06-121r3, *OpenGIS® Web Services Common Standard*
6. OGC 07-006r1 *OGC Catalog Service Implementation Specification 2.0.2*
7. OGC 09-025r1 – *WFS 2.0 Standard*
8. OGC 09-026r1 – *Filter Encoding Specification 2.0*
9. OGC 09-032, OGC® OWS-6 *SWE Event Architecture Engineering Report*
10. OGC 09-050r1, OGC® OWS-6 *Aviation Engineering Report*
11. OGC 10-060r1 OWS-7 *Event Architecture Engineering Report*
12. OGC 10-079r3 – OWS-7 *Aviation Architecture Engineering Report*
13. OGC 10-127r1 OWS-7 *Aviation Portrayal Engineering Report*
14. OGC 10-130 OWS-7 *Aviation – FUSE Deployment Integration Report*
15. OGC 10-131, OpenGIS® OWS-7 *Aviation – AIXM Assessment Report*
16. OGC 11-060, *Use of GML in Aeronautical Data* discussion paper
17. OGC 11-073, OWS-8 *WFS Guidance Engineering Report*

This document also includes several SAA-specific AIXM schema extensions listed in Annex A.

3 Terms and definitions

Several aeronautical terms are used in this ER to describe the SAA Pilot Study activities.

- AIM

The FAA Aeronautical Information Management (AIM) group is the authoritative government source for collecting, validating, storing, maintaining, and disseminating aeronautical data concerning the United States and its territories to support real-time aviation activities.

- AIXM

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

- NOTAM

Notice To AirMen are text-based messages sent to flight crews to notify them about changes to conditions or equipment relevant to their flights. NOTAMs are also used to inform other people involved with flight operations such as controllers or dispatchers.

- Digital NOTAM (dNOTAM)

Digital NOTAM messages are encoded as AIXM 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 filter expressions to retrieve only those updates that affect a specific flight.

- Airspace

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

- Special Use Airspace (SUA)

An airspace designated by the FAA for use by military or other agencies, or having other reasons to exclude general air traffic. SUA include Alert, Prohibited, Restricted, and Warning Areas, as well as Military Operations, Controlled Firing, and National Security Areas.

- Special Activity Airspace (SAA)

A term used to refer to all types of airspace and routes requiring FAA coordination and management. including SUA, Military Training Routes (MTR), and Air Traffic Control Assigned Airspace (ATCAA) used by FAA controllers to help direct and manage air traffic.

- SAA definitions

Airspace definitions are published in advance of the date that they will go into effect, which is synchronized across FAA systems on a 56-day Aeronautical Information Regulation and Control (AIRAC) update cycle. SAA definitions are considered “static data” as changes or updates are relatively infrequent.

- SAA reservations

SAA schedule requests are submitted to the FAA in advance by military air traffic control specialists who plan the missions and special operations that will take place inside the airspace. These requests include the start time, end time, and altitude boundaries expected to be used. The controlling facility for the SAA, usually an FAA Air Traffic Control Center, confirms that the scheduled reservation will not conflict with other planned activities, and can approve, modify or disapprove the schedule request. The military agency submitting the schedule request may also modify the request before the planned start time, or may cancel the request when the SAA is no longer needed. SAA reservations are considered "dynamic data" as they change much more frequently than the SAA definitions.

- AIXM temporality

AIXM includes a temporality model to represent the states and events of aeronautical features which may change over time. A distinction is made between permanent changes and temporary status. A baseline defines the initial state of a feature, such as when the feature came into existence. Changes to a baseline are defined as deltas, and are either permanent or temporary. These deltas contain information on the changes, and the relevant feature ID.

More information on the AIXM data model can be found at http://www.aixm.aero/public/subsite_homepage/homepage.html.

4 Conventions

4.1 Abbreviated Terms and Acronymns

AIM	Aeronautical Information Management
AIRAC	Aeronautical Information Regulation and Control
AIXM	Aeronautical Information Exchange Model
ARTCC	Air Route Traffic Control Center
ATCAA	Air Traffic Control Assigned Airspace
ATM	Air Traffic Management
CEP	Complex Event Processing
COI	Community of Interest
CRS	Coordinate Reference System
CSW	Catalog Service for Web
dNOTAM	digital NOTice to AirMen
DoD	Department of Defense
EML	Event Pattern Markup Language
ER	Engineering Report
ES	Event Service
ESB	Enterprise Service Bus

ESP	Event Stream Processing
FAA	Federal Aviation Administration
FES	Filter Encoding Specification
FPS	Feature Portrayal Service
GeoJSON	Geographic JavaScript Object Notation
GML	Geography Markup Language
GMT	Greenwich Mean Time
GNU	GNU's Not Unix
GPL	General Public License
HTTP	HyperText Transport Protocol
ID	Identifier
IfGI	Institute for Geoinformatics (University of Münster)
ISO	International Standardization Organization
JMS	Java Messaging Service
KML	Keyhole Markup Language
KVP	Key-Value Pair
MOA	Military Operations Area
MTR	Military Training Route
NAS	National Airspace System
NextGen	Next Generation Air Transportation System
NOTAM	NOTice To AirMen
OASIS	Organization for the Advancement of Structured Information Standards
OGC	Open Geospatial Consortium
OSGi	Open Services Gateway initiative
OWS	OGC Web Services
OWS-7	OWS testbed phase 7
SAA	Special Activity Airspace
SAS	Sensor Alert Service
SE	Symbology Encoding
SLD	Styled Layer Description
SOA	Service Oriented Architecture
SOAP	Simple Object Access Protocol
SQL	Structured Query Language
SUA	Special Use Airspace
SWE	Sensor Web Enablement
SWIM	System Wide Information Management

UCUM	Unified Codes for Units of Measure
URL	Uniform Resource Locator
UUID	Universally Unique Identifier
VFR	Visual Flight Rules
WAR	Web application ARchive
WFS	Web Feature Service
WFS-T	Web Feature Service –Transactional
WMC	Web Map Context
WMS	Web Map Service
WS-I	Web Services Interoperability
XML	Extensible Markup Language
Xpath	XML Path Language

Table 4-1. Abbreviations and Acronyms Used in this Document

5 SAA Pilot Component Architecture Overview

The System Wide Information Management (SWIM) program within FAA is a National Airspace System (NAS)-wide initiative that supports the Next Generation Air Transportation System (NextGen) goals. The goal of SWIM is to achieve systems interoperability and information management for diverse Air Traffic Management (ATM) systems, platforms and software implementations through the realization of a Service-Oriented Architecture (SOA).

The FAA SAA Dissemination Pilot will extend the benefits of the SAA SWIM services by enabling interoperability with external stakeholders. The Pilot Study objectives include:

- Focus on dissemination of SAA information (including updates) to users via OGC Web Services
- Promote the implementation of adapters to support external stakeholders
- Expose SAA information services to the NAS stakeholders with emphasis on airlines to automate flight dispatch and planning
- Develop an architecture that can accommodate future requirements.

SAA dissemination involves the access, filtering and graphical portrayal of airspace information, as well as the ability for subscribed users to receive notifications of SAA updates and reservation schedule changes. The ultimate goal is to use international geospatial standards and services to provide end-to-end information flows from Department of Defense (DoD) originators of SAA reservation requests, to airlines and other NAS stakeholders. Achieving that goal will lead to improved compliance, utilization, safety and efficiency of DoD and FAA cooperation on the use of SAA.

As part of the FAA SWIM Segment 1 activities, the Aeronautical Information Management (AIM) Community of Interest (COI) is developing the capability to exchange static and dynamic Special Activity Airspace (SAA) information between operational air traffic management systems. The Segment 1 focus is on the storage, dissemination and scheduling military reservations for the SAAs. Future segments will focus on the design, approval, and tracking of SAAs. From a technical perspective, the SAA SWIM web services are SOAP over HTTP synchronous request/response web services, the Java Messaging Service (JMS) web services for asynchronous event notifications, and the SWIM Web Services Interoperability Organization (WS-I) Basic Profile. Aeronautical Information Exchange Model (AIXM) version 5.0, which is based on Geography Markup Language (GML), was adopted as the standard data format to facilitate Service Oriented Architecture (SOA) message exchange among SWIM data stores and systems.

The Pilot Study intention to implement OGC Web Service interfaces to the SAA SWIM service endpoints (seen in Figure 5-1) was amended in response to the unavailability of Internet access to the SAA SWIM services. Instead, the Pilot Study software components used file-based SAA definition and reservation data. SAA definitions, representing the content provided by the Static Repository SWIM services, are provided as a collection of “subscriber files” containing airspace definition records in AIXM 5.0 format. SAA reservations, representing the content provided by the Operational Repository SWIM services, are obtained via access to a web service providing records of pending and active airspace reservations in HTML format. The OGC Web Feature Service (WFS) and Event

Service (ES) components translate the FAA-provided data into AIXM 5.1 format messages for dissemination to the client software for display (as shown in Figure 5-2).

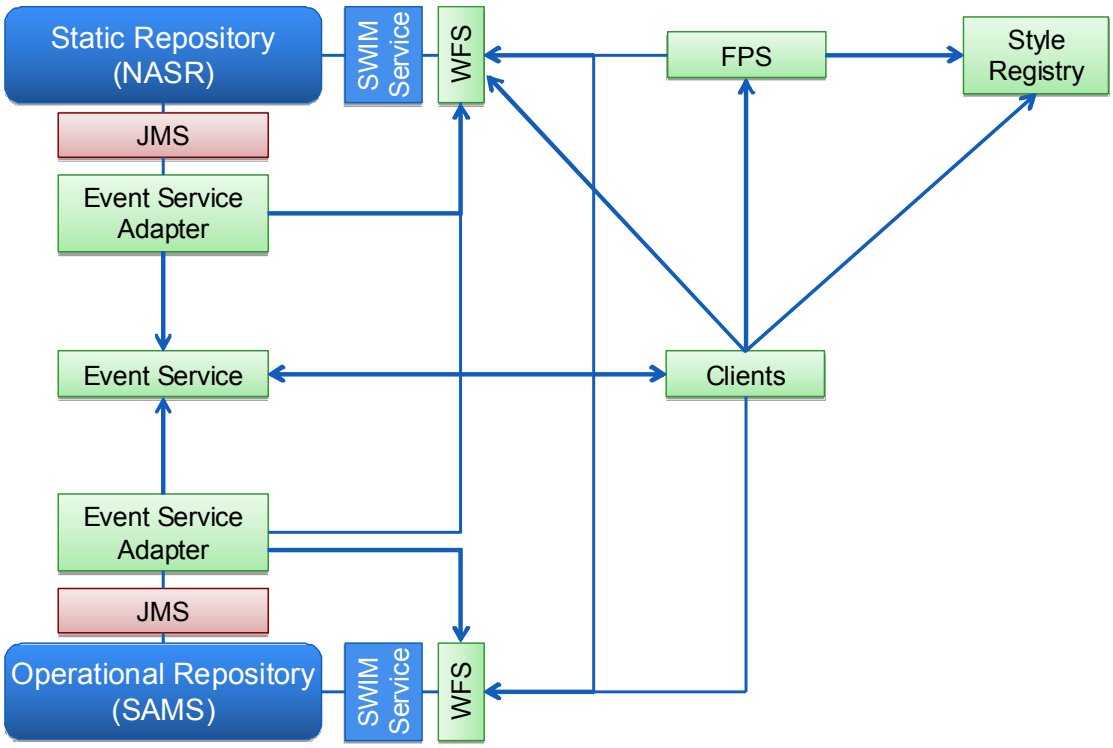

Figure 5-1. Original SAA Pilot Study Engineering Approach

As a substitute for the schedule event messages that would have been provided by the SWIM Operational Repository services, the SAA Pilot Study participants received access to an FAA web site that publishes the current set of pending and active airspace schedules each minute in HTML format. This web site is shown as "schedule feed" in Figure 5-2, and discussed further in section 7.2.

In order to identify airspace definition changes, the current and the upcoming data sets would have to be compared to identify their differences (shown as "Subscriber file diff" in Figure 5-2). These periodic updates to the airspace definitions were not implemented for the SAA Pilot Study.

The types of OGC components used in the Pilot Study and their relationships are shown below in Figure 5-2, and are briefly described in the following sections.

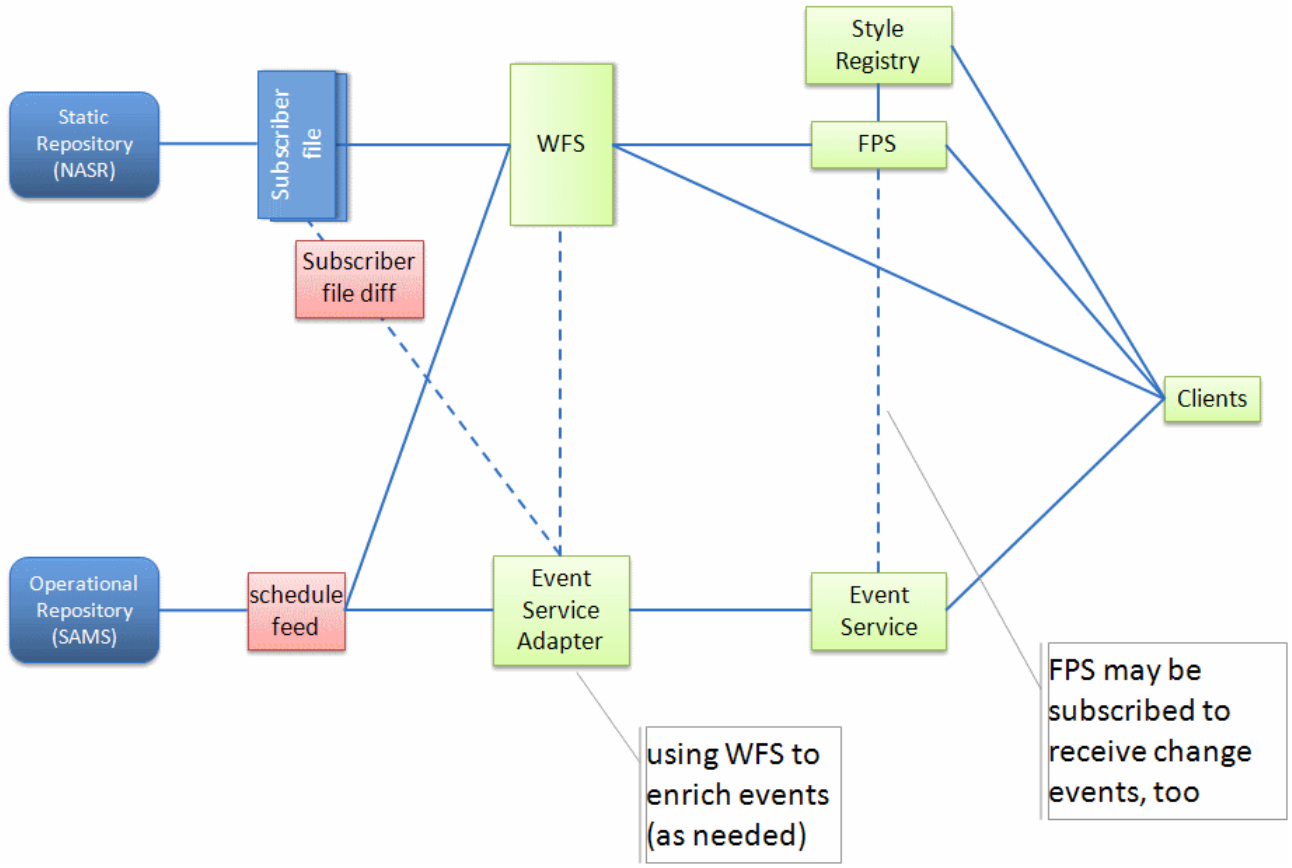

Figure 5-2. SAA Pilot Study OGC Components

5.1 Web Feature Service

A WFS (OGC 09-025r1) is a stateless request/response web service involving XML messages encoded using a GML application schema such as AIXM 5.0 or AIXM 5.1. The WFS supports HTTP GET/POST or SOAP bindings. The WFS is analogous to existing SAA request/response services but provides a more comprehensive set of operations and querying capabilities. These WFS 2.0 operations allow clients to discover, access and maintain data within a remote data store. The stored query operations, new to the WFS 2.0 specification provide significant benefits as WFS administrators can pre-define common complex queries that can reduce implementation barriers for client application development.

Query operations are underpinned by the OGC Filter Encoding 2.0 specification (OGC 09-026r1) that defines a set of filter expressions that enable fine-grained retrieval of data via any OGC web services. The WFS query operations also allow clients to perform:

- Single or multiple queries
- Join queries (optional): property, spatial, temporal
- Resolve: to include referenced features in the response
- Coordinate reference system transformation (optional)
- Define how the response is returned: Sort order or alternate response (hits) = number of features corresponding to query
- Exclude unwanted optional properties in response

The WFS enables users to perform a wide range of queries from simple queries:

- Get me all features within a bounding box
- Get me the Airspace with the identifier: eafcbdff-e8cc-4afd-beb3-3c08883b7b1d

to more complex queries:

- Get all active Airspaces within 200 nautical miles of a route
- Get the names of all military Units operated by US Navy

5.2 Event Service

As part of the OGC Web Services (OWS) initiative, the OGC has developed a service specification for event notifications based on the OASIS Web Services Notification standard. An Event Service allows users to automatically receive notifications about feature-related events, which represent updates to feature property values (state changes). This is analogous to the asynchronous SAA Service publish/subscribe (pub/sub) topics implemented using the Java Messaging Service (JMS) standard, but without vendor-specific JMS binding issues. In addition, the Event Service subscription can include filter criteria for pre-defined event topics as well as filters that operate on the spatial, temporal and/or thematic properties of events, much like the WFS filters.

Details on the Event Service implementations used in the Pilot Study can be found in section 7.1. The encoding of the events used in the Pilot Study is also described in section 7.1. The Event Service design also includes a standard "heartbeat" channel to allow subscribers to confirm the continuing availability and connectivity to the Event Service (see section 8.8).

5.3 Event Service Adapter

The Event Service Adapters, provided by Snowflake Software and IfGI, act as the data source for the OGC Event Service. The Event Service adapters integrate information from the SUA reservation schedule data feed with data derived from the BASELINE data from the SAA WFS. The Event messages were augmented with additional metadata to enable:

- **Spatial filtering within subscriptions:** by populating the bounding box geometry of airspace reservation change events generated by the Event Service Adapter
- **Identification of Event publisher:** by adding ISO encoded metadata in the Airspace timeslice

SAA airspace reservation changes (schedule events) are obtained from periodic polling and analysis of schedule status listings provided by an FAA website. An Event Service Adapter is used to infer the occurrence of an SAA event by comparing the corresponding "schedule ID" values across two sequential listings and identifying when each airspace reservation request has been submitted, approved or disapproved, and cancelled or completed. The Schedule Event Service Adapter will check for SAA reservation updates on the FAA website, integrate the information with the airspace definitions from the subscriber file (provided via WFS), and forward the event messages to the Event Service for dissemination to the application clients. Details on the available implementations can be found in section 7.2.1 (provided by IfGI), and section 7.2.2 (provided by Snowflake Software).

5.4 Style Registry

The Style Registry is an implementation of the OGC Catalog Service Implementation Specification 2.0.2, based on the OWS-7 design of an aviation portrayal registry model. The OGC Catalog Service

supports the ability to publish and search collections of descriptive information (metadata) for data, services, and related information objects. This metadata is encoded using the OGC Styled Layer Descriptor (SLD) standard to enable its use by Feature Portrayal Service (FPS) components. The Style Registry is used in the SAA Pilot Study to catalog, manage, discover and access symbols, and styles as a repository of map styling used by the FPS to support SAA geospatial data display by the OGC Application Client components.

5.5 Feature Portrayal Service

The Feature Portrayal Service (FPS) is a type of OGC Component Web Map Service (WMS) that can symbolize feature data obtained from one or more remote WFS components. The term FPS is defined in the Styled Layer Descriptor Profile of WMS (SLD-WMS), OGC Specification 05-78r4, and describes a Web Map Service capable of receiving an SLD Document. The specification actually describes two types of SLD-WMS, the Integrated WMS that is connected to a database, and the Component WMS (or FPS) which is capable of styling data from a compliant WFS. The latter is more flexible and is the type used in this trial.

An FPS allows for flexible binding of arbitrary GML data sources and applicable styling rules for inclusion in a client display. Typically, an FPS has the following characteristics:

- It has no pre-defined ‘named’ styles or layers (i.e. it acts as a portrayal engine rather than a data source).
- It only supports the WMS (OGC standard) interface.
- It can symbolize feature data from any compliant WFS component or GML data.
- It supports user-defined styles, symbols, and map layers.

Styles, encoded using OGC Symbology Encoding (SE), describe styling attributes that can be applied to particular features in the portrayal process. Symbols are generic graphical entities referenced in styles and used by the FPS in the styling process

One of the objectives of the FAA SAA Dissemination Pilot is to lower the barrier for client use of SAA information by leveraging the OGC separation of Portrayal Rules, Symbol Sets, and Portrayal specifications from underlying geospatial data. This allows users to define and apply different styles/symbols to the same feature data depending on the chosen styling rules, which may have different styles or conventions based on the type of Client or User Community. The separation is achieved by using the FPS as a portrayal engine for the SAA geospatial data to allow the generation of customizable maps with localized standards for symbology (styles and symbols) and portrayal rules.

5.6 Application Client

OGC-compatible application clients provide a user interface for the retrieval and display of geospatial features and their properties that are obtained from WFS and ES components. Vendors of several commercial OGC client applications participated in the Pilot Study to show how SAA information could be presented to the aircraft pilot or air traffic controller.

Commercial products from Envitia, Luciad, and Lufthansa Systems participated in the Pilot Study technical interoperability exercises to demonstrate how OGC standards facilitate the dissemination of information among software products developed by different vendors. Typically two general forms of client exist; the thick or rich client, typically developed using a high level language to implement

significant client-side functionality (such as the Luciad and Lufthansa Systems products), and light or browser-based clients such as Envitia, developed using JavaScript and HTML to exploit functionality in the service platform.

6 SAA Dissemination Scenarios

An operational Commercial Flight scenario was developed to demonstrate OGC component capabilities to disseminate SAA information and events. The component capabilities were described in a number of Use Cases. Additional scenario topics, queries and filters that could also be used to demonstrate features of the SAA Pilot architecture are provided in Annex E.

6.1 WFS Use Cases

- **WFS UC 1** (spatial bounding box filter)
  The WFS is queried for SAAs using a spatial filter expression. The filter expression consists of a rectangle (bounding box) that represents the query area.
  - A SAA set is returned that contains all SAAs, the horizontal extent of which intersects with the query polygon. Each SAA record in the result set includes the activity schedule.

- **WFS UC 2** (coarse flight route filter)
  The WFS is queried for SAAs using a spatiotemporal filter. The filter contains a polygon for the (horizontally) buffered flight route and a temporal interval for the time of the flight (from departure to arrival).
  - A SAA set is returned that contains all SAAs with the following properties: (1.) The horizontal extent spatially intersects with the submitted polygon, and (2.) some entry in the activity schedule intersects with the temporal interval. Each SAA record in the result set includes the activity schedule.

- **WFS UC 3** (fine trajectory filters)
  The WFS is queried for SAAs using a spatiotemporal filter expression. The filter expression consists of a disjunction of filters, each being an individual spatiotemporal filter for one leg of the flight trajectory. Each spatiotemporal filter consists of a spatial region (represented as polygon) and a temporal interval.
  - A SAA set is returned that contains all SAA's with the following properties: (1) The horizontal extent spatially intersects with at least one of the airspace polygons in the individual filters, and (2) for the same airspace, there is an entry in the activity schedule that intersects with the temporal interval of the same spatiotemporal filter. Each SAA record in the result set includes the activity schedule.

- **WFS UC 4a** (query by airspace feature property value)
  A military user wants to identify an airspace under their command that has altitudes that meet their mission requirements. They want to filter based on a specific value of the unit name – for example MCGUIRE AFB.
  - A SAA set with matching unit name is returned.

- **WFS UC 4b** (query for airspace property values)
  The military user wants to receive the upper and lower altitudes for the base geometry

component of the returned airspace.
- A SAA set with default upper and lower altitude limits for the SAA is returned.

- **WFS UC 5** (Buffered lateral spatial flight route filter)
  The WFS is queried for SAAs using a spatial filter expression. The filter expression consists of a "NOT BEYOND" operator applied to a linestring representing the lateral route of a flight and a distance (specified as x nautical miles).
  - A SAA set is returned that contains all SAAs, the horizontal extent of which intersects with the area resulting from applying a buffer of x nautical miles to the flight route. Each SAA record in the result set includes the activity schedule.

6.2 Event Service Use Cases

- **ES UC 1a** (flight route **active schedules** subscription)
  A subscription for a buffered flight route (filter expression consists of a "NOT BEYOND" operator applied to a linestring representing the lateral route of a flight and a distance) and temporal interval is registered with the Event Service, corresponding to the Operational Repository Active Schedules JMS topic.
- **ES UC 1b** (airspace **active schedules** notification)
  Upon the approval of an airspace reservation request, where the horizontal extent of the airspace intersects with the area representing the buffered flight route from a previously registered subscription and the reservation interval intersects within the subscription time interval, the subscriber is notified of the new active schedule.
- **ES UC 2a** (airspace **schedule event** subscription)
  A subscription for schedule changes for one or more airspaces is registered with the Event Service. This subscription is made on the SAA schedule event channel corresponding to the SWIM Operational Repository JMS schedule event topic.
- **ES UC 2b** (schedule **submission** notification)
  Upon the submission of an SAA reservation request, for which a subscription was previously registered, the subscriber is notified of the new schedule reservation request (Status=Pending) for that airspace.
- **ES UC 2c** (schedule **approval** notification)
  Upon the FAA approval of an SAA reservation request in the future, for which a subscription was previously registered, the subscriber is notified of the new schedule reservation (Status=Approved/Waiting) for that airspace.
- **ES UC 2d** (schedule **activation** notification)
  When the schedule reservation start time for a Waiting schedule occurs, for which a subscription was previously registered, the subscriber is notified of the activation of the reservation request (Status=Active/Hot) for that airspace.
- **ES UC 2e** (schedule **cancellation** notification)
  Upon the submission of an SAA reservation cancellation, for which a subscription was previously registered, the subscriber is notified of the cancelled reservation request (Status=Cancelled/Cold) for that airspace. According to the description of the FAA SUA Gateway feed, http://sua.faa.gov/ops/docs/suagwDataFmt.html, the status 'cold' does not exist. The cancellation can be detected by not finding the entry anymore when the schedule reservation end time did not run out.

- **ES UC 3** (unsubscription)
  A subscriber cancels an active subscription for SAA schedule events at the Event Service.
- **ES UC 4** (SAA **definition** subscription)
  A subscription for updates of the subscriber file for airspace definitions is made. This subscription is made on the SAA definition update channel (corresponding to the SWIM Static Repository JMS topic).
- **ES UC 5** (SAA **definition** notification)
  The SAA Universally Unique Identifier (UUID) for a changed SAA definition is sent to the subscriber. To obtain the details, the subscriber then submits a WFS query for the SAA definition using the SAA UUID.
- **ES UC 6a** (Event Service **heartbeat** subscription)
  A subscription for periodic heartbeat notifications is made. This subscription is made on the SAA **heartbeat** channel.
- **ES UC 6b** (Event Service **heartbeat** notification)
  The subscriber receives an Event Service heartbeat notification, indicating the Event Service is still in operation.

6.3 Commercial Flight Scenario

This scenario describes a domestic commercial flight from KORD to KLAX that uses:

- SAA Web Feature Service (WFS) for flight planning by retrieving relevant SAAs based on a spatial-temporal filter. The flight crew visually inspects the route on their enroute chart, checking SAA reservations along the route and their activation times. Airspaces are color coded according to activation times, to facilitate planning.

- SAA Event Service (ES) for monitoring of SAA status while en route. The flight crew receives notification of SAA activation that may impact their flight plan.

- WFS for flight re-planning while en route. The flight crew modifies their flight plan, using the WFS to identify SAA locations and scheduled activation, as they did in the initial flight planning stage.

6.3.1 Flight Planning

Dispatch personnel plans a flight from KORD to KLAX by first generating a route from departure to destination, then checking restrictions along the route, which involves checking for airspaces that are active at the time the aircraft would pass them. The optimized flight route is then released as part of an electronic briefing package for the cockpit crew. (WFS UC 3)

SAA on flight path: ADA EAST MOA, ADA WEST MOA - FL70-B180
Scheduling agency: McConnel AFB, 184th Tactical Fighter Group
Controlling agency: ZKC Kansas City ARTCC

6.3.2 Flight Preparation

The cockpit crew loads the briefing package into their cockpit systems. To get familiar with the route, they visually inspect the route on their enroute chart, which involves checking SAA reservations along the route and their activation times. (WFS UC 2, ES UC 1a/2a)

6.3.3 Flight Conduction

During the course of the flight, the activation schedule for an airspace on the route is changed. The Dispatch Office is notified by the ARTCC, and radios the change to the flight. The crew checks the new activation schedule and location of the airspace and finds that it does not directly affect the flight. (ES UC 1b/2b)

6.3.4 In-Flight Replanning

Severe weather is encountered and the flight has to be diverted. For planning a diversion route, the crew uses the client application, which displays on the enroute chart all SAAs, represented differently (e.g., colored) depending of their activation time (e.g., currently active / becomes active in 2/4/8 hrs). The crew plans a diversion route avoiding active airspaces. (WFS UC 1, ES UC 3, ES UC 1a/2a)

6.3.5 After Flight

The subscription is removed. (ES UC 3)

6.4 Web Registry Service Demonstrations

The Web Registry Service demonstrations include publication, discovery and life-cycle management capabilities, described briefly below. More information on the Web Registry Service design is provided in the OWS-7 publication *Aviation Portrayal ER* (10-127r1).

Discovery

Search for datasets by bbox (bounding box) and keyword (e.g. AIXM)
Retrieve dataset metadata of interest from search above
Discover styles (SLDs) and services (Envitia and Luciad FPS) associated to the dataset found above
Retrieve two different SLD docs - view as html table legend
Retrieve FPS capabilities (including service endpoints and capabilities)

Dataset Metadata

AIXM Metadata for datasets in GML/ISO 19139 encoding

Service Metadata

determine the endpoints and capabilities of services that provide AIXM data in a given area of interest, including FPS and/or WFS

Portrayal Data

get SLDs relevant to a chosen dataset
get Symbols

Data loading (publication)

load dataset metadata
load FPS capabilities and endpoint urls
load SLDs/Symbols

Lifecycle Managment

View audit trail related to SLDs and/or symbols (versions submitted/changed/updated)

Creation/Publishing

Create ebRIM model using Designer client
Publish model to Registry

Additional Functionality

notification of changes in registry (e.g. to inform FPS of new SLDs)

6.5 Demo Data Considerations

In preparing for the scenario demonstrations, two ways to provide sample data to drive the demo events were considered. Updates, at least to the dynamic data, could be simulated via a special demo schedule feed that the WFS and Event Service Adapters would access. With this approach, relevant demo events are triggered by a pre-defined schedule feed. This would require:

a separate custom feed endpoint

functionality for client providers to perform update of schedule feed upon request (exact content of feed entries to achieve desired events would need to be defined)

higher frequency of scans by ES Adapter during demo (to pick up changes immediately, since waiting several minutes to continue in the demo would not be feasible)

having the ES Adapter set to the time that is used in actual demonstration (because the ES Adapter detects hot activations based upon current time), which could be different based upon the timeframe that best suits the demo scenario.

Figure 6-1 illustrates the use of a demo schedule feed:

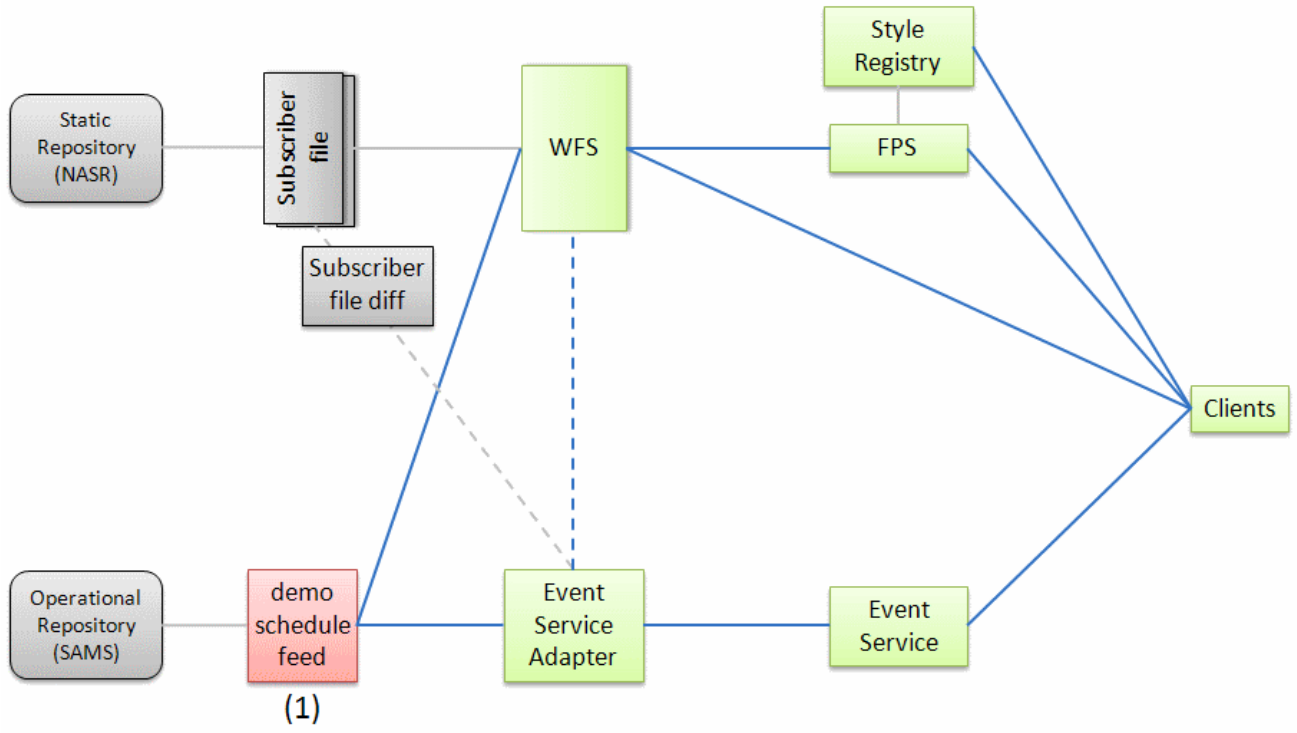

Figure 6-1. Scripted Airspace Reservation Schedule Updates

The advantage of this approach would be that the whole component chain is demonstrated, and that the demo schedule feed would automatically trigger events to be sent to the Event Service, and also insert the changes into the WFS.

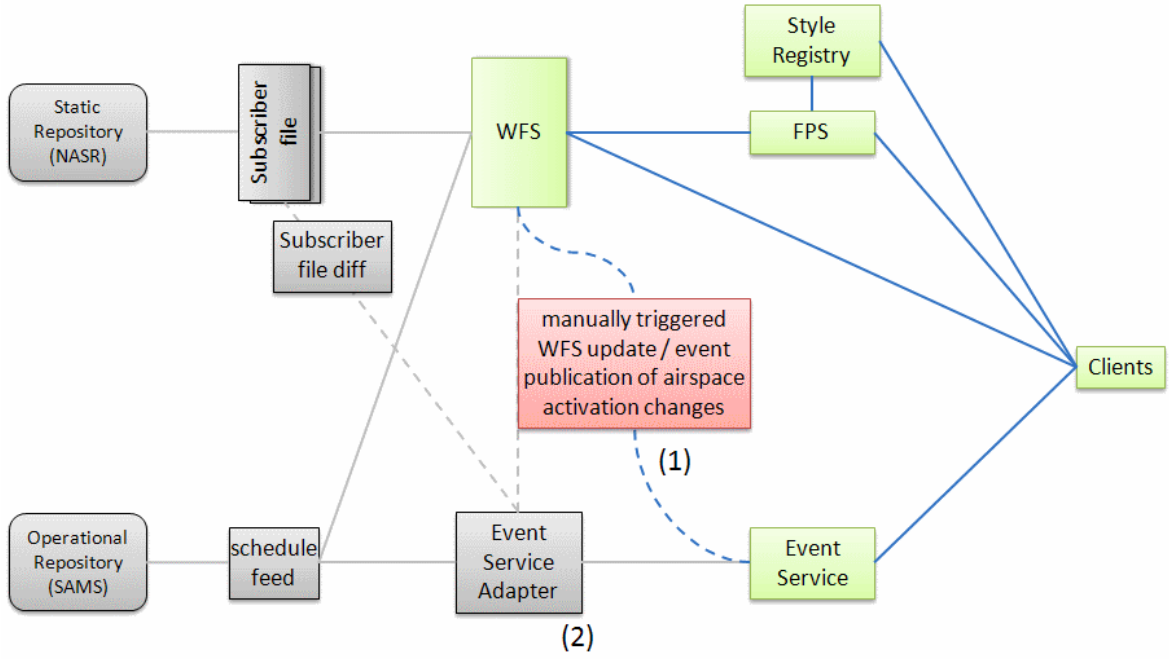

Figure 6-2. Manual triggering of WFS update/event

An alternative approach would be to use a tool to manually send the necessary events to the Event Service and to update the WFS, as seen in Figure 6-2. With this approach, simulated airspace activation changes are triggered manually, which has the following advantages:

- changes can be triggered as needed
- no need for components to synchronize and reset their state before running the demo

This would also require that the demo Event Service Adapter functionality be validated through additional testing.

To minimize the need for synchronizing timestamps among different sets of demo data for use by different components, the latter approach of manually generating event messages was adopted to demonstrate SAA data dissemination in the Pilot Study. An additional Event Service Adapter Tester component was developed to ensure that the adapters work as expected. Details of this additional component can be found in section 7.2.3.

7 SAA Pilot Components

This section describes the OGC components implemented for the SAA Pilot Study.

Note: a summary of the standards relevant for the implementation of these components is provided in chapters three and four of the Request For Quotation And Call For Participation In the FAA SAA Dissemination Pilot – Annex B

7.1 Event Service Implementation

The events on schedule changes are designed as thin events. This means that they contain only the new information, in this case the information on the scheduled activation. The information on the base feature has to be accessed at a Web Feature Service.

In contrast to thin events, a thick event would contain the base feature in addition to the activation information. The main differences are that the thin event saves bandwidth and reduces communication load, whereas the thick event provides a more complete basis for further processing and filtering tasks. In this section, only the agreed event encoding for the Pilot Study is described. A general discussion of thin and thick events, intermediate types, event enrichment as well as filtering and processing options will be provided in the OGC OWS-8 Aviation thread engineering reports.

The Aeronautical Information Exchange Model is an XML based encoding for aeronautical features (like runways, navigation aids, and many more). It is developed by the Federal Aviation Administration (FAA) and Eurocontrol. Since version 5.0 it is based on GML 3.2.1 and represents the aeronautical features as extensions of GML features. To represent varying changes to feature properties over time such as runway closures, the AIXM features are extended with a temporal validity model based on timeslices. In addition to basic feature propertie, timeslices are used to encode temporal and permanent changes to the data, and keep track of these changes.

The support of event encoding in AIXM is one important aspect of effective SAA dissemination, and was the basis for the data format representation of digital NOTAMs (Notice To AirMen). A NOTAM is a codified message structure to notify pilots when conditions regarding their flight have changed following flight plan approval. The dNOTAM AIXM schema extension includes specific elements for the encoding of the digital NOTAMs. The SAA Pilot Study investigated the use of dNOTAM formats as encoding for more general aeronautical events, including changes to special activity airspace (SAA) reservations that could impact the flight. More information on the use of AIXM to encode events can be found in the OWS-6 and -7 Aviation Engineering Reports (09-050r1 and 10-079).

7.1.1 Event Encoding

The events used in the FAA SAA Dissemination Pilot are encoded as AIXM 5.1 basic messages (XML element AIXMBasicMessage). They include the Airspace as the only mandatory feature, listing the properties affected by the schedule update and the bounding box of the airpace. Optionally, the message may contain an Event element as defined in the dNOTAM event specification v0.6. In this case, the airspace information shall also contain a link to the Event element. The XML for an example event as produced by the Snowflake Event Service Adapter is shown below.

```
<?xml version="1.0" encoding="UTF-8"?>
<...>
    <event:Event gml:id="EVENT_2338666_W">
        <gml:identifier codeSpace="http://sua.faa.gov">2338666</gml:identifier>
        <event:timeSlice>
            <event:EventTimeSlice gml:id="EVENT_TS_2338666_67704">
                <gml:validTime>
                    <gml:TimePeriod gml:id="EVENT_VT_2338666_67704">
                        <gml:beginPosition>2011-04-15T18:00:00.000Z</gml:beginPosition>
                        <gml:endPosition>2011-04-15T20:30:00.000Z</gml:endPosition>
                    </gml:TimePeriod>
                </gml:validTime>
                <aixm:interpretation>BASELINE</aixm:interpretation>
                <aixm:sequenceNumber>1</aixm:sequenceNumber>
                <aixm:correctionNumber>0</aixm:correctionNumber>
                <aixm:featureLifetime>
                    <gml:TimePeriod gml:id="EVENT_FL_2338666_67704">
                        <gml:beginPosition>2011-04-15T18:00:00.000Z</gml:beginPosition>
                        <gml:endPosition>2011-04-15T20:30:00.000Z</gml:endPosition>
                    </gml:TimePeriod>
                </aixm:featureLifetime>
                <event:encoding>DIGITAL</event:encoding>
                <event:revision>2011-04-15T17:30:03.000Z</event:revision>
            </event:EventTimeSlice>
        </event:timeSlice>
    </event:Event>
</message:hasMember>
```

```
<message:hasMember>
    <aixm:Airspace gml:id="Airspace12a754f50-26c3-4824-8a22-4f1f8ead0cff">
        <gml:identifier codeSpace="http://www.faa.gov/nasr">2a754f50-26c3-4824-8a22-
4f1f8ead0cff</gml:identifier>
        <gml:boundedBy>
            <gml:Envelope srsName="urn:ogc:def:crs:OGC:1.3:CRS84">
                <gml:lowerCorner>-115.565 33.146</gml:lowerCorner>
                <gml:upperCorner>-114.926 33.544</gml:upperCorner>
            </gml:Envelope>
        </gml:boundedBy>
        <aixm:timeSlice>
            <aixm:AirspaceTimeSlice gml:id="SCHED_TS_2339714_67702">
                <gml:validTime>
                    <gml:TimePeriod gml:id="SCHED_VT_2339714_67702">
                        <gml:beginPosition>2011-04-15T18:00:00.000Z</gml:beginPosition>
                        <gml:endPosition>2011-04-15T23:50:00.000Z</gml:endPosition>
                    </gml:TimePeriod>
                </gml:validTime>
                <aixm:interpretation>TEMPDELTA</aixm:interpretation>
                <aixm:sequenceNumber>1</aixm:sequenceNumber>
                <aixm:timeSliceMetadata>
                    <gmd:MD_Metadata>
                        <gmd:contact gco:nilReason="withheld"/>
                        <gmd:dateStamp>
                            <gco:DateTime>2011-05-06T17:36:37.000Z</gco:DateTime>
                        </gmd:dateStamp>
                        <gmd:identificationInfo>
                            <gmd:MD_DataIdentification>
                                <gmd:citation gco:nilReason="inapplicable"/>
                                <gmd:abstract gco:nilReason="inapplicable"/>
                                <gmd:pointOfContact>
                                    <gmd:CI_ResponsibleParty>
                                        <gmd:organisationName>
                                <gco:CharacterString>Snowflake
Software</gco:CharacterString>
                                        </gmd:organisationName>
                                        <gmd:role>
                                            <gmd:CI_RoleCode
codeList="http://standards.iso.org/ittf/PubliclyAvailableStandards/ISO_19139_Schemas/re
sources/Codelist/ML_gmxCodelists.xml#CI_RoleCode"
codeListValue="publisher">publisher</gmd:CI_RoleCode>
                                        </gmd:role>
                                    </gmd:CI_ResponsibleParty>
                                </gmd:pointOfContact>
                                <gmd:language>
                                    <gmd:LanguageCode
codeList="http://standards.iso.org/ittf/PubliclyAvailableStandards/ISO_19139_Schemas/re
sources/Codelist/ML_gmx Codelists.xml#LanguageCode"
codeListValue="eng">eng</gmd:LanguageCode>
                                </gmd:language>
                            </gmd:MD_DataIdentification>
                        </gmd:identificationInfo>
                    </gmd:MD_Metadata>
                </aixm:timeSliceMetadata>
```

```
            <aixm:activation>
                <aixm:AirspaceActivation gml:id="SCHED_AA_2339714_67702">
                    <aixm:status>ACTIVE</aixm:status>
                    <aixm:levels>
                        <aixm:AirspaceLayer gml:id="SCHED_LV_2339714_67702">
                            <aixm:upperLimit uom="FT">17900</aixm:upperLimit>
                            <aixm:lowerLimit uom="FT">07000</aixm:lowerLimit>
                        </aixm:AirspaceLayer>
                    </aixm:levels>
                    <aixm:extension>
                <aimsua:AirspaceActivationExtension gml:id="SCHED_RP_2339714_67702">
                        <aimsua:reservationPhase>APPROVED </aimsua:reservationPhase>
                        </aimsua:AirspaceActivationExtension>
                    </aixm:extension>
                </aixm:AirspaceActivation>
            </aixm:activation>
            <aixm:extension>
                <event:AirspaceExtension gml:id="SCHED_EV_2339714_67702">
                    <event:theEvent xlink:href="#EVENT_2339714_W"/>
                </event:AirspaceExtension>
            </aixm:extension>
        </aixm:AirspaceTimeSlice>
        </aixm:timeSlice>
    </aixm:Airspace>
  </message:hasMember>
</message:AIXMBasicMessage>
```

The first member in this example is the optional dNOTAM Event element. It describes the baseline of the event. The second member of the AIXM basic message is the mandatory part, the affected airspace. It starts with the gml:identifier. This identifier is globally unique and allows to identify the airspace feature and retrieve the base data from a WFS. The bounding box which follows the identifier is included to allow spatial filtering on the event without the need to access a WFS. This is the only place where the event design does not strictly follow the approach of a thin event.

The next section of the airspace is a new time slice which describes the activation. It holds the time for which the activation is scheduled (valid time) and is defined as a temporal update. The time slice metadata section gives details on the publisher of this new time slice. The organization name in the example above identifies the publisher as Snowflake Software.

The activation element within the airspace time slice gives detailed information on the activation of the airspace. It informs about the status (e.g. ACTIVE), the vertical limits of the activated area, and the reservation phase (e.g APPROVED or PENDING).

The last part in the airspace time slice, the extension element, is only needed if the optional dNOTAM Event is included. In that case, the extension provides a reference to the dNOTAM Event element as defined by the dNOTAM specification.

7.1.2 Event Service Operations

In general, the interaction with the Event Service contains the steps shown in Figure 7-1. Three main components are involved: a producer generating events, a consumer receiving them and the Event Service. At first the consumer (e.g. a client application) is subscribed to event of interest. This can be done by the consumer itself or by an external entity for instance an administrator. With the subscribe

request (1.0) a filter expression and the consumer endpoint are transmitted. The response (1.1) contains the information to unsubscribe later on, e.g. when a flight was cancelled.

When a new event is sent to the Event Service (2.0) it is matched against the filter expressions of all subscriptions. When the filter is satisfied, the event is forwarded to the according consumer (2.1). Note that this communication is push-based. Thus, no response is sent upon receiving an event.

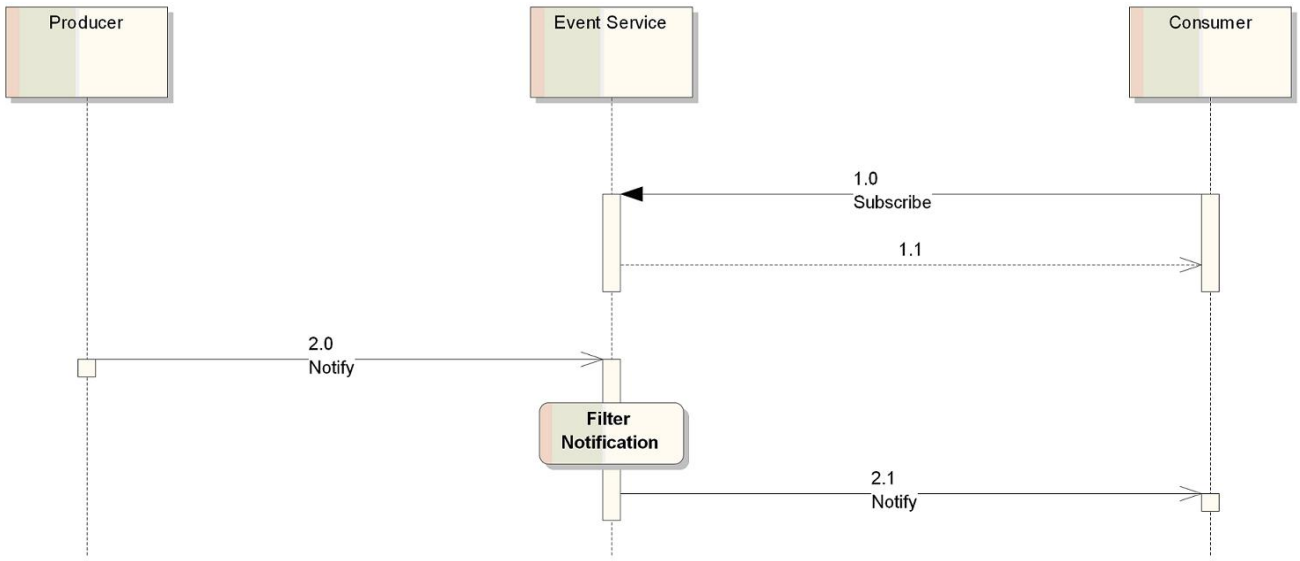

Figure 7-1. Event Service workflow

Subscriptions can be made against a specific topic, which is like an event pre-filter. For example, the subscriber may only be interested in reservation schedules for one type of airspace like Military Operations Areas (MOA). An Event Service could provide separate topics for each type of airspace, or perhaps separate topics for approved airspace reservations (in either waiting (W) or active (H) status), and for those still pending approval (P).

Filter encoding options are based on the OGC filter encoding (FES 2.0) standard, which includes filter criteria expressions that can be combinations of property values, spatial or temporal operators, or XPath XML element matching expressions.

- Property filtering offers more options than XPath. In addition to 'equals' expressions, more complex comparison expressions can also be used.

- Spatial filtering can be used to subscribe for updates in specific regions. Operators like 'intersects' or 'within distance' are available. This can be used to subscribe for SUAs that affect an airspace that is within a given distance to the flight path.

- Temporal filtering is similar to the property filtering for temporal expressions. This can for instance be used to subscribe only for SUAs that affect the flight time.

All the filter encoding expressions can be combined using logical expressions like 'AND' or 'OR'. Notifications can be triggered based on matching the subscription filter criteria, or on NOT matching any of the filter criteria.

A general (simple) Event Service workflow could look like this:

1. A client makes a subscription to an Event Service at a named endpoint. This may include a filter to specify the events of interest. The filter can use a topic expression, an OGC filter encoding based property filter or both.
2. The baseline feature data has to be requested by the client from a data store (e.g. WFS). The Event Service does not store any information about previous events. Thus, to be sure that every relevant update is received, the client should subscribe to updates and then request the current state.
3. Updates are made and the Event Server Adapter (see section 5.3) sends notifications to the Event Service.
4. The Event Service forwards the notifications to the clients according to the subscription criteria.
5. A consumer unsubscribes to an existing subscription, either to change the subscription filter criteria (the workflow would then start from the beginning) or to end the workflow.

The Event Service specification also includes Event Stream Processing (ESP) and Complex Event Processing (CEP) functions that can be used to derive further information from multiple notifications or to detect event trends, although these functions were not evaluated as part of the Pilot Study. More details on how to use the IfGI Event Service are provided in Annex D.

7.1.3 Channels and Published Events

Event channels (aka topics) are a way to provide a predefined filter. They can be compared to a TV channel where the definition of the channel defines what content is available. For instance on the weather channel one will receive weather information like forecasts, nowcasts and background information. On the sports channel one will get information on sport activities. For the FAA SAA Dissemination Pilot two channels were defined:

- estopic:heartbeat: on this channel the ‘I-am-still-alive’ messages are published (see section 8.8).
- estopic:schedules: on this channel the schedule update information is published

The namespace estopic is defined as xmlns:estopic="http://www.opengis.net/es/topics". To restrict a subscription to a specific channel, an addition to the subscription filter has to be made. The following example shows a restriction to the heartbeat channel. Full examples of the request can be found in section 7.1.4.


```
<wsnt:Filter>
    <wsnt:TopicExpression xmlns:estopic="http://www.opengis.net/es/topics"
           Dialect="http://docs.oasis-open.org/wsn/t-1/TopicExpression/Simple">
        estopic:heartbeat
    </wsnt:TopicExpression>
    <wsnt:MessageContent Dialect="http://www.opengis.net/ses/filter/level2">
    <fes:Filter>...</fes:Filter>
    </wsnt:MessageContent>
</wsnt:Filter>
```

To support the channel concept, the incoming events have to be assigned to a channel. This is done with a small addition to the notify request:

```
<wsnt:Notify>
    <wsnt:NotificationMessage>
        <wsnt:Topic xmlns:estopic=http://www.opengis.net/es/topics
            Dialect="http://docs.oasis-open.org/wsn/t-1/TopicExpression/Simple">
            estopic:schedules
        </wsnt:Topic>
        <wsnt:Message>
            ...
        </wsnt:Message>
    </wsnt:NotificationMessage>
</wsnt:Notify>
```

7.1.4 Event Service Workflow with Examples

As described before, the standard interaction with the Event Service follows this workflow:

1. Subscription
2. Notification to the Event Service
3. Internal filter matching
4. Notification of consumers
5. Unsubscription

Complete requests for step 1, 2 and 5 are listed below. The first example shows a subscribe request with a filter for schedule updates on airspaces that are not beyond (at least partly within) a distance around a flight path and have a temporal intersection with the flight period. Note that the URL in the wsa:To element has to be adjusted to the actual service URL.

```
<?xml version="1.0" encoding="UTF-8"?>
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope">
  <env:Header>
    <wsa:To xmlns:wsa="http://www.w3.org/2005/08/addressing">http://giv-ses.uni-
muenster.de:8080/SaaEventService/services/SesPortType</wsa:To>
    <wsa:Action xmlns:wsa="http://www.w3.org/2005/08/addressing">http://docs.oasis-
open.org/wsn/bw-2/NotificationProducer/SubscribeRequest</wsa:Action>
    <wsa:MessageID xmlns:wsa="http://www.w3.org/2005/08/addressing">uuid:1b4d3025-f80a-
a5b6-aa37-864c47fa1a7e</wsa:MessageID>
    <wsa:From xmlns:wsa="http://www.w3.org/2005/08/addressing">
      <wsa:Address>http://www.w3.org/2005/08/addressing/role/anonymous</wsa:Address>
    </wsa:From>
  </env:Header>
  <env:Body>
    <wsnt:Subscribe xmlns:wsnt="http://docs.oasis-open.org/wsn/b-2">
      <wsnt:ConsumerReference>
        <wsa:Address xmlns:wsa="http://www.w3.org/2005/08/addressing">
http://localhost:8082</wsa:Address>
      </wsnt:ConsumerReference>
      <wsnt:Filter>
        <wsnt:TopicExpression xmlns:estopic="http://www.opengis.net/es/topics"
Dialect="http://docs.oasis-open.org/wsn/t-1/TopicExpression/Simple">
estopic:schedules
        </wsnt:TopicExpression>
        <wsnt:MessageContent Dialect="http://www.opengis.net/ses/filter/level2">
          <fes:Filter xmlns="http://www.opengis.net/wfs"
xmlns:fes="http://www.opengis.net/fes/2.0" xmlns:gml="http://www.opengis.net/gml/3.2"
xmlns:ows="http://www.opengis.net/ows" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns:swe="http://www.opengis.net/swe/1.0.1">
            <fes:And>
              <fes:Not>
                <fes:Beyond>
                  <fes:ValueReference>geometry</fes:ValueReference>
                  <gml:LineString gml:id="ls1" srsName="urn:ogc:def:crs:OGC:1.3:CRS84">
                    <gml:coordinates 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</gml:coordinates>
                  </gml:LineString>
                  <fes:Distance uom="[nmi_i]">10</fes:Distance>
                </fes:Beyond>
              </fes:Not>
              <fes:AnyInteracts>
                <fes:ValueReference>validTime</fes:ValueReference>
                <gml:TimePeriod gml:id="tp1">
                  <gml:beginPosition>2011-04-04T20:00:00.000Z</gml:beginPosition>
                  <gml:endPosition>2011-04-04T22:00:00.000Z</gml:endPosition>
                </gml:TimePeriod>
              </fes:AnyInteracts>
            </fes:And>
          </fes:Filter>
        </wsnt:MessageContent>
      </wsnt:Filter>
    </wsnt:Subscribe>
  </env:Body>
</env:Envelope>
```

The second example shows a notify request. This request is used to send events to the Event Service or to one of the clients. The example below includes the assignment to a channel of the Event Service.

```
<?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.oasis-open.org/wsn/b-2">
   <soap:Header>
      <wsa:To>http://giv-ses.uni-
muenster.de:8080/SaaEventService/services/SesPortType</wsa:To>
      <wsa:Action>http://docs.oasis-open.org/wsn/bw-
2/NotificationConsumer/Notify</wsa:Action>
      <wsa:MessageID>uuid:1b4d3025-f80a-a5b6-aa37-864c47fa1a7e</wsa:MessageID>
      <wsa:From>
<wsa:Address>http://www.w3.org/2005/08/addressing/role/anonymous</wsa:Address>
      </wsa:From>
   </soap:Header>
   <soap:Body>
      <wsnt:Notify>
         <wsnt:NotificationMessage>
            <wsnt:Topic Dialect="http://docs.oasis-open.org/wsn/t-
1/TopicExpression/Simple" xmlns:estopic="http://www.opengis.net/es/topics">
estopic:schedules
            </wsnt:Topic>
            <wsnt:Message>
               <message:AIXMBasicMessage>
                  …
               </message:AIXMBasicMessage>
            </wsnt:Message>
         </wsnt:NotificationMessage>
      </wsnt:Notify>
   </soap:Body>
</soap:Envelope>
```

The last example shows an unsubscribe request. Note that this request is sent to a slightly different URL. It is handled by the subscription manager component. The resource ID that has to be used is contained in the subscribe response. It is used to identify the subscription that shall be terminated.

```
<?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.oasis-
open.org/wsn/b-2">
   <soap:Header>
      <wsa:To>http://giv-ses.uni-
muenster.de:8080/SaaEventService/services/SubscriptionManagerContextPath</wsa:To
      <wsa:Action>http://docs.oasis-open.org/wsn/bw-
2/SubscriptionManager/UnsubscribeRequest</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</wsa:Address
      </wsa:From>
      <muse-wsa:ResourceId xmlns:muse-wsa="http://ws.apache.org/muse/addressing'
wsa:IsReferenceParameter="true">MuseResource-5</muse-wsa:ResourceId>
   </soap:Header>
   <soap:Body>
      <wsnt:Unsubscribe/>
   </soap:Body>
</soap:Envelope>
```

7.2 Event Service Adapters

The Event Service Adapters check the airspace schedule activation feed on a regular basis, and provide events to the Event Service for any detected change. The encoding of the events is described in section 7.1.1. The airspace activation information is in a flat record structure, with the parameters (delimited by vertical bars) numbered beneath.

```
1975541|9325|H|R|3704A|2010/0304 18:00|2010/1231 23:59|000|100|A

   1   |  2 |3|4|  5  |       6       |        7      | 8 | 9 |10
```

Parameter 1 is a unique identifier for the airspace schedule, and parameter 2 identifies the affected airspace. Parameter 3 indicates the schedule status is given, where ‘H’ means Hot/Activated. Parameter 4 identifies the type of airspace being scheduled, which in this case is a Restricted area. Parameter 5 gives the airspace name, and the next two parameters define the start and end times for the activation. Parameters 8 and 9 define the vertical limits of the activated area in Flight Level. The last parameter that defines the separation rule for the activation is optional, but in this case it is an Aircraft separation rule.

The valid SUA reservation schedule feed values for schedule status are:

- P (pending approval)
- W (waiting to start)
- H (hot / active)

while the AIXM AirspaceActivation status values (CodeStatusAirspaceType) are:

- OTHER
- ACTIVE
- AVBL_FOR_ACTIVATION
- IN_USE
- INACTIVE
- INTERMITTENT

The SUA reservation schedule status values should instead be mapped to the SUA AIXM schema extension element for AirspaceActivation reservation phase (aimsua:AirspaceActivation.reservationPhase):

Status	reservationPhase
P	PENDING
W or H	APPROVED

Other reservation phase values must be inferred from the changes between two sequential instances of the SUA reservation schedule feed listings:

- reservationPhase DISAPPROVED must be deduced from the disappearance of a SUA schedule previously in P status

- reservationPhase CANCELED must be deduced from the disappearance of SUA schedule previously in W status, or previously in H status if current time is prior to reservation endTime

Since the Airspace status is the activation status property of the timeslice in effect during the validTime period, the default timeslice values for aixm:AirspaceActivation.status are:

BASELINE timeslice ... status=AVBL_FOR_ACTIVATION
(exception: when aimsua:Airspace.suaType=RA (Restricted Airspace), status=ACTIVE)
TEMPDELTA timeslice... status=ACTIVE

7.2.1 IfGI Event Service Adapter Processing

The Event Service Adapter provided by IfGI is implemented as a stand-alone Java program. Every minute it performs the following steps to generate the input for the Event Service:

1. The latest SUA schedules are retrieved from the FAA website.

2. The new schedule information is compared with the previous one. Only for changed entries an output is generated. This ensures that only new information is forwarded. At the start of the adapter all entries will be taken into account.

3. For each changed, new or removed entry the adapter checks if the activation type is supported. The information on the type is stored in the fourth parameter of each entry. The following types are supported: W, R, M, P, L and O. Other types are ignored.

4. For removed entries the time period is inspected. If the end time (parameter number seven) is in the past no event is generated as the activation ended. In other cases the activation was cancelled and an event is produced.

5. All remaining entries are translated by executing these steps:

    a. Get the airspace ID (2nd parameter) and retrieve the corresponding static data AIXM designator (UUID) of this airspace. A spreadsheet showing the AIXM UUID mappings to the SUA feed airspace ID values was provided by the FAA.

    b. Request the airspace base data from a WFS using the UUID.

    c. Build an AIXM basic message including the gml:identifier of the airspace feature, the bounding box of the airspace geometry and a new time slice for the activation information. The timeslice contains the time period of the activation as valid time, metadata identifying the publisher of this event, the activation status (ACTIVE always), the vertical limits and the reservation phase (3rd parameter).

6. The AIXM basic messages are sent as WS-Notification Notify requests to the Event Service.

7. Finally, the parsed SUA feed has to be stored to serve for the comparison in step 2 in the next iteration.

The optional dNOTAM Event element is not generated by the IfGI Event Service Adapter.

The IfGI Event Service Adapter is capable of using both the Luciad and the Snowflake WFS as feature data sources. It also supports feature collections and SAA messages as a return type for GetFeature requests.

7.2.2 Snowflake Software Event Service Adapter

The Event Service Adapter provided by Snowflake Software is comprised of several components that perform the following steps:

1. **GNU WGet cron job:** configured to request the latest SUA schedules from the SAA data feed every minute and download the .csv file to a local directory
2. **SQL*Loader script**: configured to poll the local directory and load the .csv into a SCHEDULE database table within an Oracle database
3. **Oracle triggers**: several triggers were established to be run automatically on insert of new schedules into the table:
    a. Remove schedules that have been received in previous downloads
    b. Identify schedules that have disappeared before their effective time and generate a correction schedule to state that it is cancelled/disapproved
    c. Insert new schedules into the live schedule tables and assign each schedule an identifier, sequenceNumber and correctionNumber. The live schedule tables are based on the AirspaceUsage AIXM 5.0 schema to enable improved querying by the WFS adapter.

4. **GO Publisher Agent:** once the mappings required to transform the schedule data into event messages were configured using GO Publisher Desktop, these were deployed within GO Publisher Agent. GO Publisher Agent was configured to automatically publish an Event message into a local directory triggered by the insert of a new schedule into the live tables.
5. **Event Pusher:** is a standalone JAVA program that polls the local directory and pushes new Event messages to the Event Service. Once pushed the Event Pusher moves the Event messages into an archive directory.

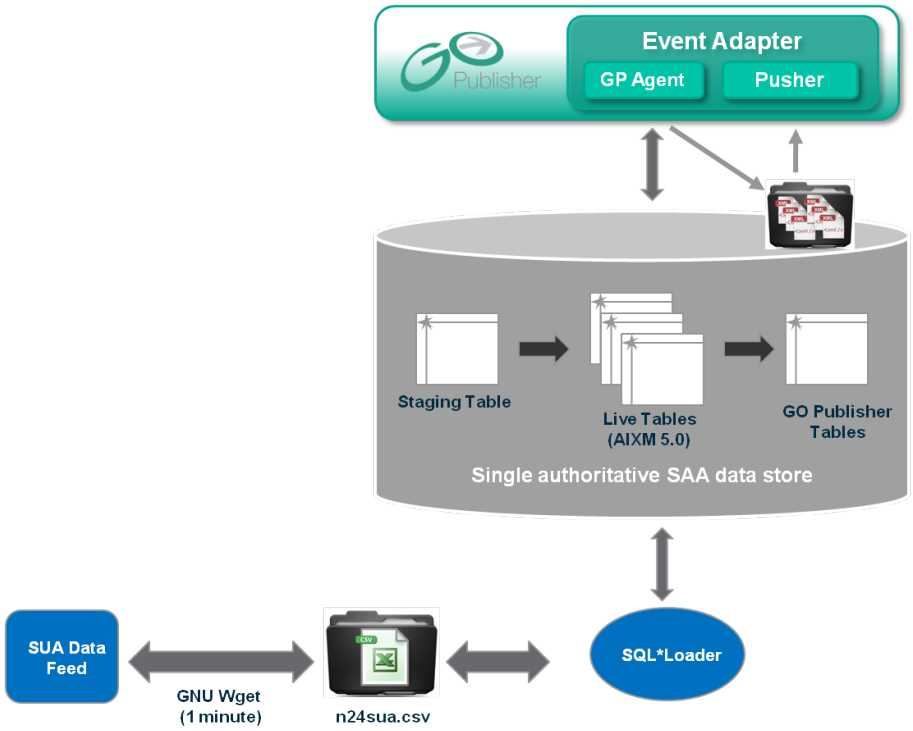

Figure 7-2. Snowflake Software Event Service Adapter Component Architecture

The Snowflake Software Event Service Adapter does not integrate with the WFS, instead both the static and dynamic SAA data are integrated into a single data store. This was required to enable simultaneous access to Events via both the WFS and Event Service.

Snowflake Software opted to test the draft version of the Digital NOTAM Event Specification v0.6 as the encoding for the Event messages. The Digital NOTAM Event Specification v0.6 requires that an Event message contains two features:

1. **Event:** this feature contains a BASELINE timeslice containing information describing the Event that created the update to the AIXM feature
2. **Airspace:** contains a TEMPDELTA timeslice containing the following properties:
    a. **Activation:** containing the schedule
    b. **Timeslice metadata:** containing information describing who is responsible for providing the Event
    c. **boundedBy:** containing a bounding box geometry (generated from the Airspace geometry table in the static data tables) defining the extent of the Airspace to support spatial filtering within the Event subscription

The schedule data received from the SUA data feed contains reservation schedules for a wide range of special activity airspace, not all of which were applicable to the Pilot Study. Therefore, the Snowflake Event Service adapter was configured to publish only Events relating to the airspaces relevant to the pilot. All other reservation schedules were ignored by GO Publisher Agent.

Due to resource constraints, Snowflake was not able to develop the processes required to detect schedules that had disappeared from the schedule file between downloads. This would be important for a real-world implementation as AIXM 5.1 requires that any change to a feature is explicitly published.

7.2.3 Event Service Adapter Testing

As the scenario demonstrations are based on predefined events and not on the input of the Event Service Adapters, IfGI developed a tool to test the adapters separately. This tool is designed as a web service, accepting WS-Notification Notify requests. This way no changes at the adapters need to be made in order to perform the tests.

The main purpose of the test is to verify that both the IfGI and the Snowflake Event Service Adapters provide the same information during a fixed time frame. To accomplish this, each received event is analyzed to identify its affected airspace (using the gml:identifier), the valid time, the activation status, the vertical limits and the reservation phase. Ideally, one event from each adapter (identified via the timeslice metadata) is received for each set of the above mentioned properties.

The test results can be displayed in a browser showing the number of events received from both adapters, and those only from either IfGI or Snowflake. In addition, the time since the start of the test is displayed, and every received event can be accessed.

In order to ensure that the events that are received should have a counterpart from the other adapter, the testing tool has to ignore inputs as long as only one adapter is online. Also, when an adapter begins sending events, the tool has to ignore the first set of messages. This is because in the first iteration, the adapters generate events for every entry in the SUA feed, which means they generate events for activations that were defined already in the past. If both adapters do not start in exactly the same moment, the events that are received will most likely not match. To avoid this and to generate meaningful comparisons, all input in the first few minutes after both adapters are started are ignored.

The first runs of the test program identified some differences in the behavior of the two Event Service Adapters. The vertical limits were interpreted differently (resolution: interpret as flight level) and the status was set differently for pending activations (resolution: use ACTIVE).

In the later runs with updated event service adapters and an updated event monitor the events produced by the two adapters were matched successfully. However, the monitoring reports show a number of matched events and a number of events produced by ifgi only. The reason is that the adapters behave differently when the activation status is changed from W (waiting) to H (hot / activated). The ifgi adapter is designed to produce an event for every change in the activation feed. As the ststus is changed, a new event is generated and published. The advantage is that every change in the feed is published.

The disadvantage is that this results in two events with the same content. This happens due to the fact that the three possible states in the activation feed (namely P for pending, W for waiting and H for hot / activated) are translated to only two reservation phase values in AIXM. P is translated to PENDING, whereas W and H are translated to APPROVED. When an activation is changed from W to H (this happens when the start time is reached) the same event would be generated. As most clients already received this information and check for the start time themselves, the Snowflake Event Service Adapter suppresses these duplicate events.

Below, test results for a period of over 48 hours are shown:

- Events received from both adapters: 911
- Events received from ifgi only: 1,994
- Events received from Snowflake only: 10

As you can see, there were 911 new activations and activation that were approved. In addition, 1,994 schedules switched to hot during the testing period. The ten events that were produced by the Snowflake adapter only as for the affected airspaces no AIXM designator is given in the airspace mapping file. Thus, the ifgi-adapter cannot build events for these entries.

All in all, the adapter testing showed that the two event service adapters work properly and produce the same output. All differences are either intended or a result of incomplete base data.

The program used to perform the above mentioned tests is available at 52°North:
https://wiki.52north.org/pub/Sensornet/SensorEventService/SAAEventMonitor.war

7.3 Snowflake WFS

The WFS 2.0 Adapter provided by Snowflake Software is composed of the following components (Figure 7-3):

1. **GO Loader:** was used to configure, build and maintain the static data within an AIXM 5.0 database schema, supplied in the SAA subscriber files within the Oracle database.
2. **Oracle triggers:** once the data is loaded into the staging tables using GO Loader, various scripts are triggered to:
    a. Remove duplicate features and timeslices loaded in previous files
    b. Automatically generate a correction BASELINE timeslice on receipt of a new update timeslice
3. **Validation and QA process:** a manual validation and QA process was performed post-load to identify any issues with the data received including:
    a. Identify features that contain updated properties but neither the sequenceNumber or correctionNumber were incremented
    b. Identify features that have disappeared between files
    c. Visually inspect the airspace geometries to identify whether the geometries were valid. This was required to identify issues with arcByCentrePoint geometries where start and end positions were inverted.

    Figure 7-4 shows the results of this validation process.
4. **GO Publisher WFS:** transforms the data on-the-fly into AIXM 5.1 for access via the WFS 2.0 interface

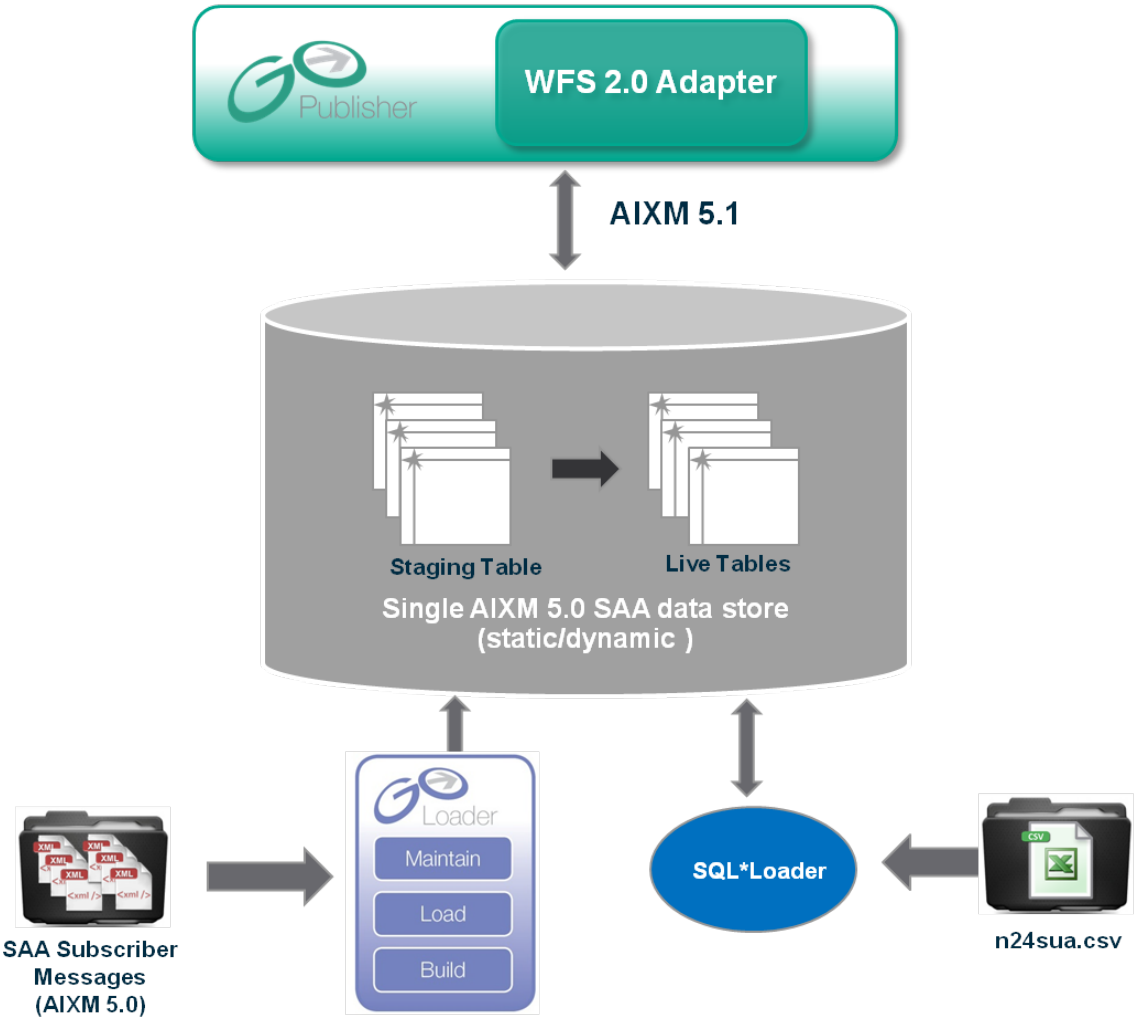

Figure 7-3. Snowflake Software WFS Adapter Component Architecture

SAA Subscriber messages

AIRAC Cycle	Total No. Messages	New	Missing	Updated	Erroneous (corrections)	Test (Removed)
13-JAN-2011	86	86	-	-	2	23
10-FEB-2011*	1005	942	0	63	-	
10-MAR-2011	957	897	48	60	1	
07-APR-2011	952	0	5	19	1	

•The 10-FEB-2011 files were not loaded – missing sequenceNumber/correctionNumber

Figure 7-4. Summary of SAA subscriber messages received and processed

The objective of the Snowflake Software WFS Adapter was to maintain the SAA messages received in conformance to the AIXM 5 Temporality Model and to provide access to the full history of the AIXM features through the WFS 2.0 adapter. Therefore, the static and dynamic data from both SAA data sources were integrated and made accessible via a single WFS 2.0 adapter (Figure 7-5).

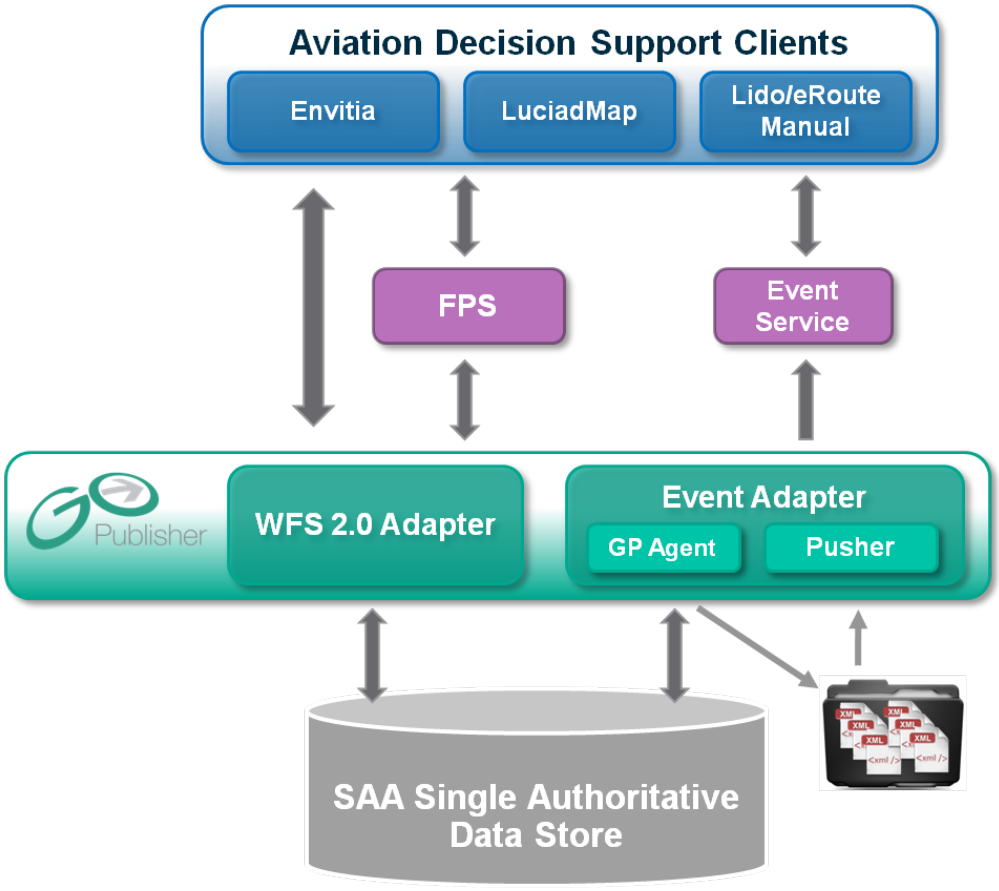

Figure 7-5. Snowflake Software SAA component architecture

GO Publisher Desktop was used to configure the schema transformations required to publish the SAA subscriber messages and schedule data into the SAA AIXM 5.0 specification. The data was configured to publish a single AIXM feature containing multiple BASELINE and TEMPDELTA timeslices.

Example: QUAIL MOA Airspace

```
wfs:member>
    <aixm:Airspace gml:id="Airspace1_95b5cd27-f2e7-40eb-a195-17f129625ebf">
        <gml:identifier codeSpace="http://www.faa.gov/nasr">95b5cd27-f2e7-40eb-a195-
17f129625ebf</gml:identifier>
        <aixm:timeSlice>
            <aixm:AirspaceTimeSlice gml:id="SCHED_TS_2383562_154404">
                <gml:validTime>
                    <gml:TimePeriod gml:id="SCHED_TP_2383562_154404">
                        <gml:beginPosition>2011-06-01T18:10:00.000Z</gml:beginPosition>
                        <gml:endPosition>2011-06-01T20:00:00.000Z</gml:endPosition>
                    </gml:TimePeriod>
```

```
            </gml:validTime>
            <aixm:interpretation>TEMPDELTA</aixm:interpretation>
            <aixm:sequenceNumber>2</aixm:sequenceNumber>
            <aixm:activation/>
        </aixm:AirspaceTimeSlice>
    </aixm:timeSlice>

    <aixm:timeSlice>
        <aixm:AirspaceTimeSlice gml:id="SCHED_TS_2384124_155004">
            <gml:validTime>
                <gml:TimePeriod gml:id="SCHED_TP_2384124_155004">
                    <gml:beginPosition>2011-06-01T22:15:00.000Z</gml:beginPosition>
                    <gml:endPosition>2011-06-02T00:15:00.000Z</gml:endPosition>
                </gml:TimePeriod>
            </gml:validTime>
            <aixm:interpretation>TEMPDELTA</aixm:interpretation>
            <aixm:sequenceNumber>3</aixm:sequenceNumber>
            <aixm:activation/>
        </aixm:AirspaceTimeSlice>
    </aixm:timeSlice>
    <aixm:timeSlice>
        <aixm:AirspaceTimeSlice gml:id="Airspace1_TS11338">
            <gml:validTime>
                <gml:TimePeriod gml:id="Airspace1_TS1_TP11338">
                    <gml:beginPosition>2011-03-10T05:00:00.000Z</gml:beginPosition>
                    <gml:endPosition indeterminatePosition="unknown"/>
                </gml:TimePeriod>
            </gml:validTime>
            <aixm:interpretation>BASELINE</aixm:interpretation>
            <aixm:sequenceNumber>1</aixm:sequenceNumber>
            <aixm:correctionNumber>0</aixm:correctionNumber>
            <aixm:featureLifetime/>
            <aixm:designator>MQUAIL</aixm:designator>
            <aixm:name>QUAIL MOA, AZ</aixm:name>
            <aixm:geometryComponent/>
            <aixm:activation/>
            <aixm:annotation/>
            <aixm:extension/>
            <aixm:extension/>
        </aixm:AirspaceTimeSlice>
    </aixm:timeSlice>
  </aixm:Airspace>
</wfs:member>
```

Snowflake opted to encode all BASELINE and TEMPDELTA timeslices within a single feature to ensure that when a client performed a query to select active Airspaces, the WFS response contained the correct features. Within OWS-7, Snowflake identified that if a WFS provides access to only BASELINE and TEMPDELTA timeslices encoded within separate instances of a feature, then the response to such queries is imprecise.

However, issues were identified with this configuration approach:

1. More information is returned than clients request as the WFS does not provide functionality to subset the content of a complex property

2. Clients struggle to parse features containing large numbers of timeslices as load times increase, because this requires high memory usage which reduces performance

To improve client performance and demonstrate that a WFS can be quickly and easily configured to meet different end user requirements, multiple end points were provided:

- **Full history end points**: these WFS end points provided access to features containing 1 or more timeslices representing the history of the feature from Jan 2011
- **Current history end points**: these WFS end points provided access to features containing one or more timeslices whose validTime/beginPosition is greater than or equal to "now"

Snowflake Software also identified the potential need for additional features as part of future development, including the integration of event features to provide a pull endpoint for event messages, and development of interpretation rules for handling the creation of AIXM 5.1 timeslices.

7.4 Luciad WFS, FPS and WMS

Luciad contributed three services to demonstrate the dissemination of SAA data:

- An OGC Web Feature Service to serve the SAA data in AIXM 5.1,
- An OGC Feature Portrayal Service to render the SAA data served by the WFS using OGC Styled Layer Descriptors,
- An OGC Web Map Service to serve Visual Flight Rules (VFR) aeronautical charts.

These services are built with the LuciadFusion technology, Luciad's product to efficiently handle geospatial data in networked environments. Figure 7-6 shows an overview of the Luciad server architecture.

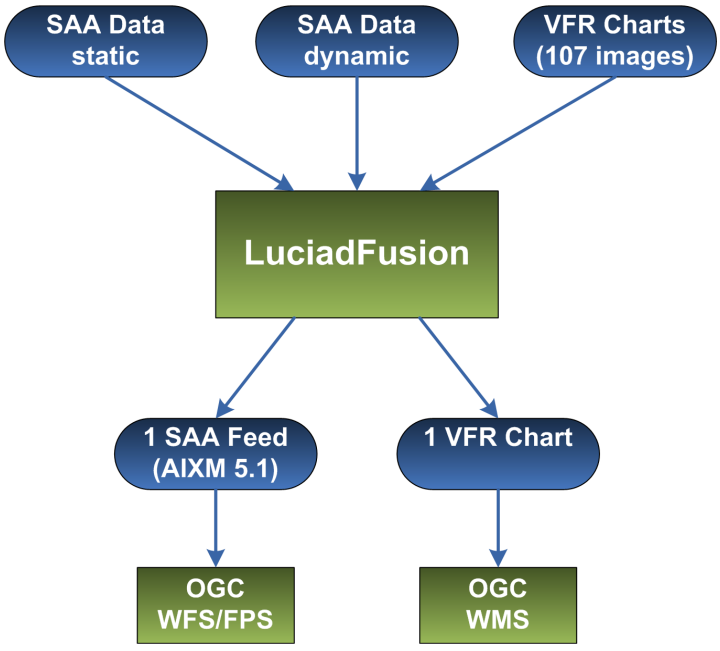

Figure 7-6. Server Architecture

LuciadFusion enables connection to various data sources, on-the-fly data fusion and redistribution of the result to various clients. The benefits that were demonstrated in this project include on-the-fly

data conversion, OGC-compliant services, enhanced filtering capabilities, customized on-the-fly visualization, increased ease of access, and high-performance access to raster data. The following paragraphs and sections discuss the project requirements and the provided solutions in more detail.

Three different data sources were made available by the FAA for the Pilot Study. A first data source represents static SAA definition data, provided in an FAA extension of the AIXM 5.0 data format. The extension is used to represent all SAA- and SUA-specific properties, along with the default AIXM properties. The data itself is delivered as a set of files, with each file describing one airspace.

A second data source represents information about scheduled changes to the SAA activation status. This data is published through a dynamic text feed, available on the web. Through LuciadFusion, a transformation pipeline was set up to fuse the static and dynamic SAA data together and transform the result to an FAA extension of the AIXM 5.1 format. The fusion of the static and dynamic data is done by converting the dynamic SAA data into AIXM 5.1 tempdelta time slices, which are then combined with the AIXM 5.1 baseline timeslices representing the static SAA data. The resulting setup enabled the on-the-fly processing of the incoming data and streaming to the various OGC service endpoints.

A third data source represents VFR sectional chart data, delivered as a set of disparate images in the GeoTIFF format for various regions in North-America. LuciadFusion is used here to process the source images (stripping off legends and redundant borders) and fuse them together into one coherent, tiled image with multiple levels of details, allowing efficient loading and distribution of the data.

The following sections describe in more detail how these processed data sources are made available to the clients through a set of OGC services.

7.4.1 Luciad WFS Component

A first component provided by Luciad is an OGC Web Feature Service to access the SAA data. The WFS made it possible to provide filtering capabilities for the SAA data, through the OGC Filter Encoding language. We were able to show filtering on properties, spatial filtering to retrieve airspaces that potentially affect a flight rout, and temporal filtering.

One of the challenges with SAA geometry is the handling of the separate components that are combined using constructive geometry operations, such as unions and subtractions. LuciadFusion supports the additional processing required for exact spatial queries on such complex airspaces. This allows you to evaluate exactly whether a point lies inside or outside of a particular airspace.

The supplied data also contained links between features, for instance from an air traffic control service to the airspace for which it provides a traffic separation service. These links pose additional issues when they are encountered by client applications. This issue was avoided by the WFS by automatically converting all links to local links to features that were included with the response.

7.4.2 Luciad FPS Component

In addition to the WFS component, Luciad also provides an OGC Feature Portrayal Service (FPS) to portray the SAA data. Whereas the WFS caters to the needs of heavyweight applications capable of parsing AIXM 5 data, an FPS service takes away the burden of parsing and visualizing the raw AIXM 5 data. Instead, it remotely loads and renders the data, and returns the result as an image that can be shown by any modern web browser.

The FPS is built using the OGC-standards based components provided by LuciadFusion. The data is requested from the WFS using a standard WFS client, decoded into an AIXM 5 domain model, and finally visualized with an SLD implementation optimized to render AIXM 5.

The FPS was provided as an in-kind contribution because it entirely relied on standards and did not need any code specific to the project. In fact, the same setup is also used in other OGC projects.

With the FPS, it was shown how a very simple client application can still offer advanced filtering and styling capabilities, without having to implement support for the AIXM 5 format. The supported scenarios were:

1. Visualization of predefined data with predefined styles; this is also known as a classic WMS, publishing named layers and named styles.
2. The use of an SLD as a style catalog, provided in a catalogue service (CSW); in this mode, the FPS defines the data source (named layer) but the user defines the rendering style (user style).
3. The use of an SLD to specify custom layers and styles; this is the 'real' FPS mode, where the user defines both the data source (user layer) and rendering style (user style).

The only drawback to the FPS setup is the additional overhead introduced by the need to retrieve the data from the WFS. This overhead could potentially be reduced by setting up an FPS that shares a data store with a WFS, or possibly by introducing an additional caching mechanism. Both approaches would be fully transparent to the user.

Another useful enhancement to increase the dissemination of SAA data would be support for vector-based output formats, such as KML and GeoJSON. This could also benefit map display performance, as the client application would no longer need to request new images every time a user zooms or pans.

7.4.3 Luciad WMS Components

Luciad provides two Web Mapping Service (WMS) components to the Pilot Study. The first WMS component is part of the FPS described above, and exposes the SAA data as a WMS layer. It uses the same data back-end as the WFS to transform the source SAA data to AIXM 5.1, before rendering it with a default style.

A second WMS component provides access to the VFR aeronautical chart data. This data is published as a single layer, representing the result of the fused VFR chart images. This layer can be used by client applications as a background layer.

7.5 INdicio Web Registry Service

INdicio™ is a Style Registry that implements the Open Geospatial Consortium (OGC) Web Registry Service specification (also known as the Catalogue Service for the Web). The Indicio Web Registry is a web-based registry service that implements the CSW-ebRIM 1.0.1 specification (OGC document 07-110r4). CSW-ebRIM is the ebRIM profile of the Catalogue 2.0.2 specification (OGC document 07-006r1). The INdicio Web Registry is used for hosting Styling information encoded using the OGC Styled Layer Descriptor (SLD) specifications that are used by the Envitia Feature Portrayal Service (FPS) component in the SAA Pilot Study.

Demonstration of the Web Registry Service capabilities are largely independent from the Commercial Flight scenario used by the other service components. The resources instances contained in the Web Registry are constrained to the repository items corresponding to the ebRIM model, illustrated in the INdicio Designer Client diagram shown below:

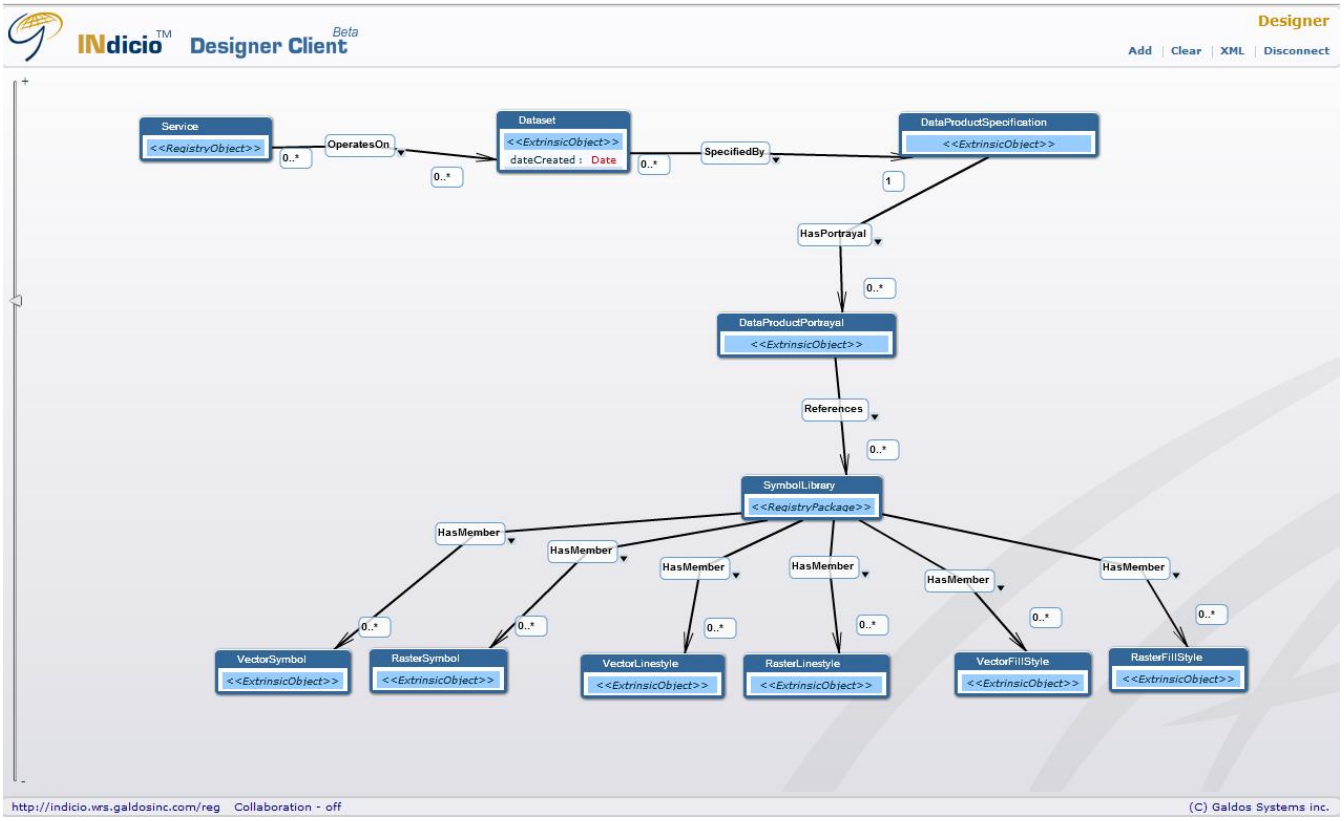

Figure 7-7. INdicio Web Registry Service Repository Items Used in Pilot Study

In short, the Web Registry Service resources are limited to the following:

SLD (repository item for DataProductPortrayal in diagram above)
Symbol set (repository item for SymbolLibrary in diagram above)
AIXM schemas (remote link to official schemas corresponding to DataProductspecification in diagram above)
Dataset metadata (repository item for DataSet in diagram above)
Service metadata (repository item for Service in diagram above)

7.6 Lufthansa Systems Client Implementation

Lido/eRouteManual is a commercial off-the-shelf product by Lufthansa Systems FlightNav. It is a software charting solution for fully electronic operation on the flight deck that is in use by many major airlines worldwide.

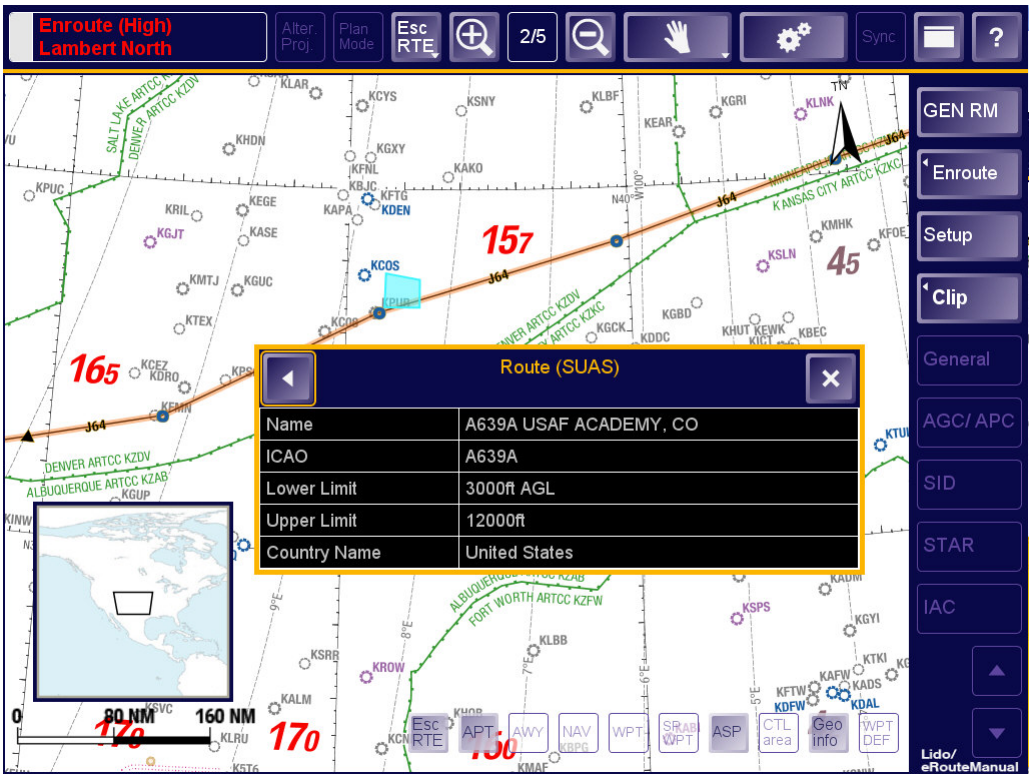

Figure 7-8. Lido/eRouteManual Client Display

Lido/eRouteManual contains terminal and approach charts as well as seamless, worldwide enroute charts. It supports pilots in all phases of the flight -- from taxi, departure, enroute to arrival and approach. All navigational information, route segments, topographical information, etc., are shown precisely to scale and the appropriate chart projection is selected automatically. Information on airports, navigation aids, waypoints, airways, airspaces and other objects can be looked up to view frequencies, coordinates, tracks, altitudes, and so on.

Lufthansa Systems already contributed to the OGC OWS-6 testbed by demonstrating the feasibility of integrating Digital NOTAM information into Lido/eRouteManual's charts for inflight display.

In the frame of the FAA SAA Dissemination Pilot, Lufthansa Systems FlightNavhttps://portal.opengeospatial.org/twiki/bin/edit/SAAPilot/FlightNav?topicparent=SAAPilot.SaaPilotClientLufthansaSys;nowysiwyg=1 extends the functionality of and develops interfaces for Lido/eRouteManual to fully support all client functionality of the technical infrastructure:

- WFS module to formulate WFS queries and request SAA (schedule) information from the FAA SAA WFS Webservices

- Event Service module to subscribe to and unsubscribe from and receive event notification from the FAA SAA Event Services
- AIXM module to enable usage of many AIXM features

The back-end functionality enables two new user functions in eRouteManual, which are developed concurrently:

- Upon entering or loading a flight route, the **SUA Function** retrieves all SUAs along the route, for which a reservation is scheduled that could impact the flight, and displays their names in a tabular list. The user can select each list entry, highlight and "jump to" the respective airspace as well as access additional information such as the scheduled reservation times. The data used for this function is dynamically updated by accessing the WFS and Event Service.

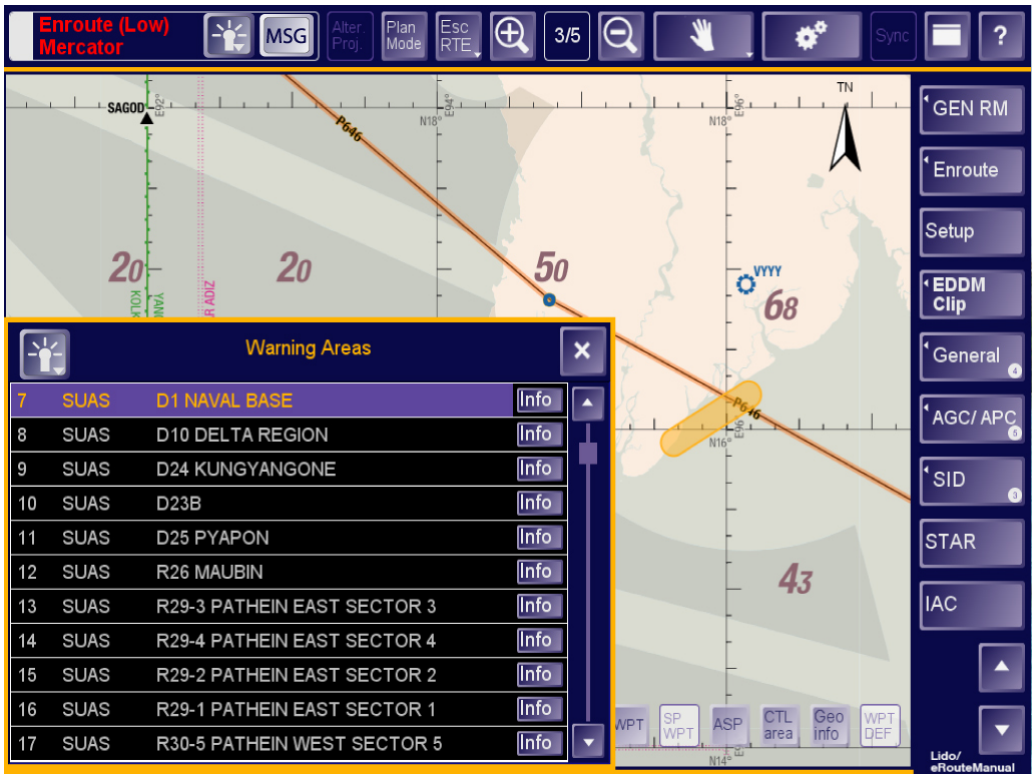

Figure 7-9. Active Special Use Airspace Reservations Along the Route

The **Notification Function** provides the user with dynamic information about airspace proximity and data changes. When the aircraft approaches an active airspace or when reservation schedules for airspaces along the defined routes are changed, the notification function informs the user.

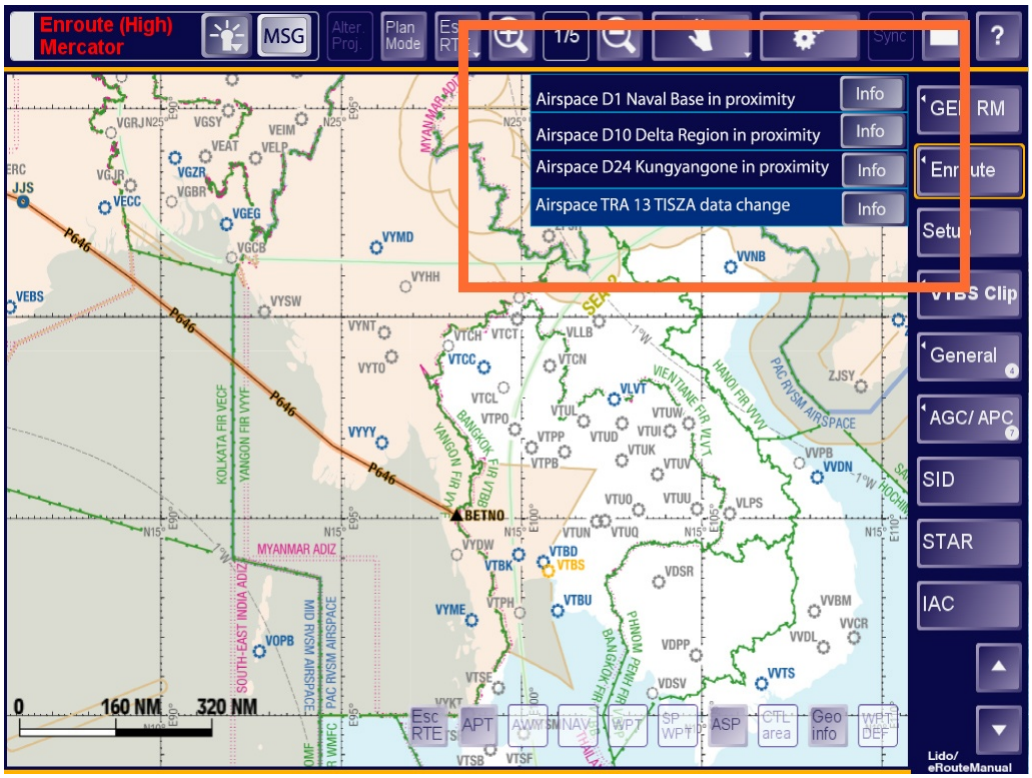

Figure 7-10. Notification of Airspace Reservation Change

7.7 Envitia FPS, WPS and Client Implementation

The Envitia components aim to demonstrate how a fairly high level of application functionality can be delivered flexibly, via a browser, exploiting the full power of an SDI based service platform. The demonstration uses Web Feature Services from Luciad and Snowflake to supply the data, and this is exploited via the Envitia Portrayal service which supports SLD-WMS (FPS) and WPS End Points. It also exploits the GALDOS registry as the source for the page contents presented (which is defined using another OGC Standard, the Web Map Context (WMC). The subsystem also exploits two other services, a 'gazetteer service' delivered via a WFS and an 'annotation service' delivered via a WFS-T. Both of these are supplied by a Geoserver based WFS set up on the Envitia hosting environment. The configuration is shown below.

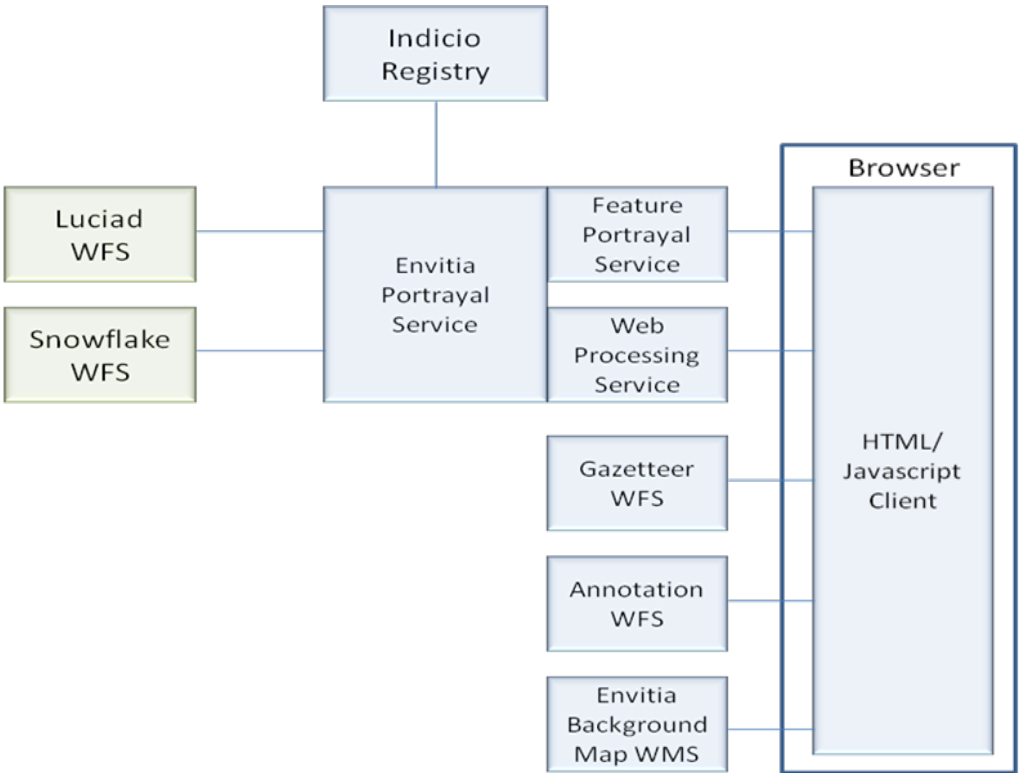

Figure 7-11. Envitia Client and Services

The Envitia Portrayal Service

The Envitia Portrayal Service is capable of accessing Web Feature Servers, Web Map Servers and Web Coverage Servers in order to obtain data to portray in a client. Typically it is configured to access specific services (rather than receiving the definition in the SLD) so that it can cache data in the servers (it periodically updates, looking for changes). It is thus typically a hybrid of the SLD-WMS Integrated and Component models of a SLD-WMS. This mode of operation means the FPS can cache data but there is also the flexibility to easily re-direct it to another WFS. The FPS can also cache the SLDs (based on URL) maximizing the performance, but this is only really safe if the SLD is in the registry. A more detailed treatment of the rationale around this is given in OGC Document 10-127r1 (OWS-7 *Aviation Portrayal ER*). The FPS interface supports Get FeatureInfo allowing the user to click on an item and see the attribution present.

The Envitia Web Processing Service

The Envitia Web Processing service is an implementation of the OGC Web Processing Service. This is an abstract service, which needs to be specialized by an application profile. Envitia has implemented a 'Cross-section' Web Processing Service, which visualizes the layers available in the configured Feature Portrayal Service in X-Z rather than X-Y. As such it acts like a WMS except the display axis is X-Z rather than Z-Y. The display shows the cross-section along the great circle line between the two points specified. This view is also capable of visualizing the Special Activity Airspace in X-Z giving a clear view of the airspace restrictions along the route.

```
http://service.domain.name/wps.aspx?
    request=Execute&
    service=WPS&
    identifier=CrossSection&
    version=1.0.0&
    datainputs=
        Width=1340;
        Height=500;
        Layer=Layers.xxxxx;
        startLon=-16.819514142336;
        startLat=49.984774178832;
        endLon=-16.819514142336;
        endLat=49.984774178832;
        MinDepthBound=-1000;
        MaxDepthBound=0;
        Graticle=1&
      RawDataOutput=
        image=@
        mimetype=image/png@

        encoding=binary
```

The Envitia Background Map Service

This service is portraying three background map options, a basic vector map, a shaded terrain map based on DTED and "World Raster Aeronautical Charts" for the USA which were downloaded from FAA VFR Sectional Charts site. The service used is a static configuration (named layer) of the Envitia FPS as this configuration is the highest performance option.

Geo-Server Web Feature Service

This service is used to provide a range of gazetteers, but in particular an airport gazetteer. It is also used to provide a transactional WFS (WFS-T) used to store annotations entered by the user, so they can be accessed by other users.

The Envitia Javascript Client

The Envitia browser-based client, built on the Dojo Toolkit and OpenLayers, is used in the SAA Pilot to implement the SAA review and update scenario. The goal is to provide an interface which will run in any standard browser environment, without the need for plug-ins. It provides a configurable interface allowing easy re-configuration and focused functionality. The client can be accessed from the following URL: http://217.33.30.10/FAASAA This link brings up the contents page of the demonstration. On this page are links to the various configured views set up on the portal. View content is defined using Web Map Context documents (which is stored in the Indicio Registry and passed on the command line to the portal).

The user can locate a specific location (airport) with the gazetteer. Pan to that airport and then review the airspace. To help the review a number of different styling options can be selected (these are defined by SLD. The user can then measure or draw a line across which to analyze and the Cross-section is displayed with the vertical airspace. The user can then add an 'annotation' and create graphics and text which can then be saved. It may be a new temporary airspace, or it could be a comment on an existing airspace.

Overall the portal shows a high level of functionality, is simple to use, and builds on existing and stable OGC services.

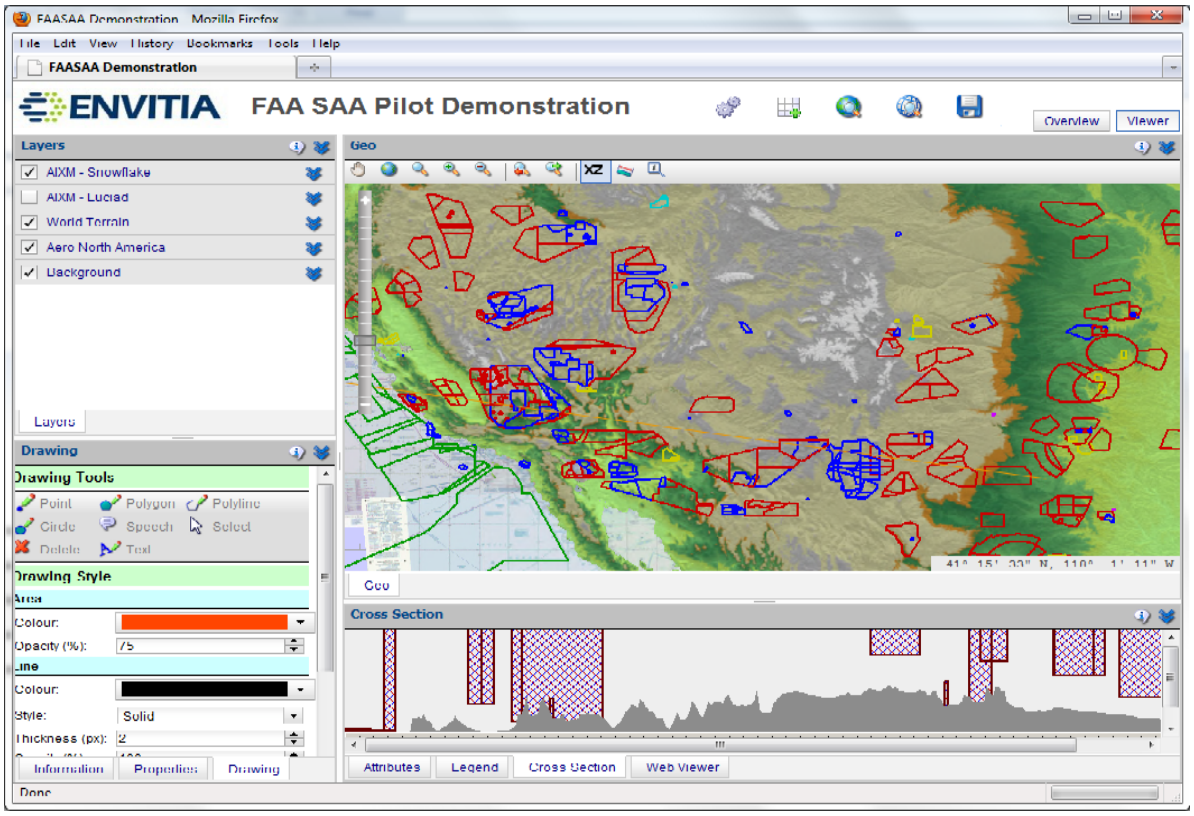

Figure 7-12. Envitia Browser-Based Client Display

7.8 Luciad Client Implementation

Luciad contributed a client application to access the SAA data through the various OGC services used in the project. The application is built with the LuciadMap software suite, offering a number of OGC-standards based components and an application framework for rapid application development. Figure 7-13 shows the client and the services with which it interacts. The following sections highlight the features of the client application relevant for the SAA Pilot.

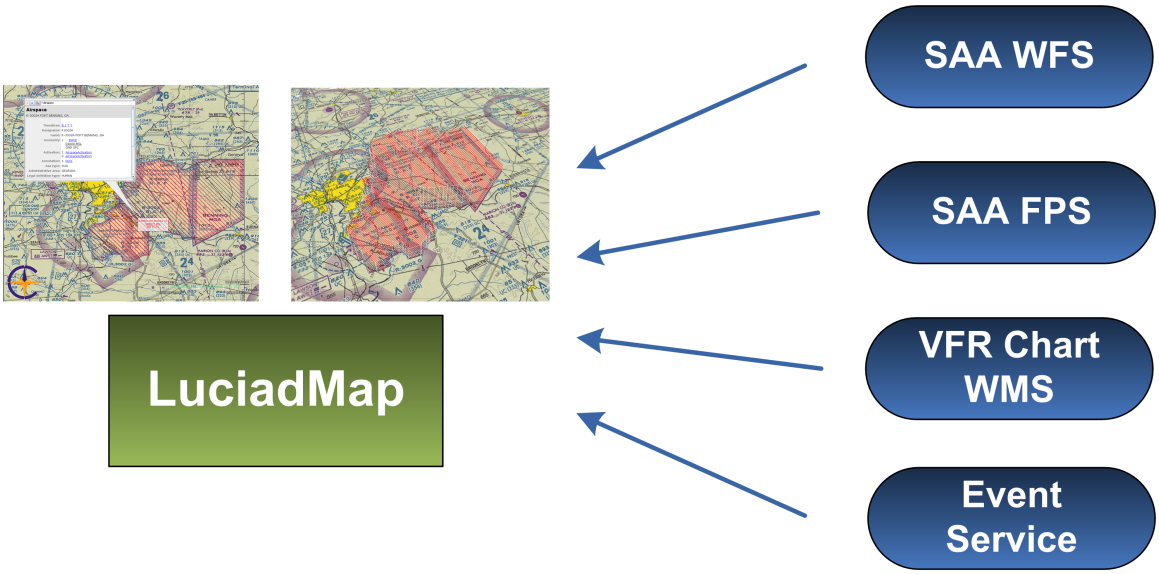

Figure 7-13. Client-Service Interaction Overview

The application provides its users with flexible means for efficient queries to WFS services for SAA data. These queries can be based on a geographical region and/or a time interval. Additionally, an interface to the Event Service guarantees that the user is kept informed of any updates to the SAA data.

The client supports four key features for effective use of the SAA data set:

1. 3D visualization
2. Time-dependent styling
3. Data browsing
4. User-defined styling

The first feature is 3D visualization. The airspaces can be rendered as 3D volumes, based on the 3D constructive geometry operations specified in the data. These operations are used a lot in SAA airspaces and sometimes result in difficulties when trying to give a correct view of the airspace in 2D. Visualizing the data in 3D gives the user an optimal view on both the shape and the height of an airspace.

An example is illustrated in Figure 7-14. In this example, the airspace geometry is defined by a base geometry and a circular subtraction. Due to a difference between the height of the base geometry and the subtracted geometry, the operation results in a blind hole. In 2D, it is not possible to describe such a blind hole without adding annotations to the map. This is not a problem in 3D, where you can clearly see the result of the subtraction.

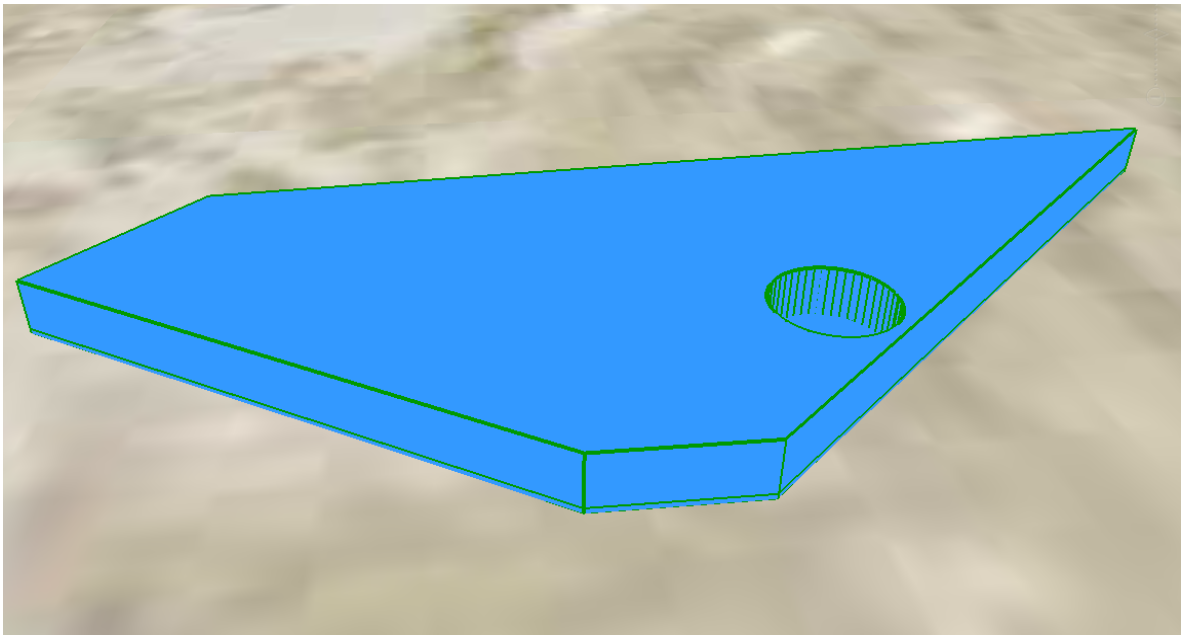

Figure 7-14. Airspace with partial hole in 3D

A second feature is time-dependent styling, using the activation status and time before activation to choose a specific rendering style for an airspace. The time-dependent styling is very relevant as the most important use cases for flight planners are to know when an SAA is active. By using specific styles for the SAA based on their status, the flight planners get visual aids allowing them to work more efficiently. The information on the status of an SAA is contained in the extensive AIXM 5 temporality model. The resulting feature is illustrated in Figure 7-15. Airspaces that are active are displayed in red, while inactive airspaces are displayed in grey. Additionally, a color code is used for airspaces that will become active within 8 hours. This gives a visual indication of how much time is left before the airspace is activated. This enables flight planners to simulate their flight efficiently without having to dig through the data themselves.

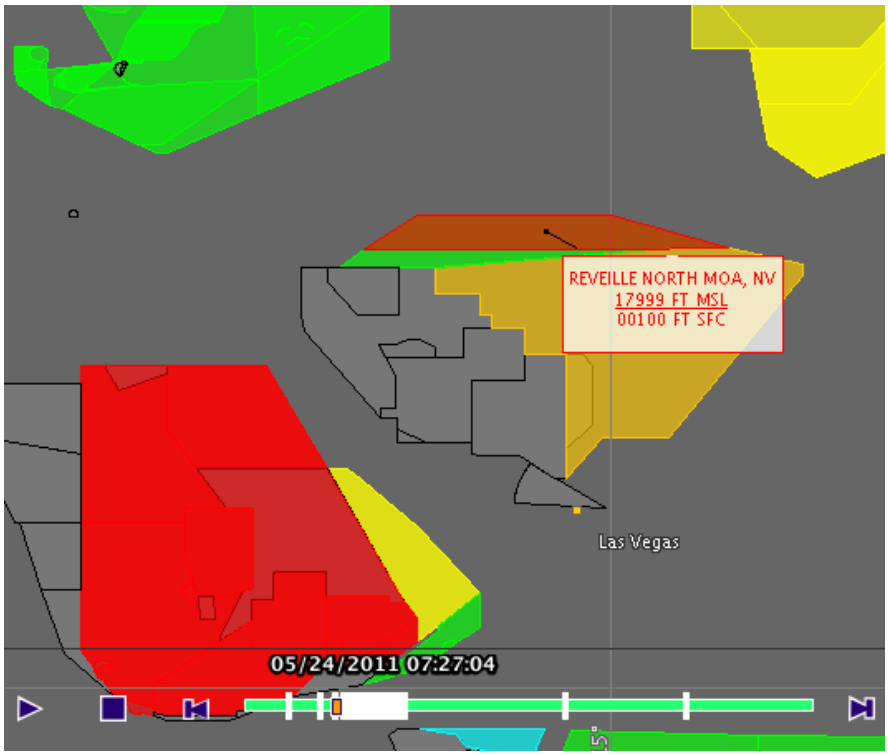

Figure 7-15. Time slider with active and inactive airspaces

Next to a visual appearance on the map, users may also be interested in accessing the properties of the SAA data. To support this use case, the client application displays a convenient data browsing component for a selected airspace, illustrated in figure Figure 7-16. This component gives access to all available data properties in a way that is similar to a web browser. Users can simply follow a link from top level features to nested features or from one feature to a linked feature.

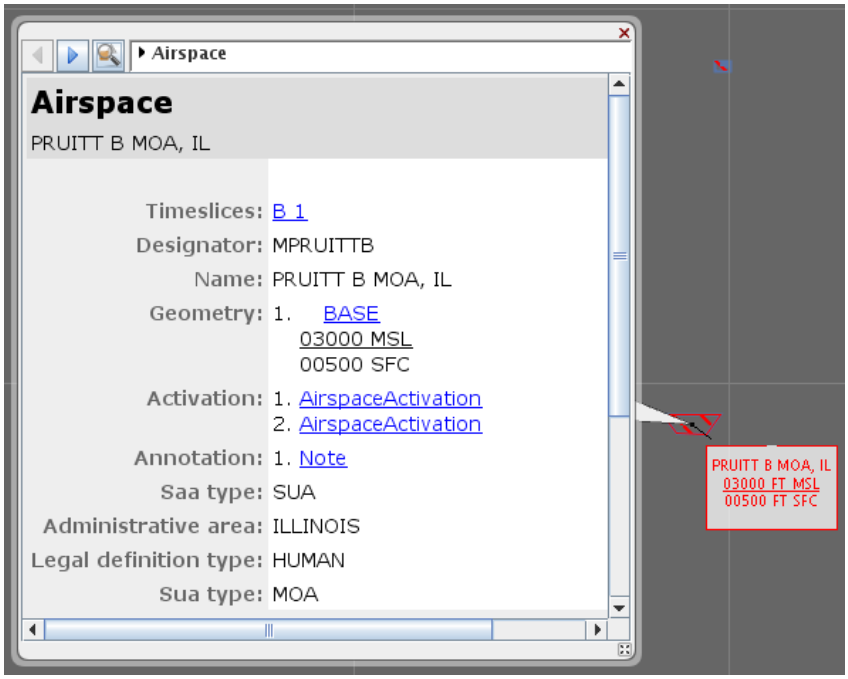

Figure 7-16. SAA data browsing

One style to visualize the SAA data does not fit the needs of all users. A flight planner wants to know what airspaces affect a potential flight route; an airspace designer may want to see the separate SAA components; while another user might be interested in the different types of airspaces. All of these needs can be fulfilled in one application by allowing custom user-defined styles.

These styles can be defined in the OGC Symbology Encoding format, and stored in an online Web Registry Service. This enables you to define a set of styles that can be shared between users. The use of an open format ensures that they can be used across a range of applications.

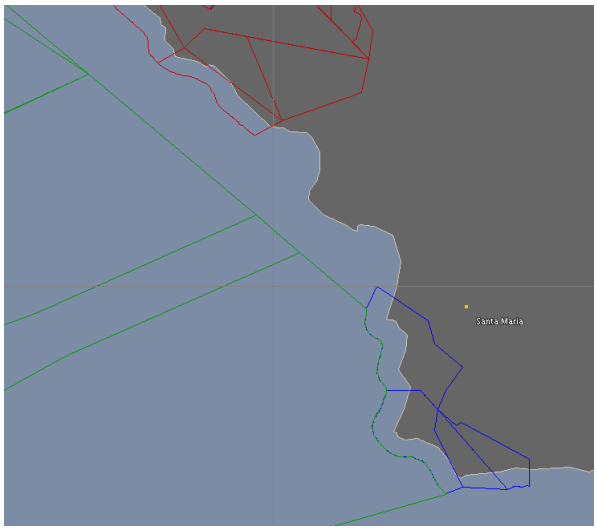

Figure 7-17. SAA type-based styling

7.9 Component integration with FUSE ESB

Section 8.12 describes the feasibility analysis for FUSE Enterprise Service Bus (ESB) integration performed by the service component providers. They were not required to deploy their component on a FUSE ESB, although they were required to at least investigate and document the feasibility of doing so.

In general, the service component providers indicated that FUSE integration should be feasible, but would likely require some redesign of their software. The LuciadFusion service components (WFTS, FPS and WMS) and the Snowflake Software WFS have already been successfully deployed on the FUSE platform.

In the future, the deployment of all relevant service components on the same FUSE ESB should be investigated. This was not required for the SAA Pilot Study.

8 Architecture/Implementation Issues and Lessons Learned

A number of interesting discussions arose during the SAA Pilot Study that led to agreement on the need for Change Request submissions to the OGC and other standards bodies, and that suggested that some alternative approaches would work better than others. These issues and lessons learned are described below.

8.1 Rename FES 2.0 Temporal Filter Operator

A convenience filter: AnyInteracts was added to the Filter Encoding 2.0 specification which combines the following temporal queries into a single query: Begins OR Ends OR During OR Equals OR Contains OR Overlaps OR OverlappedBy OR BegunBy OR EndedBy.

During the development of the scenarios, it emerged that the name of the FES 2.0 AnyInteracts filter expression does not correspond with terminology commonly used to describe its functionality. The term intersects was commonly used. Therefore, a change request was submitted to the FE 2.0 specification to request that the AnyInteracts filter expression be re-named to TIntersects to ensure that it is inline with common terminology and is more intuitive to ensure proper use.

8.2 Interpretation of Arc by Center Point

The AIXM5 airspace data used in this Pilot Study makes extensive use of GML ArcByCenterPoint to construct surfaces. AIXM5 airspaces are defined as GML surfaces that usually only have an exterior ring. This exterior rings is constructed by concatenating a number of segments. Usually these segments are either simple line strings, which are lists of points, or Arc by center points. These arc by center points are defined by a center point, a radius, a start angle and an end angle.

The SAA Pilot participants recognized that the specification of ArcByCenterPoint in the GML 3.2.1 standard (OGC 07-036) is ambiguous and that therefore a corrigendum is required.

Details on the appropriate interpretation for ArcByCenterPoint are discussed in the "Use of GML in aeronautical data" OGC discussion paper. A formal corrigendum proposal has been submitted to the OGC.

8.3 Interpretation of DWithin geometry filter operator

The semantics of the DWithin spatial filter operator in FES 2.0, as used in the WFS, are underspecified.

1. The OGC/ISO FES 'DWithin' Operator is currently defined as: DWithin(A,B,D) = Within(Buffer(A,D),B) in agreement with ISO WFS/FES EC).

2. OGC/ISO FES 'DWithin' Operator currently doesn't map faithfully to Oracle's 'SDO_WITHIN_DISTANCE'. In fact, SDO_WITHIN_DISTANCE is equivalent to: DWithin(A,B,D) = Distance(A,B) < D, which disagrees with the ISO WFS/FES EC interpretation.

3. If Distance = 0, then DWithin = Within

If we interpret OGC Filter/ISO FDIS 19143 as written, rather than relying on an informal clarification, the Filter 2.0 specification does not provide a rigorous definition of DWithin (as compared to Simple Features for SQL), Instead, it provides a text description containing the phrase "within a specified distance." The phrase "within distance" is defined in the oxford dictionary and implies Distance(A,B) < D.

From discussions at the OGC Technical Committee meetings in Bonn, it seems that the first of the following two interpretations (considered before) is more widely implemented (e.g. Oracle and GeoAPI) and seems to be the preferred choice.

1. DWithin(A,B,D) = Distance(A,B) < D
2. DWithin(A,B,D) = Within(DBuffer(A),B)

Note that when the geometry 'B' is a point (which is commonly the case, when B = a person or vehicle location) the two definitions above are equivalent because a point B is within DBuffer(A) if an only if Distance(A,B) < D. Of course, the oxford dictionary example used an instance where B is just a point geometry; e.g., "He lives within a few miles of Oxford."

As we also know, the two definitions above are not equivalent if B is a higher dimensional geometry such as a line or polygon. Clearly we need to support this more general use case in the SAA Pilot, where B is a higher dimensional geometry.

It appears that the Oracle operator SDO.within.distance is equivalent to:

Distance(A,B) =< D (similar to interpretation #1 above)

But this subtle difference to interpretation #1 above isn't really an issue, because we do not operate in a spatial realm of infinite precision anyway, between round-off measurement errors and the inexactness of floating point arithmetic, the comparison operators =< and < will rarely ever evaluate differently in practice.

The recommendation is to follow interpretation #1 and document a change request against the OGC Filter 2.0 specification to include a clarification to this effect.

8.4 Event Origin Identification

To allow clients to identify the origin of an event, a timeslice metadata section has been added to the AIXM 5.1 schema used in the Pilot Study. The data identification element provides a point of contact to identify the publisher of the event. Multiple options to include this information in an SAA Message were discussed:

- Use the feature metadata property of the dNOTAM Event element
- Use a gml:description element
- Use the message metadata property of the root element, the AIXM basic message.
- Use the timeslice metadata property.

The first option was rejected as the dNOTAM Event element is not mandatory, and thus might not be included in all events. The gml:description element was not selected, as it does not provide a

machine readable template to encode the publisher information. The option of using the metadata section of the AIXM basic message was rejected, since it is possible that a message might contain multiple member features with different publishers. In addition, message metadata provides metadata on the message itself, which does not necessarily indicate the entity that published the actual events that are included in the message. Therefore, the remaining alternative to use the metadata section of the airspace timeslice was chosen.

Indeed, metadata at the time slice level seems to make the most sense, as both messages and features are just containers that are not necessarily stored by the WFS. However, this results in a potential conflict in how to interpret this metadata:

- metadata describing the timeslice itself, such as the author of the time slice
- metadata describing the feature that can also change over time, such as the bounding box.

Both types of metadata would currently be stored in the timeslice metadata. As a result, to add metadata that describes the author of a timeslice, it is necessary copy and also include the bounding box metadata from the related BASELINE timeslice, otherwise it would be overwritten by the new metadata when generating a feature snapshot. In conclusion, the Pilot Study agreed to add ISO encoded metadata identifying the source of the temporal data to the timeslice that represents the actual change; i.e., the timeslice in the AIXM feature that is transported in the AIXM message.

8.5 Web Feature Service

Although the pilot successfully demonstrated that it is possible to develop OGC Web Feature Services that serve SAA AIXM 5.1 format data using existing SAA AIXM 5.0 data feeds, there were several issues and lessons learned:

1. **SUA Schedule Data Feed**
    a. SUA Data Feed does not explicitly publish when a schedule is cancelled/disapproved. It is simply absent from subsequent files
    b. Converting the upper limit levels from FL to FT could not be achieved correctly as there is an anomaly in how the upper altitude limit is handled in the SUA data feed when the airspace is defined as “up to but not including” resulting in inconsistent values in the BASELINE and TEMPDELTA timeslices. In addition, flight level and feet are incompatible units of measure, therefore a conversion is impossible and should not even be performed. Filter expressions should also use flight level as unit of measure for a given comparison value.
2. **SAA Subscriber Messages**
    a. SAA Subscriber Message feed was in development during the pilot so we received messages containing test data which had to be subsequently removed, edited or updated
    b. How to deal with SAA messages that disappeared between AIRAC cycles was not resolved. No process was defined describing how these should be interpreted and dealt with within the AIXM 5 Temporality Model.
3. **WFS Adapter**
    a. Support for SNAPSHOT timeslices is essential for decision-support scenarios
    b. Alternatives to encoding multiple timeslices within a single feature instance within a WFS should be investigated in OWS-8
    c. Difficulties were identified when retrieving Airspaces based on Level properties which contain a mixture of heights and code values using filter parameters alone

8.6 Event Service Adapter

This section describes discussions and decisions regarding the behavior of the Event Service Adapters.

8.6.1 Changes in schedule feed

In the schedule feed (seen in Figure 5-2. SAA Pilot Study OGC Components), some entries are marked as pending. Thus, these schedule updates are not approved yet and might not become active. The question was whether these entries shall be translated into AIXM and sent to the Event Service or not.

The decision was made to include these entries for the following reasons:

- Some applications might need these notifications. New pending activations could be sent to a decision maker who could then approve or disapprove them in a timely manner.
- The information on a possible activation of an airspace is also of interest as the airspace can be avoided in an earlier stage of the planning process.
- Clients not interested in pending activations can add a filter criterion to their subscription to excluded these events as this information is available in the reservation phase element.

8.6.2 Error handling

When an Event Service Adapter assembles an Event message for delivery to the Event Service, it is possible that the Event Service may not be able to process or retrieve sufficient information from the WFS to supply missing BASELINE details. In case of Event Service errors (such as no SaaMessage, or no corresponding airspace gml:identifier is available from the WFS), it was decided that no event would be generated for the Pilot Study implementation.

These types of errors were occurring due to data inconsistencies between the static airspace definitions and the airspace schedule reservations, and were not expected to occur in a production environment. Without eliminating these error states, the client software would require additional logic to deal with these exceptions, which should never appear in the operational system. It was agreed that the Pilot Study would not develop solutions for errors that are due to the prototypical state of the system, and that would not be present in the operational version.

8.7 Subscriber File Data Issues and Lessons Learned

8.7.1 Date Time Format

The subscriber file uses a date format based on http://www.w3.org/TR/xmlschema-2/#date. For example,

```
<validTime xmlns=" http://www.opengis.net/gml/3.2 ">
<TimePeriod ns:id="ID1_OrganisationAuthority1_TS1_TP1" xmlns:ns="
   http://www.opengis.net/gml/3.2 ">
<beginPosition>2010-02-11-04:00</beginPosition>
<endPosition indeterminatePosition="unknown"/>
</TimePeriod>
</validTime>
```

The -04:00 beginPosition element is a timezone offset from Greenwich Mean Time (GMT–4 hours). This is not to be confused with a dateTime format which has seconds:

```
<beginPosition>2010-02-11T04:00:00</beginPosition>
```

8.7.2 Text in Altitude Fields

AIXM allows special altitude values in text, such as GND or UNL, rather than just integer values. This causes a problem with WFS/FES filtering based on altitude properties.

8.7.3 Geospatial Filtering with Compound Objects

Filtering based on altitude values for airspace features with multiple geometry objects can also be an issue. One example is to query for an upper altitude that is less than 20000' and a lower altitude greater than 25000'. There should be no airspace that matches this query, but SUA W-386 does. It consists of three airspace volumes, one with an upper altitude of 1,999', and another with a lower altitude of 60,001', so when these volumes are evaluated independently they satisfy the filter criteria. In order to correctly apply altitude filters for airspace with multiple volumes, the filter criteria should be applied to the boundaries of the full 3D airspace (or perhaps just the airspace bounding box); otherwise additional processing on the results of a filter operation is needed.

8.8 Event Service Heartbeat Messages

As the Event Service is push-based, clients would not know if the service is still running when no notifications are received. This may happen due to a malfunction of the service, or when there are simply no events to handle. In the FAA SAA Dissemination Pilot, three proposals on how to determine if the service is still alive were discussed:

8.8.1 GetCapabilities approach

In this approach, clients could periodically request the service capabilities from the Event Service. It would offer the following advantages:

- Clients can specify by themselves in which interval the capabilities are requested.
- No additional implementation work has to be done as the GetCapabilities operation is already available.

The disadvantages of this approach are:

- The service capabilities are quite large for a simple message indicating that the service is still alive.

- If the service is implemented in a modular fashion there might be the risk that the capabilities handling are still working although the stream processing and notification modules are not. Thus, the result might not be reliable.

To overcome the disadvantages a specific operation could be available to check the service state. This would produce a small response to avoid communication overhead, and could be implemented to guarantee the full functionality of the service. However, would require additional implementation effort, and the Event Service specification would have to be extended.

8.8.2 Periodic heartbeat approach

In this approach to notify clients of Event Service availability, an external component could be used to send heartbeat messages to the Event Service. These messages would be available on a dedicated topic. Clients could then subscribe to this topic and receive the heartbeat messages. This option comes with the following advantages:

- There is only a low implementation effort to build the external component, no change at the event service is necessary.
- The push mechanism is used ensuring that the relevant sub-module of the implementation is tested.

The disadvantage of this approach is that clients cannot specify the heartbeat frequency.

To overcome the disadvantage of the predefined frequency, two solutions are possible. First, multiple heartbeat topics could be provided, e.g. for a heartbeat every minute, every ten minutes and every hour. The second option would make use of the Event Stream Processing functionality of the IfGI Event Service implementation. Using a complex filter statement, clients could subscribe to receive heartbeats only at desired intervals.

8.8.3 Client-specific heartbeat approach

In this option, the heartbeats are generated by each client interested in the state of the Event Service. Each client would send a heartbeat including a unique client ID and subscribe for its own heartbeat. This approach offers the following advantages:

- No implementation effort on the server side.
- The push mechanism is used ensuring that the relevant sub-module of the implementation is tested.

The disadvantages of this approach are:

- There is implementation effort on the client side to produce the heartbeat events.
- When many clients use the heartbeat function, this option could lead to a high filtering load on the Event Service, especially when the heartbeats are sent in high frequencies.

8.8.4 Recommended Option

For the FAA SAA Dissemination Pilot, the decision was made to implement the second option, the externally generated period heartbeat. The primary reason for this was that it does not impose much implementation effort for the service and client developers. In addition, all the appropriate sub-

components of the Event Service are used, thus ensuring a high significance of the heartbeat messages. The disadvantage of clients not being able to specify the heartbeat interval was acknowledged, but considered not significant for the Pilot Study implementation.

To produce the heartbeats, a small Java program was implemented. It can be configured to send messages to a given URL in a given interval. The messages look like this:

```
<hb:Heartbeat xmlns:hb="de.IfGI.swsl.SES.heartbeat">
    <hb:date>2011-04-05T12:39:37.275Z</hb:date>
</hb:Heartbeat>
```

The only content of the heartbeat message is the time and date of the generation. This allows clients to verify that the Event Service is not overloaded.

8.9 Persistence of Historical Timeslices

The AIXM 5.1 conceptual model states that a real-world aeronautical object should be represented by a single feature containing one or more timeslice objects representing the state and/or events that have occurred or will occur during its lifespan. However, this conceptual representation poses a challenge for implementing systems using OGC WFS 2.0.

The OGC WFS 2.0 assumes that the features contained within the service conform to the ISO 19109 General Feature Model and is designed to retrieve complete features or values for a single specified property. There is no operation within the WFS to return a feature containing a subset of objects. Therefore, if an aeronautical object is represented as a single feature containing multiple timeslice objects, then the service will return all the timeslices. This becomes problematic with airspace reservation TEMPDELTA timeslices collected over time, which generally are requested (created) on a daily basis.

After discussion of several alternative ways for the WFS to support features with more than one timeslice, it was decided to persist all available timeslices, and to return them all in response to a feature request. This approach may be suboptimal for large scale WFS implementations where large numbers of BASELINE timeslices may exist, leading to very large feature definitions when encoded in AIXM5.

This approach also implies the WFS will have a semantic responsibility to ensure that the collection of timeslices associated with a particular feature always represent a current valid state; e.g., at least one BASELINE, PERMDELTA or TEMPDELTA timeslice must be valid at any point in time. More information on the issue of subsetting timeslice information of a returned AIXM feature is provided in the OWS-8 *WFS Guidance Engineering Report*.

8.10 Performance Aspects of Timeslice Use in Feature Portrayal Service

There are two distinct approaches to exploiting the time element in AIXM within the Feature Portrayal Service. There may in the end not be too much to choose between them, but if the FPS provides data caching, it is possible that one approach is preferred to the other.

The FPS has two main methods of selecting temporal data. The first is to specify an OGC Filter in the FPS Request. This filter would then select a specific timeslice for visualisation. These are

characterised by the structure of the request to the FPS, the resulting request to the WFS, and caching of the WFS response by the FPS. It is possible to define a filter request in the FPS which is passed along to the WFS to return a specific set of data for a given time.

AIXM includes the construct of 'Timeslice' within featureType. Thus if a simple featureType is requested, the result will include a number of timeslices. Each timeslice is then accessed via an XPath expression. The alternative is to have a more selective Filter expression which can specify an individual timeslice. The result is then rendered directly.

In the first approach, all data is read up front and cached. Further requests (as the user pans along the timeline) do not trigger a new WFS Query, as the FPS spots that the URL and query string are unchanged. Hence individual timeline increments are smooth, although the initial wait to retrieve the data is longer.

The alternative is to continually modify the time interval or value defined in the WFS Request. The result of this approach is that a WFS request will be executed each time the user increments the timeline selection. Given the latency involved in repeated WFS queries, the simple featureType approach may not be as practical as using an OGC filter, and can result in a jerky visual display.

8.11 AIXM 5.1 Representation Issues

8.11.1 Defining inactive time for airspaces

The issue was raised that the static definition of an airspace from the subscriber file does not have the status defined for when it is 'INACTIVE'. This issue should be resolved when converting the airspace definition from AIXM 5.0 to AIXM 5.1 through the use of the timeslice Excluded attribute.

An airspace that has inactive periods of time should have a static definition that explicitly states this. This is done by creating an additional AirspaceActivation that has Timesheet objects for the same periods of time where the airspace is defined as 'ACTIVE' or 'AVBL_FOR_ACTIVATION'. These AirspaceActivation objects would have a status = 'INACTIVE' and the Timesheet objects would have Excluded = 'YES'. This would clearly define what the airspace status is for any time of the day.

8.11.2 Defining "other times by NOTAM"

Defining that an airspace has "other times by NOTAM" in AIXM 5.1 is similar to how we define inactive times. This is done by creating an additional AirspaceActivation that has Timesheet objects for the same periods of time where the airspace is defined as 'AVBL_FOR_ACTIVATION'. These AirspaceActivation objects would have a status = 'AVBL_FOR_ACTIVATION' with Excluded = 'YES' and issueNotam = 'YES'.

8.12 FUSE integration

Table 8-1 summarizes the feasibility analysis for FUSE ESB integration performed by the service component providers.

Service Provider	FUSE integration feasible (yes/no)	Component deployed on FUSE ESB (yes/no)	Comment
Snowflake WFS	yes	no	
Snowflake Event Service Adapter	yes	no	
Luciad WFS	yes	yes	
Luciad FPS	yes	yes	
Luciad WMS	yes	yes	
IfGI Event Service Adapter	should work	no	Not tested as the Event Service itself is not deployed in FUSE
IfGI Event Service	should work	no	Not deployed and tested as the whole software project would have to be re-engineered.
Envitia FPS	yes	no	
Galdos Style Registry	yes	no	

Table 8-1. FUSE ESB Integration Summary

8.12.1 Snowflake Software

The successful deployment of the Snowflake WFS to the FUSE ESB platform was described in the *OWS-7 Aviation - FUSE Deployment Integration Report* (OGC 10-130). Porting the Snowflake Event Service Adapter to FUSE ESB was not attempted in this Pilot Study.

8.12.2 Luciad

The LuciadFusion-based service components (WFS, FPS and WMS) have been successfully deployed on the FUSE ESB platform.

The service components are built using Java Servlet technology and are packaged as Web Application Archives (WARs). To realize deployment on the FUSE ESB platform, the WARs needed to be integrated into OSGi bundles. This was done using the bnd tool, delivered with a FUSE ESB installation. The bnd tool analyzes one or more jars in a WAR and then adds entries to the jar manifest files to express the dependencies. No modification was needed to the source code.

The main technical challenge was to provide a proper bind configuration file for each of the WARs. This configuration file specifies a number of options that influence how the manifest is generated. The difficulty in creating these files is the very limited feedback generated by the bnd tool in case

something is wrong. As a result, creating these files is a trial-and-error process requiring many iterations before the manifest is generated correctly.

Even though the services could be deployed on the FUSE ESB platform, this approach was not used in practice. This is mainly because the deployment process was already in place for the LuciadFusion-based services (using a Java servlet container), and the benefits of using FUSE ESB did not apply for this small-scale setup.

8.12.3Institue for Geoinformatics (IfGI)

As shown in the table above, the Event Service was not deployed into the FUSE ESB. The reason for this is the design of the implementation and the used libraries. For the WS-Notification support, IfGI makes use of the MUSE library. In addition, the software project and its referenced libraries are managed via Apache Maven.

According to the MUSE documentation, projects based on MUSE should be deployable as they are into FUSE. However, this does not work with the Event Service implementation. A simple deployment of the WAR file was not successful. In order to reach compatibility, the Event Service project would have to be re-engineered completely. This task was not feasible to be performed within the SAA Pilot Study timeframes. Because of this, the integration was not achieved and no final statement about the feasibility can be made, as potential side effects when re-engineering the software cannot be foreseen.

The Event Service Adapter developed by IfGI is not implemented as a web service. It is a stand-alone Java program. Because of this and because of the fact that the Event Service is not deployed in FUSE for the Pilot Study, no feasibility study for the integration of the Adapter was made. However, as the adapter is rather simple, a refactoring as a FUSE compatible service should not be a problem.

9 Conclusions

Despite the need to work around scope and requirements changes resulting from SAA source data limitations, the SAA Pilot Study represents another successful OGC Interoperability Program Initiative. OGC standards-based web service components were used to support SAA data dissemination, including translations of AIXM 5.0 format source data into AIXM 5.1 format messages, AIXM 5.1 temporality encodings, support for SAA AIXM schema extensions, and demonstration and test of the draft Digital NOTAM Event Specification. Other significant accomplishments included:

- Implemented WFS and ES support for complex airspace geometries, aggregated airspaces, and improved filter capabilities in spatial queries
- Interfaced WFS and ES components with available data sources using adapter components
- Implemented support for portrayal of AIXM 5.1 data (via WMS, FPS and Registry), including time-dependent map display styling using SLD
- Demonstrated the value of geospatial and temporal data fusion for the display and use of airspace activation data in realistic aviation scenarios

The SAA Pilot Study found that the AIXM and OGC standards used for their component implementations are mature enough for industry adoption, although they also found that several technical issues require some additional clarification, including:

- Interpretation of Arc by Center Point encodings
- Interpretation of DWithin geometry filter operator
- Event Service (and potentially other OGC service) need for heartbeat messages
- Filter Encoding (FES 2.0) temporal operator changes (AnyInteracts renamed to TIntersects)
- Interpretation and conventions when using AIXM temporal model timesheets

The SAA Pilot Study demonstrated the value of OGC open standards compliance in facilitating cost-effective multi-vendor tool integration. Using the OGC standards allowed for rapid implementation of the ability to merge static and dynamic SAA data, including event notifications, into a single AIXM-format representation. This kind of data integration enabled value-added functions such as airspace proximity notification, and enhanced onboard situational awareness and decision support. Use of the OGC standards provided flexible attribute, spatial and temporal information retrievals, although fine-grained retrieval of AIXM feature timeslices still needs some additional work.

The Pilot Study also demonstrated a variety of OGC standards-based client component interfaces, including a lightweight, browser-based client, and thick client tools with more advanced data analysis and rendering capabilities. Using both AIXM 5.0 and AIXM 5.1 SAA schema extensions provided a good beta test for the use of AIXM-format airspace definitions and airspace reservations, and provided insight to industry on how to leverage these AIXM-compliant formats for data dissemination using OGC web service components.

Annex A
SAA XML Schema Extensions

This Annex includes two sets of four XML schema documents representing the SAA and SUA schema extensions for both AIXM 5.0 and AIXM 5.1, listed below.

AIXM 5.0 SAA Extensions	AIXM 5.1 SAA Extensions
SAA-DataTypes.xsd	SAA-DataTypes.xsd
SAA-Feature.xsd	SAA-Feature.xsd
SAA-Message.xsd	SAA-Message.xsd
SUA-Feature.xsd	SUA-Feature.xsd

These SAA-specific schema extend the baseline AIXM 5.0 and AIXM 5.1 schema that are available from http://www.aixm.aero. A folder that contains each version-specific set of four schema extensions needs to have subfolders named ISO_19136_Schemas, ISO_19139_Schemas, and xlink; with each subfolder containing the appropriate GML and xlink schemas, in order for all of the SAA XML schema extension references to be resolved.

AIXM 5.0 – SAA-DataTypes.xsd

```
<?xml version="1.0" encoding="windows-1252"?>
<!-- Generated from Rational Rose 2006.0.0.060314 -->
<!-- Script: AicmToXmlSchema.A.ebs -->
<!-- Date:      09-17-2009  11:16:19 -->
<!-- Model:     E:\AIXM\design\AIXM-5-0-200803061118.mdl -->
<!-- Component: Logical View::AIXM Application Schemas::AIM SAA::SAA Data Types -->
<!-- Version:  -->
<!-- XML-Schema level supported is specified by W3C -->
<!--    http://www.w3.org/2001/XMLSchema/ -->
<schema xmlns:aimsaa="urn:us:gov:dot:faa:aim:saa" xmlns="http://www.w3.org/2001/XMLSchema"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:gml="http://www.opengis.net/gml/3.2"
xmlns:xlink="http://www.w3.org/1999/xlink" targetNamespace="urn:us:gov:dot:faa:aim:saa"
elementFormDefault="qualified" attributeFormDefault="unqualified">
    <annotation>
        <appinfo>

    <gml:gmlProfileSchema>http://www.aixm.aero/schema/5.0/profile/gml4aixm.xsd</gml:gmlProfileSchema>
        </appinfo>
    </annotation>
    <annotation>
        <appinfo source="www.aixm.aero/schema/5.0">AIXM_DataTypes.xsd</appinfo>
    </annotation>
    <simpleType name="CodeSpecialActivityAirspaceType_base">
        <restriction base="xsd:string">
            <enumeration value="SUA">
                <annotation>
                    <documentation/>
                </annotation>
            </enumeration>
            <enumeration value="ATCAA"/>
            <enumeration value="TFR"/>
            <enumeration value="ALTRV"/>
            <enumeration value="LAA"/>
            <enumeration value="SAA_COMPONENT">
                <annotation>
```

```
                <documentation/>
            </annotation>
        </enumeration>
    </restriction>
</simpleType>
<simpleType name="CodeSpecialActivityAirspaceType">
    <union memberTypes="aimsaa:CodeSpecialActivityAirspaceType_base xsd:string"/>
</simpleType>
<simpleType name="CodeSpecialActivityAirspaceStatusType_base">
    <restriction base="xsd:string">
        <enumeration value="HOT"/>
        <enumeration value="COLD"/>
        <enumeration value="PARTIAL"/>
    </restriction>
</simpleType>
<simpleType name="CodeSpecialActivityAirspaceStatusType">
    <union memberTypes="aimsaa:CodeSpecialActivityAirspaceStatusType_base xsd:string"/>
</simpleType>
<simpleType name="CodeSeparationRuleType_base">
    <restriction base="xsd:string">
        <enumeration value="AIRCRAFT"/>
        <enumeration value="OTHER"/>
        <enumeration value="UNSPECIFIED"/>
    </restriction>
</simpleType>
<simpleType name="CodeSeparationRuleType">
    <union memberTypes="aimsaa:CodeSeparationRuleType_base xsd:string"/>
</simpleType>
<simpleType name="CodeAdminAreaType">
    <restriction base="xsd:string">
        <enumeration value="ALASKA"/>
        <enumeration value="ALABAMA"/>
        <enumeration value="ARKANSAS"/>
        <enumeration value="AMERICAN SAMOA"/>
        <enumeration value="ARIZONA"/>
        <enumeration value="CALIFORNIA"/>
        <enumeration value="COLORADO"/>
        <enumeration value="N MARIANA ISLANDS"/>
        <enumeration value="CONNECTICUT"/>
        <enumeration value="CANAL ZONE"/>
        <enumeration value="DIST. OF COLUMBIA"/>
        <enumeration value="DELAWARE"/>
        <enumeration value="FLORIDA"/>
        <enumeration value="GEORGIA"/>
        <enumeration value="GUAM"/>
        <enumeration value="HAWAII"/>
        <enumeration value="IOWA"/>
        <enumeration value="IDAHO"/>
        <enumeration value="ILLINOIS"/>
        <enumeration value="INDIANA"/>
        <enumeration value="KANSAS"/>
        <enumeration value="KENTUCKY"/>
        <enumeration value="LOUISIANA"/>
        <enumeration value="MASSACHUSETTS"/>
        <enumeration value="MARYLAND"/>
        <enumeration value="MAINE"/>
        <enumeration value="MICHIGAN"/>
        <enumeration value="MINNESOTA"/>
        <enumeration value="MISSOURI"/>
        <enumeration value="MISSISSIPPI"/>
        <enumeration value="MONTANA"/>
        <enumeration value="NORTH CAROLINA"/>
        <enumeration value="NORTH DAKOTA"/>
        <enumeration value="NEBRASKA"/>
        <enumeration value="NEW HAMPSHIRE"/>
        <enumeration value="NEW JERSEY"/>
        <enumeration value="NEW MEXICO"/>
        <enumeration value="NEVADA"/>
        <enumeration value="NEW YORK"/>
        <enumeration value="OFFSHORE ATLANTIC"/>
        <enumeration value="OFFSHORE CARIB"/>
        <enumeration value="OFFSHORE GULF"/>
```

```
            <enumeration value="OFFSHORE PACIFIC"/>
            <enumeration value="OHIO"/>
            <enumeration value="OKLAHOMA"/>
            <enumeration value="OREGON"/>
            <enumeration value="PENNSYLVANIA"/>
            <enumeration value="PUERTO RICO"/>
            <enumeration value="RHODE ISLAND"/>
            <enumeration value="SOUTH CAROLINA"/>
            <enumeration value="SOUTH DAKOTA"/>
            <enumeration value="TENNESSEE"/>
            <enumeration value="TEXAS"/>
            <enumeration value="UTAH"/>
            <enumeration value="VIRGINIA"/>
            <enumeration value="VIRGIN ISLANDS"/>
            <enumeration value="VERMONT"/>
            <enumeration value="WASHINGTON"/>
            <enumeration value="WISCONSIN"/>
            <enumeration value="WAKE ISLAND"/>
            <enumeration value="WEST VIRGINIA"/>
            <enumeration value="WYOMING"/>
            <enumeration value="MIDWAY ATOLL"/>
            <enumeration value="BRITISH WEST INDIES"/>
            <enumeration value="BAHAMA ISLANDS"/>
            <enumeration value="INTERNATIONAL"/>
        </restriction>
    </simpleType>
    <simpleType name="CodeLegalDefinitionType_base">
        <restriction base="xsd:string">
            <enumeration value="GENERATED"/>
            <enumeration value="HUMAN"/>
        </restriction>
    </simpleType>
    <simpleType name="CodeLegalDefinitionType">
        <union memberTypes="aimsaa:CodeLegalDefinitionType_base xsd:string"/>
    </simpleType>
    <simpleType name="CodeSpecialUseAirspaceType_base">
        <restriction base="xsd:string">
            <enumeration value="MOA">
                <annotation>
                    <documentation/>
                </annotation>
            </enumeration>
            <enumeration value="NSA">
                <annotation>
                    <documentation/>
                </annotation>
            </enumeration>
            <enumeration value="CFA">
                <annotation>
                    <documentation/>
                </annotation>
            </enumeration>
            <enumeration value="PA">
                <annotation>
                    <documentation/>
                </annotation>
            </enumeration>
            <enumeration value="RA">
                <annotation>
                    <documentation/>
                </annotation>
            </enumeration>
            <enumeration value="WA">
                <annotation>
                    <documentation/>
                </annotation>
            </enumeration>
            <enumeration value="AA">
                <annotation>
                    <documentation/>
                </annotation>
```

```
            </enumeration>
        </restriction>
    </simpleType>
    <simpleType name="CodeSpecialUseAirspaceType">
        <union memberTypes="aimsaa:CodeSpecialUseAirspaceType_base xsd:string"/>
    </simpleType>
</schema>
```

AIXM 5.0 – SAA-Feature.xsd

```
<?xml version="1.0" encoding="utf-8"?>
<!-- Generated automatically by Rational Rose 2006.0.0.060314 -->
<!-- Date:      01-21-2010  12:46:08 -->
<!-- Model:     E:\AIXM\design\AIXM-5-0-200803061118.mdl -->
<!-- Component: Logical View::AIXM Application Schemas::AIM SAA::SAA Feature -->
<!-- Version:  -->
<!-- XML-Schema level supported is specified by W3C -->
<!--     http://www.w3.org/2001/XMLSchema/ -->
<schema xmlns:aimsaa="urn:us:gov:dot:faa:aim:saa" xmlns="http://www.w3.org/2001/XMLSchema"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:gml="http://www.opengis.net/gml/3.2"
xmlns:xlink="http://www.w3.org/1999/xlink" xmlns:aixm="http://www.aixm.aero/schema/5.0"
targetNamespace="urn:us:gov:dot:faa:aim:saa" elementFormDefault="qualified"
attributeFormDefault="unqualified">
    <import namespace="http://www.opengis.net/gml/3.2" schemaLocation="./ISO_19136_Schemas/gml.xsd"/>
    <import namespace="http://www.w3.org/1999/xlink" schemaLocation="./xlink/xlinks.xsd"/>
    <import namespace="http://www.aixm.aero/schema/5.0" schemaLocation="./AIXM_DataTypes.xsd"/>
    <import namespace="http://www.aixm.aero/schema/5.0" schemaLocation="./AIXM_Features.xsd"/>
    <import namespace="http://www.aixm.aero/schema/5.0"
schemaLocation="./AIXM_AbstractGML_ObjectTypes.xsd"/>
    <include schemaLocation="./SAA-DataTypes.xsd"/>
    <annotation>
        <appinfo>

    <gml:gmlProfileSchema>http://www.aixm.aero/schema/5.0/profile/gml4aixm.xsd</gml:gmlProfileSchema>
        </appinfo>
    </annotation>
    <annotation>
        <appinfo source="www.aixm.aero/schema/5.0">AIXM_Features.xsd</appinfo>
    </annotation>
    <element name="AirspaceExtension" type="aimsaa:AirspaceExtensionType"
substitutionGroup="aixm:AbstractAirspaceExtension"/>
    <complexType name="AirspaceExtensionType">
        <complexContent>
            <extension base="aixm:AbstractExtensionType">
                <sequence>
                    <group ref="aimsaa:AirspaceExtensionPropertyGroup"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
    <group name="AirspaceExtensionPropertyGroup">
        <sequence>
            <element name="saaType" nillable="true" minOccurs="0">
                <complexType>
                    <simpleContent>
                        <extension base="aimsaa:CodeSpecialActivityAirspaceType">
                            <attribute name="nilReason" type="gml:NilReasonEnumeration"/>
                        </extension>
                    </simpleContent>
                </complexType>
            </element>
            <element name="timeInAdvance" nillable="true" minOccurs="0">
                <complexType>
                    <simpleContent>
                        <extension base="aixm:ValDurationType">
                            <attribute name="nilReason" type="gml:NilReasonEnumeration"/>
                        </extension>
                    </simpleContent>
                </complexType>
            </element>
```

```
            <element name="administrativeArea" nillable="true" minOccurs="0">
                <complexType>
                    <simpleContent>
                        <extension base="aimsaa:CodeAdminAreaType">
                            <attribute name="nilReason" type="gml:NilReasonEnumeration"/>
                        </extension>
                    </simpleContent>
                </complexType>
            </element>
            <element name="city" nillable="true" minOccurs="0">
                <complexType>
                    <simpleContent>
                        <extension base="aixm:TextNameType">
                            <attribute name="nilReason" type="gml:NilReasonEnumeration"/>
                        </extension>
                    </simpleContent>
                </complexType>
            </element>
            <element name="conditionalExclusion" nillable="true" minOccurs="0">
                <complexType>
                    <simpleContent>
                        <extension base="aixm:CodeYesNoType">
                            <attribute name="nilReason" type="gml:NilReasonEnumeration"/>
                        </extension>
                    </simpleContent>
                </complexType>
            </element>
            <element name="legalDefinitionType" nillable="true" minOccurs="0">
                <complexType>
                    <simpleContent>
                        <extension base="aimsaa:CodeLegalDefinitionType">
                            <attribute name="nilReason" type="gml:NilReasonEnumeration"/>
                        </extension>
                    </simpleContent>
                </complexType>
            </element>
        </sequence>
    </group>
    <element name="TimesheetExtension" type="aimsaa:TimesheetExtensionType"
substitutionGroup="aixm:AbstractTimesheetExtension"/>
    <complexType name="TimesheetExtensionType">
        <complexContent>
            <extension base="aixm:AbstractExtensionType">
                <sequence>
                    <group ref="aimsaa:TimesheetExtensionPropertyGroup"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
    <group name="TimesheetExtensionPropertyGroup">
        <sequence>
            <element name="intermittent" nillable="true" minOccurs="0">
                <complexType>
                    <simpleContent>
                        <extension base="aixm:CodeYesNoType">
                            <attribute name="nilReason" type="gml:NilReasonEnumeration"/>
                        </extension>
                    </simpleContent>
                </complexType>
            </element>
            <element name="timeOffset" nillable="true" minOccurs="0">
                <complexType>
                    <simpleContent>
                        <extension base="aixm:ValDurationType">
                            <attribute name="nilReason" type="gml:NilReasonEnumeration"/>
                        </extension>
                    </simpleContent>
                </complexType>
            </element>
        </sequence>
    </group>
```

```
    <element name="AirspaceUsageExtension" type="aimsaa:AirspaceUsageExtensionType"
substitutionGroup="aixm:AbstractAirspaceUsageExtension"/>
    <complexType name="AirspaceUsageExtensionType">
        <complexContent>
            <extension base="aixm:AbstractExtensionType">
                <sequence>
                    <group ref="aimsaa:AirspaceUsageExtensionPropertyGroup"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
    <group name="AirspaceUsageExtensionPropertyGroup">
        <sequence>
            <element name="daylightSavings" nillable="true" minOccurs="0">
                <complexType>
                    <simpleContent>
                        <extension base="aixm:CodeYesNoType">
                            <attribute name="nilReason" type="gml:NilReasonEnumeration"/>
                        </extension>
                    </simpleContent>
                </complexType>
            </element>
            <element name="approver" nillable="true" minOccurs="0">
                <complexType>
                    <complexContent>
                        <extension base="aixm:ContactInformationPropertyType">
                            <attribute name="nilReason" type="gml:NilReasonEnumeration"/>
                        </extension>
                    </complexContent>
                </complexType>
            </element>
            <element name="creator" nillable="true" minOccurs="0">
                <complexType>
                    <complexContent>
                        <extension base="aixm:ContactInformationPropertyType">
                            <attribute name="nilReason" type="gml:NilReasonEnumeration"/>
                        </extension>
                    </complexContent>
                </complexType>
            </element>
            <element name="lastModifier" nillable="true" minOccurs="0">
                <complexType>
                    <complexContent>
                        <extension base="aixm:ContactInformationPropertyType">
                            <attribute name="nilReason" type="gml:NilReasonEnumeration"/>
                        </extension>
                    </complexContent>
                </complexType>
            </element>
        </sequence>
    </group>
    <complexType name="SaaGroupPropertyType">
        <sequence>
            <element ref="aimsaa:SaaGroup"/>
        </sequence>
        <attributeGroup ref="gml:OwnershipAttributeGroup"/>
    </complexType>
    <element name="SaaGroup" type="aimsaa:SaaGroupType"/>
    <complexType name="SaaGroupType">
        <sequence>
            <group ref="aimsaa:SaaGroupPropertyGroup"/>
            <element name="extension" minOccurs="0" maxOccurs="unbounded">
                <complexType>
                    <sequence>
                        <element ref="aimsaa:AbstractSaaGroupExtension"/>
                    </sequence>
                    <attributeGroup ref="gml:OwnershipAttributeGroup"/>
                </complexType>
            </element>
        </sequence>
    </complexType>
    <group name="SaaGroupPropertyGroup">
```

```
        <sequence>
            <element name="name" nillable="true" minOccurs="0">
                <complexType>
                    <simpleContent>
                        <extension base="aixm:TextNameType">
                            <attribute name="nilReason" type="gml:NilReasonEnumeration"/>
                        </extension>
                    </simpleContent>
                </complexType>
            </element>
            <element name="groupMember" type="aixm:AirspacePropertyType" nillable="true"
minOccurs="0" maxOccurs="unbounded"/>
        </sequence>
    </group>
    <element name="AbstractSaaGroupExtension" type="aixm:AbstractExtensionType" abstract="true"
substitutionGroup="aixm:AbstractExtension"/>
    <element name="InformationServiceExtension" type="aimsaa:InformationServiceExtensionType"
substitutionGroup="aixm:AbstractInformationServiceExtension"/>
    <complexType name="InformationServiceExtensionType">
        <complexContent>
            <extension base="aixm:AbstractExtensionType">
                <sequence>
                    <group ref="aimsaa:InformationServiceExtensionPropertyGroup"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
    <group name="InformationServiceExtensionPropertyGroup">
        <sequence>
            <element name="channelAllocation" nillable="true" minOccurs="0" maxOccurs="unbounded">
                <complexType>
                    <complexContent>
                        <extension base="aimsaa:RadioCommunicationChannelAllocationPropertyType">
                            <attribute name="nilReason" type="gml:NilReasonEnumeration"/>
                        </extension>
                    </complexContent>
                </complexType>
            </element>
        </sequence>
    </group>
    <complexType name="RadioCommunicationChannelAllocationPropertyType">
        <sequence>
            <element ref="aimsaa:RadioCommunicationChannelAllocation"/>
        </sequence>
        <attributeGroup ref="gml:OwnershipAttributeGroup"/>
    </complexType>
    <element name="RadioCommunicationChannelAllocation"
type="aimsaa:RadioCommunicationChannelAllocationType"/>
    <complexType name="RadioCommunicationChannelAllocationType">
        <sequence>
            <group ref="aimsaa:RadioCommunicationChannelAllocationPropertyGroup"/>
        </sequence>
    </complexType>
    <group name="RadioCommunicationChannelAllocationPropertyGroup">
        <sequence>
            <element name="associatedAirspace" type="aixm:AirspacePropertyType" nillable="true"
minOccurs="0"/>
            <element name="allocatedChannelDetails" nillable="true" minOccurs="0"
maxOccurs="unbounded">
                <complexType>
                    <complexContent>
                        <extension base="aimsaa:SaaRadioCommunicationChannelPropertyType">
                            <attribute name="nilReason" type="gml:NilReasonEnumeration"/>
                        </extension>
                    </complexContent>
                </complexType>
            </element>
        </sequence>
    </group>
    <element name="AbstractRadioCommunicationChannelAllocationExtension"
type="aixm:AbstractExtensionType" abstract="true" substitutionGroup="aixm:AbstractExtension"/>
```

```
    <complexType name="SaaRadioCommunicationChannelPropertyType">
        <sequence>
            <element ref="aimsaa:SaaRadioCommunicationChannel"/>
        </sequence>
        <attributeGroup ref="gml:OwnershipAttributeGroup"/>
    </complexType>
    <element name="SaaRadioCommunicationChannel" type="aimsaa:SaaRadioCommunicationChannelType"/>
    <complexType name="SaaRadioCommunicationChannelType">
        <sequence>
            <group ref="aimsaa:SaaRadioCommunicationChannelPropertyGroup"/>
        </sequence>
    </complexType>
    <group name="SaaRadioCommunicationChannelPropertyGroup">
        <sequence>
            <element name="communicationAllowed" nillable="true" minOccurs="0">
                <complexType>
                    <simpleContent>
                        <extension base="aixm:CodeMilitaryStatusType">
                            <attribute name="nilReason" type="gml:NilReasonEnumeration"/>
                        </extension>
                    </simpleContent>
                </complexType>
            </element>
            <element name="sectors" nillable="true" minOccurs="0">
                <complexType>
                    <simpleContent>
                        <extension base="aixm:TextNameType">
                            <attribute name="nilReason" type="gml:NilReasonEnumeration"/>
                        </extension>
                    </simpleContent>
                </complexType>
            </element>
            <element name="altitudes" nillable="true" minOccurs="0">
                <complexType>
                    <simpleContent>
                        <extension base="aixm:TextNameType">
                            <attribute name="nilReason" type="gml:NilReasonEnumeration"/>
                        </extension>
                    </simpleContent>
                </complexType>
            </element>
            <element name="charted" nillable="true" minOccurs="0">
                <complexType>
                    <simpleContent>
                        <extension base="aixm:CodeYesNoType">
                            <attribute name="nilReason" type="gml:NilReasonEnumeration"/>
                        </extension>
                    </simpleContent>
                </complexType>
            </element>
            <element name="associatedChannel" type="aixm:RadioCommunicationChannelPropertyType"
nillable="true" minOccurs="0"/>
        </sequence>
    </group>
</schema>
```

AIXM 5.0 – SAA-Message.xsd

```
<?xml version="1.0" encoding="utf-8"?>
<!-- Generated automatically by Rational Rose 2006.0.0.060314 -->
<!-- Date:      09-10-2009  10:50:39 -->
<!-- Model:     E:\AIXM\design\AIXM-5-0-200803061118.mdl -->
<!-- Component: Logical View::AIXM Application Schemas::AIM SAA::SAA Messages -->
<!-- Version:  -->
<!-- XML-Schema level supported is specified by W3C -->
<!--     http://www.w3.org/2001/XMLSchema/ -->
<schema xmlns:aimsaa="urn:us:gov:dot:faa:aim:saa" xmlns:aimsua="urn:us:gov:dot:faa:aim:saa:sua"
xmlns="http://www.w3.org/2001/XMLSchema" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:gml="http://www.opengis.net/gml/3.2" xmlns:xlink="http://www.w3.org/1999/xlink"
```

```
xmlns:aixm="http://www.aixm.aero/schema/5.0" targetNamespace="urn:us:gov:dot:faa:aim:saa"
elementFormDefault="qualified" attributeFormDefault="unqualified">
    <import namespace="http://www.opengis.net/gml/3.2" schemaLocation="./ISO_19136_Schemas/gml.xsd"/>
    <import namespace="http://www.w3.org/1999/xlink" schemaLocation="./xlink/xlinks.xsd"/>
    <import namespace="http://www.aixm.aero/schema/5.0" schemaLocation="./AIXM_DataTypes.xsd"/>
    <import namespace="http://www.aixm.aero/schema/5.0" schemaLocation="./AIXM_Features.xsd"/>
    <import namespace="http://www.aixm.aero/schema/5.0"
schemaLocation="./AIXM_AbstractGML_ObjectTypes.xsd"/>
    <import namespace="urn:us:gov:dot:faa:aim:saa:sua" schemaLocation="./SUA-Feature.xsd"/>
    <include schemaLocation="./SAA-DataTypes.xsd"/>
    <include schemaLocation="./SAA-Feature.xsd"/>
    <annotation>
        <appinfo source="urn:eurocontrol:specification:aixm:schema:xsd:SAA-Message.xsd:">SAA-
Message.xsd</appinfo>
    </annotation>
    <group name="SaaMessagePropertyGroup">
        <sequence>
            <element name="hasMember" type="aimsaa:SaaMessageComponentPropertyType" minOccurs="0"
maxOccurs="unbounded"/>
        </sequence>
    </group>
    <complexType name="SaaMessageType">
        <complexContent>
            <extension base="aixm:AbstractAIXMMessageType">
                <sequence>
                    <group ref="aimsaa:SaaMessagePropertyGroup"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
    <element name="SaaMessage" type="aimsaa:SaaMessageType" substitutionGroup="gml:AbstractFeature"/>
    <complexType name="SaaMessageComponentPropertyType">
        <complexContent>
            <extension base="gml:AbstractFeatureMemberType">
                <choice>
                    <element ref="aixm:Airspace"/>
                    <element ref="aixm:Unit"/>
                    <element ref="aixm:AirspaceUsage"/>
                    <element ref="aixm:RadioCommunicationChannel"/>
                    <element ref="aixm:AirTrafficControlService"/>
                    <element ref="aixm:InformationService"/>
                    <element ref="aixm:OrganisationAuthority"/>
                    <element ref="aixm:GeoBorder"/>
                    <element ref="aixm:AirportHeliport"/>
                    <element ref="aixm:Navaid"/>
                    <element ref="aixm:DesignatedPoint"/>
                    <element ref="aimsaa:SaaGroup"/>
                </choice>
                <attributeGroup ref="gml:AssociationAttributeGroup"/>
            </extension>
        </complexContent>
    </complexType>
    <group name="SaaScheduleMessagePropertyGroup">
        <sequence>
            <element name="hasMember" type="aimsaa:SaaScheduleMessageComponentPropertyType"
minOccurs="0" maxOccurs="unbounded"/>
        </sequence>
    </group>
    <complexType name="SaaScheduleMessageType">
        <complexContent>
            <extension base="aixm:AbstractAIXMMessageType">
                <sequence>
                    <group ref="aimsaa:SaaScheduleMessagePropertyGroup"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
    <element name="SaaScheduleMessage" type="aimsaa:SaaScheduleMessageType"
substitutionGroup="gml:AbstractFeature"/>
    <complexType name="SaaScheduleMessageComponentPropertyType">
        <complexContent>
```

```
            <extension base="gml:AbstractFeatureMemberType">
                <choice>
                    <element ref="aixm:AirspaceUsage"/>
                </choice>
                <attributeGroup ref="gml:AssociationAttributeGroup"/>
            </extension>
        </complexContent>
    </complexType>
</schema>
```

AIXM 5.0 – SUA-Feature.xsd

```
<?xml version="1.0" encoding="utf-8"?>
<!-- Generated automatically by Rational Rose 2006.0.0.060314 -->
<!-- Date:      04-17-2009  15:00:04 -->
<!-- Model:     E:\AIXM\design\AIXM-5-0-200803061118.mdl -->
<!-- Component: Logical View::AIXM Application Schemas::AIM SAA::SUA Feature -->
<!-- Version:  -->
<!-- XML-Schema level supported is specified by W3C -->
<!--    http://www.w3.org/2001/XMLSchema/ -->
<schema xmlns:aimsua="urn:us:gov:dot:faa:aim:saa:sua" xmlns:aimsaa="urn:us:gov:dot:faa:aim:saa"
xmlns="http://www.w3.org/2001/XMLSchema" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:gml="http://www.opengis.net/gml/3.2" xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:aixm="http://www.aixm.aero/schema/5.0" xmlns:ns1="urn:us:gov:dot:faa:aim:saa:sua"
targetNamespace="urn:us:gov:dot:faa:aim:saa:sua" elementFormDefault="qualified"
attributeFormDefault="unqualified">
    <import namespace="http://www.opengis.net/gml/3.2" schemaLocation="./ISO_19136_Schemas/gml.xsd"/>
    <import namespace="http://www.aixm.aero/schema/5.0" schemaLocation="./AIXM_DataTypes.xsd"/>
    <import namespace="http://www.aixm.aero/schema/5.0" schemaLocation="./AIXM_Features.xsd"/>
    <import namespace="http://www.aixm.aero/schema/5.0"
schemaLocation="./AIXM_AbstractGML_ObjectTypes.xsd"/>
    <import namespace="http://www.w3.org/1999/xlink" schemaLocation="./xlink/xlinks.xsd"/>
    <import namespace="urn:us:gov:dot:faa:aim:saa" schemaLocation="./SAA-Feature.xsd"/>
    <import namespace="urn:us:gov:dot:faa:aim:saa" schemaLocation="./SAA-DataTypes.xsd"/>
    <annotation>
        <appinfo>

    <gml:gmlProfileSchema>http://www.aixm.aero/schema/5.0/profile/gml4aixm.xsd</gml:gmlProfileSchema>
        </appinfo>
    </annotation>
    <annotation>
        <appinfo source="www.aixm.aero/schema/5.0">AIXM_Features.xsd</appinfo>
    </annotation>
    <element name="AirspaceExtension" type="aimsua:AirspaceExtensionType"
substitutionGroup="aixm:AbstractAirspaceExtension"/>
    <complexType name="AirspaceExtensionType">
        <complexContent>
            <extension base="aixm:AbstractExtensionType">
                <sequence>
                    <group ref="aimsua:AirspaceExtensionPropertyGroup"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
    <group name="AirspaceExtensionPropertyGroup">
        <sequence>
            <element name="suaType" nillable="true" minOccurs="0">
                <complexType>
                    <simpleContent>
                        <extension base="aimsaa:CodeSpecialUseAirspaceType">
                            <attribute name="nilReason" type="gml:NilReasonEnumeration"/>
                        </extension>
                    </simpleContent>
                </complexType>
            </element>
            <element name="lightsOut" nillable="true" minOccurs="0">
                <complexType>
                    <simpleContent>
                        <extension base="aixm:CodeYesNoType">
                            <attribute name="nilReason" type="gml:NilReasonEnumeration"/>
```

```
                    </extension>
                </simpleContent>
            </complexType>
        </element>
        <element name="ICAO_NOTAM" nillable="true" minOccurs="0">
            <complexType>
                <simpleContent>
                    <extension base="aixm:CodeYesNoType">
                        <attribute name="nilReason" type="gml:NilReasonEnumeration"/>
                    </extension>
                </simpleContent>
            </complexType>
        </element>
        <element name="separationRule" nillable="true" minOccurs="0">
            <complexType>
                <simpleContent>
                    <extension base="aimsaa:CodeSeparationRuleType">
                        <attribute name="nilReason" type="gml:NilReasonEnumeration"/>
                    </extension>
                </simpleContent>
            </complexType>
        </element>
    </sequence>
</group>
</schema>
```

AIXM 5.1 – SAA-DataTypes.xsd

```
<?xml version="1.0" encoding="windows-1252"?>
<!-- Generated from Rational Rose 2006.0.0.060314 -->
<!-- Script: AIXM-DataTypeGenerator.ebs -->
<!-- Date:      02-08-2011  17:28:58 -->
<!-- Model:     C:\AIXM\design\5.1\AIXM-5-1-20100201.mdl -->
<!-- Component: Logical View::AIXM Application Schemas::AIM SAA::SAA-DataTypes -->
<!-- Version:  -->
<!-- XML-Schema level supported is specified by W3C -->
<!--     http://www.w3.org/2001/XMLSchema/ -->
<schema xmlns:aimsaa="urn:us:gov:dot:faa:aim:saa:5.1" xmlns="http://www.w3.org/2001/XMLSchema"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:gml="http://www.opengis.net/gml/3.2"
xmlns:xlink="http://www.w3.org/1999/xlink" targetNamespace="urn:us:gov:dot:faa:aim:saa:5.1"
elementFormDefault="qualified" attributeFormDefault="unqualified">
    <annotation>
        <appinfo>

    <gml:gmlProfileSchema>http://www.aixm.aero/schema/5.1/profile/gml4aixm.xsd</gml:gmlProfileSchema>
        </appinfo>
    </annotation>
    <annotation>
        <appinfo source="www.aixm.aero/schema/5.1">AIXM_DataTypes.xsd</appinfo>
    </annotation>
    <simpleType name="CodeSpecialActivityAirspaceType">
        <restriction base="xsd:string">
            <enumeration value="SUA"/>
            <enumeration value="ATCAA"/>
            <enumeration value="TFR"/>
            <enumeration value="ALTRV"/>
            <enumeration value="LAA"/>
            <enumeration value="SAA_COMPONENT"/>
        </restriction>
    </simpleType>
    <simpleType name="CodeSpecialUseAirspaceType">
        <restriction base="xsd:string">
            <enumeration value="MOA"/>
            <enumeration value="NSA"/>
            <enumeration value="CFA"/>
            <enumeration value="PA"/>
            <enumeration value="RA"/>
            <enumeration value="WA"/>
            <enumeration value="AA"/>
        </restriction>
    </simpleType>
    <simpleType name="CodeLegalDefinitionType">
        <restriction base="xsd:string">
            <enumeration value="GENERATED"/>
            <enumeration value="HUMAN"/>
        </restriction>
    </simpleType>
    <simpleType name="CodeAdminAreaType">
        <restriction base="xsd:string">
            <enumeration value="ALASKA"/>
            <enumeration value="ALABAMA"/>
            <enumeration value="ARKANSAS"/>
            <enumeration value="AMERICAN SAMOA"/>
            <enumeration value="ARIZONA"/>
            <enumeration value="CALIFORNIA"/>
            <enumeration value="COLORADO"/>
            <enumeration value="CONNECTICUT"/>
            <enumeration value="CANAL ZONE"/>
            <enumeration value="DIST. OF COLUMBIA"/>
            <enumeration value="DELAWARE"/>
            <enumeration value="FLORIDA"/>
            <enumeration value="GEORGIA"/>
            <enumeration value="GUAM"/>
            <enumeration value="HAWAII"/>
            <enumeration value="IOWA"/>
```

```
            <enumeration value="IDAHO"/>
            <enumeration value="ILLINOIS"/>
            <enumeration value="INDIANA"/>
            <enumeration value="KANSAS"/>
            <enumeration value="KENTUCKY"/>
            <enumeration value="LOUISIANA"/>
            <enumeration value="MASSACHUSETTS"/>
            <enumeration value="MARYLAND"/>
            <enumeration value="MAINE"/>
            <enumeration value="MICHIGAN"/>
            <enumeration value="MINNESOTA"/>
            <enumeration value="MISSOURI"/>
            <enumeration value="MISSISSIPPI"/>
            <enumeration value="MONTANA"/>
            <enumeration value="N MARIANA ISLANDS"/>
            <enumeration value="NORTH CAROLINA"/>
            <enumeration value="NORTH DAKOTA"/>
            <enumeration value="NEBRASKA"/>
            <enumeration value="NEW HAMPSHIRE"/>
            <enumeration value="NEW JERSEY"/>
            <enumeration value="NEW MEXICO"/>
            <enumeration value="NEVADA"/>
            <enumeration value="NEW YORK"/>
            <enumeration value="OFFSHORE ATLANTIC"/>
            <enumeration value="OFFSHORE CARIB"/>
            <enumeration value="OFFSHORE GULF"/>
            <enumeration value="OFFSHORE PACIFIC"/>
            <enumeration value="OHIO"/>
            <enumeration value="OKLAHOMA"/>
            <enumeration value="OREGON"/>
            <enumeration value="PENNSYLVANIA"/>
            <enumeration value="PUERTO RICO"/>
            <enumeration value="RHODE ISLAND"/>
            <enumeration value="SOUTH CAROLINA"/>
            <enumeration value="SOUTH DAKOTA"/>
            <enumeration value="TENNESSEE"/>
            <enumeration value="TEXAS"/>
            <enumeration value="UTAH"/>
            <enumeration value="VIRGINIA"/>
            <enumeration value="VIRGIN ISLANDS"/>
            <enumeration value="VERMONT"/>
            <enumeration value="WASHINGTON"/>
            <enumeration value="WISCONSIN"/>
            <enumeration value="WAKE ISLAND"/>
            <enumeration value="WEST VIRGINIA"/>
            <enumeration value="WYOMING"/>
            <enumeration value="MIDWAY ATOLL"/>
            <enumeration value="BRITISH WEST INDIES"/>
            <enumeration value="BAHAMA ISLANDS"/>
            <enumeration value="INTERNATIONAL"/>
        </restriction>
    </simpleType>
    <simpleType name="CodeAirspaceReservationPhaseType">
        <restriction base="xsd:string">
            <enumeration value="PENDING"/>
            <enumeration value="APPROVED"/>
            <enumeration value="DISAPPROVED"/>
            <enumeration value="CANCELED"/>
        </restriction>
    </simpleType>
    <simpleType name="CodeReservationUserActionType">
        <restriction base="xsd:string">
            <enumeration value="ACKNOWLEDGE"/>
            <enumeration value="APPROVE"/>
            <enumeration value="CANCEL"/>
            <enumeration value="CREATE"/>
            <enumeration value="DISAPPROVE"/>
            <enumeration value="GO_COLD"/>
            <enumeration value="GO_HOT"/>
            <enumeration value="MODIFY"/>
        </restriction>
```

```
    </simpleType>
    <simpleType name="CodeTimeInAdvanceType">
        <union>
            <simpleType>
                <restriction base="xsd:string">
                    <enumeration value="NOTAM"/>
                    <enumeration value="AMENDMENTS"/>
                    <enumeration value="ENTRY_NOTIFICATION"/>
                    <enumeration value="EXIT_NOTIFICATION"/>
                    <enumeration value="FILE_FLIGHT_PLAN"/>
                    <enumeration value="WEEKLY_SCHEDULE"/>
                </restriction>
            </simpleType>
            <simpleType>
                <restriction base="string">
                    <pattern value="OTHER(:(\w|_){1,58})?"/>
                </restriction>
            </simpleType>
        </union>
    </simpleType>
    <simpleType name="CodeSeparationStandardType">
        <restriction base="xsd:string">
            <enumeration value="AVIATION"/>
            <enumeration value="NON-AVIATION"/>
            <enumeration value="UNSPECIFIED"/>
        </restriction>
    </simpleType>
</schema>
```

AIXM 5.1 – SAA-Feature.xsd

```
<?xml version="1.0" encoding="utf-8"?>
<!-- Generated automatically by Rational Rose 2006.0.0.060314 -->
<!-- Date:      02-08-2011  17:22:14 -->
<!-- Model:     C:\AIXM\design\5.1\AIXM-5-1-20100201.mdl -->
<!-- Component: Logical View::AIXM Application Schemas::AIM SAA::SAA-Feature -->
<!-- Version:  -->
<!-- XML-Schema level supported is specified by W3C -->
<!--     http://www.w3.org/2001/XMLSchema/ -->
<schema xmlns:aimsaa="urn:us:gov:dot:faa:aim:saa:5.1" xmlns="http://www.w3.org/2001/XMLSchema"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:gml="http://www.opengis.net/gml/3.2"
xmlns:xlink="http://www.w3.org/1999/xlink" xmlns:aixm="http://www.aixm.aero/schema/5.1"
targetNamespace="urn:us:gov:dot:faa:aim:saa:5.1" elementFormDefault="qualified"
attributeFormDefault="unqualified">
    <import namespace="http://www.opengis.net/gml/3.2" schemaLocation="./ISO_19136_Schemas/gml.xsd"/>
    <import namespace="http://www.w3.org/1999/xlink" schemaLocation="./xlink/xlinks.xsd"/>
    <import namespace="http://www.aixm.aero/schema/5.1" schemaLocation="./AIXM_DataTypes.xsd"/>
    <import namespace="http://www.aixm.aero/schema/5.1" schemaLocation="./AIXM_Features.xsd"/>
    <import namespace="http://www.aixm.aero/schema/5.1"
schemaLocation="./AIXM_AbstractGML_ObjectTypes.xsd"/>
    <include schemaLocation="./SAA-DataTypes.xsd"/>
    <annotation>
        <appinfo>

    <gml:gmlProfileSchema>http://www.aixm.aero/schema/5.0/profile/gml4aixm.xsd</gml:gmlProfileSchema>
        </appinfo>
    </annotation>
    <annotation>
        <appinfo source="www.aixm.aero/schema/5.0">AIXM_Features.xsd</appinfo>
    </annotation>
    <complexType name="RadioCommunicationChannelAllocationPropertyType">
        <complexContent>
            <extension base="aixm:AbstractAIXMPropertyType">
                <sequence>
                    <element ref="aimsaa:RadioCommunicationChannelAllocation"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
```

```
    <element name="RadioCommunicationChannelAllocation"
type="aimsaa:RadioCommunicationChannelAllocationType"/>
    <complexType name="RadioCommunicationChannelAllocationType">
        <complexContent>
            <extension base="aixm:AbstractAIXMObjectType">
                <sequence>
                    <group ref="aimsaa:RadioCommunicationChannelAllocationPropertyGroup"/>
                    <element name="extension" minOccurs="0" maxOccurs="unbounded">
                        <complexType>
                            <sequence>
                                <element
ref="aimsaa:AbstractRadioCommunicationChannelAllocationExtension"/>
                            </sequence>
                            <attributeGroup ref="gml:OwnershipAttributeGroup"/>
                        </complexType>
                    </element>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
    <group name="RadioCommunicationChannelAllocationPropertyGroup">
        <sequence>
            <element name="allocatedChannelDetails"
type="aimsaa:SaaRadioCommunicationChannelPropertyType" nillable="true" minOccurs="0"
maxOccurs="unbounded"/>
            <element name="associatedAirspace" type="aixm:AirspacePropertyType" nillable="true"
minOccurs="0"/>
        </sequence>
    </group>
    <element name="AbstractRadioCommunicationChannelAllocationExtension"
type="aixm:AbstractExtensionType" abstract="true" substitutionGroup="aixm:AbstractExtension"/>
    <complexType name="SaaRadioCommunicationChannelPropertyType">
        <complexContent>
            <extension base="aixm:AbstractAIXMPropertyType">
                <sequence>
                    <element ref="aimsaa:SaaRadioCommunicationChannel"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
    <element name="SaaRadioCommunicationChannel" type="aimsaa:SaaRadioCommunicationChannelType"/>
    <complexType name="SaaRadioCommunicationChannelType">
        <complexContent>
            <extension base="aixm:AbstractAIXMObjectType">
                <sequence>
                    <group ref="aimsaa:SaaRadioCommunicationChannelPropertyGroup"/>
                    <element name="extension" minOccurs="0" maxOccurs="unbounded">
                        <complexType>
                            <sequence>
                                <element
ref="aimsaa:AbstractSaaRadioCommunicationChannelExtension"/>
                            </sequence>
                            <attributeGroup ref="gml:OwnershipAttributeGroup"/>
                        </complexType>
                    </element>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
    <group name="SaaRadioCommunicationChannelPropertyGroup">
        <sequence>
            <element name="communicationAllowed" type="aixm:CodeMilitaryStatusType" nillable="true"
minOccurs="0"/>
            <element name="sectors" type="aixm:TextNameType" nillable="true" minOccurs="0"/>
            <element name="altitudes" type="aixm:TextNameType" nillable="true" minOccurs="0"/>
            <element name="charted" type="aixm:CodeYesNoType" nillable="true" minOccurs="0"/>
            <element name="associatedChannel" type="aixm:RadioCommunicationChannelPropertyType"
nillable="true" minOccurs="0"/>
        </sequence>
    </group>
```

```
    <element name="AbstractSaaRadioCommunicationChannelExtension" type="aixm:AbstractExtensionType"
abstract="true" substitutionGroup="aixm:AbstractExtension"/>
    <element name="InformationServiceExtension" type="aimsaa:InformationServiceExtensionType"
substitutionGroup="aixm:AbstractInformationServiceExtension"/>
    <complexType name="InformationServiceExtensionType">
        <complexContent>
            <extension base="aixm:AbstractExtensionType">
                <sequence>
                    <group ref="aimsaa:InformationServiceExtensionPropertyGroup"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
    <group name="InformationServiceExtensionPropertyGroup">
        <sequence>
            <element name="channelAllocation"
type="aimsaa:RadioCommunicationChannelAllocationPropertyType" nillable="true" minOccurs="0"
maxOccurs="unbounded"/>
        </sequence>
    </group>
    <element name="AirspaceExtension" type="aimsaa:AirspaceExtensionType"
substitutionGroup="aixm:AbstractAirspaceExtension"/>
    <complexType name="AirspaceExtensionType">
        <complexContent>
            <extension base="aixm:AbstractExtensionType">
                <sequence>
                    <group ref="aimsaa:AirspaceExtensionPropertyGroup"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
    <group name="AirspaceExtensionPropertyGroup">
        <sequence>
            <element name="saaType" type="aimsaa:CodeSpecialActivityAirspaceType" nillable="true"
minOccurs="0"/>
            <element name="administrativeArea" type="aimsaa:CodeAdminAreaType" nillable="true"
minOccurs="0"/>
            <element name="city" type="aixm:TextNameType" nillable="true" minOccurs="0"/>
            <element name="legalDefinitionType" type="aimsaa:CodeLegalDefinitionType" nillable="true"
minOccurs="0"/>
            <element name="timeAhead" type="aimsaa:TimeInAdvancePropertyType" nillable="true"
minOccurs="0" maxOccurs="unbounded"/>
        </sequence>
    </group>
    <complexType name="ConditionalAirspaceExclusionPropertyType">
        <complexContent>
            <extension base="aixm:AbstractAIXMPropertyType">
                <sequence>
                    <element ref="aimsaa:ConditionalAirspaceExclusion"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
    <element name="ConditionalAirspaceExclusion" type="aimsaa:ConditionalAirspaceExclusionType"
substitutionGroup="aixm:AbstractPropertiesWithSchedule"/>
    <complexType name="ConditionalAirspaceExclusionType">
        <complexContent>
            <extension base="aixm:AbstractPropertiesWithScheduleType">
                <sequence>
                    <group ref="aimsaa:ConditionalAirspaceExclusionPropertyGroup"/>
                    <element name="extension" minOccurs="0" maxOccurs="unbounded">
                        <complexType>
                            <choice>
                                <element
ref="aimsaa:AbstractConditionalAirspaceExclusionExtension"/>
                            </choice>
                            <attributeGroup ref="gml:OwnershipAttributeGroup"/>
                        </complexType>
                    </element>
                </sequence>
            </extension>
        </complexContent>
```

```
    </complexType>
    <group name="ConditionalAirspaceExclusionPropertyGroup">
        <sequence>
            <element name="whenActive" type="aixm:CodeYesNoType" nillable="true" minOccurs="0"/>
        </sequence>
    </group>
    <element name="AbstractConditionalAirspaceExclusionExtension" type="aixm:AbstractExtensionType"
abstract="true" substitutionGroup="aixm:AbstractExtension"/>
    <element name="AirspaceVolumeExtension" type="aimsaa:AirspaceVolumeExtensionType"
substitutionGroup="aixm:AbstractAirspaceVolumeExtension"/>
    <complexType name="AirspaceVolumeExtensionType">
        <complexContent>
            <extension base="aixm:AbstractExtensionType">
                <sequence>
                    <group ref="aimsaa:AirspaceVolumeExtensionPropertyGroup"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
    <group name="AirspaceVolumeExtensionPropertyGroup">
        <sequence>
            <element name="upperLimitInclusive" type="aixm:CodeYesNoType" nillable="true"
minOccurs="0"/>
            <element name="maximumLimitInclusive" type="aixm:CodeYesNoType" nillable="true"
minOccurs="0"/>
            <element name="lowerLimitInclusive" type="aixm:CodeYesNoType" nillable="true"
minOccurs="0"/>
            <element name="minimumLimitInclusive" type="aixm:CodeYesNoType" nillable="true"
minOccurs="0"/>
        </sequence>
    </group>
    <element name="AirspaceLayerExtension" type="aimsaa:AirspaceLayerExtensionType"
substitutionGroup="aixm:AbstractAirspaceLayerExtension"/>
    <complexType name="AirspaceLayerExtensionType">
        <complexContent>
            <extension base="aixm:AbstractExtensionType">
                <sequence>
                    <group ref="aimsaa:AirspaceLayerExtensionPropertyGroup"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
    <group name="AirspaceLayerExtensionPropertyGroup">
        <sequence>
            <element name="upperLimitInclusive" type="aixm:CodeYesNoType" nillable="true"
minOccurs="0"/>
            <element name="lowerLimitInclusive" type="aixm:CodeYesNoType" nillable="true"
minOccurs="0"/>
        </sequence>
    </group>
    <element name="AirspaceActivationExtension" type="aimsaa:AirspaceActivationExtensionType"
substitutionGroup="aixm:AbstractAirspaceActivationExtension"/>
    <complexType name="AirspaceActivationExtensionType">
        <complexContent>
            <extension base="aixm:AbstractExtensionType">
                <sequence>
                    <group ref="aimsaa:AirspaceActivationExtensionPropertyGroup"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
    <group name="AirspaceActivationExtensionPropertyGroup">
        <sequence>
            <element name="creationDate" type="xsd:dateTime" nillable="true" minOccurs="0"/>
            <element name="approvalDate" type="xsd:dateTime" nillable="true" minOccurs="0"/>
            <element name="lastModifiedDate" type="xsd:dateTime" nillable="true" minOccurs="0"/>
            <element name="creator" type="aixm:ContactInformationPropertyType" nillable="true"
minOccurs="0"/>
            <element name="lastModifier" type="aixm:ContactInformationPropertyType" nillable="true"
minOccurs="0"/>
```

```
            <element name="approver" type="aixm:ContactInformationPropertyType" nillable="true"
minOccurs="0"/>
        </sequence>
    </group>
    <complexType name="TimeInAdvancePropertyType">
        <complexContent>
            <extension base="aixm:AbstractAIXMPropertyType">
                <sequence>
                    <element ref="aimsaa:TimeInAdvance"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
    <element name="TimeInAdvance" type="aimsaa:TimeInAdvanceType"/>
    <complexType name="TimeInAdvanceType">
        <complexContent>
            <extension base="aixm:AbstractAIXMObjectType">
                <sequence>
                    <group ref="aimsaa:TimeInAdvancePropertyGroup"/>
                    <element name="extension" minOccurs="0" maxOccurs="unbounded">
                        <complexType>
                            <sequence>
                                <element ref="aimsaa:AbstractTimeInAdvanceExtension"/>
                            </sequence>
                            <attributeGroup ref="gml:OwnershipAttributeGroup"/>
                        </complexType>
                    </element>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
    <group name="TimeInAdvancePropertyGroup">
        <sequence>
            <element name="type" type="aimsaa:CodeTimeInAdvanceType" nillable="true" minOccurs="0"/>
            <element name="timeInAdvance" type="aixm:ValDurationType" nillable="true" minOccurs="0"/>
            <element name="day" type="aixm:CodeDayType" nillable="true" minOccurs="0"/>
            <element name="timeOfDay" type="aixm:TimeType" nillable="true" minOccurs="0"/>
        </sequence>
    </group>
    <element name="AbstractTimeInAdvanceExtension" type="aixm:AbstractExtensionType" abstract="true"
substitutionGroup="aixm:AbstractExtension"/>
    <element name="AirspaceGeometryComponentExtension"
type="aimsaa:AirspaceGeometryComponentExtensionType"
substitutionGroup="aixm:AbstractAirspaceGeometryComponentExtension"/>
    <complexType name="AirspaceGeometryComponentExtensionType">
        <complexContent>
            <extension base="aixm:AbstractExtensionType">
                <sequence>
                    <group ref="aimsaa:AirspaceGeometryComponentExtensionPropertyGroup"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
    <group name="AirspaceGeometryComponentExtensionPropertyGroup">
        <sequence>
            <element name="conditionalExclusion"
type="aimsaa:ConditionalAirspaceExclusionPropertyType" nillable="true" minOccurs="0"
maxOccurs="unbounded"/>
        </sequence>
    </group>
    <element name="TimesheetExtension" type="aimsaa:TimesheetExtensionType"
substitutionGroup="aixm:AbstractTimesheetExtension"/>
    <complexType name="TimesheetExtensionType">
        <complexContent>
            <extension base="aixm:AbstractExtensionType">
                <sequence>
                    <group ref="aimsaa:TimesheetExtensionPropertyGroup"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
    <group name="TimesheetExtensionPropertyGroup">
```

```
        <sequence>
            <element name="issueNotam" type="aixm:CodeYesNoType" nillable="true" minOccurs="0"/>
        </sequence>
    </group>
    <complexType name="SaaGroupPropertyType">
        <attributeGroup ref="gml:OwnershipAttributeGroup"/>
        <attributeGroup ref="gml:AssociationAttributeGroup"/>
    </complexType>
    <element name="SaaGroup" type="aimsaa:SaaGroupType"
substitutionGroup="aixm:AbstractAIXMFeature"/>
    <complexType name="SaaGroupType">
        <complexContent>
            <extension base="aixm:AbstractAIXMFeatureType">
                <sequence>
                    <element name="timeSlice" type="aimsaa:SaaGroupTimeSlicePropertyType"
maxOccurs="unbounded"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
    <complexType name="SaaGroupTimeSlicePropertyType">
        <sequence>
            <element ref="aimsaa:SaaGroupTimeSlice"/>
        </sequence>
        <attributeGroup ref="gml:OwnershipAttributeGroup"/>
    </complexType>
    <element name="SaaGroupTimeSlice" type="aimsaa:SaaGroupTimeSliceType"
substitutionGroup="gml:AbstractTimeSlice"/>
    <complexType name="SaaGroupTimeSliceType">
        <complexContent>
            <extension base="aixm:AbstractAIXMTimeSliceType">
                <sequence>
                    <group ref="aimsaa:SaaGroupPropertyGroup"/>
                    <element name="extension" minOccurs="0" maxOccurs="unbounded">
                        <complexType>
                            <sequence>
                                <element ref="aimsaa:AbstractSaaGroupExtension"/>
                            </sequence>
                            <attributeGroup ref="gml:OwnershipAttributeGroup"/>
                        </complexType>
                    </element>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
    <group name="SaaGroupPropertyGroup">
        <sequence>
            <element name="name" type="aixm:TextNameType" nillable="true" minOccurs="0"/>
            <element name="groupMember" type="aixm:AirspacePropertyType" nillable="true"
minOccurs="0" maxOccurs="unbounded"/>
            <element name="activation" type="aixm:AirspaceActivationPropertyType" nillable="true"
minOccurs="0" maxOccurs="unbounded"/>
        </sequence>
    </group>
    <element name="AbstractSaaGroupExtension" type="aixm:AbstractExtensionType" abstract="true"
substitutionGroup="aixm:AbstractExtension"/>
</schema>
```

AIXM 5.1 – SAA-Message.xsd

```
<?xml version="1.0" encoding="utf-8"?>
<!-- Generated automatically by Rational Rose 2006.0.0.060314 -->
<!-- Date:      02-11-2011  11:30:04 -->
<!-- Model:     C:\AIXM\design\5.1\AIXM-5-1-20100201.mdl -->
<!-- Component: Logical View::AIXM Application Schemas::AIM SAA::SAA-Message -->
<!-- Version:  -->
<!-- XML-Schema level supported is specified by W3C -->
<!--    http://www.w3.org/2001/XMLSchema/ -->
```

```
<schema xmlns:aimsaa="urn:us:gov:dot:faa:aim:saa:5.1"
xmlns:aimsua="urn:us:gov:dot:faa:aim:saa:sua:5.1" xmlns="http://www.w3.org/2001/XMLSchema"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:gml="http://www.opengis.net/gml/3.2"
xmlns:xlink="http://www.w3.org/1999/xlink" xmlns:aixm="http://www.aixm.aero/schema/5.1"
targetNamespace="urn:us:gov:dot:faa:aim:saa:5.1" elementFormDefault="qualified"
attributeFormDefault="unqualified">
    <import namespace="http://www.opengis.net/gml/3.2" schemaLocation="./ISO_19136_Schemas/gml.xsd"/>
    <import namespace="http://www.w3.org/1999/xlink" schemaLocation="./xlink/xlinks.xsd"/>
    <import namespace="http://www.aixm.aero/schema/5.1" schemaLocation="./AIXM_DataTypes.xsd"/>
    <import namespace="http://www.aixm.aero/schema/5.1" schemaLocation="./AIXM_Features.xsd"/>
    <import namespace="http://www.aixm.aero/schema/5.1"
schemaLocation="./AIXM_AbstractGML_ObjectTypes.xsd"/>
    <import namespace="urn:us:gov:dot:faa:aim:saa:sua:5.1" schemaLocation="./SUA-Feature.xsd"/>
    <include schemaLocation="./SAA-DataTypes.xsd"/>
    <include schemaLocation="./SAA-Feature.xsd"/>
    <annotation>
        <appinfo source="urn:eurocontrol:specification:aixm:schema:xsd:SAA-Message.xsd:">SAA-
Message.xsd</appinfo>
    </annotation>
    <group name="SaaMessagePropertyGroup">
        <sequence>
            <element name="hasMember" type="aimsaa:SaaMessageComponentPropertyType" minOccurs="0"
maxOccurs="unbounded"/>
        </sequence>
    </group>
    <complexType name="SaaMessageType">
        <complexContent>
            <extension base="aixm:AbstractAIXMMessageType">
                <sequence>
                    <group ref="aimsaa:SaaMessagePropertyGroup"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
    <element name="SaaMessage" type="aimsaa:SaaMessageType" substitutionGroup="gml:AbstractFeature"/>
    <complexType name="SaaMessageComponentPropertyType">
        <complexContent>
            <extension base="gml:AbstractFeatureMemberType">
                <choice>
                    <element ref="aixm:Airspace"/>
                    <element ref="aixm:RadioCommunicationChannel"/>
                    <element ref="aixm:InformationService"/>
                    <element ref="aixm:AirTrafficControlService"/>
                    <element ref="aixm:AirportHeliport"/>
                    <element ref="aixm:Navaid"/>
                    <element ref="aixm:DesignatedPoint"/>
                    <element ref="aixm:GeoBorder"/>
                    <element ref="aixm:Unit"/>
                    <element ref="aixm:OrganisationAuthority"/>
                    <element ref="aimsaa:SaaGroup"/>
                </choice>
                <attributeGroup ref="gml:AssociationAttributeGroup"/>
            </extension>
        </complexContent>
    </complexType>
</schema>
```

AIXM 5.1 – SUA-Feature.xsd

```
<?xml version="1.0" encoding="utf-8"?>
<!-- Generated automatically by Rational Rose 2006.0.0.060314 -->
<!-- Date:      02-08-2011  18:07:45 -->
<!-- Model:     C:\AIXM\design\5.1\AIXM-5-1-20100201.mdl -->
<!-- Component: Logical View::AIXM Application Schemas::AIM SAA::SUA-Feature -->
<!-- Version:  -->
<!-- XML-Schema level supported is specified by W3C -->
<!--     http://www.w3.org/2001/XMLSchema/ -->
<schema xmlns:aimsua="urn:us:gov:dot:faa:aim:saa:sua:5.1"
xmlns:aimsaa="urn:us:gov:dot:faa:aim:saa:5.1" xmlns="http://www.w3.org/2001/XMLSchema"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:gml="http://www.opengis.net/gml/3.2"
```

```
xmlns:xlink="http://www.w3.org/1999/xlink" xmlns:aixm="http://www.aixm.aero/schema/5.1"
targetNamespace="urn:us:gov:dot:faa:aim:saa:sua:5.1" elementFormDefault="qualified"
attributeFormDefault="unqualified">
    <import namespace="http://www.opengis.net/gml/3.2" schemaLocation="./ISO_19136_Schemas/gml.xsd"/>
    <import namespace="http://www.aixm.aero/schema/5.1" schemaLocation="./AIXM_DataTypes.xsd"/>
    <import namespace="http://www.aixm.aero/schema/5.1" schemaLocation="./AIXM_Features.xsd"/>
    <import namespace="http://www.aixm.aero/schema/5.1"
schemaLocation="./AIXM_AbstractGML_ObjectTypes.xsd"/>
    <import namespace="http://www.w3.org/1999/xlink" schemaLocation="./xlink/xlinks.xsd"/>
    <import namespace="urn:us:gov:dot:faa:aim:saa:5.1" schemaLocation="./SAA-Feature.xsd"/>
    <import namespace="urn:us:gov:dot:faa:aim:saa:5.1" schemaLocation="./SAA-DataTypes.xsd"/>
    <annotation>
        <appinfo>

    <gml:gmlProfileSchema>http://www.aixm.aero/schema/5.0/profile/gml4aixm.xsd</gml:gmlProfileSchema>
        </appinfo>
    </annotation>
    <annotation>
        <appinfo source="www.aixm.aero/schema/5.0">AIXM_Features.xsd</appinfo>
    </annotation>
    <element name="AirspaceExtension" type="aimsua:AirspaceExtensionType"
substitutionGroup="aixm:AbstractAirspaceExtension"/>
    <complexType name="AirspaceExtensionType">
        <complexContent>
            <extension base="aixm:AbstractExtensionType">
                <sequence>
                    <group ref="aimsua:AirspaceExtensionPropertyGroup"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
    <group name="AirspaceExtensionPropertyGroup">
        <sequence>
            <element name="suaType" type="aimsaa:CodeSpecialUseAirspaceType" nillable="true"
minOccurs="0"/>
            <element name="separationStandard" type="aimsaa:CodeSeparationStandardType"
nillable="true" minOccurs="0"/>
        </sequence>
    </group>
    <element name="AirspaceActivationExtension" type="aimsua:AirspaceActivationExtensionType"
substitutionGroup="aixm:AbstractAirspaceActivationExtension"/>
    <complexType name="AirspaceActivationExtensionType">
        <complexContent>
            <extension base="aixm:AbstractExtensionType">
                <sequence>
                    <group ref="aimsua:AirspaceActivationExtensionPropertyGroup"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
    <group name="AirspaceActivationExtensionPropertyGroup">
        <sequence>
            <element name="reservationID" type="aixm:TextNameType" nillable="true" minOccurs="0"/>
            <element name="reservationPhase" type="aimsaa:CodeAirspaceReservationPhaseType"
nillable="true" minOccurs="0"/>
            <element name="reservationUserAction" type="aimsaa:CodeReservationUserActionType"
nillable="true" minOccurs="0"/>
            <element name="liveFire" type="aixm:CodeYesNoType" nillable="true" minOccurs="0"/>
            <element name="lightsOut" type="aixm:CodeYesNoType" nillable="true" minOccurs="0"/>
            <element name="sorties" type="aixm:NoNumberType" nillable="true" minOccurs="0"/>
            <element name="separationStandard" type="aimsaa:CodeSeparationStandardType"
nillable="true" minOccurs="0"/>
            <element name="aircraftInvolved" type="aimsua:AircraftGroupPropertyType" nillable="true"
minOccurs="0" maxOccurs="unbounded"/>
        </sequence>
    </group>
    <complexType name="AircraftDetailPropertyType">
        <complexContent>
            <extension base="aixm:AbstractAIXMPropertyType">
                <sequence>
                    <element ref="aimsua:AircraftDetail"/>
```

```
                </sequence>
            </extension>
        </complexContent>
    </complexType>
    <element name="AircraftDetail" type="aimsua:AircraftDetailType"/>
    <complexType name="AircraftDetailType">
        <complexContent>
            <extension base="aixm:AbstractAIXMObjectType">
                <sequence>
                    <group ref="aimsua:AircraftDetailPropertyGroup"/>
                    <element name="extension" minOccurs="0" maxOccurs="unbounded">
                        <complexType>
                            <sequence>
                                <element ref="aimsua:AbstractAircraftDetailExtension"/>
                            </sequence>
                            <attributeGroup ref="gml:OwnershipAttributeGroup"/>
                        </complexType>
                    </element>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
    <group name="AircraftDetailPropertyGroup">
        <sequence>
            <element name="callSign" type="aixm:TextNameType" nillable="true" minOccurs="0"/>
        </sequence>
    </group>
    <element name="AbstractAircraftDetailExtension" type="aixm:AbstractExtensionType" abstract="true"
substitutionGroup="aixm:AbstractExtension"/>
    <complexType name="AircraftGroupPropertyType">
        <complexContent>
            <extension base="aixm:AbstractAIXMPropertyType">
                <sequence>
                    <element ref="aimsua:AircraftGroup"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
    <element name="AircraftGroup" type="aimsua:AircraftGroupType"/>
    <complexType name="AircraftGroupType">
        <complexContent>
            <extension base="aixm:AbstractAIXMObjectType">
                <sequence>
                    <group ref="aimsua:AircraftGroupPropertyGroup"/>
                    <element name="extension" minOccurs="0" maxOccurs="unbounded">
                        <complexType>
                            <sequence>
                                <element ref="aimsua:AbstractAircraftGroupExtension"/>
                            </sequence>
                            <attributeGroup ref="gml:OwnershipAttributeGroup"/>
                        </complexType>
                    </element>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
    <group name="AircraftGroupPropertyGroup">
        <sequence>
            <element name="aircraftType" type="aixm:CodeAircraftType" nillable="true" minOccurs="0"/>
            <element name="flyingUnit" type="aixm:TextNameType" nillable="true" minOccurs="0"/>
            <element name="indicatedAirSpeed" type="aixm:ValSpeedType" nillable="true"
minOccurs="0"/>
            <element name="aircraft" type="aimsua:AircraftDetailPropertyType" nillable="true"
minOccurs="0" maxOccurs="unbounded"/>
        </sequence>
    </group>
    <element name="AbstractAircraftGroupExtension" type="aixm:AbstractExtensionType" abstract="true"
substitutionGroup="aixm:AbstractExtension"/>
</schema>
```

Annex B
SAA Schema Extension UML Models

This annex provides UML class diagram models of the SAA and SUA XML schema extensions in Annex A.

B.1 AXIM 5.0 SAA UML Diagrams

The following figures illustrate the extensions made to AIXM 5.0 for Special Activity Airspace (SAA) and Special Use Airspace (SUA). The core AIXM objects and features are colored yellow and the extensions are colored blue.

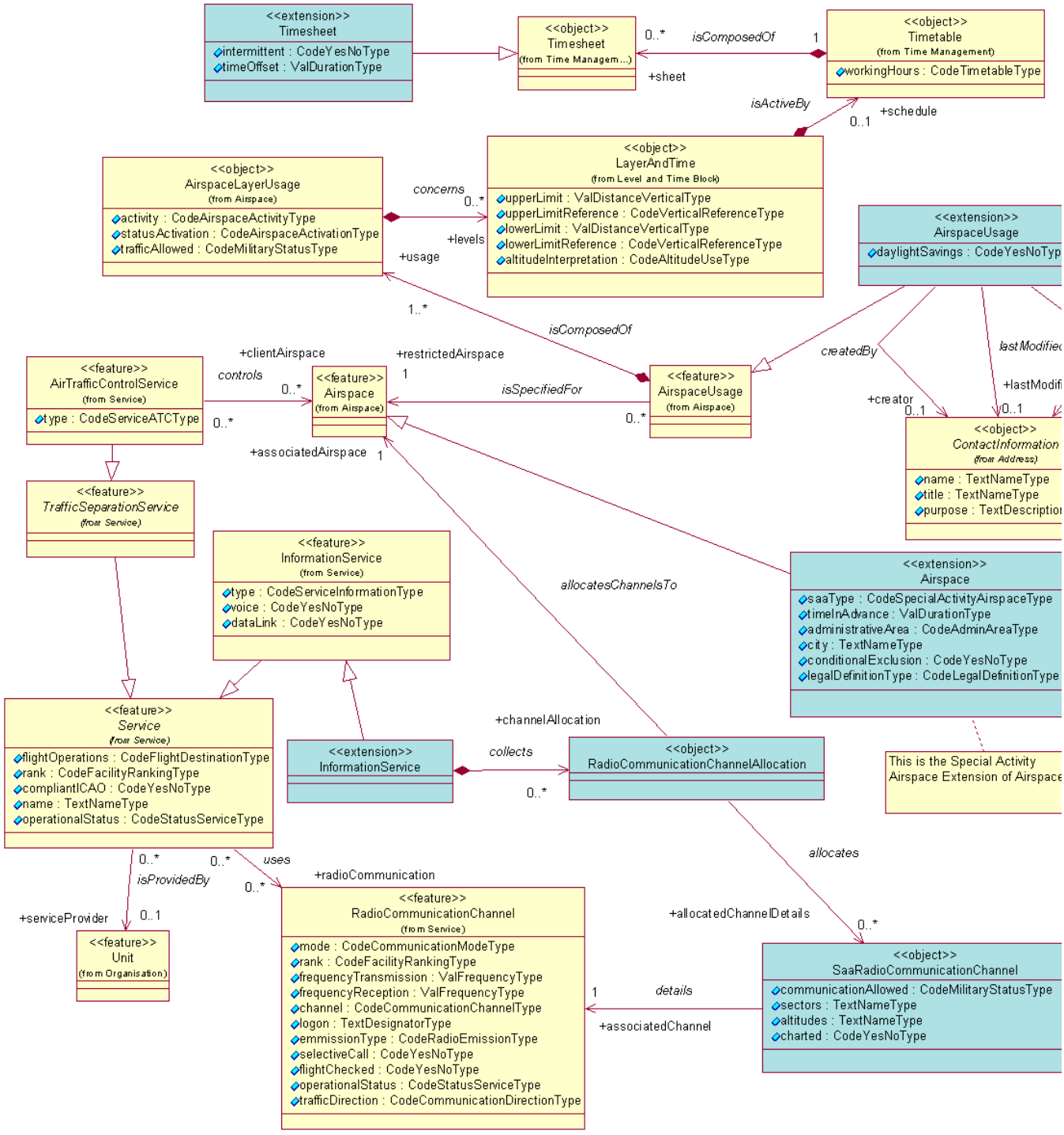

Figure B-1. SAA Feature Extension AIXM 5.0

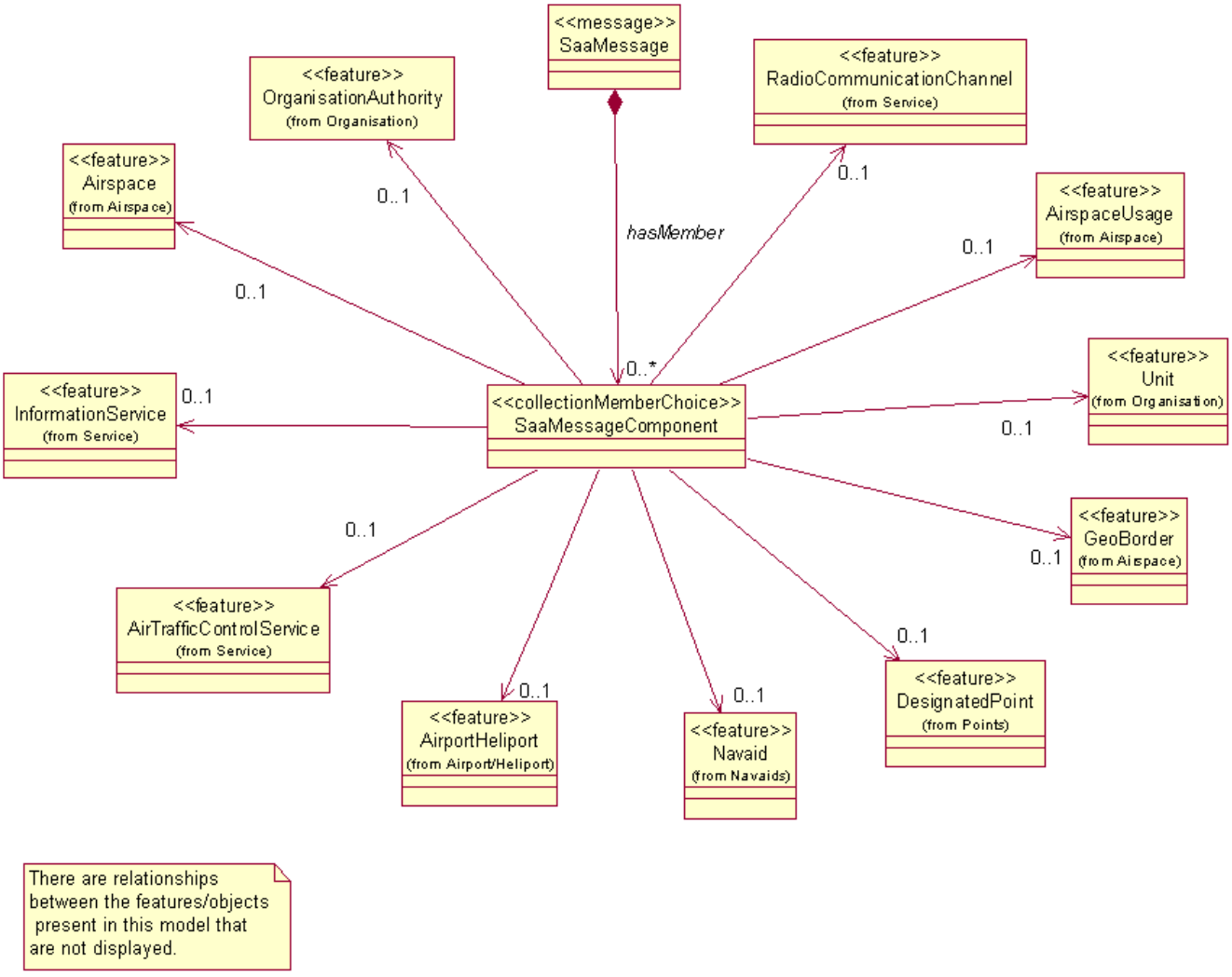

Figure B-2. SAA Message Extension AIXM 5.0

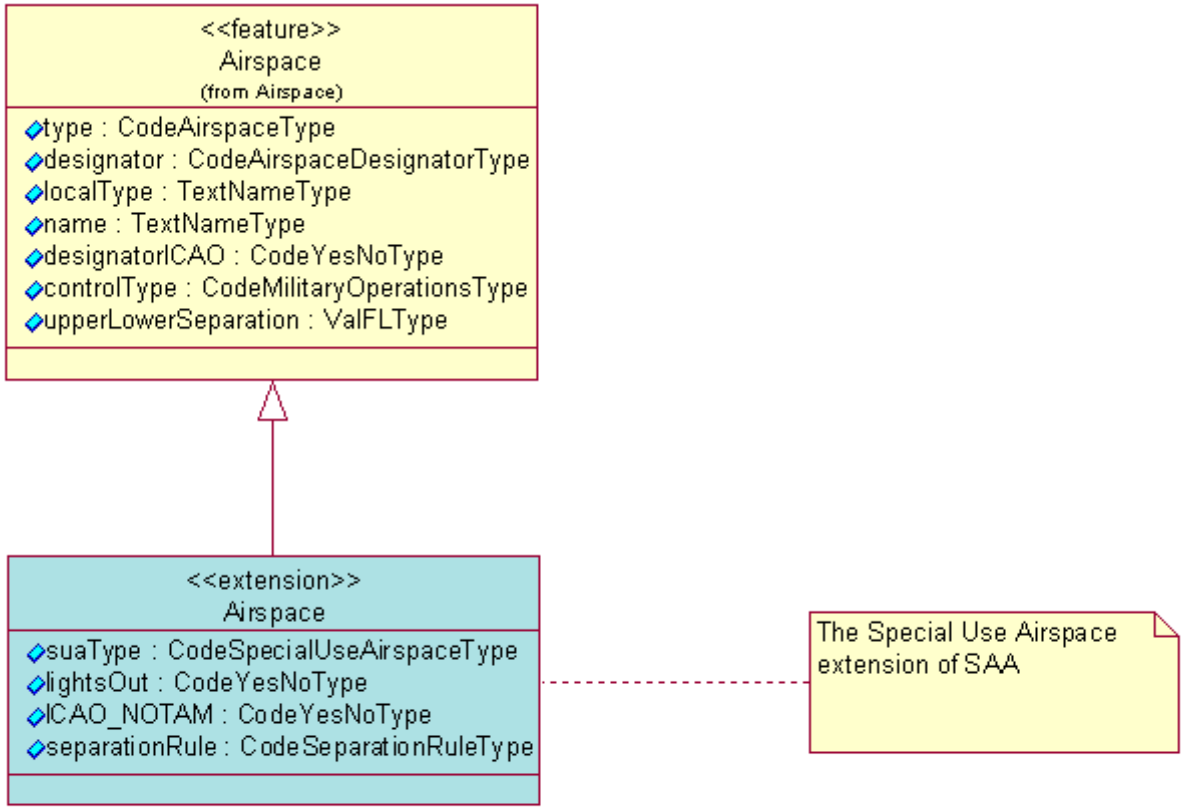

Figure B-3. SUA Feature Extension AIXM 5.0

B.2 AXIM 5.1 SAA UML Diagrams

The following figures illustrate the extensions made to AIXM 5.1 for Special Activity Airspace (SAA) and Special Use Airspace (SUA). The core AIXM objects and features are colored yellow and the extensions are colored blue.

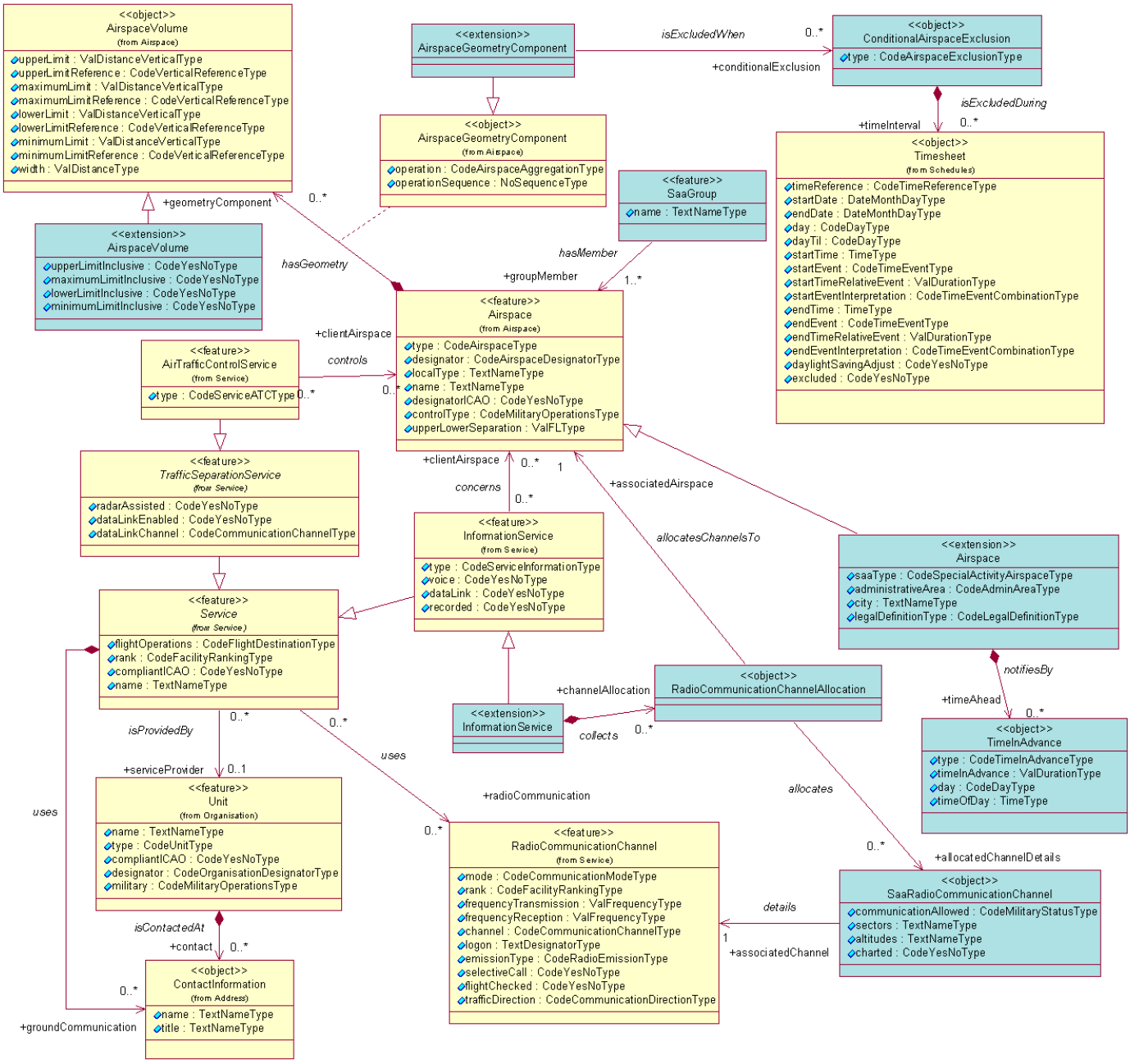

Figure B-4. SAA Airspace Feature Extension AIXM 5.1

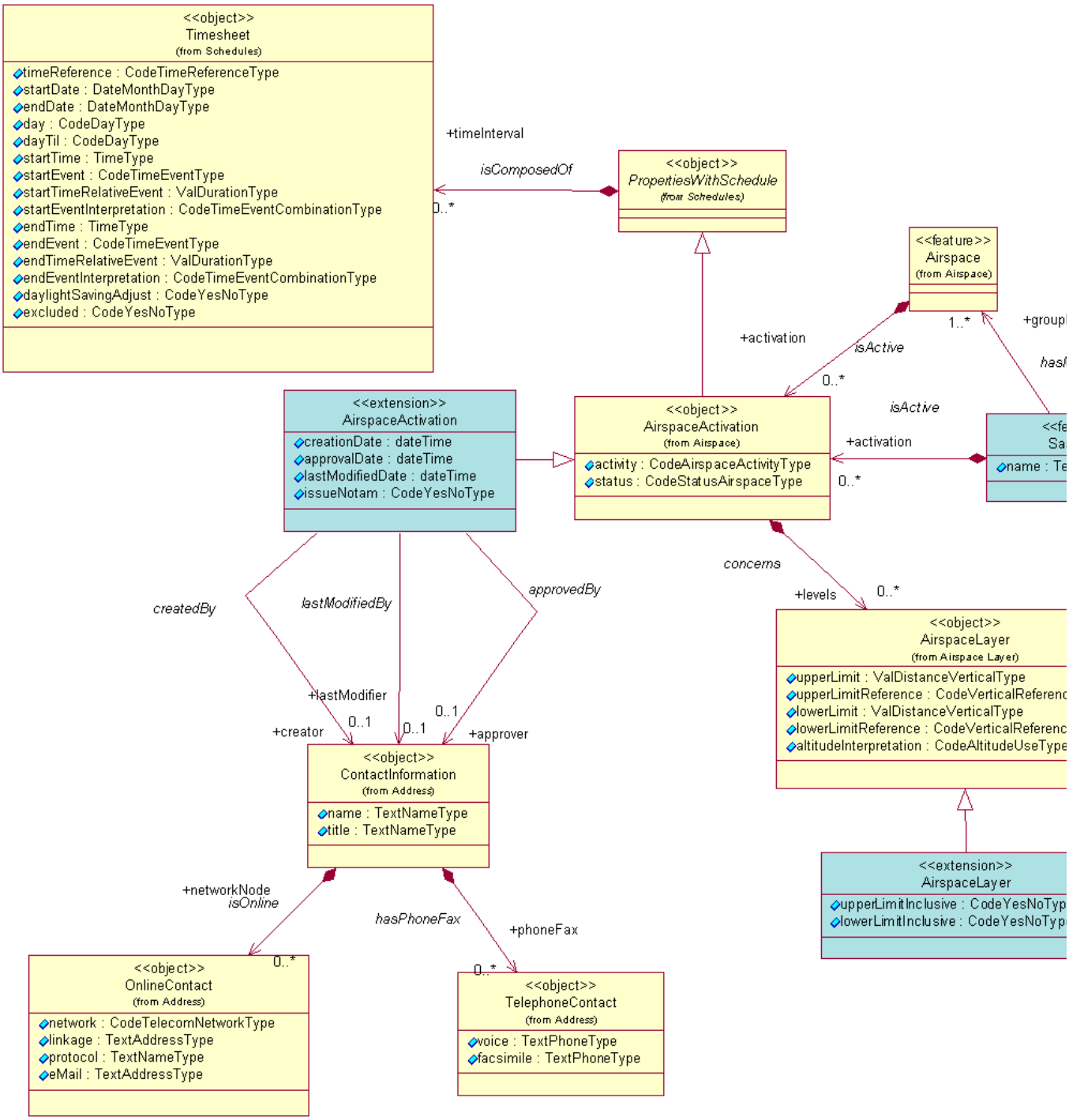

Figure B-5. SAA Airspace Activation Feature Extension AIXM 5.1

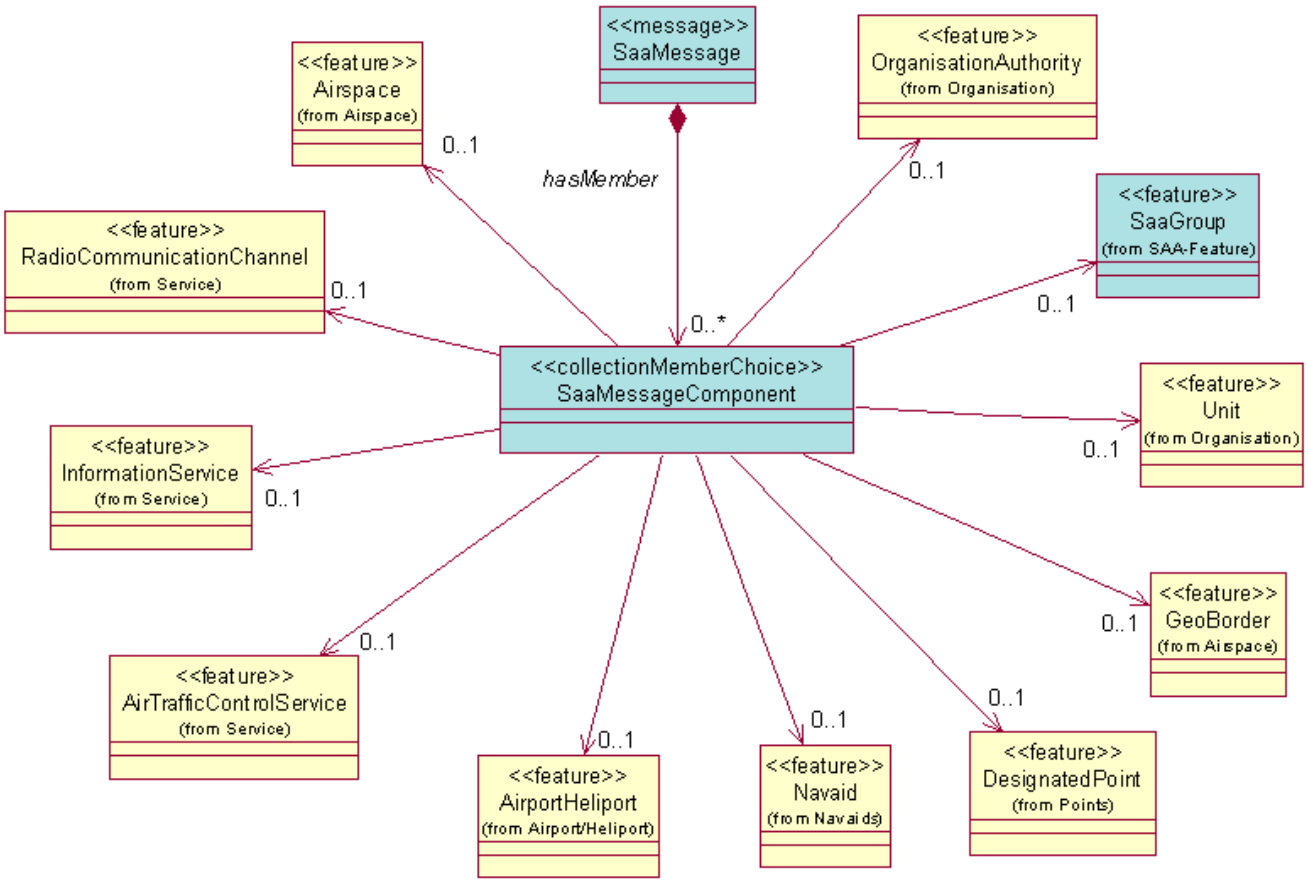

Figure B-6. SAA Message Extension AIXM 5.1

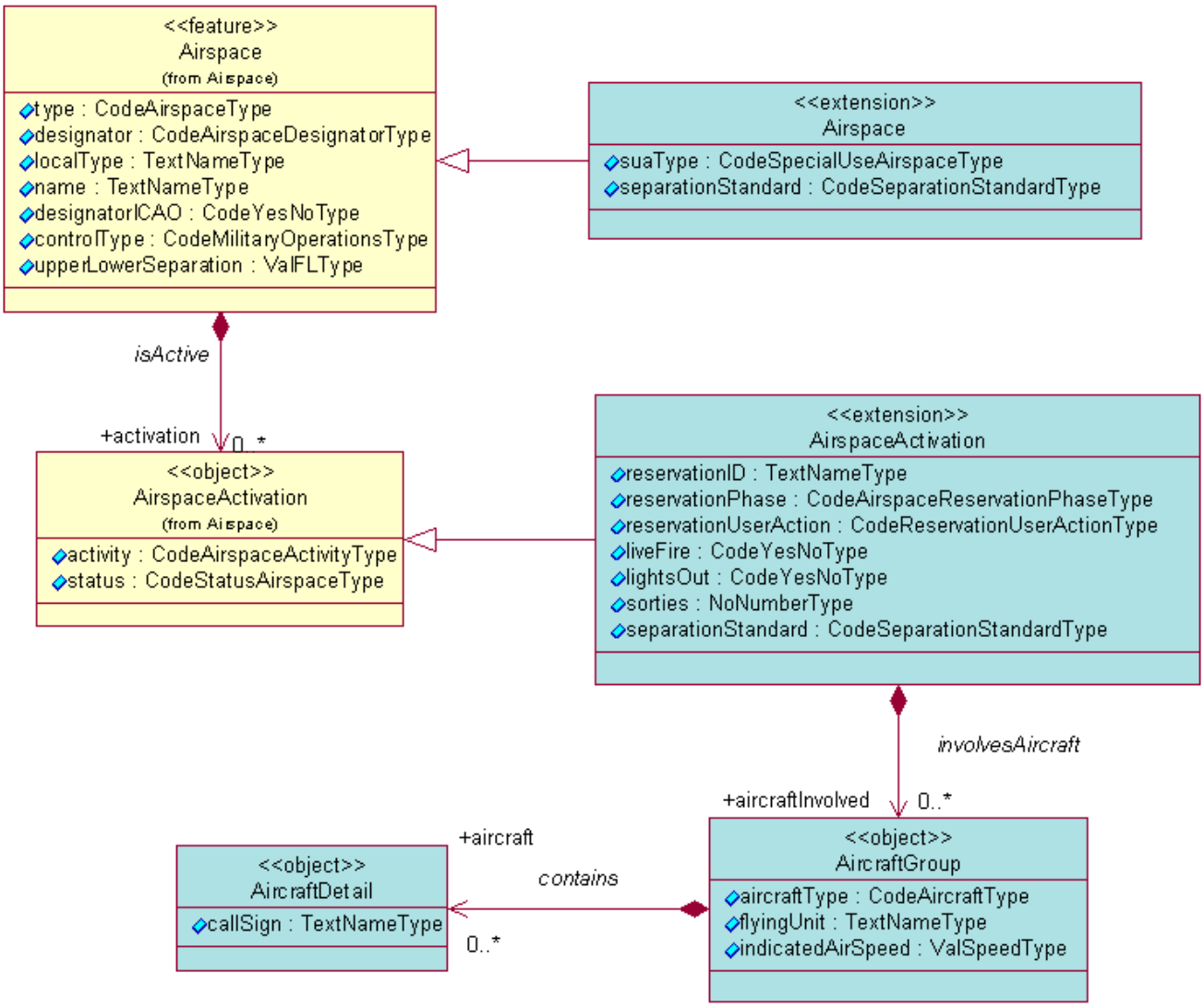

Figure B-7. SUA Feature Extension AIXM 5.1

Annex C
SAA Timesheets in AIXM 5.1

This annex describes how to define the status of an airspace. There are two different ways in AIXM to accomplish this. The first is using the Timesheet AIXM object to define the status for a static airspace definition. The second is to use a TEMPDELTA timeslice to set the status of an airspace for an individual airspace reservation.

C.1. Airspace Status in Static Definitions

The static definition of an SUA contains the times of use for the airspace. The times of use indicate the period when the using agency is authorized to schedule the use of a SUA.

The times of use are defined in the static definition of the airspace through the AirspaceActivation and Timesheet objects.

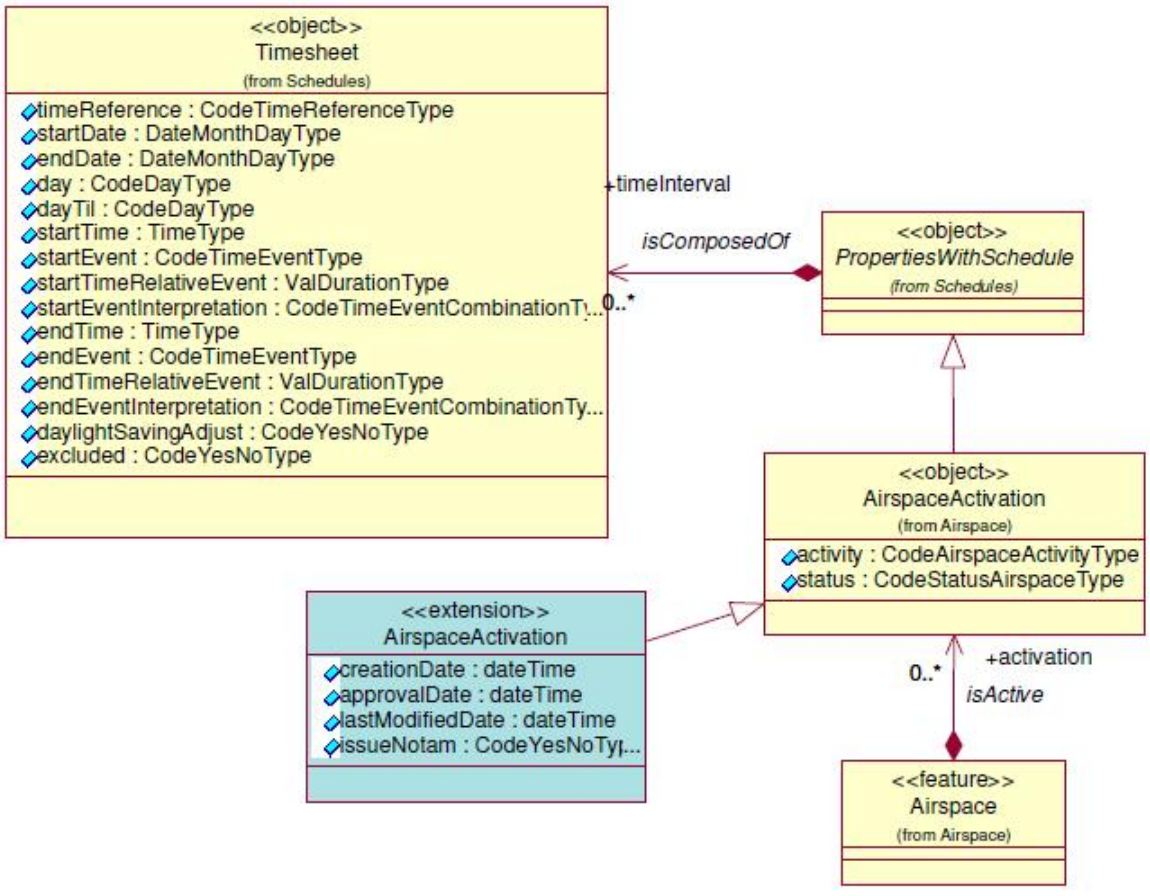

Figure C-1 - AIXM 5.1 AirspaceActivation for Static Definitions

The static definition is a BASELINE timeslice of the Airspace feature. This can contain one or more AirspaceActivation objects that set the status of the airspace prior to any operational reservations of the airspace. The static definition can set the status of the airspace to the following:

- AVBL_FOR_ACTIVATION – The airspace can be scheduled for activation.
- ACTIVE – The airspace is active.
- INACTIVE – The airspace cannot be scheduled for activation.

If an airspace is always active (such as a prohibited area) or always available for activation then there is no reason to have any Timesheet objects and they should be omitted to facilitate processing of the data set. The airspace should simply have an AirspaceActivation object with the status set. It's interpreted that the airspace's status is applied for the duration of the timeslice.

When an airspace is not always the same status in the static definition, timesheets need to be used. When timesheets are used, the compilation of all the AirspaceActivation objects and timesheets should represent the status of the airspace at any given time of the timeslice. The easiest way to make sure that all times are accounted for is through the use of the excluded attribute. The excluded attribute is used to indicate that the time block indicated by the current Timesheet is not set to the status in the AirspaceActivation. The following example illustrates this:

Example 1 – Basic airspace schedule

For an airspace with published times of use as 0800-2200 Monday-Saturday, two different AirspaceActivation objects would be used to encode this schedule. The first is to define the hours that the airspace is available for activation, using a separate timesheet for each day. The second is to define the time outside of these hours, to indicate the time periods that the airspace is inactive. An easy way to define the inactive time periods is to repeat all of the timesheets from the first AirspaceActivation, but then add the excluded attribute to each of the timesheets to indicate INACTIVE.

The objects and attributes for this example would be the following within a BASELINE timeslice of the airspace:

AirspaceActivation 1; status = AVBL_FOR_ACTIVATION

	Timesheet 1	Timesheet 2	Timesheet 3	Timesheet 4	Timesheet 5	Timesheet 6
day	MON	TUE	WED	THU	FRI	SAT
startTime	08:00	08:00	08:00	08:00	08:00	08:00
endTime	22:00	22:00	22:00	22:00	22:00	22:00
excluded	NO	NO	NO	NO	NO	NO

AirspaceActivation 2; status = INACTIVE

	Timesheet 1	Timesheet 2	Timesheet 3	Timesheet 4	Timesheet 5	Timesheet 6
day	MON	TUE	WED	THU	FRI	SAT
startTime	08:00	08:00	08:00	08:00	08:00	08:00
endTime	22:00	22:00	22:00	22:00	22:00	22:00

excluded	YES	YES	YES	YES	YES	YES

Example 2 – Same status at all times

For an airspace with the published times of use "Continuous," only one AirspaceActivation object is used, with status = ACTIVE, and no timesheets.

This would also be how you represent available for activation at any time without issuing a NOTAM. In that case, the status would be set to 'AVBL_FOR_ACTIVATION'.

C.2. The issueNotam Flag in Static Definitions

The issueNotam attribute in the SAA extension of the AirspaceActivationhttps://portal.opengeospatial.org/twiki/bin/edit/SAAPilot/AirspaceActivation?topicparent=SAAPilot.SaaTimesheetUsage;nowysiwyg=1 object is used to indicate when a NOTAM needs to be issued when an SAA is scheduled. The following examples illustrate the four possible scenarios in which the issueNotam flag would be used, and how they would be represented using AIXM 5.1 timesheets:

a) The SAA is available to be schedule at any time by NOTAM

- AirspaceActivation:
  - status = AVBL_FOR_ACTIVATION
  - issueNotam = YES

b) The SAA is available to be scheduled during a time period by NOTAM. This requires two AirspaceActivation:

- AirspaceActivation 1:
  - status = AVBL_FOR_ACTIVATION
  - issueNotam = YES

	Timesheet
day	MON
startTime	09:00
endTime	17:00
excluded	NO

- AirspaceActivation 2:
  - status = INACTIVE

	Timesheet
day	MON
startTime	09:00
endTime	17:00
excluded	YES

c) The SAA is available to be scheduled during one time period by NOTAM and during a different time period without NOTAM. This requires three AirspaceActivation, with a total of four timesheets:

- AirspaceActivation 1 :
  - status = AVBL_FOR_ACTIVATION

	Timesheet
day	ANY
startTime	09:00
endTime	16:59
excluded	NO

- AirspaceActivation 2:
  - status = AVBL_FOR_ACTIVATION
  - issueNotam = YES

	Timesheet
day	ANY
startTime	17:00
endTime	22:00
excluded	NO

- AirspaceActivation 3:
  - status = INACTIVE

	Timesheet 1	Timesheet 2
day	ANY	ANY
startTime	09:00	17:00
endTime	16:59	22:00
excluded	YES	YES

d) The SAA is available to be scheduled during set time periods, and can be scheduled at other times by NOTAM. This requires two AirspaceActivation:

- AirspaceActivation:
  - status = AVBL_FOR_ACTIVATION

	Timesheet
day	MON
startTime	09:00
endTime	17:00
excluded	NO

- AirspaceActivation 2:
  - status = AVBL_FOR_ACTIVATION
  - issueNotam = YES

	Timesheet
day	MON

startTime	09:00
endTime	17:00
excluded	YES

C.3. Airspace Status in Operational Schedules

Unlike the static definition, setting the status of an airspace for an individual activation does not use the Timesheet object. An individual airspace reservation is represented by a TEMPDELTA airspace timeslice containing an AirspaceActivation object setting the status. If the reservation is for a piece of the airspace, then an AirspaceLayer object is used to define the altitudes that are being reserved.

Figure C-2 illustrates the important objects for an operational airspace reservation in UML.

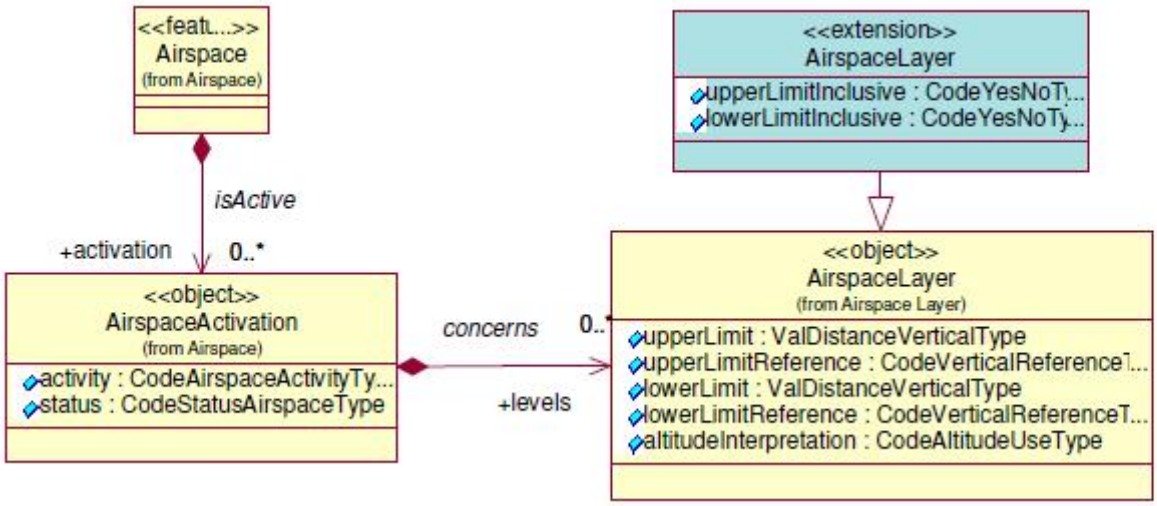

Figure C-2 - AirspaceActivation used for Airspace Reservation

So an individual airspace reservation can be represented as simply as a single Airspace timeslice with an AirspaceActivation object (not including the additional data elements used in the SUA extension):

- Airspace
  - Timeslice
    - TimePeriod: begin and end times
    - interpretation = TEMPDELTA
    - AirspaceActivation
      - status = ACTIVE

Annex D
Web Feature and Event Service Implementation Details

D.1. WFS Capabilities

This section provides examples of additional WFS functionality not included in the Commercial Flight scenario that would be beneficial to meeting FAA operational requirements.

D.1.1 GetCapabilities Operation

The GetCapabilities operation enables clients to discover what services and feature types are available. It is also used to evaluate the operations and filter expressions supported by the service and any constraints that apply to operations or query parameters.

D.1.2 DescribeFeatureType Operation

The DescribeFeatureType operation enables clients to discover what parameters a feature has. This is primarily used to enable the WFS to validate the content of a transaction operation.

D.1.3 GetFeature Operation

The GetFeature operation is the key request type for retrieving AIXM 5.1 features. The GetFeature operation supports two types of query:

1. **Ad hoc queries:** are user defined queries not known by the server before they are invoked.
2. **Stored queries:** are pre-defined, parameterized queries stored on the server that can be re-used within a client

A WFS that supports both types of query offers more functionality than one that just supports ad hoc queries. Stored queries offer a wide range of benefits on top of ad hoc queries:

- Common, complex queries can be simplified
- Allow the WFS administrator to better control the service
- Simulate current SAA web service requests
- Reduce implementation barriers for client application development or migration onto OGC WFS

NOTE: A WFS can be configured to support Stored Queries only.

Additional query parameters not associated with selection can also be included in the GetFeature operation. These include:

Type	Parameter	Description
Projection Parameters	PropertyName	Identifies a subset of optional feature properties to be included in the response[2]. If this property is not included in the request, the

Type	Parameter	Description
		response shall include all mandatory and optional properties by default.
Standard Resolve Parameters	resolve	Controls whether and which resource references are to be resolved (local, remote, all or none). NOTE 1: If no resolve parameters are specified then the server shall not resolve any references by default
	resolveDepth	resolveDepth defines the depth to which nested references shall be resolved. For example, if a feature contains a property that contains a reference to another feature, which contains a property that references another resource then this property specifies whether or not to resolve the properties in the nested referenced resources. Default value = *
	resolvePath	Limits the resource resolution to the property specified for the resolvePath
	resolveTimeout	Controls how long a server shall wait to receive a response when resolving resource references. This is specified in seconds
CRS Transformation Parameters	srsName	If a WFS declares support for multiple coordinate reference systems, the srsName may be used to perform a coordinate reference system transformation to the features returned in the response. The value may be the wfs:DefaultCRS or any of the wfs:OtherCRS values listed in the GetCapabilities. If no srsName parameter is defined in the query expression, then the wfs:DefaultCRS shall be used. NOTE 1: this property is not supported by KVP encoded requests Example: srsName=”urn:ogc:def:crs:EPSG::4326”
Standard Presentation Parameters	count	This parameter is used to limit the number of features to be included in the response. There is no predefined default value and if this value is absent then the server shall return all features corresponding to the request query expression
	startIndex	Indicates the index from which the server shall begin presenting the results in the response. The default value is 1if not specified
	outputFormat	Specifies the format used to encode the response. The default value is “application/gml+xml; version 3.2” if not specified. This parameter should only be required where a server supports the ability to encode the result in multiple formats.
	resultType	If the ResultType parameter is not expressed, then the response shall contain a wfs:FeatureCollection containing all features corresponding to the request query expression. If the resultType=hits then the response shall be a count of the number of features corresponding to the request query expression. This is useful to identify how many features may be returned before submitting the response.
Sorting Clause	sortBy	SortBy is used to specify one or more property values that should be used to order (ASC or DESC) the features contained in the response. If the sort order is not specified then the default ordering shall be ascending. NOTE: Sorting is only supported for single query expressions

Table D-1. WFS GetFeature Query Parameters

Example requests demonstrating these additional query parameters:

D.1.4 Projection parameters:

The projection parameter – PropertyName can be included in the GetFeature request to select which optional properties that should be included in the response, where a client does not need all the feature properties to be returned.

NOTE 1: if a client only wants to return value(s) for a single property, then the GetPropertyValue request should be used.

NOTE 2: in many WFS implementations the projection parameter cannot be used to limit the response to include specific optional properties of complex properties (i.e. a selected aixm:timeslice). The AIXM model uses a single feature with multiple timeslices. WFS queries currently return the whole feature, and thus querying on a property value which is matched by any timeslice in a feature will return the whole feature and all its timeslices. There is an agenda item at the OGC Bonn Aviation Working Group to discuss this issue.

Within the pilot, only mandatory properties were provided so this parameter cannot be demonstrated with the SAA WFS. The OWS-7 AIXM 5.1 WFS includes the optional feature properties: boundedBy and featureMetadata. So the example shall demonstrate how to define whether one or none of these optional properties are included in the response.

Example 1: All properties are returned

Request:

http://demo.snowflakesoftware.com:8081/AIXM51_WFS2/GOPublisherWFS?service=wfs&version=2.0.0&request=GetFeature&count=1&typename=aixm:Airspace

Response:

```
<message:AIXMBasicMessage xsi:schemaLocation="http://www.aixm.aero/schema/5.1/message
AIXM_BasicMessage.xsd" gml:id="gml_fc_1">
    <message:hasMember>
        <aixm:Airspace gml:id="urn-
x:ows7:snowflake:sua:DEVILS_LAKE_EAST_MOA_ND:Airspace01:2">
            <gml:identifier codeSpace="http://www.faa.gov/nasr"/>
            <gml:boundedBy/>
                <...>
            <aixm:featureMetadata/>
            <aixm:timeSlice/>
                <...>
        </aixm:Airspace>
    </message:hasMember>
</message:AIXMBasicMessage>
```

Example 2: Include only the optional boundedBy property in the response

Request:
http://demo.snowflakesoftware.com:8081/AIXM51_WFS2/GOPublisherWFS?service=wfs&version=2.0.0&request=GetFeature&count=1&typename=aixm:Airspace&propertyName=gml:boundedBy

Response:

```
<message:AIXMBasicMessage xsi:schemaLocation="http://www.aixm.aero/schema/5.1/message
AIXM_BasicMessage.xsd" gml:id="gml_fc_1">
    <message:hasMember>
        <aixm:Airspace gml:id="urn-
x:ows7:snowflake:sua:DEVILS_LAKE_EAST_MOA_ND:Airspace01:2">
            <gml:boundedBy/>
            <aixm:timeSlice/>
        </aixm:Airspace>
    </message:hasMember>
</message:AIXMBasicMessage>
```

Example 3: No optional properties are returned (using propertyName parameter)

http://demo.snowflakesoftware.com:8081/AIXM51_WFS2/GOPublisherWFS?service=wfs&version=2.0.0&request=GetFeature&count=1&typename=aixm:Airspace&propertyName=aixm:timeSlice

```
<message:AIXMBasicMessage xsi:schemaLocation="http://www.aixm.aero/schema/5.1/message
AIXM_BasicMessage.xsd" gml:id="gml_fc_1">
    <message:hasMember>
        <aixm:Airspace gml:id="urn-
x:ows7:snowflake:sua:DEVILS_LAKE_EAST_MOA_ND:Airspace01:2">
            <aixm:timeSlice/>
        </aixm:Airspace>
    </message:hasMember>
</message:AIXMBasicMessage>
```

D.1.5 Standard resolve parameters:

Many AIXM features contain one or more properties that reference to another AIXM feature. For example, the AirTrafficControlService contains two properties that reference other features:

- **serviceProvider:** contains a reference to the Unit feature
- **clientAirspace:** contains a reference to the Airspace feature under control of the AirTrafficControlService

The WFS 2.0 specification states that a WFS should support the ability to resolve local references. Using the standard resolve parameters, the client can control how the service returns resolved references in the response.

If the client requests that local references should be resolved then the response shall contain:

- wfs:member [1..*]: containing all of the features that correspond to a request, and
- wfs:additionalObjects [1]: containing all of the referenced features. It may also contain additional features if any of the resolved features contain properties that reference other features.

Example 5: Get the AirTrafficControlService by RESOURCEID

In this example, the server resolves all references contained within both the requested feature (AirTrafficControlService) and the resolved features (i.e. Unit). Note that OrganisationAuthority feature has been resolved for the Unit feature

Request:

http://demo.luciad.com:8080/SaaWFS/wfs?REQUEST=GetFeature&TYPENAMES=AirTrafficControlService&VERSION=2.0.0&SERVICE=WFS&RESOURCEID=urn.uuid.518938a6-1ab0-4bfd-8afe-7321de915f25&RESOLVE=local

```
<wfs:FeatureCollection <!—schema declarations omitted for brevity-->
<wfs:member>
    <ns5:AirTrafficControlService gml:id="urn.uuid.518938a6-1ab0-4bfd-8afe-
7321de915f25">
      <gml:identifier codeSpace="http://www.faa.gov/nasr">518938a6-1ab0-4bfd-
8afe-7321de915f25</gml:identifier>
      <ns5:timeSlice>
        <ns5:AirTrafficControlServiceTimeSlice gml:id="urn.uuid.518938a6-
1ab0-4bfd-8afe-7321de915f25_1">
          <gml:validTime>
            <gml:TimePeriod gml:id="urn.uuid.518938a6-1ab0-4bfd-8afe-
7321de915f25_2">
              <gml:beginPosition>2011-03-10-05:00</gml:beginPosition>
              <gml:endPosition
indeterminatePosition="unknown"></gml:endPosition>
            </gml:TimePeriod>
          </gml:validTime>
          <ns5:interpretation>BASELINE</ns5:interpretation>
          <ns5:sequenceNumber>1</ns5:sequenceNumber>
          <ns5:correctionNumber>0</ns5:correctionNumber>
          <ns5:featureLifetime>
            <gml:TimePeriod gml:id="urn.uuid.518938a6-1ab0-4bfd-8afe-
7321de915f25_3">
              <gml:beginPosition>2011-03-10-05:00</gml:beginPosition>
              <gml:endPosition
indeterminatePosition="unknown"></gml:endPosition>
            </gml:TimePeriod>
          </ns5:featureLifetime>
```

Response:

```
<ns5:serviceProvider xlink:href="#urn.uuid.82A95872-9182-2362-E044-00212803DA06"
xsi:nil="true"/>
          <ns5:type>ACS</ns5:type>
          <ns5:clientAirspace xlink:href="#urn.uuid.7404b12b-e47c-4419-bddb-
a9007f14ce33" xsi:nil="true"/>
        </ns5:AirTrafficControlServiceTimeSlice>
      </ns5:timeSlice>
    </ns5:AirTrafficControlService>
  </wfs:member>
  <wfs:additionalObjects>
    <wfs:ValueCollection>
      <wfs:member>
        <ns5:Airspace gml:id="urn.uuid.7404b12b-e47c-4419-bddb-a9007f14ce33"/>
      </wfs:member>
       <wfs:member>
        <ns5:Unit gml:id="urn.uuid.82A95872-9182-2362-E044-00212803DA06">
          <gml:identifier codeSpace="http://www.faa.gov/nasr">82A95872-9182-2362-E044-
00212803DA06</gml:identifier>
          <ns5:timeSlice>
            <ns5:UnitTimeSlice gml:id="urn.uuid.82A95872-9182-2362-E044-
00212803DA06_1">
              <gml:validTime>
                <gml:TimePeriod gml:id="urn.uuid.82A95872-9182-2362-E044-
00212803DA06_2">
                  <gml:beginPosition>2010-02-11-04:00</gml:beginPosition>
                  <gml:endPosition indeterminatePosition="unknown"></gml:endPosition>
                </gml:TimePeriod>
              </gml:validTime>
              <ns5:interpretation>BASELINE</ns5:interpretation>
              <ns5:sequenceNumber>1</ns5:sequenceNumber>
              <ns5:correctionNumber>0</ns5:correctionNumber>
              <ns5:featureLifetime>
                <gml:TimePeriod gml:id="urn.uuid.82A95872-9182-2362-E044-
00212803DA06_3">
                  <gml:beginPosition>2010-02-11-04:00</gml:beginPosition>
                  <gml:endPosition indeterminatePosition="unknown"></gml:endPosition>
                </gml:TimePeriod>
              </ns5:featureLifetime>
              <ns5:name>JACKSONVILLE</ns5:name>
              <ns5:type>ARTCC</ns5:type>
              <ns5:compliantICAO>NO</ns5:compliantICAO>
              <ns5:designator>ZJX</ns5:designator>
              <ns5:military>CIVIL</ns5:military>
              <ns5:ownerOrganisation xlink:href="#urn.uuid.82A95872-8EC5-2362-E044-
00212803DA06" xsi:nil="true"/>
            </ns5:UnitTimeSlice>
          </ns5:timeSlice>
        </ns5:Unit>
      </wfs:member>
            <wfs:member>
        <ns5:OrganisationAuthority gml:id="urn.uuid.82A95872-8EC5-2362-E044-
00212803DA06">
                  </ns5:OrganisationAuthority>
      </wfs:member>
    </wfs:ValueCollection>
  </wfs:additionalObjects>
</wfs:FeatureCollection>
```

D.1.6 Sorting Clause:

The sorting clause enables the server to order the features contained in the response (ascending or descending) based on the values of a specific property.

Example 6: Select 10 Airspace features ordered by gml:identifier (ascending)

Request:

http://demo.snowflakesoftware.com:8081/SAA_AIXM51_FeatureCollection/FAA_SAA_DisseminationPilot?service=wfs&version=2.0.0&request=GetFeature&count=10&typename=aixm:Airspace&sortBy=gml:identifier

Response:

```
<wfs:FeatureCollection xsi:schemaLocation="urn:us:gov:dot:faa:aim:saa:5.1 SAA-
Message.xsd http://www.opengis.net/wfs/2.0 http://schemas.opengis.net/wfs/2.0/wfs.xsd"
numberMatched="unknown" numberReturned="999999" timeStamp="9999-01-01T00:00:00.000">
    <wfs:member>
        <aixm:Airspace gml:id="Airspace1_00df955f-aa65-4ddc-a2ef-68fe4a24c567"/>
    </wfs:member>
    <wfs:member>
        <aixm:Airspace gml:id="Airspace1_00ece7aa-bdcc-43f6-8c07-822227bef318"/>
    </wfs:member>
    <wfs:member>
        <aixm:Airspace gml:id="Airspace1_0113d386-5377-419d-9b70-0eabff6c71d7"/>
    </wfs:member>
    <wfs:member>
        <aixm:Airspace gml:id="Airspace1_0128a6a2-ce79-4a21-8a55-ee2ea75f8dcc"/>
    </wfs:member>
    <wfs:member>
        <aixm:Airspace gml:id="Airspace1_015b4b48-cbb1-47d1-bda7-b6b5b2ba7a55"/>
    </wfs:member>
    <wfs:member>
        <aixm:Airspace gml:id="Airspace1_0178f2ba-df47-45fc-a87d-ba3074116d18"/>
    </wfs:member>
    <wfs:member>
        <aixm:Airspace gml:id="Airspace1_01f6738f-be05-4a6c-b25b-f3494cb57c7d"/>
    </wfs:member>
    <wfs:member>
        <aixm:Airspace gml:id="Airspace1_035360bd-db03-4d3f-853a-4103dc82f438"/>
    </wfs:member>
    <wfs:member>
        <aixm:Airspace gml:id="Airspace1_03895df2-41c8-4467-93c4-70aeffddca90"/>
    </wfs:member>
    <wfs:member>
        <aixm:Airspace gml:id="Airspace1_03ed2e4d-c165-46e9-ace7-30e126d0ee37"/>
    </wfs:member>
</wfs:FeatureCollection>
```

D.1.7 GetPropertyValue Operation

The GetPropertyValue operation supports a similar set of request parameters to the GetFeature operation. However, the objective of the GetPropertyValue operation is to return only the value or values of the specified property.

The GetPropertyValue operation also supports ad hoc and stored queries. Within the pilot only ad hoc GetPropertyValue operations were demonstrated.

A typical GetPropertyValue request identified in the pilot is:

- List the names of SAA Units based on one or more property values
- Get the transmission frequency of a specific RadioCommunicationChannel

Example 7: Get the names of all MILOPS units

Request:

```
<?xml version="1.0" encoding="UTF-8"?>

<wfs:GetPropertyValue service="WFS" version="2.0.0"

outputFormat="application/gml+xml; version=3.2"
valueReference="//aixm:name"

xmlns:wfs="http://www.opengis.net/wfs/2.0"
xmlns:aixm="http://www.aixm.aero/schema/5.1"
xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:ogc="http://www.opengis.net/ogc"
xmlns:gml="http://www.opengis.net/gml/3.2"
xmlns:fes="http://www.opengis.net/fes/2.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://www.opengis.net/wfs/2.0
                     http://schemas.opengis.net/wfs/2.0/wfs.xsd">
<wfs:Query typeNames="aixm:Unit" handle="Q01"><fes:Filter>
<fes:PropertyIsEqualTo>
    <fes:ValueReference>aixm:timeSlice//aixm:type</fes:ValueReference>
    <fes:Literal>MILOPS</fes:Literal>
</fes:PropertyIsEqualTo>
</fes:Filter></wfs:Query>
</wfs:GetPropertyValue>
```

Response:

```
?xml version="1.0" encoding="UTF-8"?><wfs:ValueCollection
xmlns:wfs="http://www.opengis.net/wfs/2.0" numberReturned="493">
<wfs:member>
<aixm:name xmlns:aixm="http://www.aixm.aero/schema/5.1">USAF, 25TH AIR
DIVISION, MCCHORD AFB</aixm:name>
</wfs:member>
<wfs:member>
<aixm:name xmlns:aixm="http://www.aixm.aero/schema/5.1">NOT
AVAILABLE</aixm:name>
</wfs:member>
<wfs:member>
<aixm:name xmlns:aixm="http://www.aixm.aero/schema/5.1">STRIKE FIGHTER WING
PACIFIC FLEET, NAVAL AIR STATION</aixm:name>
</wfs:member>
<wfs:member>
<aixm:name xmlns:aixm="http://www.aixm.aero/schema/5.1">NOT
AVAILABLE</aixm:name>
</wfs:member>
<wfs:member>
<aixm:name xmlns:aixm="http://www.aixm.aero/schema/5.1">USMC, U.S. MARINE
CORPS AIR STATION CHERRY POINT</aixm:name>
</wfs:member>
```

```
<wfs:member>
<aixm:name xmlns:aixm="http://www.aixm.aero/schema/5.1">NOT
AVAILABLE</aixm:name>
</wfs:member>
<wfs:member>
<aixm:name xmlns:aixm="http://www.aixm.aero/schema/5.1">120 FIGHTER
INTERCEPTOR GROUP, ANG</aixm:name>
</wfs:member>
<wfs:member>
<aixm:name xmlns:aixm="http://www.aixm.aero/schema/5.1">NOT
AVAILABLE</aixm:name>
</wfs:member>
<wfs:member>
<aixm:name xmlns:aixm="http://www.aixm.aero/schema/5.1">U.S. ARMY
UMATILLA CHEMICAL DEPOT</aixm:name>
</wfs:member>

<!—Truncated for brevity →
```

Example 8: Get transmission frequency of RadioCommunicationChannel with identifier: 9C0D0B68-A8CD-3E45-E044-00212803DA06

Request

```
<?xml version="1.0" encoding="UTF-8"?>
<wfs:GetPropertyValue
service="WFS"
version="2.0.0"
outputFormat="application/gml+xml; version=3.2"
valueReference="//aixm:frequencyTransmission"
xmlns:wfs="http://www.opengis.net/wfs/2.0"
xmlns:aixm="http://www.aixm.aero/schema/5.1"
xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:ogc="http://www.opengis.net/ogc"
xmlns:gml="http://www.opengis.net/gml/3.2"
xmlns:fes="http://www.opengis.net/fes/2.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.opengis.net/wfs/2.0
                    http://schemas.opengis.net/wfs/2.0/wfs.xsd">
<wfs:Query typeNames="aixm:RadioCommunicationChannel "
handle="Q01"><fes:Filter>
<fes:PropertyIsEqualTo>
<fes:ValueReference>//gml:identifier</fes:ValueReference>
   <fes:Literal>9C0D0B68-A8CD-3E45-E044-00212803DA06</fes:Literal>
</fes:PropertyIsEqualTo></fes:Filter></wfs:Query>
</wfs:GetPropertyValue>
```

Response

```
<?xml version="1.0" encoding="UTF-8"?>
<wfs:ValueCollection xmlns:wfs="http://www.opengis.net/wfs/2.0"
numberReturned="1">
    <wfs:member>
        <aixm:frequencyTransmission
xmlns:aixm="http://www.aixm.aero/schema/5.1"
uom="MHZ">134.45</aixm:frequencyTransmission>
    </wfs:member>
</wfs:ValueCollection>
```

D.1.8 Ad hoc queries

Ad hoc queries enable the WFS to be highly flexible, enabling them to simultaneously support a wide range of user requirements from:

1. **Data exchange use cases:**
   - Get all the current SAA message features or get the current SAA message features within an area of interest or by identifier
   - Get the latest Airspace activations
   - Get only the Airspace and Unit features updated in the last AIRAC cycle
2. **Decision-support use cases:**
   - Get me all the active SAA Airspaces between t1 and t2 within an area of interest
   - Get me all runways and taxiways that are active at Airport KBOS at t3

Within the pilot, the scenarios above demonstrated the ability for the WFS to support decision-support use cases.

D.2. Event Service Capabilities

The Event Service provided by the Institute for Geoinformatics (IfGI) of the University of Münster, Germany is based on the Sensor Event Service (SES) component used in the previous OGC OWS-6 and OWS-7 aviation threads. The IfGI Event Service is a stand-alone implementation of a publish-subscribe service. This means that the service is not implemented on top of another service (stand-alone) and that its main purpose is to provide access to data using the publish-subscribe communication pattern. The service is available as open source (GPLv2) via the 52° North initiative (https://52north.org/communities/sensorweb/ses/0.0.1/index.html).

The service was originally developed as an implementation of the Sensor Event Service (SES) specification published as an OGC discussion paper (OGC 08-133). This service is designed as an experimental successor of the Sensor Alert Service (SAS) with a focus on an enhanced set of functionalities and a wider use of existing standards. For instance, the publish-subscribe communication definition uses the OASIS Web Services Notification (WS-N) standard. For the definition of subscription filters, the service accepts XPaths, the OGC Filter Encoding and the Event Pattern Markup Language (EML).

The main modification of the IfGI Event Service for the FAA SAA Dissemination Pilot, compared to the OWS-7 version, was the implementation of a new parser for the events. In the Pilot Study the decision was made to use AIXM 5.1 basic messages as the root element, and airspace time slices for the encoding. Also, more AIXM properties were parsed. In the previous OWS testbeds, the relevant information in an event was the affected area, the time for which the data is valid and the feature identifier. In the FAA SAA Dissemination Pilot, airspace activation specific information was also used, including the affected flight levels, the activation status and the reservation phase. Further modifications concern the use of different topics and the filtering capabilities. To support all of the scenarios, methods like the temporal AnyInteracts filter were implemented (see section 8.1).

D.2.1 Using the Event Service

The details necessary to work with the IfGI Event Service are provided in this section. This includes the available properties for the filter expressions, available topics, and detailed examples for the service operations.

The Event Service implementation is available as open source software at 52°North. On their web site, there is a dedicated section on the Sensor Event Service (SES): https://52north.org/communities/sensorweb/ses/0.0.1/index.html

The SES implementation contains the additions made to support the aviation requirements including the FAA SAA Dissemination Pilot work. Installation guides for the service deployment and source code compilation are available besides snapshot releases in the downloads section.

D.2.2 Property Access

In general the following properties are parsed form the incoming events and can be accessed in filter encoding statements within an element:

geometry: the provided bounding box of the affected airspace

startTime: the begin position of the valid time of the according time slice

endTime: the end position of the valid time of the according time slice

validTime: the time interval of the valid time, can be used in the AnyInteracts filter

status: the activation status (e.g. ACTIVE)

reservationPhase: the activation's reservation phase (e.g. PENDING)

identifierValue: the value of the gml:identifier element of the airspace

identifierCodeSpace: the code space of the identifier

lowerLimit: the lower limit of the activation level

upperLimit: the upper limit of the activation level

In a filter statement they are accessed using the value reference element. This may look like this:

```
<fes:ValueReference>status</fes:ValueReference>
```

The identifier consists of two parts: the code space and the actual value. Both together are globally unique and thus, both should be used when building a filter for a specific airspace. However, they are provided as separate properties as a combination to a single one (e.g. by concatenation) can result in ambiguous values.

The values for the lower and upper limit are interpreted as meter unless a different unit of measure is specified. The reason for this behavior is that the Event Service does only use SI unit for internal matching. This allows a correct comparison of different compatible units, for instance feet and meters or meters and nautical miles. The examples below show how a filter can look like. The first example just uses meters, the second uses a Sensor Web Enablement (SWE) common data type to define the unit of measure.


```
<fes:Filter xmlns:fes="http://www.opengis.net/fes/2.0">
   <fes:PropertyIsGreaterThan>
      <fes:ValueReference>upperLimit</fes:ValueReference>
      <fes:Literal 100</fes:Literal>
   </fes:PropertyIsGreaterThan>
</fes:Filter>

<fes:Filter xmlns:fes="http://www.opengis.net/fes/2.0"

   xmlns:swe="http://www.opengis.net/swe/1.0.1" >
   <fes:PropertyIsGreaterThan>
      <fes:ValueReference>upperLimit</fes:ValueReference>
      <fes:Literal>
         <swe:Quantity>
            <swe:uom code="[ft_i]"/>
            <swe:value>100</swe:value>
         </swe:Quantity>
      </fes:Literal>
   </fes:PropertyIsGreaterThan>
</fes:Filter>
```

All Unified Code for Units of Measure (UCUM) codes can be used as a value for the unit code. In addition some further common codes can be supported. For this Pilot Study the only additional codes are 'nautical mile' and 'ft'. The table below lists some popular codes for spatial and flight level filters. More information regarding UCUM codes can be found here: http://aurora.regenstrief.org/~ucum/ucum.html.

Code	Description
m	UCUM code for meter, the SI unit for length and used as default
km	UCUM code for kilometer, equates to 1000 meters
[ft_i]	UCUM code for foot, common unit for flight levels
ft	Alternative code for foot, not UCUM[1]
[mi_i]	UCUM code for an international mile
[nmi_i]	UCUM code for a nautical mile[2]
nautical mile	Nautical mile as represented in some database systems, commonly used.

Table D-2. Event Service Unit Codes

[1] The UCUM interpretation of 'ft' is 'femto tonne' which the same a a 'nano gram' thus 1*10^-9 grams.

[2] The UCUM code 'nm' cannot be used for nautical miles as it collides with the code for nano meters.

Annex E
SAA Dissemination Additional Scenarios

In addition to the Commercial Flight scenario (section 6.3) used to demonstrate the fusion of static and dynamic SAA information, many additional operational concepts and component features were identified that could be used to exercise and demonstrate the SAA Pilot component architecture.

E.1. Mission Planning Exercises

The mission planner of the Air Armament Center at Eglin AFB needs to schedule a low-altitude training mission in one of its SUAs. They have 15 SUAs (7 MOAs and 8 restricted areas). They set a query to the WFS for all SUAs that have a using agency of USAF, AIR ARMAMENT CENTER, EGLIN AFB, and optionally a SUA type of MOA or R. They ask for the low and high altitudes to be returned, so that they can identify which SUAs have the low altitudes that they need for their training mission.

A mission planner is scheduling a training mission at Coastal 1 East. Since this will be a training mission, and the novice pilots may fly beyond the borders of the MOA, the mission planner sets a spatial query for any SUAs that border Coastal 1 East, and that have a scheduled mission during the desired training times.

E.2. Retrieval of airspaces active at a given time instant / during a time period

Users often need to know which airspaces are active at a given time instant or during a given time period. This can for example be used in flight planning.

The user may restrict the scope of airspaces to one (defined via its name/identifier) or to those that satisfy a spatial filter. This requires creation of a query which takes both an airspace's timesheet with the regular activation schedule AND the ad-hoc activations into account.

Performing this kind of query at the WFS would prevent the need for clients to do this kind of computation by themselves.

E.3. Logistics Supply Mission

An Air Mobility Command pilot at Eglin AFB is notified that some equipment is needed immediately at McGuire AFB. The local Air Base ops are closed. The pilot uses the Internet to identify airspace scheduled by McGuire that involve altitudes below 10000 feet MSL, and to obtain SAA reservation schedules for these airspace that will be active within the next 24 hours. He obtains flight plan approval from the local ARTCC, and his departure is scheduled for three hours in the future. Just before boarding the airplane, the pilot receives notification of a new SAA activation that impacts his flight plan. These pre-flight changes are seen as highlighted modifications on a display showing both his original flight plan and the new route.

Scenario Step Description	Involved Components
The pilot at Eglin AFB receives orders to fly supplies to McGuire AFB. The supplies are already onboard a plane without fuel.	-
The pilot determines that the local Air Base dispatch operations personnel are gone for the day, and he will need to plan the flight himself.	-
The pilot submits a web service query to identify SAA that may impact his flight between the two airfields.	WFS
The pilot subscribes to SAA schedule event notifications for the en-route SAA.	Event Service
To plan the landing approach, the pilot submits a web service query to identify airspace scheduled by McGuire AFB that involve altitudes below 10,000 ft MSL.	WFS
The pilot subscribes to schedule event notifications for the SAA scheduled by McGuire AFB.	Event Service
The pilot receives event notifications of SAA reservations expected to be active within the next 6 hours.	Event Service Adapter, Event Service
The pilot plans his flight and determines the amount of fuel required.	-
The pilot calls ZJX (Jacksonville) to submit his flight plan request. The approved flight plan is scheduled for departure in 3 hours.	-

E.4. VFR GPS Chart Production

This scenario describes the interaction between a client production data maintenance application and the FAA SAA WFS and Event Service for the ingestion of AIXM SAA data for use in the client end product.

The client end product contains SAA static data that is depicted graphically as geographic boundaries. Those boundaries have associated SAA Schedule data information displayed graphically as well.

The first example describes the initial population of the client data store with both Baseline SAA static data information and Baseline SAA Schedule data information. It also establishes the subscription to the Event Services for future notification of changes as they occur and retrieval of the changed information respectively.

The second and third examples, describe the Event Service Adapter processing for both SAA static data and SAA Schedule data changes.

In all cases, the client production process ingests the SAA AIXM data, validates it is compliant with the published AIXM version, and further converts the AIXM data into appropriate formats for the client end product. The final output for demonstration will be a graphical display of the SAA static and Schedule data as appropriate.

1. External client subscribes to WFS and Event Service
    - WFS and Event Service authenticates subscription
    - WFS and Event Service acknowledges connection
2. External client requests Baseline SAA static data from WFS
    - WFS sends SAA static data to client
    - External client validates schema version
3. External client requests SAA data from WFS
    - WFS sends SAA data to client
    - External client validates schema version
4. External client processes SAA data into internal production processes

E.5. SAA Update Demo

1. Review of Airspaces over existing charts

Users with some responsibility for the update of the SAA Charts or preparing temporary updates require a simple and pervasive mechanism for accessing charts without needing an installed client.

The user will go to a landing page and decide on the mode they wish to use. They may enter a specific airport of interest to start with, or may wish simply pan manually to a specific area of interest.

Users will typically want to view the data over a map, possibly with different airspace types highlighted in different ways. The map tells part of the story, but users can get a better assessment of the vertical displacement of airspace with a vertical profile.

Displaying the vertical profile on a great circle line between, for example two airports may be helpful in visualizing the airspace present. It may also be valuable to limit the airspaces displayed to a certain time range.

2. Definition of Issues/Temporary Changes to the Airspace

Users may wish to indicate problems with an existing airspace by adding an 'observation' of some form, or may wish to define a temporary special activity airspace such as Time of Use Altitude Reservation (AltRev) and Temporary Flight Restriction (TFR) areas.

Other users should be able to load these additional temporary airspace definitions and overlay them in a different color over the base set of restrictions.