OGC Testbed-14

Semantically Enabled Aviation Data Models Engineering Report

Table of Contents

Publication Date: 2019-02-07

Approval Date: 2018-12-13

Submission Date: 2018-11-27

Reference number of this document: OGC 18-035

Reference URL for this document: http://www.opengis.net/doc/PER/t14-D002

Category: Public Engineering Report

Editor: Stephane Fellah

Title: OGC Testbed-14: Semantically Enabled Aviation Data Models Engineering Report

---

OGC Engineering Report

COPYRIGHT

WARNING

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

LICENSE AGREEMENT

Permission is hereby granted by the Open Geospatial Consortium, ("Licensor"), free of charge and subject to the terms set forth below, to any person obtaining a copy of this Intellectual Property and any associated documentation, to deal in the Intellectual Property without restriction (except as set forth below), including without limitation the rights to implement, use, copy, modify, merge, publish, distribute, and/or sublicense copies of the Intellectual Property, and to permit persons to whom the Intellectual Property is furnished to do so, provided that all copyright notices on the intellectual property are retained intact and that each person to whom the Intellectual Property is furnished agrees to the terms of this Agreement.

If you modify the Intellectual Property, all copies of the modified Intellectual Property must include, in addition to the above copyright notice, a notice that the Intellectual Property includes modifications that have not been approved or adopted by LICENSOR.

THIS LICENSE IS A COPYRIGHT LICENSE ONLY, AND DOES NOT CONVEY ANY RIGHTS UNDER ANY PATENTS THAT MAY BE IN FORCE ANYWHERE IN THE WORLD. THE INTELLECTUAL PROPERTY IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. THE COPYRIGHT HOLDER OR HOLDERS INCLUDED IN THIS NOTICE DO NOT WARRANT THAT THE FUNCTIONS CONTAINED IN THE INTELLECTUAL PROPERTY WILL MEET YOUR REQUIREMENTS OR THAT THE OPERATION OF THE INTELLECTUAL PROPERTY WILL BE UNINTERRUPTED OR ERROR FREE. ANY USE OF THE INTELLECTUAL PROPERTY SHALL BE MADE ENTIRELY AT THE USER’S OWN RISK. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR ANY CONTRIBUTOR OF INTELLECTUAL PROPERTY RIGHTS TO THE INTELLECTUAL PROPERTY BE LIABLE FOR ANY CLAIM, OR ANY DIRECT, SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES, OR ANY DAMAGES WHATSOEVER RESULTING FROM ANY ALLEGED INFRINGEMENT OR ANY LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR UNDER ANY OTHER LEGAL THEORY, ARISING OUT OF OR IN CONNECTION WITH THE IMPLEMENTATION, USE, COMMERCIALIZATION OR PERFORMANCE OF THIS INTELLECTUAL PROPERTY.

This license is effective until terminated. You may terminate it at any time by destroying the Intellectual Property together with all copies in any form. The license will also terminate if you fail to comply with any term or condition of this Agreement. Except as provided in the following sentence, no such termination of this license shall require the termination of any third party end-user sublicense to the Intellectual Property which is in force as of the date of notice of such termination. In addition, should the Intellectual Property, or the operation of the Intellectual Property, infringe, or in LICENSOR’s sole opinion be likely to infringe, any patent, copyright, trademark or other right of a third party, you agree that LICENSOR, in its sole discretion, may terminate this license without any compensation or liability to you, your licensees or any other party. You agree upon termination of any kind to destroy or cause to be destroyed the Intellectual Property together with all copies in any form, whether held by you or by any third party.

Except as contained in this notice, the name of LICENSOR or of any other holder of a copyright in all or part of the Intellectual Property shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Intellectual Property without prior written authorization of LICENSOR or such copyright holder. LICENSOR is and shall at all times be the sole entity that may authorize you or any third party to use certification marks, trademarks or other special designations to

indicate compliance with any LICENSOR standards or specifications.

This Agreement is governed by the laws of the Commonwealth of Massachusetts. The application to this Agreement of the United Nations Convention on Contracts for the International Sale of Goods is hereby expressly excluded. In the event any provision of this Agreement shall be deemed unenforceable, void or invalid, such provision shall be modified so as to make it valid and enforceable, and as so modified the entire Agreement shall remain in full force and effect. No decision, action or inaction by LICENSOR shall be construed to be a waiver of any rights or remedies available to it.

None of the Intellectual Property or underlying information or technology may be downloaded or otherwise exported or reexported in violation of U.S. export laws and regulations. In addition, you are responsible for complying with any local laws in your jurisdiction which may impact your right to import, export or use the Intellectual Property, and you represent that you have complied with any regulations or registration procedures required by applicable law to make this license enforceable.

Chapter 1. Summary

This Engineering Report (ER) summarizes the OGC Testbed-14 findings and recommendations to “semantically enable” existing data and metadata models used in the aviation industry. Examples of such data and metadata models include Aeronautical Information Exchange Model (AIXM) [1], Weather Information Exchange Model (WXXM) [2], Flight Information Exchange Model (FIXM) [3],Web Service Description Document (WSDD), Service Description Conceptual Model (SDCM) [4]). These models use Linked Data standards to represent this information and aim to improve the search and discovery of services and information in the aviation domain using the System Wide Information Management (SWIM) environment. This report provides a review of the existing data models and explore different approaches to provide a semantic representation of the current metadata and data models used in the aviation domain. The ER also discusses the role and importance of the controlled vocabularies.

1.1. Requirements & Research Motivation

The current SWIM developed by Federal Aviation Administration (FAA) provides an API to search and discover web services. The current service model is based on the SDCM model, which is inspired by the Semantic Markup for Web Services (OWL-S) [5] specification of the World Wide Web Consortium (W3C). However, the response returned by the API is encoded in Extensible Markup Language (XML) and uses XML Schema for validation. The description of the service is focused on defining how the service input/output can be constructed and manipulated (e.g., XML schemas, Web Service Description Language (WSDL) [6] files) but has limited semantics about the information it operates on. The current service description standards (e.g., WSDL and XML Schema) operate almost entirely at the syntactic level, focusing only on describing exposed functionality (methods signatures, input/output types) and failing to capture enough semantics (i.e., they define structure, not meaning). The standards for free-text, human-consumable documents, (e.g., FAA’s WSDD), support a sufficient amount of semantics but are not suitable for automated discovery and provide very limited support for semantic interoperability.

The following requirements were to be addressed in the OGC Testbed-14:

- Testbed-14 shall provide recommendations for making existing data models used in aviation industry (e.g., AIXM, WXXM, and FIXM) “semantically enabled”.
- The data models shall be enabled to present their contents in formats suitable for adaptation by Semantic Web technologies, including considerations for role and applicability of ontologies and linked data approaches to complex information realms such as SWIM.
- The work shall provide a clear definition of next steps.

1.2. Prior-After Comparison

The topic of semantic-enablement of models used in the domain of aviation has been explored previously in OGC Testbeds (OGC 16-039 [7],OGC 17-036 [8]). In past demonstrations, analyses recommended the use of run-time registries and complex use cases for service discovery and data taxonomy/ontology. However, much of the information exchanged within the FAA National Airspace System (NAS) [9] System-Wide Information Management (SWIM) network is made up of

various data models using XML schema encoding (such as AIXM), which addresses only the structure and syntax of information exchanged between systems, but not the semantic aspect of the model. Testbed-12 [7] and 13 [8] have made progress toward the semantic enablement of the controlled vocabularies using the Simple Knowledge Organization System (SKOS) [10] encoding but these vocabularies were still referred from the XML document based on structure and syntax. This hybrid approach does not allow the usage of off-the-shelf solutions for Linked Data such as linking heterogeneous domain entities, deductive reasoning and unified access to information. Systems are currently built around specific data models and unable to communicate and link to each other causing duplication of information and making difficult to search and discover information that are relevant for users.

The goal of this ER is to formulate an approach to semantically enable the different data models, taxonomy and service descriptions that can incorporate enough semantic metadata, including descriptive metadata, geospatial-temporal characteristics, quality information for fitness-of-use) about the services and data information, so that they can facilitate the integration of information and services, improve search and discovery in the current registry, and increase the level of automation (reasoning, access by agents) in systems.

1.3. Recommendations for Future Work

The solutions described in this engineering report may provide further insights if implemented as a greater solution for information registries. Furthermore, implementation of the recommendations for SDCM will provide a path forward for prototyping and implementation of SWIM registries and search/discovery of services and information.

This engineering report proposes the following tasks to be explored for future testbeds:

- Controlled Vocabulary Management: The management and governance of controlled vocabularies play a crucial role in the semantic "tagging" of resources managed by NAS (Datasets, Services, Maps, Layers, Documents,etc..) and the search and discovery of these resources. Vocabularies also enable semantic enrichment, reasoning and provide a common framework to represent information based on Linked Data representation facilitating the integration of information silos. Future testbeds should investigate the requirements, design and implementation of an API for a controlled vocabulary services that manages ontologies and taxonomies (encoded in SKOS).
- Development of a Semantic Registry Information Model (SRIM) [11] Application Profile for SWIM that extends the current SRIM model (which is based on a number of well-known vocabularies such as the W3C Data Catalog (DCAT) [12], GeoDCAT-AP [13], Dublin Core, SKOS [10], Provenance, Authoring and Versioning (PAV) ontology [14]) to describe services and datasets specific for the aviation domain and that aligns with the current SDCM effort.
- SRIM Semantic Registry implementation [11] of the SWIM Application Profile that showcases the search and discovery of the assets managed by the SWIM ecosystem, focusing in particular to Services and Datasets.
- Investigation of the modularization and encoding of aviation ontologies using the NASA ATM ontology as a starting point and identification of gaps with the current existing standards (AIXM, WXXM,FIXM).
- Demonstration of the use of semantic enablement of aviation data using Linked Data

representation based on the ontologies developed in the previous recommendations by deploying a Linked Data API that offers a Representational State Transfer (REST) API and a GeoSPARQL Endpoint), as well as demonstration of the link to the Dataset metadata description registered in a Semantic Registry.

1.4. What does this ER mean for the Working Group and OGC in general

This engineering report documents the findings, approaches and recommendations to semantically enable existing data models used in the aviation domain. The semantic-enablement of these existing models will improve search and discovery of aviation-related information using a semantic registry.

The chosen working group for review of this ER is the Geosemantics Domain Working Group (DWG). This work may also be applicable to the Aviation DWG which is co-sponsored by the FAA and EUROCONTROL.

The scope of the Geosemantics DWG is any aspect of semantic conceptual modeling and formal representation of aviation data models which advances the geospatial interoperability mission of OGC. A particular focus will be the adoption or development of tools and methods in support of these activities. It is the mission of the Geosemantics DWG to establish an interoperable and actionable semantic framework for representing the geospatial knowledge domains of information communities as well as mediating between them. This ER will address the need for semantically enablement of aviation data models. The use of semantics will enable better descriptions of aviation-related assets, including OGC web services in the FAA's SWIM registry, datasets, maps, layers, etc.

1.5. Document contributor contact points

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

Contacts

Name	Organization
Stephane Fellah	Image Matters LLC
Yann Le Franc	e-Science Data Factory

1.6. Foreword

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

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

supporting documentation.

Chapter 2. References

- RDF Schema 1.1, W3C Recommendation 25 February 2014 , available online at http://www.w3.org/TR/rdf-schema/
- OWL 2 Web Ontology Language Document Overview (Second Edition)], W3C Recommendation 11 December 2012, available online at https://www.w3.org/TR/owl2-overview/
- OWL 2 Web Ontology Language, Direct Semantics (Second Edition), W3C Recommendation 11 December 2012, available online at http://www.w3.org/TR/2012/REC-owl2-direct-semantics-20121211/
- OWL 2 Web Ontology Language, Structural Specification and Functional-Style Syntax (Second Edition), W3C Recommendation 11 December 2012, available online at http://www.w3.org/TR/2012/REC-owl2-syntax-20121211/
- SKOS Simple Knowledge Organization System Reference, W3C Recommendation 18 August 2009, available online at http://www.w3.org/TR/2009/REC-skos-reference-20090818/
- Data Catalog Vocabulary (DCAT), W3C Recommendation 16 January 2014, available online at https://www.w3.org/TR/vocab-dcat/
- Web Services Description Language (WSDL) Version 2.0 Part 1: Core Language, W3C Recommendation 26 June 2007 , available online at https://www.w3.org/TR/wsdl20/
- OGC 12-168r6 OGC Catalogue Services 3.0, Open Geospatial Consortium, available online at http://docs.opengeospatial.org/is/12-168r6/12-168r6.html
- Gleaning Resource Descriptions from Dialects of Languages (GRDDL), W3C Recommendation REC-grddl-20070911, September 2007. available online at https://www.w3.org/TR/grddl/

Chapter 3. Terms and definitions

For the purposes of this report, the definitions specified in Clause 4 of the OWS Common Implementation Standard OGC 06-121r9 [https://portal.opengeospatial.org/files/?artifact_id=38867&version=2] shall apply. In addition, the following terms and definitions apply.

3.1. Semantics

```
A conceptualization of the implied meaning of information that requires words and/or
symbols within a usage context.
```

3.2. Service Description

```
The information needed in order to use, or consider using, a service.
```

3.3. Service-Oriented Architecture (SOA)

```
A paradigm for organizing and utilizing distributed capabilities that may be under the
control of different ownership domains. A SOA provides a uniform means to offer,
discover, interact with, and use capabilities to produce desired effects consistent
with measurable preconditions and expectations.
```

3.4. Registry

```
An enabling infrastructure that uses a formal registration process to store, catalog,
and manage metadata relevant to a service. A registry supports the search,
identification, and understanding of resources, as well as query capabilities.
```

3.5. System Wide Information Management (SWIM)

```
A concept using Service Oriented Architecture to facility the exchange Air Traffic
Management information amongst stakeholders in the aviation domain such as Air
Navigation Service Providers, airports, and airspace users.
```

3.6. Taxonomy

```
A system or controlled list of values by which to categorize or classify objects.
```

3.7. Web Service

A platform-independent, loosely-coupled software component designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format. Other systems interact with the Web service in a manner prescribed by its description by means of XML-based messages conveyed using Internet transport protocols in conjunction with other Web-related standards.

Chapter 4. Abbreviated Terms

- ATM Air Traffic Management
- ICAO International Civil Aviation Organization
- FAA Federal Aviation Administration (United States)
- NAS National Airspace System (United States)
- NSRR NAS Service Registry and Repository
- OWL Web Ontology Language (W3C)
- OWL-S Semantic Markup for Web Services
- OWS OGC Web Service
- RDF Resource Description Framework (W3C)
- RDFS Resource Description Framework Schema (W3C)
- SDCM Service Description Conceptual Model
- SOA Service Oriented Architecture
- SWIM System Wide Information Management
- WSDOM Web Service Description Ontological Model

Chapter 5. Overview

The ER is composed of the following sections:

Section 6 provides a review of the existing data models currently used by FAA and ontologies that are relevant for semantic-enablement of the aviation domain.

Section 7 provides a comparison of existing data-centric services approach with semantic-enabled services approach and highlights the benefits to use the latter approach. Two levels of semantic enablement are identified:

- Metadata level semantic enablement to support search and discovery of information
- Instance level semantic enablement to support data integration and reasoning.

Section 8 focuses on the Metadata-level semantic enablement. It discusses some of the issues found currently in existing metadata standards in particular ISO 19115. The section provides a review of relevant semantic metadata standards (DCAT, GeoDCAT,Project Open Data, SRIM) and discusses the approaches to semantic-enable metadata on the current infrastructure.

Section 9 focuses on the Data-level semantic enablement. It discusses the issue of data-silos and how semantics can enable the integration of data silos. An inventory of different technical approaches to semantic-enable the existing infrastructure is given with discussions of pro/cons/applicability of each approach.

Section 10 addresses the role of Controlled Vocabularies in the aviation domain. It provides a classification of the different types of controlled vocabularies. It also discusses some improvements that can be done on existing vocabularies, provides recommendations about the governance and lifecycle of controlled vocabularies, and highlights some best practices for ontology design.

Annex A provides some metadata samples illustrating the description of datasets and their relationships to services.

Chapter 6. Review of Data Models

This section provides a review of the existing data models currently used by FAA and the aviation community.

6.1. Information Exchange Models

The first set of data models reviewed related to information exchange models. There are three primary standards currently used by the industry (FIXM, AIXM, and WXXM). The models have an abstract model defined in UML and an encoding based on XML schema. There is currently no encoding based on linked data standards such as the Web Ontology Language (OWL) [15] or RDF Schema [16].

6.1.1. Flight Information Exchange Model (FIXM)

The Flight Information Exchange Model (FIXM) is an exchange model capturing Flight and Flow information that is globally standardized. It captures data related to flights throughout their lifecycle from pre-flight planning through departure, en-route, and arrival segments, including data on aircraft location, aircraft characteristics, and cargo contents. The model is decomposed in modular packages (Figure 1) encoded in XML Schema.

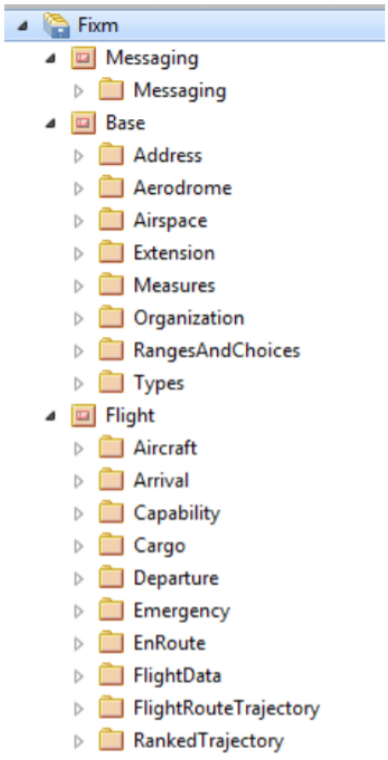

Figure 1. FIXM Core Package Structure

6.1.2. Aeronautical Information Exchange (AIXM) Model

The AIXM model describes airspace routes, procedures, boundaries, fixes, navaids, and airport surface elements (runways, gates, etc.). The objective of the Aeronautical Information Exchange Model (AIXM) is to enable the provision in digital format of the aeronautical information that is in the scope of Aeronautical Information Services (AIS). The AIS information/data flows are

increasingly complex and made up of interconnected systems. They involve many actors including multiple suppliers and consumers. There is also a growing need in the global Air Traffic Management (ATM) system for high data quality and for cost efficiency.

6.1.3. Weather Information Exchange Model (WXXM)

The WXXM is designed to enable the management and distribution of weather data in digital format (XML). WXXM version 2.0 is based on Geography Markup Language (GML) and is one of the GML Application Schemas. It has been developed by the FAA and the European Organization for the Safety of Air Navigation (EUROCONTROL). WXXM is a member of a family of data models designed for use in aviation safety, notably AIXM and FIXM.

WXXM models meteorological observations, forecasts, and measurement procedures for weather features and phenomena with spatial and temporal extent. Independent from governmental standardization efforts, the airline industry has been developing Airline Industry Data Model (AIDM).

6.1.4. NASA Air Traffic Management (ATM) Model

The National Aeronautics and Space Administration (NASA) has developed a prototype data integration system that demonstrates how ontologies can be used to integrate, query, and search over various sources of heterogeneous air traffic management (ATM) data, including data from FAA, National Oceanic and Atmospheric Administration (NOAA), NASA, and other providers. In this prototype, a common ontology is used to bridge multiple types of aviation data models and enable cross-dataset querying. In general, cross-dataset querying is very challenging due to differing data formats, nomenclature, and organizational structure. Data can only be combined from multiple sources by expending significant effort to write customized code that integrates selected data on an as-needed, piecemeal basis. To address this problem, NASA is building an integrated data source that obviates this need to write special- purpose, one-off integration code.

As part of this approach, an overarching data model (the NASA ATM Ontology [1: https://data.nasa.gov/ontologies/atmonto/NAS]) has been designed and implemented to serve as a backbone upon which to overlay data from multiple sources. The ontology data model is scoped sufficiently broadly to interconnect data from several different aviation realms, including flight, traffic management, aeronautical information, weather, and carrier operations. The ontology is currently populated with instance data corresponding to over 100K flights arriving and departing the three major airports in the New York Metroplex (KEWR, KJFK, KLGA) during July 2014. This data is encoded in approximately 250M triple statements and incorporates the following sources: flight track data from FAA's ASDI (Aircraft Situation Display to Industry) data feed; airport weather from the NOAA's METeorological Aerodrome Reports (METAR) and Terminal Aerodrome Forecasts (TAF) data feeds; information on traffic management initiatives from FAA's Command Center website; historical traffic counts and delay statistics from the Department of Transportation (DOT)'s Aviation System Performance Metrics (ASPM)database; infrastructure data on US routes, fixes, airways, and sectors used by FAA's En Route Automation Modernization (ERAM) system; aircraft information from FAA's aircraft registry and from the Commercial Aviation Safety Team (CAST)/International Civil Aviation Organization (ICAO) common aircraft taxonomy; and web-based information on airlines, airport terminals and gates. Of the 250M triples, about 99.5% represent specific flight, weather, or traffic management data. These are dominated overwhelmingly by

triples capturing flight track parameters from ASDI – such as altitude, location, and groundspeed – captured at the rate of one per minute of flight. The remaining 0.5% of triples capture mostly static aeronautical infrastructure information (e.g., data on airspace routes, sectors, and fixes, NAS facilities, airports, aircraft makes and models, airlines, etc.).

The figures below (all extracted from the abovementioned NASA Technical Memo) provide some intuition about how the classes and properties represent the structure, content, and relationships found in ATM data. Only a subset of available figures is shown here; please consult the NASA Technical Memo for additional illustrations.

Figure 2 illustrates the key relationships among selected New York area airspace structure and facility instances. Sector nas:ZNYsector075 is one of the sectors located in the New York Air Route Traffic Control Center (ARTCC) instance nas:ZNYcenter. Sector 075 is composed of two stacked horizontal layers of airspace, each represented by a shear-sided polygon of a certain height (only one polygon is depicted in the figure). The ZNY ARTCC has operational agreements with the FAA command center and the New York Terminal Radar Approach Control (TRACON) facility, which in turn has agreements with each of the airports in its territory. The ZNY Tier 1 structure contains all ARTCCs that directly border the ZNY ARTCC. (Note that only a small subset of instances is illustrated in order to keep the figure uncluttered and readable.)

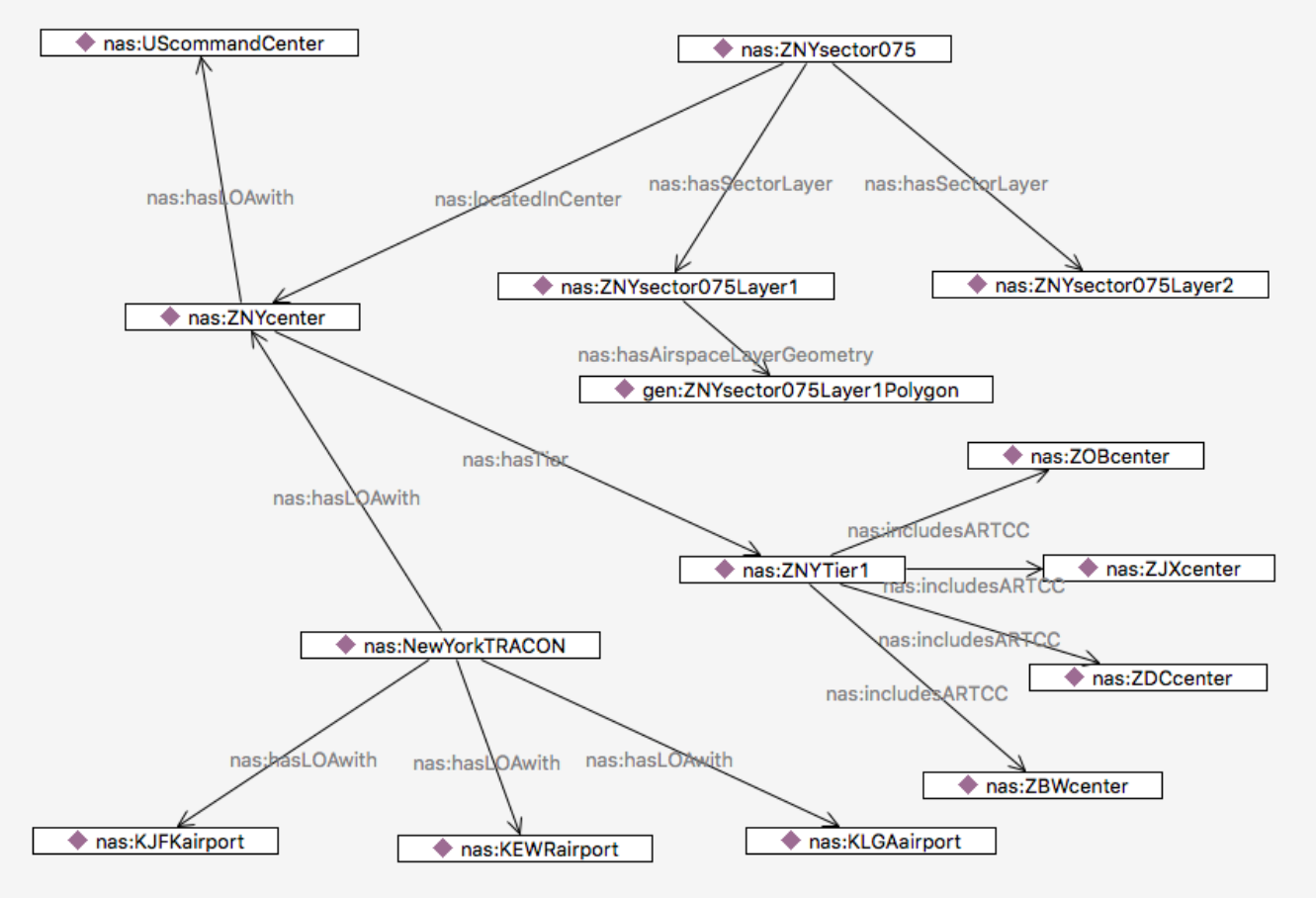

Figure 2. ATM Ontology- Structure of the NAS

Figure 3 illustrates the basic components of the ontology representation of a flight — in this illustration, Delta Airlines flight DAL435 on 2014-07-15. The flight is associated with its departure and arrival airports; the aircraft, aircraft type, and operating carrier; and the actual and planned flight route.

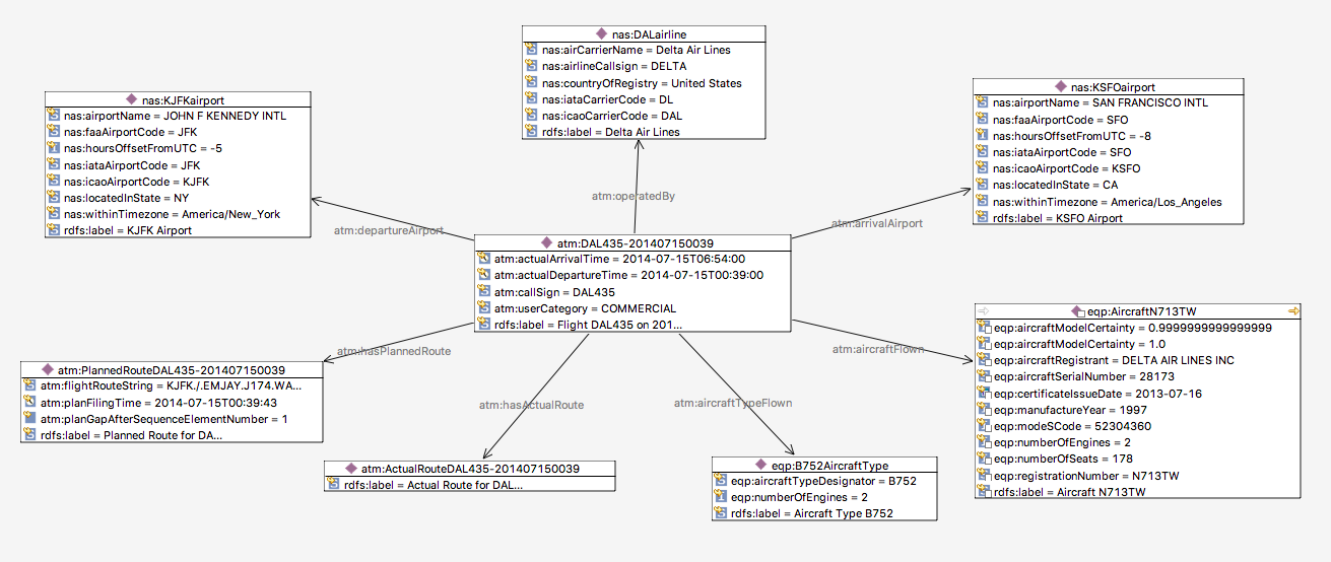

Figure 3. Flight Structure

Figure 4 illustrates the relationships among instances of aircraft, carrier, flight, model, manufacturer, and other classes associated with Delta Airlines flight DAL435 on 2014-07-15 (shown in Figure 3, above). The aircraft flown for this flight is N713TW, a Boeing model 757-2Q8, one of the B757-200 family of aircraft. The aircraft family is represented as a model class and the specific model is represented as an instance of that class. The FAA also designates an aircraft type, which may cover models in multiple aircraft families. The aircraft type for model 757-2Q8 is B752. Associated with type B752 aircraft are a set of instances that describe the engine type, wake turbulence category, and weight class of all B752 type aircraft.

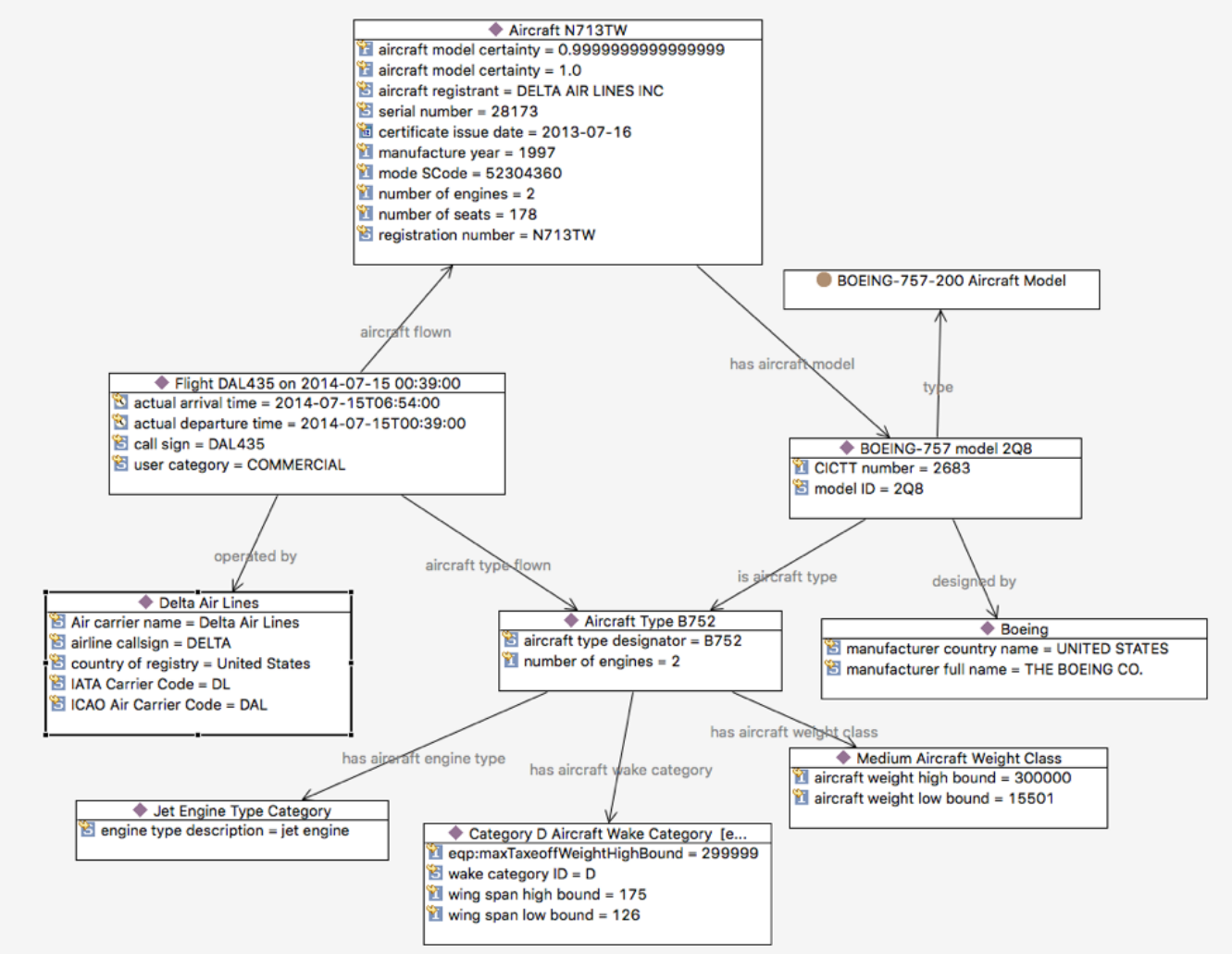

Figure 4. Relationships among aircraft, carrier, flight, model, manufacturer, and related classes

Figure 5 illustrates how the actual and planned flight routes for American Airlines Flight #AAL335 are connected to the flight instance (atm:AAL335-201407150017), which is depicted at the root of the tree structure shown. The actual flight route is represented as a sequence of track points (atm:AircraftTrackPoint). Each track point represents a specific reporting time when the aircraft's fix and speed is captured and relayed to ground systems. The track points are each linked to an instance of atm:LatLonFix, which stores the latitude, longitude, and altitude. The planned flight root is detailed in Figure 6 below.

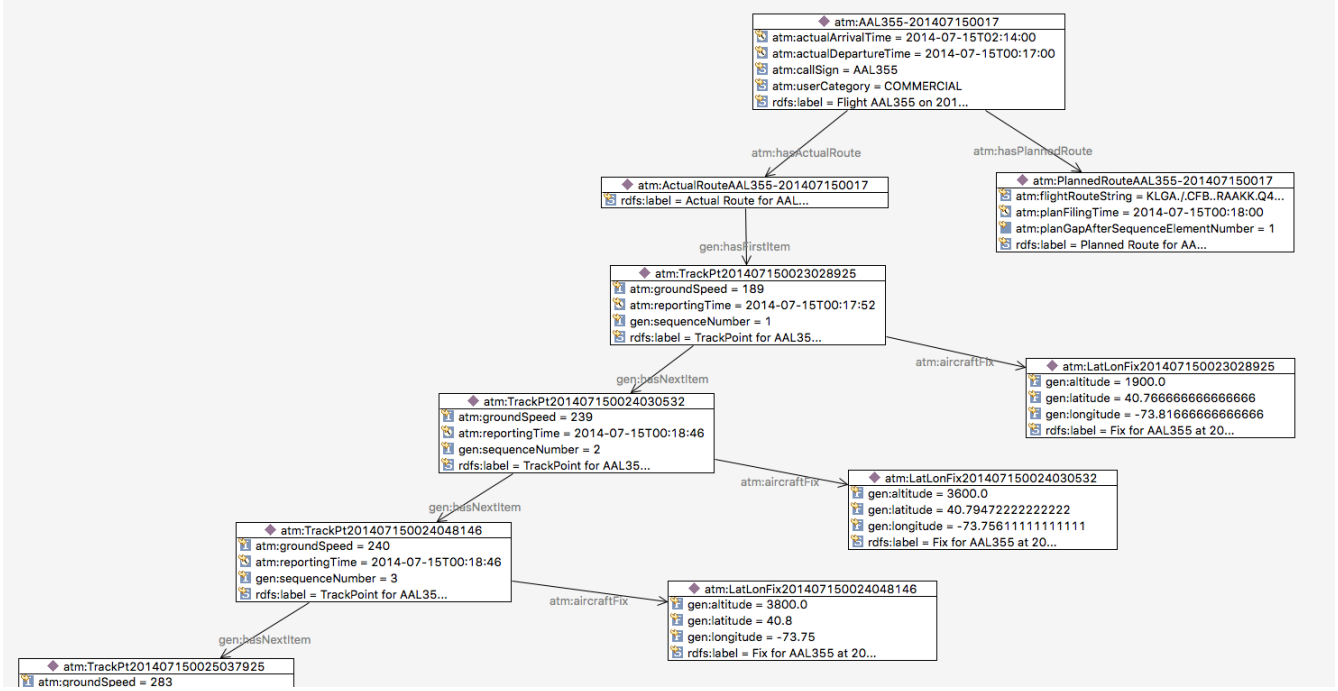

Figure 5. Structure of an Actual Flight Route (Flight Trajectory)

Figure 6 elaborates on the representation for the flight plan (atm:PlannedRouteAAL335-201407150017) shown at the root of this tree and also in Figure 5. The flight plan includes a flight route string 'KLGA./.CFB..RAAKK.Q436.EMMMA.WYNDE5.KORD', which is a representation of the path to be flown through the airspace. The flight route string is initially filed by the pilot prior to takeoff and modified as needed en route. The root node in the graph contains this route string as a property, but the string is also converted into an explicit sequence of 'container' nodes that include (via the property atm:hasNavElement) the major navigational components through which the flight is planned to progress: the originating airport (KLGA); a VOR fix (CFB); a portion of a high-altitude flight route (Q436); a traverse through a standard terminal arrival route (STAR WYNDE5); and the destination airport (KORD).

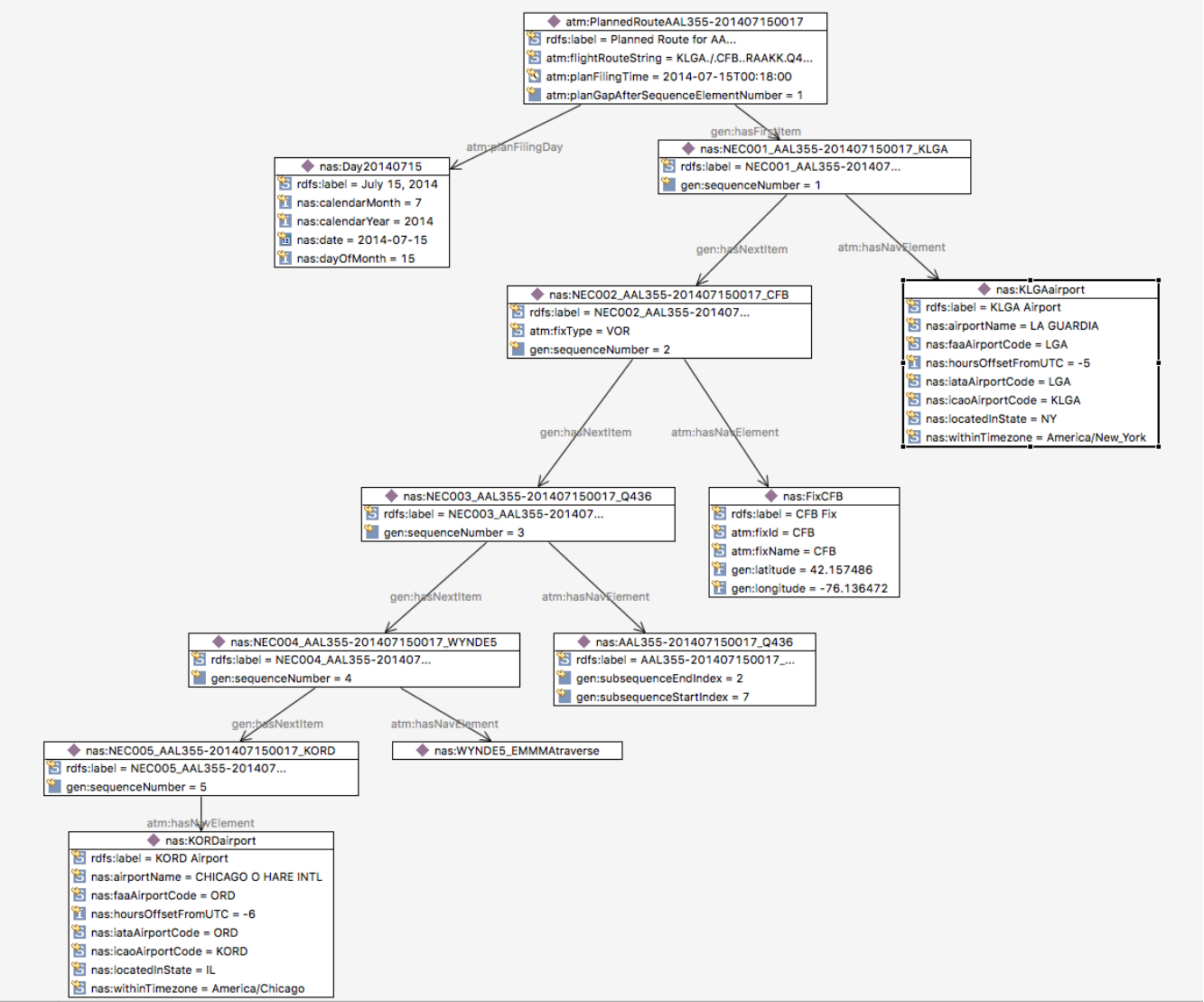

Figure 6. Example of Flight Plan

6.2. Service description models

6.2.1. Service Description Conceptual Model (SDCM)

System Wide Information Management (SWIM) is a data distribution framework started by FAA and EUROCONTROL and adopted by ICAO. The goal of the framework is to standardize mechanisms for delivery of ATM data between producers and consumers using a subscription-based Service-Oriented Architecture (SOA). As part of the SWIM architecture, data providers create services to access their data. For the FAA, these services are published in the NAS Services Registry/Repository (NSRR). NSSR is a catalog of all SWIM services and provides documentation on various aspects of each service, including its provider, functionality, quality characteristics, interface, and implementation.

As part of this effort, FAA SWIM Program and the Single European Sky Air Traffic Management (ATM) Research Programme (SESAR) Joint Undertaking (SJU) have developed a conceptual model describing services semantically called **Service Description Conceptual Model (SDCM)** [4] [17].

SDCM provides a graphical and lexical representation of the properties, structure, and interrelationships of all service metadata elements, collectively known as a Service Description. The

objectives of the SDCM are:

- To define a conceptual model of a Service Description based on consistent application of SOA principles;
- To establish adequate and consistent semantics for concepts used in documentation for SOA-based services;
- To advance a common and shared understanding of SOA concepts among international partners;
- To promote a technological means for describing all relevant aspects of a service in a manner suitable for both human-readable and machine-processable representations.

The SDM adheres to the following design principles:

- It shall be based on widely-used industry service description standards and models (e.g., OWL-S, WSDL, OASIS SOA Reference Model);
- It shall be extensible to allow deriving and adding more elements to address organization-specific tasks;
- It shall be neutral to any organizational governance model (e.g., in SESAR or FAA SWIM programs);
- It shall be agnostic to the service technological solution (e.g., it shall not be tailored to method, message or resource oriented implementations);
- It shall be vendor neutral (e.g., it shall not support any proprietary implementation of UML and/or vocabularies).

SDCM is based on the OWL-S model. The three classes composing this model describe what the service does (service profile), how to access it (service model) and how it works (service grounding) (Figure 7).

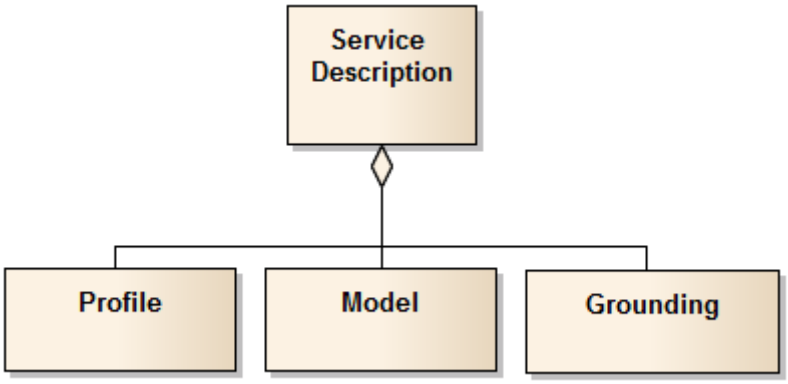

Figure 7. SDCM Service Description

The Service Profile contains mainly metadata information (provider's information, function, quality of service, security, ...) that are needed for search and discovery (Figure 8). This is part of the service description that is highly relevant for registration of services and to enable semantic search and discovery of services. In this ER, we show how the profile information can be used by a semantic registry by providing the alignment of service profile to semantic registry model. ServiceProfile class provides a good extension point. This class acts as container for the most

relevant metadata, such as service functions and real-world effects, quality of service, security parameters, service providers and consumers. The service profile class can be enriched with additional classes, properties and individuals to accommodate the specific needs for search and discovery of services in a semantic registry. Individuals of "ServiceProfile" type can contain one or many properties depicting the concept of "Service function" associated with "real-world effects". In order to extend the ontology to provide controlled vocabulary (values) for these concepts, domain experts need to define taxonomy of business functions and real world effects based on aviation traffic (management) domain needs.

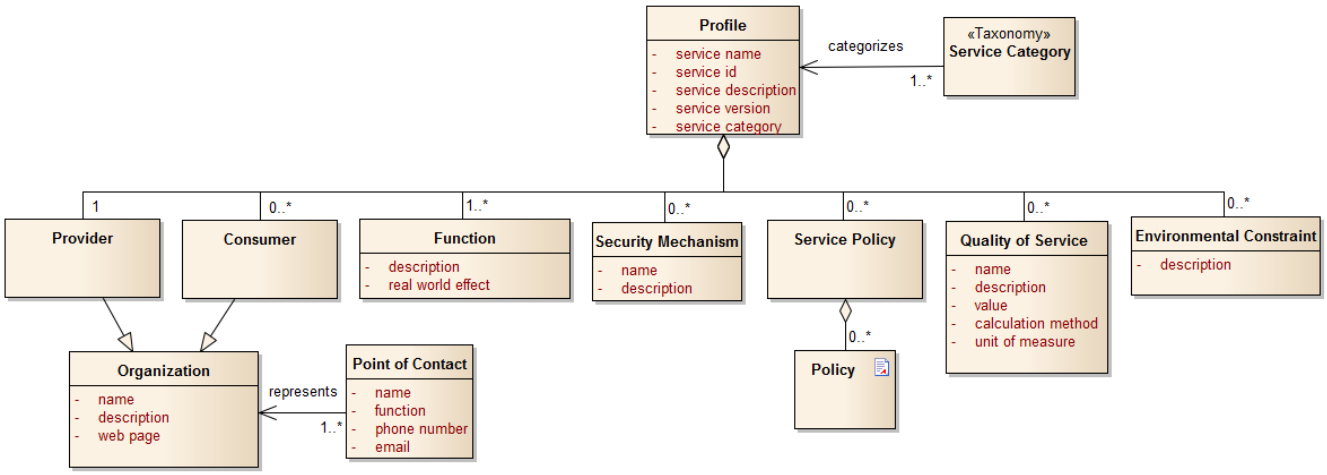

Figure 8. SDCM Service Profile

The SDCM Model describes to a service requester *how* to construct an invocation message and interpret a response message (img_sdcm_model). This part of the model is mostly technical and is used by a software agent to communicate with the service.

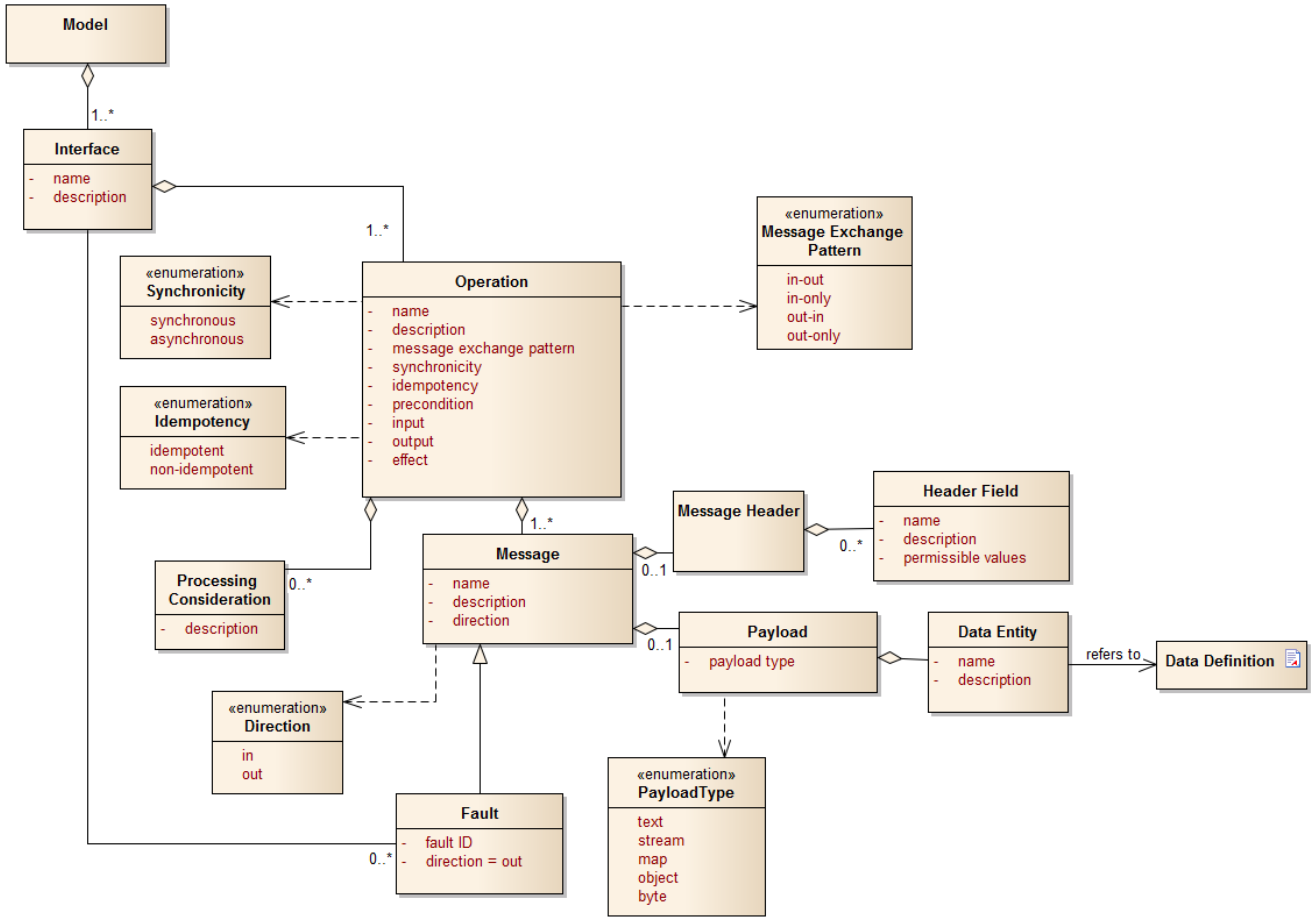

Figure 9. SDCM Model

The SDCM grounding describes the means by which the service is invoked, including the underlying technology protocols and network locations of the service (Figure 10). This part is very technical and is usually leveraged by software agents to communicate with the service.

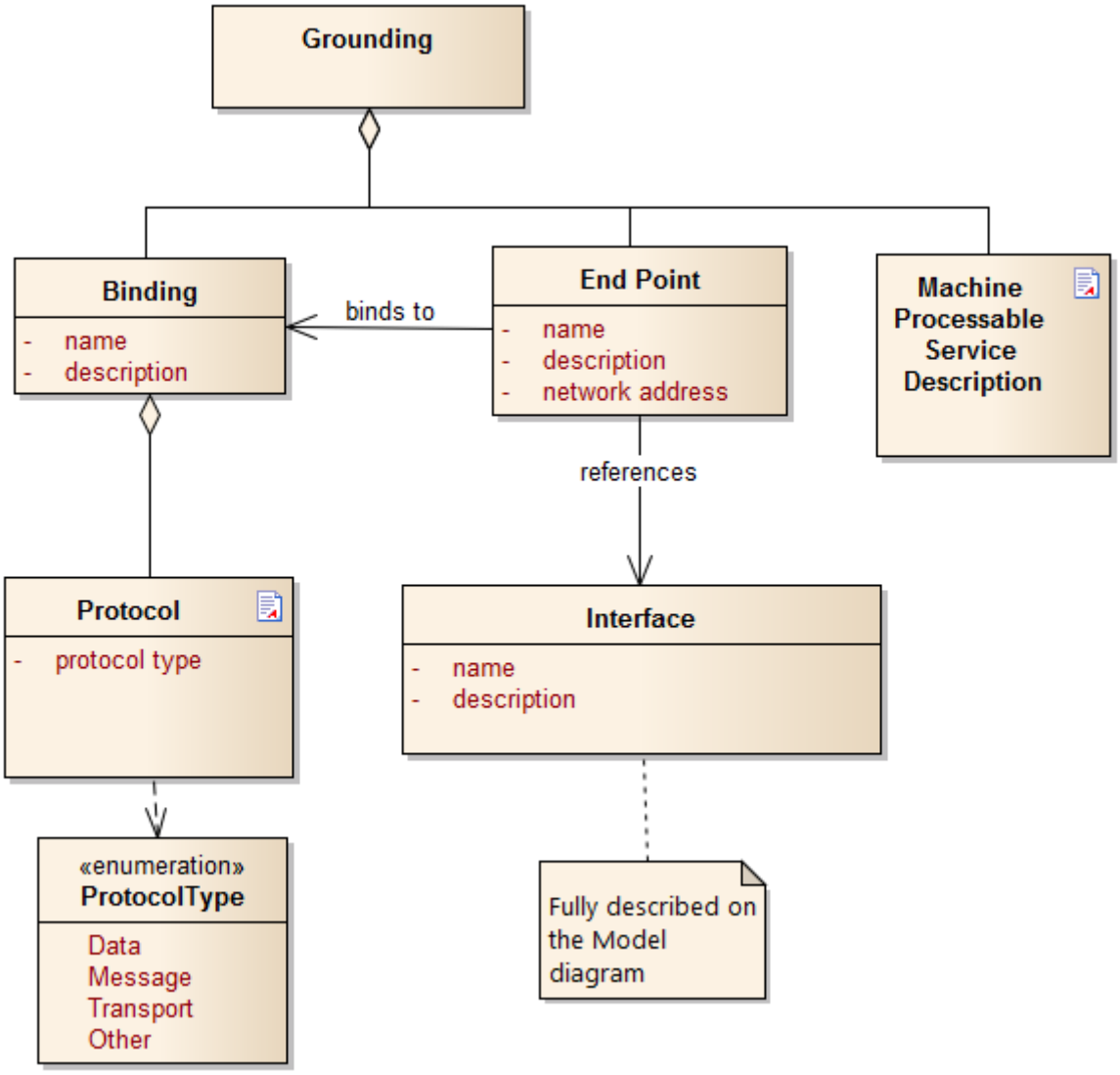

Figure 10. SDCM Grounding

6.2.2. Web Service Description Ontological Model (WSDOM)

The **Web Service Description Ontological Model (WSDOM)** [18] can be considered an "RDF" realization of the SDCM conceptual model. WSDOM standardizes information and metadata pertinent to describing SWIM services and facilitates interchange of service data between service providers and the FAA. The intent behind the ontology is to make service definitions clear, unambiguous, and discoverable by both humans and computer systems. WSDOM consists of ontology classes covering the key notions of service profile, service interface, service implementation, stakeholder, and document. WSDOM is patterned after the OWL-S semantic web services description ontology.

WSDOM was developed before SDCM and was written in OWL. For many people in the industry, OWL ontology was too technical to be understood by a large audience. To address this gap and make the service description more "readable", the Service Description Conceptual Model was created. That said, SDCM 2.0 is more recent than WSDOM 1.0, and therefore better reflects the current NSRR data structure. WSDOM 2.0 is currently being developed and not been aligned yet with the latest version of SDCM.

The WSDOM ontology was designed based on the OWL-S service ontology and considering the

service taxonomies used by FAA. The WSDOM 1.1 release consists of six OWL files and three RDF files. The three RDF files provide ontological representation of FAA standard taxonomies.

The structure follows the OWL-S model defining the Service ontology as being the top level ontology with three related classes: *ServiceProfile* (maps to OWL-S ServiceProfile), *ServiceInterface* (maps to OWL-S ServiceModel) and *ServiceImplementation* (maps to OWL-S ServiceGrounding) (Figure 11).

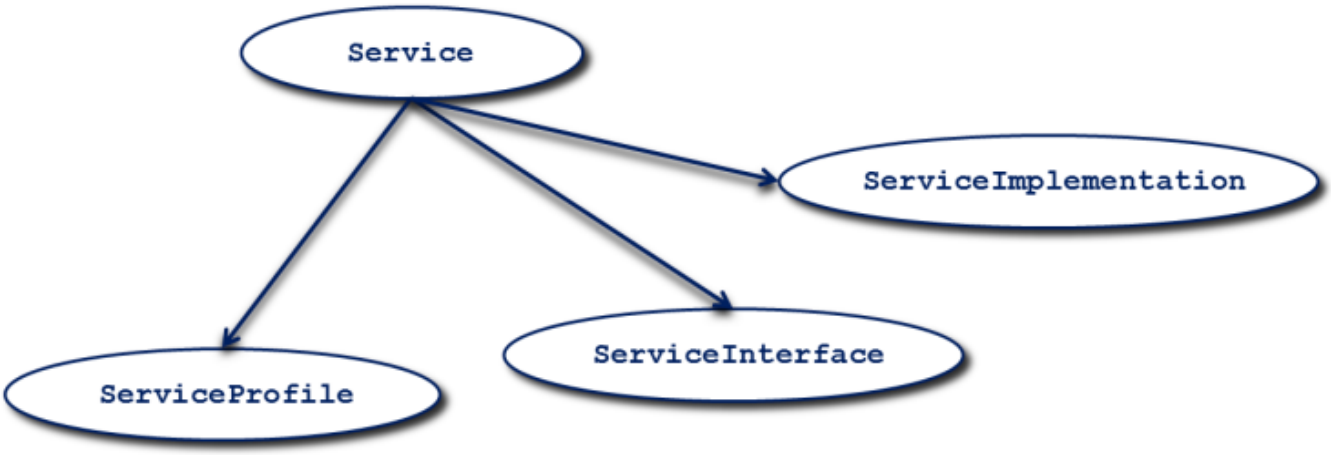

Figure 11. WSDOM Model

The *ServiceProfile* presents metadata about the service. This information is highly relevant for search of services and information. Figure 12 shows the information of the service profile that can be leveraged in a semantic registry.

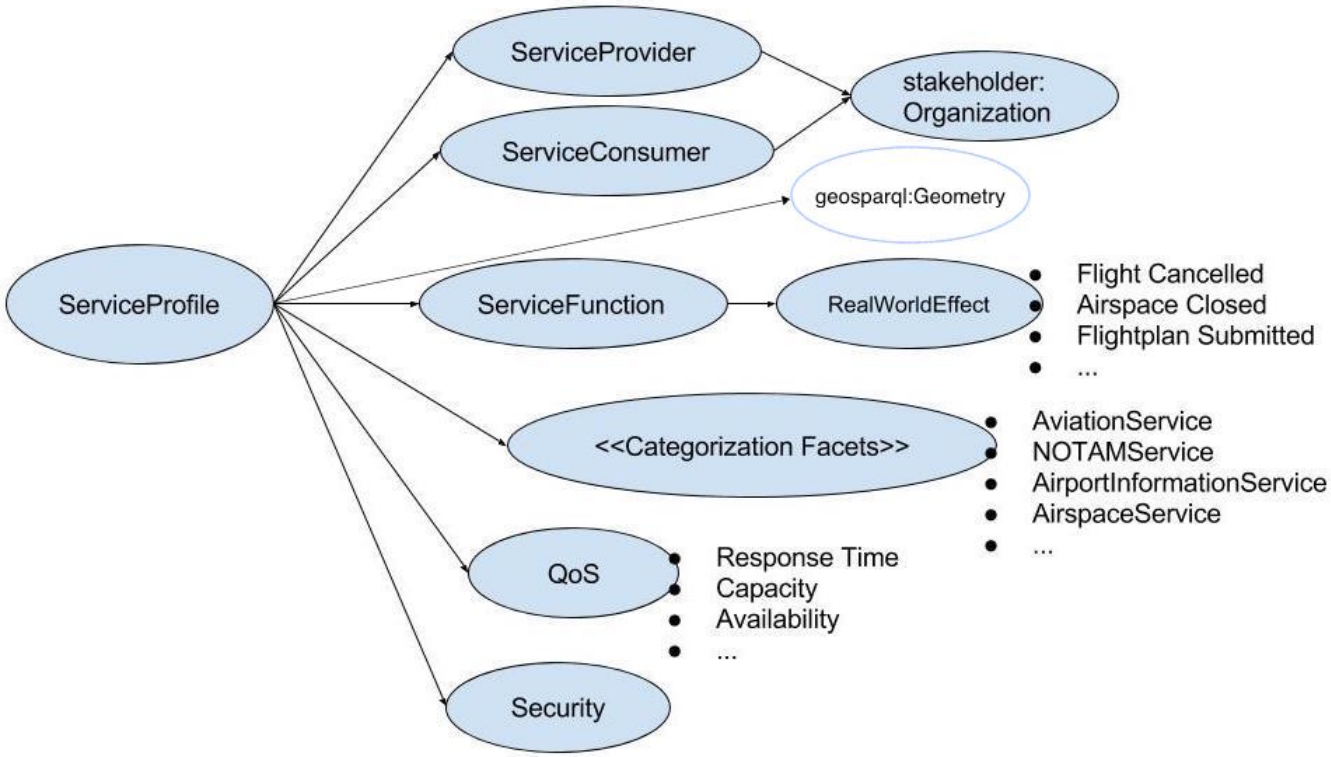

Figure 12. WSDOM Profile

ServiceInterface (maps to OWL-S ServiceModel) and ServiceImplementation (maps to OWL-S ServiceGrounding) describe how to access the service (its API and endpoints). This information is relevant when access to the service needs to be automated (Figure 13).

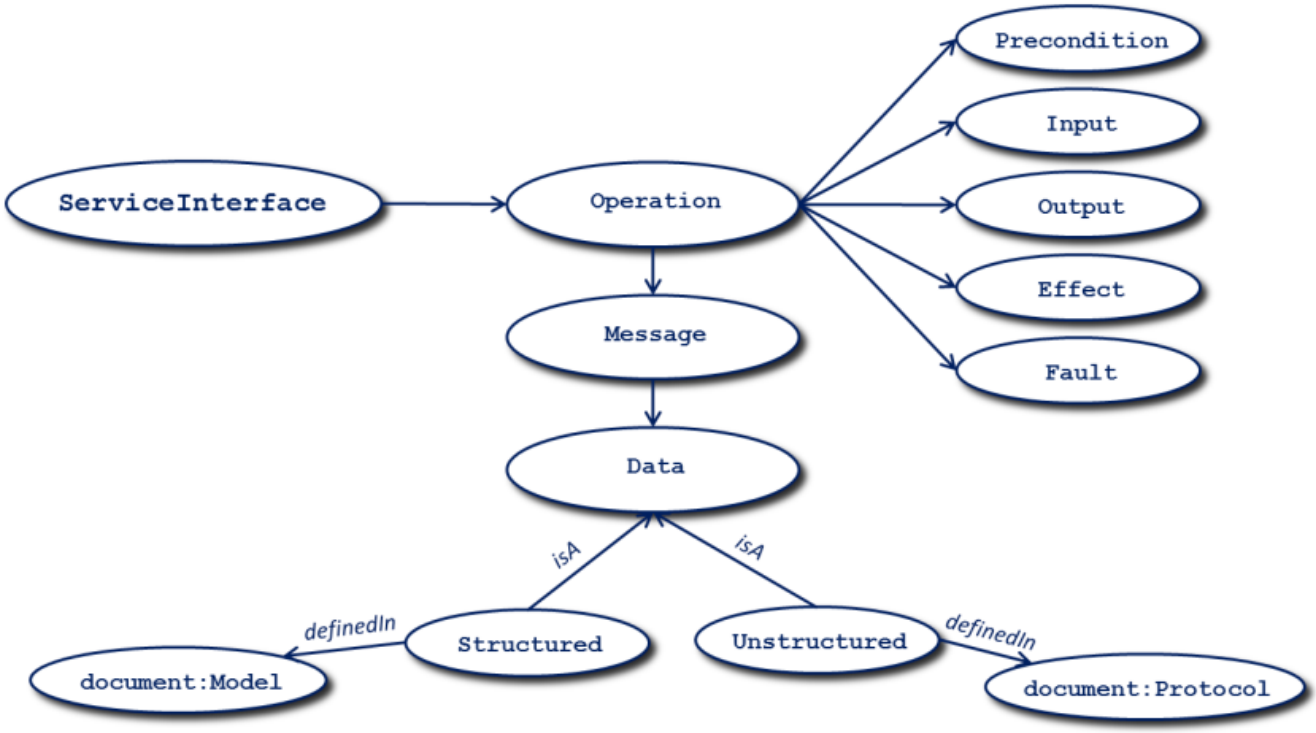

Figure 13. WSDOM Interface

6.2.3. SWIM Documentation Controlled Vocabulary (FAA)

FAA has developed a controlled vocabulary (CV), based on the Simple Knowledge Organization System (SKOS), that provides a single source for terms and definitions used in SWIM-related documentation [19]. SKOS (Simple Knowledge Organization System) is a W3C standardized RDF ontology for representing controlled vocabularies, thesauri, and taxonomies. The SWIM CV includes textual definitions of SWIM-related terms and their connections to broader, narrower, and otherwise related terms within the CV. SWIM documentation and web pages can reference term definitions in the CV using resolvable Universal Resource Identifiers (URIs).

Table 1 describes the list of taxonomies currently used by SWIM. These taxonomies were developed during previous OGC testbeds.

Table 1. Taxonomies currently used by SWIM

Name	Description
SWIM Service Product	A classification of services based on the type of SWIM data product that they deliver.
Service Availability Status	A classification of services based on their current, past, or future availability for provisioning.
Service Interface	Type A classification of services based on the type of technological solution that they deploy.
Flight Phase	A classification of services or other artifacts based on the flight phase, or period within a flight, during which the artifacts are used or provide support. See also [1].
US Airways	A classification of US Airways based on airway identification prefixes.

Name	Description
ICAO Region	A classification of geographical regions as defined by the International Civil Aviation Organization (ICAO).
US Flight Information Region	A classification of the US Flight Information Regions as defined by the International Civil Aviation Organization (ICAO).
Airspace Class	A classification of airspaces as defined and applied by the US Federal Aviation Administration (FAA).

The concepts of these taxonomies are encoded directly as skos:Concept, making it difficult to distinguish what category of concepts (ServiceProduct, ICAORegion, etc.) they denote. The next section addresses this gap in relation to controlled vocabularies (see also The role of Controlled Vocabularies).

Chapter 7. Semantic Enablement Approaches

This section introduces the term “Data-Centric Services” to generally describe classic data processing approaches, whereupon the focus is on creating, storing, sharing, and exploiting data. Within the National Airspace System (NAS), this is the world of FIXM, AIXM, WXXM, databases, and countless data capture, storage, discovery, navigation, visualization, analytics, and exchange services. By in large, in this world, aviation data is used to characterize air traffic information, weather information, and infrastructure information.

This section also introduces the term “Semantic-Enabled Services” (a.k.a. ontology-based services or knowledge-enabled services) to generally describe processing approaches that focus on creating, storing, sharing, and exploiting machine-encoded knowledge. This is the world of ontologies and formal taxonomies, graph databases, and ontology-based services. In this world, there are two basic approaches to knowledge-enabling the aviation enterprise:

1. Convert data to knowledge - structured data can be transformed into explicit knowledge, and then enriched to provide improved coherence and context.
2. Add a knowledge layer to data - a knowledge layer can be added to structured data, thus enriching the in situ data with improved semantic coherence and context, without the need of transforming 'physically' the data into knowledge (causing duplication of information). In both cases, knowledge enablement means the introduction of ontology, data-to-ontology mappings , and ontology-based services.

The National Airspace System (NAS), SWIM and indeed most IT modernization projects employ “Data-Centric Approaches” to manage and exploit information assets. Normally these efforts start by defining and standardizing logical data models, which in turn drive the implementation of dictionaries (what to call things), metadata (a means to discover things) and services (how to access and process things). Data-Centric Approaches tend to restrict research and development efforts to mostly syntactic and schematic considerations that are data-driven. These approaches focus on gaining community consensus about well-defined entities, taxonomies, properties, and relations, towards the goal of establishing common schemata and protocols for interoperability. Resulting implementations are limited by their syntactic and schematic boundaries. Further, resulting implementations fall short of providing sufficient semantics (explicit meaning of concepts) and context (relevant information that clarifies the meaning, significance and relevance of concepts and concept patterns, i.e., supplementary information that places concepts in a useful context).

These circumstances can be improved by taking a Semantic-Enabled Approach, which adds a machine-encoded, semantics-based knowledge layer to NAS data and services. Semantics are required to take aviation domain interoperability to the next level. Semantics are also key to enabling computer-assisted autonomous operations. A Knowledge-Enabled Approach captures and exploits semantics and associated context, as machine-encoded knowledge, thus contributing to a rich, conceptual model that builds upon SWIM logical data model foundations. This approach will help ensure that NSG objects and entities are properly interpreted and employed, in a useful mission context.

Again, Semantic-Enabled Approaches add a machine-encoded knowledge layer to existing

environments. This layer captures the conceptual meanings, significance, relevance and relationships of all relevant information required for use in a target mission domain. The layer also encodes business rules that are used by enhanced reasoning services to draw inferences and make sense of mission concepts. Finally, the knowledge layer integrates with an array of analytical and reasoning services to enhance information triage, fusion, situation awareness, and sense making. The layer facilitates a higher level of utility, automation, interoperability, extensibility and flexibility, beyond that which is possible with data-centric approaches. It also dramatically reduces the cognitive burden on users.

Some notable advantages of this approach are:

1. The knowledge layer provides a more user friendly and mission-relevant unified conceptual view of all pertinent data and services for a target mission domain. Users are able to interact with familiar, well-understood concepts, rather than deal at the level of fixed data schemata with its commensurate semantic and contextual ambiguities. This allows users to obtain relevant actionable information more readily, which in turn leads to making better decisions more quickly. User productivity and effectiveness increases with the reduction in cognitive burden.
2. There is substantially more flexibility in accommodating data evolution, variety and veracity. It is easier to add new data sets, make changes to existing data sets, and accommodate data set differences. Changes in information requirements will first affect the “knowledge layer”, reducing effects on databases and software, as only the mapping of the existing data to knowledge representation needs to be changed.
3. Business rules are encoded on top of the knowledge layer, where they can be more easily changed without changing business applications. (Current approaches require that business logic must be wired into business applications… a costly, inflexible approach.)
4. The vastly improved filtering, fusion, analytics and reasoning services provide an enhanced information triage mechanism for quickly finding “the needle in the haystack”, locating patterns across a complex array of data, and making inferences that are not directly exposed in data (i.e., “determine the haystack from the needles”). User productivity and effectiveness increases with the increased level of automation.

Table 2 presents the main issues addressed by a Semantic-Enabled Approach, along with its benefits.

Data-Centric Issues with Corresponding Semantic-Enabled Approach and Benefits

Issues with Data-Centric Approaches	Semantic-Enabled Approach	Value Proposition
Data model standardization relies upon homogeneous data description and organization. *This imposes strict adherence to a standard that is defined at the syntactic-schematic level, whereupon it's harder to achieve consensus and less flexible. Modelers struggle between producing simpler models for which it is easier to gain consensus, but harder to achieve desired business reality, versus those seeking richer models that are closer to reality but have unwanted complexity.*	• A semantic-enabled approach employs a standards-based formal, sharable framework that provides a conceptual domain model to accommodate various business needs. Decentralized model extensions can be accommodated without adversely affecting existing information infrastructure.	• Allows decentralized extensions of the domain model • Accommodates heterogeneous implementations of the domain model (lessens impact on systems; reduces cost) • Shareable machine-processable model and business rules; reduces required code base
Data-centric approaches increase the chance for multiple interpretations and misinterpretations of data. *Data interpretation requires knowledge of its semantics (e.g., meanings, significance, and relevance) and surrounding context. Data-centric approaches are unable to capture these semantics and context, which are in turn required for automated fusion, analytics, and reasoning.*	• Semantic-enabled approaches encode data characteristics in ontology. By formalizing the semantic and business rules unambiguously in a declarative ontology, software can use off-the-shelf semantic components to interpret, infer and validate domain data, thus reducing interpretation errors.	• Increased software maintainability • Improved data interpretation and utility • Actionable information for the decision maker

Issues with Data-Centric Approaches	Semantic-Enabled Approach	Value Proposition
Data model implementations have limited support for business rules, and lack expressiveness. *Data-centric implementations encode business rules using software or database programming languages. Additional programming is necessary to apply business rules when using the data. Robust conceptual and contextual meanings of information may not be captured in the model. The risk is high for inconsistent conceptual encoding and interpretation in each implemented system.*	• Standards-based knowledge encoding (OWL, SPARQL Rules) captures formal conceptual models and business rules, providing explicit, unambiguous meanings for use in automated systems. With richer semantic and contextual expressiveness, automated systems are less complex to design and develop. Proper interpretation and use is more consistent across enterprise systems.	• Reduction of software and associated development cost • Conceptual models and rules that provide enhanced meaning, thus reducing the burden on users • Unambiguous interpretation of domain model; greater consistency in use
Data-centric implementations are inflexible when data requirements change. *Whenever business rules and semantic meaning are encoded in a programming language, changes impact the full development life cycle for software and data. When the change includes a conceptual change (new/enhanced business concept), the full standardization process must also be executed.*	• The semantic-enabled approach uses an ontology that contains a flexible, versatile conceptual model that can better accommodate the requirements of each stakeholder in the business domain. Changes or extensions are integrated and implemented by enhancing the domain ontology and updating the semantic mapping to the data store. Older concepts and heterogeneity can still be supported by using different mappings.	• Increased flexibility to accommodate stakeholder needs; Decentralized and organic evolution of the domain model • Changes only impact affected stakeholders, not others; reduces software updates • Software adapts to domain model as ontology evolves • The enterprise can better keep up with changing environment/requirements

Issues with Data-Centric Approaches	Semantic-Enabled Approach	Value Proposition
Data-centric approaches require additional software development (tools) to integrate across several domains. *The heterogeneity of data schemas and business models in alien domains requires additional software development (or tools) to integrate and make sense of the alien data in the local domain.*	• Standards-based knowledge technologies provide the means to exchange knowledge, schemas/rules and query knowledge. Knowledge representations are non-disruptively layered on top of existing information assets. They **enhance** rather than displace existing information resources. • Using semantic-enabled technologies, different data sources can be accessed in their native form, and linked-fused-reasoned about on-the-fly for more creative uses.	• Leverages and enhances "As-Is" enterprise; preserves investments • Open standards enhance interoperability (the knowledge layer is the "last rung in the interoperability ladder") • Enables use of diverse Web and enterprise data sources for full business context; enhances business applications • Easy to work across different domains and answer more challenging, relevant business questions
Data-centric approaches require that data inferencing and validation rules are encoded in software, or delegated to human-intensive validation processes. *Reliable data that is essential for critical systems, inferencing, and effective decision support, requires rules that support inferencing and validation.*	• Semantic-enabled approaches use a formal language (OWL) that provides well-defined semantics in a form compliant with off-the-shelf software that automates data inferencing and validation. • Semantic-enabled approaches can accommodate situations where information may be missing or incomplete.	• Employs off-the-shelf software for inferencing and validation • Reduction of validation and testing in the development process • Uses all available data from sources, including inferences, while accommodating cases of missing/incomplete information

Issues with Data-Centric Approaches	Semantic-Enabled Approach	Value Proposition
Data-centric approaches presume a priori knowledge of data utility. *Semantics are pre-wired into applications based upon data verbosity, conditions and constraints. Changes in data directly impact code.*	• Encoding the conceptual model and rules explicitly using OWL enables rapid integration of new/changed data. Software accesses data through the "knowledge layer" where it's easier to accommodate changes without rewriting software.	• Reduced software maintenance owing to emergent data perturbations • Software quickly adapts to evolving domain model • New information are readily introduced and understood in their broader domain context

The next sections distinguish two levels of semantic enablement:

- Metadata level semantic enablement to support search and discovery of information
- Instance level semantic enablement to support data integration and reasoning.

Chapter 8. Metadata level semantic enablement

Semantic metadata plays a central role in facilitating the search, discovery and the assessment of geospatial assets (such as datasets, services, portrayal information, schemas, maps, layers), and the integration of these assets in a specific mission or domain. There are a number of standards, formats and APIs that provide the metadata for these assets, but in order to perform efficient search providing relevant results, there is a need to convert this information into a **unified machine readable semantic representation**. It is this conversion that enables the discovery of relevant resources that satisfy the requirement criteria of the end user. As the computing community increases its understanding of the kind of metadata information needed to perform better and smarter search, the National Airspace System (NAS) needs a model that accommodates extensions over time without breaking the proposed architecture.

The current SWIM architecture is mostly focused on the description of Services. While the model is based on OWL-S ontology, it is encoded in XML and validated with XML Schema, which reduces the benefits of using ontology, reasoning and integration by linking to other domains and vocabularies. Previous testbeds integrated controlled vocabularies in the XML representation but cannot be leveraged by off-the-shelf semantic tools (reasoner, RDF API) to perform inference and linking information between systems. Data are still living in silos and are confined in the scope of XML documents.

> **RECOMMENDATION**: Define an alternative representation of Service description in the service registry using Linked Data formats such as RDF/XML, Turtle, and JavaScript Object Notation for Linked Data (JSON-LD), so as to facilitate the integration of controlled vocabularies described in SKOS and OWL, linkage to other resources such as Dataset descriptions and reasoning. If possible, make the URL of these resources resolvable through a **REST API**.

While the SWIM Registry [9] has done an excellent job at describing services [7],[20], they do not address the needs of end-users, who are looking for information that answers their questions: Where can I find the flight plan for a given flight? What are the departures time for a given flight in a given airport? Are there any air show events in my area? When issuing a search, the end-user is not focused on the details of "how" to access the information (which service metadata provides) and "how the data are encoded", but more about finding the "relevant" information that fits their needs (topic, spatial, temporal coverages, quality information). At present, the description of information is barely addressed in the service description as it is mostly limited to tags and some controlled vocabulary terms, making the search of relevant information very difficult. The current service descriptions are not able to answer questions such as: what are the spatial, temporal, topical coverages and quality information of the datasets that services operate on? Typically, end-users are looking for information and once they discover the information, they want to know how to access it, i.e. what are the distributions available (download, service API, web site).

Data and services are the Ying and Yang of Information Systems. The balance needs to be restored in the current SWIM infrastructure by providing metadata about the information, on which the different services operates on.

RECOMMENDATION: Restore the balance of metadata description by providing more metadata about datasets/information

8.1. Issues with existing metadata standards

To discover the information needed, enough metadata needs to be given to assess the relevance of the information. There are number of metadata standards that are used to describe geospatial datasets. The most popular ones are ISO 19115 and ISO 19139. Regionally, popular ones include and the Content Standard for Digital Geospatial Metadata (CSGDM) in the United States and the metadata specification of the Infrastructure for Spatial Information in the European Community (INSPIRE) in Europe. However, these standards are defined in XML using XML Schema for validation. While these standards help to provide metadata about datasets and services, they are still falling short into helping the user to find the relevant information because they are document centric and do not enforce proper use of semantic information. They fail to properly leverage the use of controlled vocabularies using well-grounded terms. The following issues relevant to the controlled vocabularies in these standards have been identified during Testbed-12 [OGC 16-059] and are summarized below. The goal of enumerating these issues is to avoid the NAS falling in the same interoperability traps that other systems using these standards fell in.

8.1.1. Identification of Resources

Issue	Identification of Resources
Description	There is no consistent way of defining the identifiers for different resources (e.g. organizations, datasets, services, controlled vocabularies, etc.)
Why it is a problem?	Inability to link information and allow reusability. Resource information (concepts) are duplicated several times in different documents with variations of the same information. Updating this information is difficult to perform across all repositories. Need authoritative unambiguous references.
Recommendations	• Each resource should use a unique URI that is resolvable. • A policy needs to be put in place to manage the URI schemes of different types of resources. • The maintenance of the information for each resolvable URI should be decentralized to the authoritative party for the resource.
Benefits	A new policy to define URI Sets for NAS assets would provide a consistent mean to make these trusted assets available for efficient, widespread discovery and re-use. This will encourage reuse and limit duplication.

8.1.2. Resolvable URI

Issue	Resolvable URI
Description	Identifiers used in the metadata document (ex. ISO 19139) are often internal (e.g., a primary key in a store implementation) and not accessible as unambiguous web resources.
Why it is a problem?	The lack of consistent machine-resolvable URIs impedes interoperability and limits automation (concepts must be grounded with unambiguous meaning for services to interpret and respond). Grounded URIs will also help humans better understand important concepts.
Recommendations	• Make links resolvable and semantically-grounded URIs with the right information to support human and machine exploitation (for controlled vocabularies, licenses, organizations, etc.) • Make the information accessible for both human consumption (HTML) and machine-understanding (Linked Data).
Benefits	Enables the exploration of a “unified knowledge graph” that links and describes resources. Allows users to search, discover and navigate through “Concept Space”, whereupon each concept is resolvable to a grounded (unambiguous) resource for consistent human and machine understanding.

8.1.3. Multilingual Support

Issue	Multilingual Support
Description	Many existing standards (e.g. ISO 19115, CSDGM) do not enable the support of translations of human readable text in multiple languages. Language is handled at document level, not field level.
Why it is a problem?	Users who do not understand the language of the information producer will not be able to discover relevant data for their tasks. Air staff and travelers are typically from different countries.
Recommendations	Opt for an implementation that natively provides multilingual support (such as Linked data) or provide guidelines for how to handle multiple languages for example through JavaScript Object Notation (JSON) protocols.

8.1.4. External Resource Descriptions

Issue	External Resource Descriptions
Description	• A number of properties refer to external resources (homepage, landing page, online resource for contact, page about document, reference to metadata document). • Standards such as POD model these resources using a simple URL assigned to a property. This prevents for adding additional properties such as title, description, format or role of the document that helps the user to understand the meaning of the URL
Why it is a problem?	External resources modeled as a URL value inhibits the capture of additional information to help the role and meaning of the external (auxiliary) resource in the context of a given resource
Recommendations	• Model external resources as objects when their role is ambiguous. • If the property referring to a resource URL is unambiguous (homepage), use the URL directly.

8.1.5. Controlled Vocabulary Management

Issue	Controlled Vocabulary Management
Description	• Controlled vocabularies are not made publicly available or are not resolvable (where is the National Map Theme Thesaurus?) • Lack unique identifier for controlled vocabulary (e.g., GCMD, Global Change Master Directory) • Lack unique identifier for keyword concepts (e.g., Paris, France vs. Paris, TX) • Duplication of concepts (keywords) from different taxonomies, e.g., National Map Theme Thesaurus contains “Elevation” and NGDA Portfolio Theme refers to it as “Elevation Theme”. Are they the same concept and meaning?. • Tendency to use alternative spellings for the same concept (e.g., US and United States)
Why it is a problem?	• Can’t perform semantic search • Lack consistent use of concepts (keywords) across metadata documents • Ambiguity in the meaning of concepts (lack of grounded concepts)

Issue	Controlled Vocabulary Management
Recommendations	• Define concepts in SKOS encoding with unique identifiers that are resolvable • Group alternate labels or translations under the same concept • Provide SKOS mappings to other vocabularies to enable semantic search across taxonomies • Make controlled vocabularies publicly available and uniquely identified with a resolvable URL.
Benefits	• Allows reusability of controlled vocabularies • Less verbose document • Unambiguous interpretation of key concepts • Inference enabled by using standard SKOS semantics (semantic search) • Enables Multilingual search by concept

8.1.6. Keywords Types

Issue	Keyword Types
Description	The list of keyword types in many standards is limited to a few categories (e.g. discipline, strata, topic, place, temporal in ISO 19115).
Why it is a problem?	Inability to accommodate new types of concepts such as audience, function, subject, topic, etc..
Recommendations	• Provide a mechanism to extend the list of keyword types in standards using SKOS controlled vocabularies • Define the keyword types in a controlled vocabulary to make them uniquely identifiable and resolvable • Refer to the keyword type by resolvable URL
Benefits	• Provides an extensibility mechanism to accommodate other types of concepts (Audience, Function, Purpose, etc.). • Allows reusability of keyword types

8.1.7. Keyword Labeling Inconsistencies

Issue	Keyword Labeling Inconsistencies
Description	In some instances, multiple labels are encoded as one keyword (e.g., 'list of all US states' is one keyword).

Issue	Keyword Labeling Inconsistencies
Why it is a problem?	While this is fine for doing lexical-based text search, it is not sufficient when supporting semantic search, where each concept must be grounded to a unique meaning.
Recommendations	• Each keyword should refer to one concept only • In addition to a label, use a URI to refer to a concept
Benefits	• Less verbose document • Enables inference by using standard SKOS semantics

8.1.8. Authority for Controlled Vocabularies

Issue	Authority for Controlled Vocabularies
Description	The ISO 19139 standard uses the list of topic categories in the standard ISO 19115. There is a SKOS encoding available in the European Registry located at: http://inspire.ec.europa.eu/metadata-codelist/TopicCategory. The mapping to Semantic Registry uses this URI to reference dcat:theme.
Why it is a problem?	If no authority is responsible of the management of controlled vocabularies, the vocabularies will not be reused and risk to be duplicated.
Recommendations	• There is a need for a registry of controlled vocabularies that are reusable across agencies. • FAA could host controlled vocabularies encoded in SKOS.
Benefits	The taxonomy is maintained by the authority that defines the standard and thus will favor reusability of the vocabularies among information producers.

Without addressing these issues in the metadata standards, interoperability at the semantic level will not be achieved. Fortunately, there are ongoing efforts using Linked Data standards to semantically describe metadata about datasets, that are addressing these gaps (DCAT, Project Open Data, GeoDCAT-AP and OGC SRIM). The US Government has enforced the use of Project Open Data for all US agencies and the European Union has enforced the use of DCAT to describe metadata about datasets used by the different EU members. The policies have proven to be successful and have been adopted rapidly by the different agencies. For example, the European Data Portal is implementing the DCAT-AP as the common vocabulary for harmonizing descriptions of over 258,000 datasets harvested from 67 data portals of 34 countries.

RECOMMENDATION: Use Linked Data standards to describe metadata of the NAS Assets

8.2. Relevant ontologies

This section reviews relevant ontologies to encode metadata for NAS assets.

8.2.1. DCAT

DCAT is an RDF vocabulary designed to facilitate interoperability between data catalogs published on the Web. By using DCAT to describe datasets in data catalogs, publishers increase discoverability and enable their applications to easily consume metadata from multiple catalogs. It further enables decentralized publishing of catalogs and facilitates federated dataset search across sites. Aggregated DCAT metadata can serve as a manifest file to facilitate digital preservation. Figure 14 illustrates defines core DCAT model.

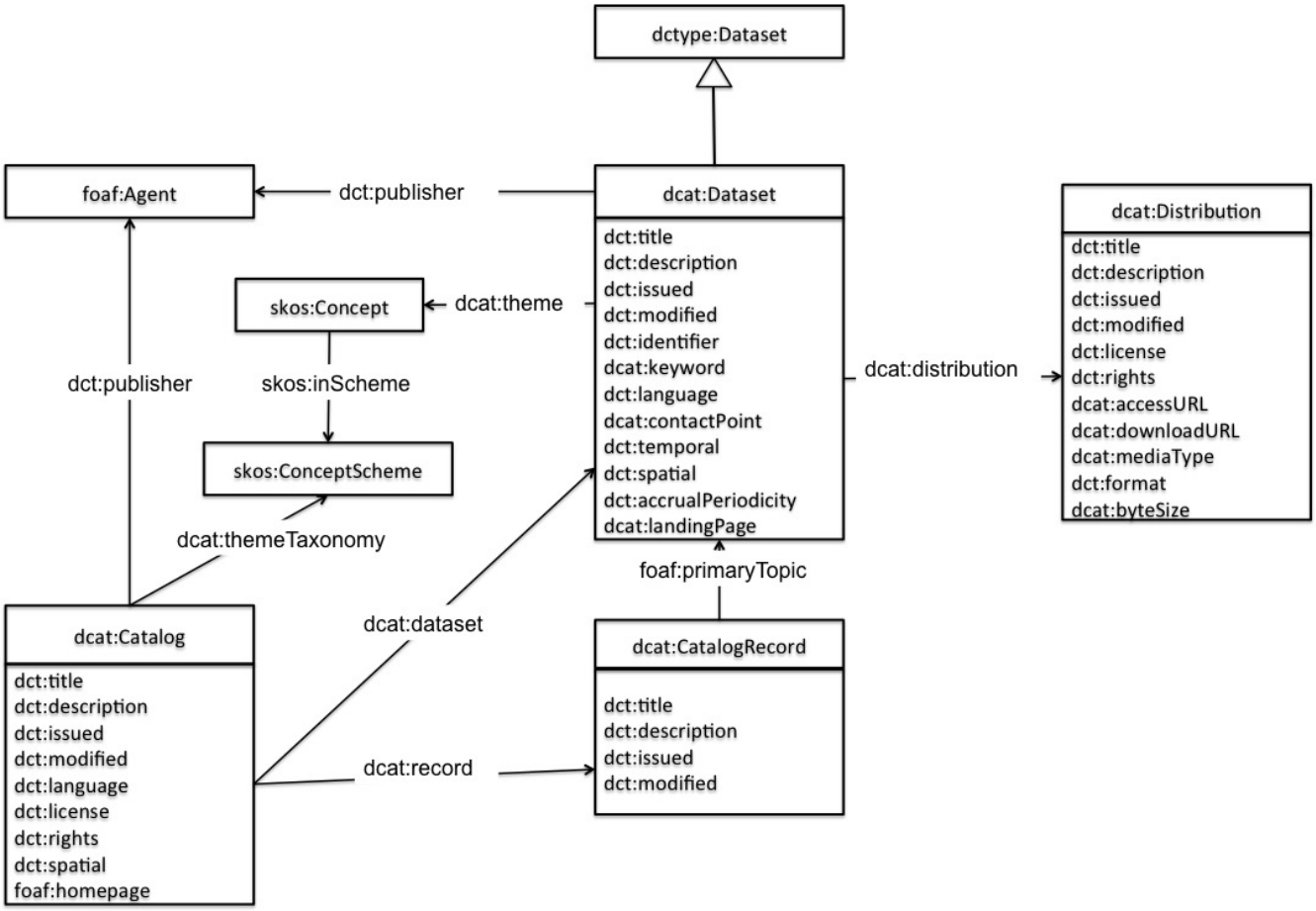

Figure 14. DCAT Model

DCAT is highly relevant for NAS as it provides a set of core classes and properties to describe datasets. To accommodate the specificities for specific domains, application profiles for DCAT have been emerging over time (DCAT-AP,GeoDCAT-AP, StatDCAT-AP, DCAT-AP-NO,DCAT-AP-IT,DCAT-AP.de,TransportDCAT-AP, EPOS-DCAT-AP,POD). Typically, application profiles for DCAT are defined by reusing existing classes and properties from different ontologies and assigning them an obligation level defined as mandatory, recommended or optional. The following subsections document the application profiles that are the most relevant for the NAS.

NOTE	At time of this writing, there is a W3C Working group working on a new revision of DCAT (1.1) that better address some of the gaps identified over time. It will address better the relationships between service and datasets. This work has a lot of overlap with the OGC SRIM Model and reconciliation between both models will need to occur once DCAT 1.1 becomes an official W3C standard. The new specification also attempts to define best practices for managing DCAT application profiles.

8.2.2. DCAT-AP

The **DCAT Application profile for data portals in Europe** (DCAT-AP) is a specification based on the Data Catalogue vocabulary (DCAT [http://www.w3.org/TR/vocab-dcat/]) for describing public sector datasets in Europe. Its basic use case is to enable cross-data portal search for data sets and to allow public sector data to be easily searchable across borders and sectors. This can be achieved by the exchange of descriptions of datasets among data portals.

In February 2015, the ISA² programme [http://ec.europa.eu/isa/isa2/index_en.htm] of the European Commission has started an activity to revise the DCAT-AP, based on experience gained since its development in 2013. The outcome of this effort was the publication of DCAT-AP 1.1.

The European Data Portal [http://europeandataportal.eu/] is implementing the DCAT-AP as the common vocabulary for harmonizing descriptions of over 258,000 datasets harvested from 67 data portals from 34 countries. The DCAT-AP is used in the Open Data Support [https://joinup.ec.europa.eu/community/ods/description] service initiated by the European Commission with the purpose of realizing the vision of European data portals.

8.2.3. Asset Description Metadata Schema (ADMS)

ADMS is a profile of DCAT that is used to describe *semantic assets* (or just 'Assets'). These assets are defined as highly reusable metadata (e.g. xml schemata, generic data models) and reference data (e.g. code lists, taxonomies, dictionaries, vocabularies) that are used for eGovernment system development.

The ADMS model is intended to facilitate federation and co-operation. Like DCAT, ADMS has the concepts of a repository (catalog), assets within the repository that are often conceptual in nature, and accessible realizations of those assets, known as distributions. An asset may have zero or multiple distributions. As an example, a W3C namespace document can be considered to be a Semantic Asset that is typically available in multiple distributions, one or more machine processable versions and one in HTML for human consumption. An asset without any distributions is effectively a concept with no tangible realization, such as a planned output of a working group that has not yet been drafted.

ADMS is an RDF vocabulary with an RDF schema available at its namespace http://www.w3.org/ns/adms [http://www.w3.org/ns/adms] . The original ADMS specification published by the European Commission [ADMS1 [https://www.w3.org/TR/vocab-adms/#bib-ADMS1]] includes an XML schema that also defines all of the controlled vocabularies and cardinality constraints associated with the original document. The model is shown in Figure 15.

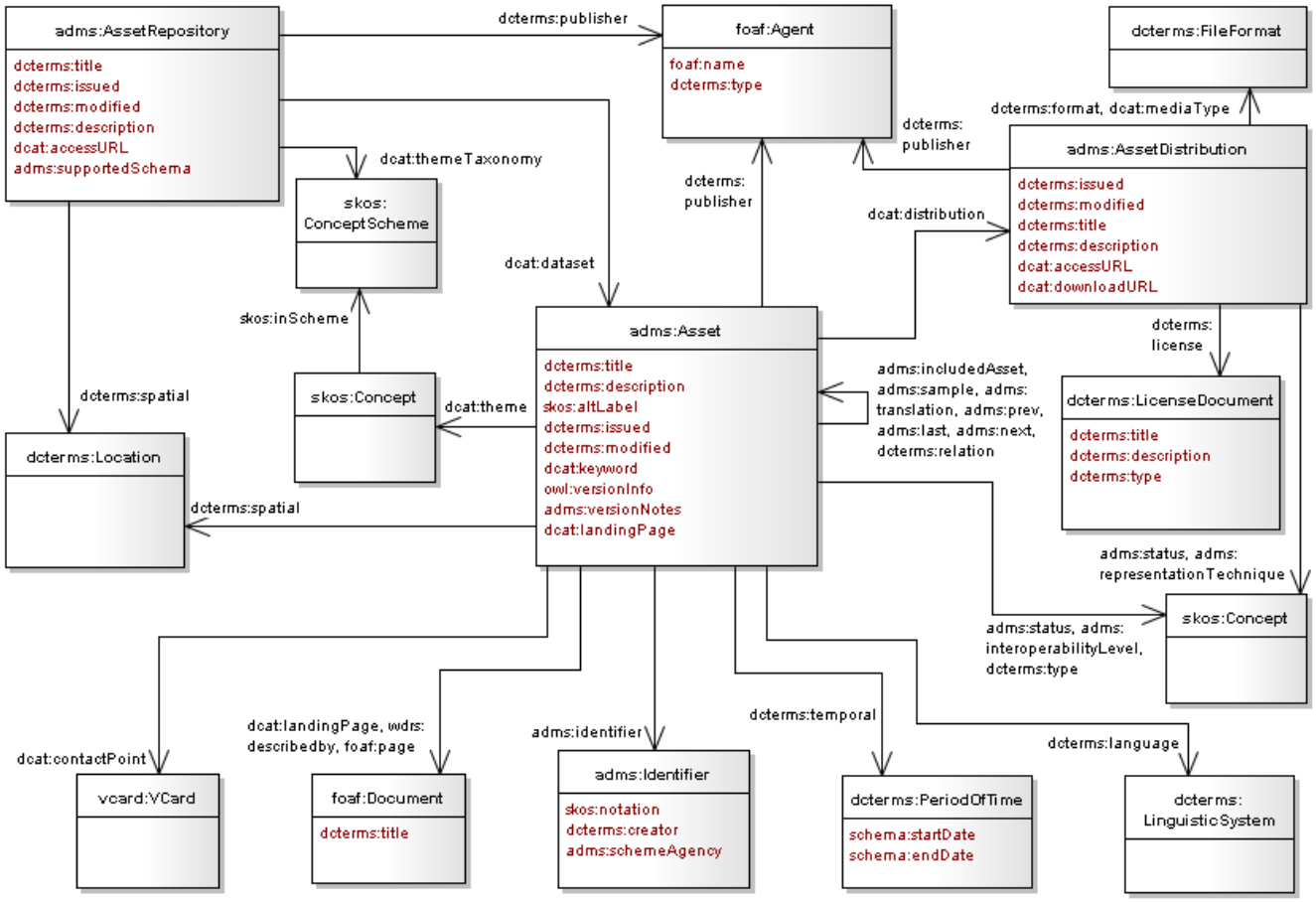

Figure 15. ADMS Model

This model is relevant for the SWIM architecture as the current registry contains information about schemas and service definitions (such as WSDL document). ADMS provides a set of terms that could be used to better describe the metadata information about these type of assets.

8.2.4. Project Open Data (POD)

Project Open Data [21] provides the implementation guide and associated resources for the Federal Executive Order on open data and data management, M-13-13 [https://project-open-data.cio.gov/policy-memo/] "Managing Information as an Asset," which includes the standardized metadata schema that all CFO Act agencies are required to use to publish their enterprise data inventories.

The Project Open Data Metadata Schema [https://project-open-data.cio.gov/v1.1/schema/] is a JSON-based implementation of the W3C DCAT vocabulary [http://www.w3.org/TR/vocab-dcat/]. This standard is currently implemented by multiple data catalog platforms as well as state and local governments. POD is based on the DCAT model but it enforces the encoding in JSON-LD. POD defines a JSON-LD context to map the JSON document back to Linked data representation.

The POD Model is shown in Figure 16

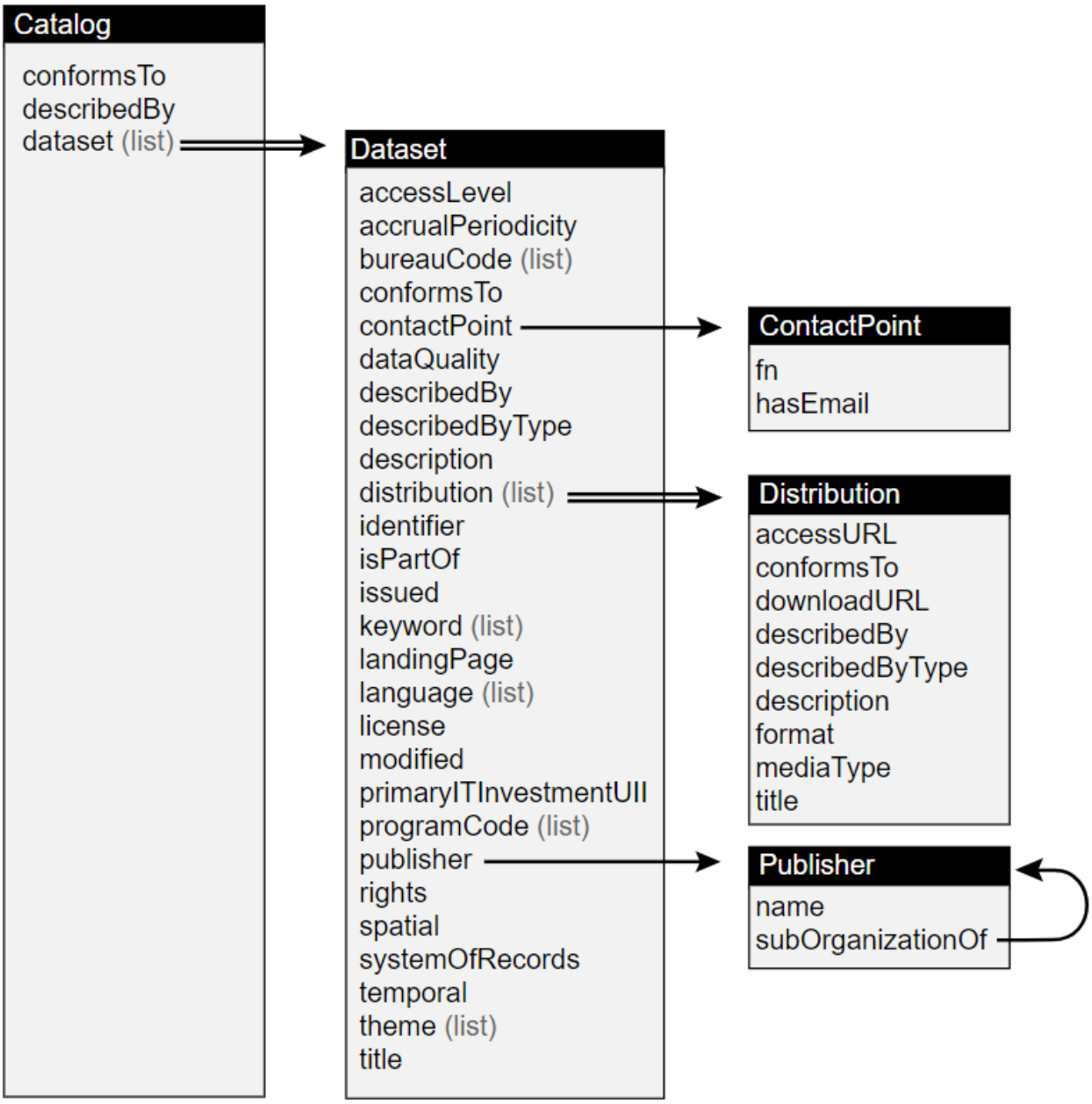

Figure 16. Project Open Data (POD) Model

Typically, POD documents are published in a Web Accessible Folder (WOF) and harvested by catalogs such as data.gov. The intent of POD is to lower the bar of complexity needed to represent data information by providing guidelines and recommended metadata. This improves search and discovery for datasets within the US government.

Below is an example of a POD dataset published by FAA on data.gov (which can be found at: https://catalog.data.gov/dataset/flight-schedule-monitor)

Sample POD Dataset published by FAA


```
{
   "@type":"dcat:Dataset",
   "QualityResponsibleAgency":"Federal Aviation Administration",
   "accessLevel":"non-public",
   "accrualPeriodicity":"R/P1D",
```

```
   "additionalMetadata":"N/A",
   "agencyDataSeriesURL":null,
   "agencyProgramURL":null,
   "analysisUnit":null,
   "bibliographicCitation":null,
   "bureauCode":[
      "021:12"
   ],
   "category":"Raw data",
   "categoryDesignation":"Research",
   "collectionInstrument":null,
   "collectionMode":"Electronic",
   "comments":null,
   "confidentiality":false,
   "contactPoint":{
      "@type":"vcard:Contact",
      "fn":"Mojdeh Supola",
      "hasEmail":"mailto:Mojdeh.Supola@faa.gov"
   },
   "dataQuality":false,
   "describedBy":null,
   "description":"The Flight Schedule Monitor (FSM) is the main tool for the traffic
management specialist at the FAA David J. Hurley Air Traffic Control System Command
Center (ATCSCC) to monitor, model, and implement Ground Delay Program (GDP)
operations.  FAA and airlines use FSM to monitor demand through receipt of FSM demand
pictures of airports updated every 5 minutes.  FSM constructs  if scenarios for best
options (i.e., best parameters) prior to making a GDP decision.  Modeling may be used
by: (1) the ARTCC Traffic Management Coordinator (TMC) to request ATCSCC
implementation of a GDP in the event of significant congestion or if a demand
imbalance is projected at an en route fix, route, or sector; (2) the ATCSCC to
determine Air Route Traffic Control Center (ARTCC) start times, Airport Arrival Rate
(AAR), and other parameters for a particular GDP scenario; and (3) the Airlines to see
the effects of canceling or delaying a specific flights under a GDP.",
   "dictionaryList":null,
   "guidelineCompliance":false,
   "identifier":"1736.0",
   "issued":"2014-11-21",
   "keyword":[
      "flight",
      "traffic",
      "route",
      "airport",
      "weather",
      "airspace",
      "surface"
   ],
   "landingPage":null,
   "language":[
      "en-US"
   ],
   "license":"https://project-open-data.cio.gov/unknown-license/",
```

```
    "modified":"2015-02-09",
    "numberOfDatasets":1,
    "organizationId":null,
    "organizationName":"Department of Transportation / Federal Aviation
 Administration",
    "phone":"202-267-1026",
    "primaryITInvestmentUII":null,
    "programCode":[
       "021:001"
    ],
    "publisher":{
       "@type":"org:Organization",
       "name":"Federal Aviation Administration",
       "subOrganizationOf":{
          "@type":"org:Organization",
          "name":"Department of Transportation",
          "subOrganizationOf":{
             "@type":"org:Organization",
             "name":"U.S. Government"
          }
       }
    },
    "references":null,
    "rights":"This data requires review prior to Open Data publication.",
    "spatial":null,
    "subagency":"Federal Aviation Administration",
    "systemOfRecords":"This data requires review prior to Open Data publication.",
    "theme":[
       "Transportation"
    ],
    "title":"Flight Schedule Monitor -"
 }
```


While POD is a huge step toward enabling search and discovery of datasets and facilitates the integration with existing tools, it falls short of addressing geospatial aspects of datasets (which SRIM and GeoDCAT address). It also does not provide good guidance of the use of controlled vocabularies due to the lack of policies and solutions to manage and access controlled vocabularies (a section of this ER is dedicated on this issue). There is also no standard in place to describe services in detail. It is currently restricted to simple accessURL in dcat:Distribution.

8.2.5. GeoDCAT-AP

GeoDCAT-AP [https://joinup.ec.europa.eu/asset/dcat_application_profile/description#Geo-DCAT-AP] is defined as an extension of DCAT-AP [https://joinup.ec.europa.eu/asset/dcat_application_profile/description#DCAT-AP] for describing geospatial datasets, dataset series, and services. It provides an RDF syntax binding for the union of metadata elements defined in the core profile of ISO 19115:2003 [http://www.iso.org/iso/catalogue_detail?csnumber=26020] and those defined in the framework of the INSPIRE Directive [http://inspire.ec.europa.eu/]. Its basic use case is to make spatial datasets, data series, and services searchable on general data portals, thereby making geospatial information more searchable across

borders and sectors. This can be achieved by the exchange of descriptions of datasets among data portals. GeoDCAT-AP has been submitted to OGC as a candidate for standardization of geospatial metadata using Linked Data standards.

8.2.6. SRIM

During OGC Testbed-12, a number of metadata specifications were reviewed, including the W3C standard DCAT, DCAT-AP, GeoDCAT-AP, ADMS, Project Open Data 1.1, Dublin Core, ISO 19115, and ISO 19119 to identify the common and relevant metadata information needed for search and discovery, but also to identify the gaps to address the requirements to describe dataset, service, portrayal information, schema and schema mapping, vocabularies and other future asset types. It quickly emerged that the DCAT standard and its different application profiles were data-centric and insufficient to describe metadata for portrayal information, schemas and services. The goal was not to define a new standard, but to leverage existing standards to define an application profile of DCAT that could accommodate the schema, service, portrayal and other asset type information needed to enable relevant search and discovery and filling the gaps by introducing additional properties and fields, all while preserving backward compatibility with existing standards.

This analysis resulted in a new application profile of DCAT called the **Semantic Registry Information Model (SRIM) Core Model** [11], referred to later in the document as **SRIM Core**. SRIM Core is defined as a superset of DCAT and its existing application profiles (DCAT-AP, GeoDCAT-AP, ADMS), and it introduces a superclass of dcat:Dataset called srim:Item. The ontology draws from multiple well-established standards such as W3C, DCAT, Project Open Data 1.1, DCAT-AP, GeoDCAT-AP, VCard, Dublin Core, and PAV, but also addresses some gaps in the standards, such as descriptions of web services (for example OGC Web Map Service (WMS), Web Feature Service (WFS)), richer description for geospatial data, additional metadata to model schema, schema mapping, and portrayal information in order to enable better semantic search of resources that fit the mission of the users. The ontology draws also on the geospatial metadata standard ISO 19115. SRIM enables the integration of different metadata providers (Catalog Service for the Web (CSW), CKAN, POD Web Accessible Folder (WAF), WMS, Web Coverage Service (WCS)) by providing a common core vocabulary to describe resources (data, services, vocabularies, map, layers, schemas, etc) and accommodate the specificities of each source by leveraging the built-in extensibility mechanism in OWL. The integration is done through the use of a semantic bridge that maps the syntactic metadata (JSON, XML based) to a semantic representation based on the SRIM model. The SRIM Core model has been extended by introducing **SRIM application profiles** to represent other kinds of geospatial assets such as schemas and portrayal information (see section for Semantic Mediation and Semantic Portrayal Service in Testbed-12 ER 16-056).

The purpose of SRIM Core is to define a common interchange metadata format for geospatial portals. In order to achieve this, SRIM defines a set of classes and properties, grouped into mandatory, recommended, and optional. Such classes and properties correspond to information on register items and registers that are shared by many data portals, aiding interoperability. Although SRIM is designed to be independent from its actual implementation, RDF and Linked Data are the reference technologies used to perform the modeling and preserve the semantic fidelity of the conceptual model. However, to facilitate a wide adoption, an encoding based on JSON is provided, which could be converted transparently back to a semantic model using a JSON-LD context. The JSON encoding has been closely aligned with the Project Open Data (POD) metadata schema 1.1 standard. However, to accommodate some of the requirements needed by the Semantic Registry

Service, the model was extended, and sometimes modified where needed. An application profile of SRIM has been used for Geoplatform.gov [https://www.geoplatform.gov/], which manages all the geospatial assets of all the US government agencies. The profile has been extended to manage layers, maps and galleries. The geoplatform.gov provides importer of existing ISO 19115-x/ISO 19139 standards to SRIM and object editor to annotate assets with controlled vocabularies.

Figure 17 shows the core model of SRIM. The core mode defines all the properties that are common for most assets of any domain. The core model introduces the concept of srim:Item, which is a superclass of dcat:Dataset. It leverages geospatial extension from GeoDCAT-AP and introduces concepts of spatial extent. Attributions information are based on PROV-O and Dublin Cores. Versioning information is based on PAV ontology.

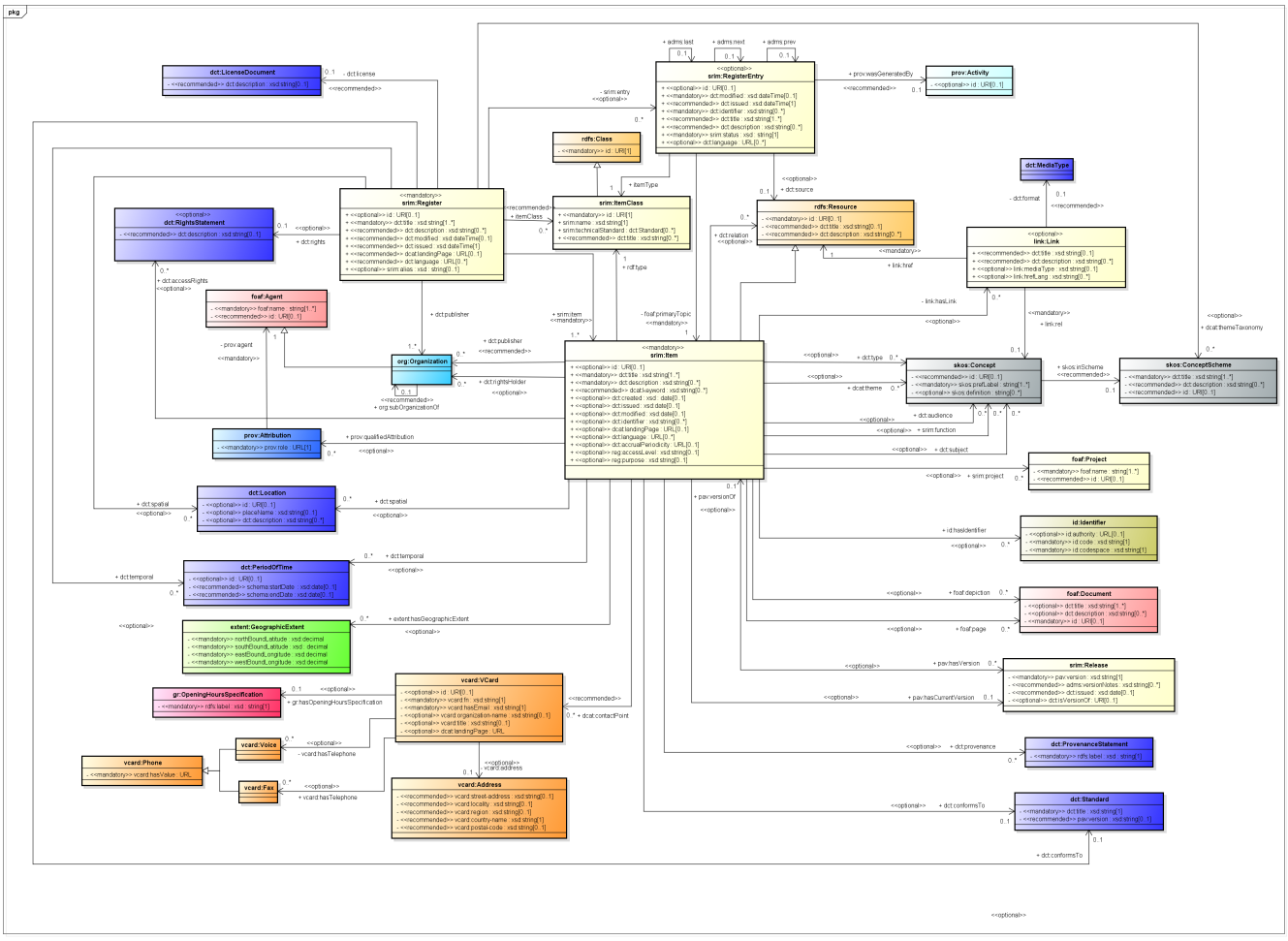

Figure 17. Core SRIM Model

Figure 18 shows the SRIM application profile to describe dataset and services. Contrary to DCAT, which uses distribution to model both services and downloadable resources,the SRIM profile makes the Service as a first class object directly subclass of srim:Item. To accommodate the variety of service APIs (using SOAP, REST, OGC services), the model introduces the concept of API Document (*srim:APIDocument*), which can point to a formal specification of the API such as ISO 19115/19119, Open API (a.k.a. Swagger), WSDL or OWL-S. By introducing this concept, we have the ability to describe service API in an extensible way. The model is highly relevant for the SWIM registry as it provides a bridge between the FAA service registry and datasets metadata using Linked data approach.

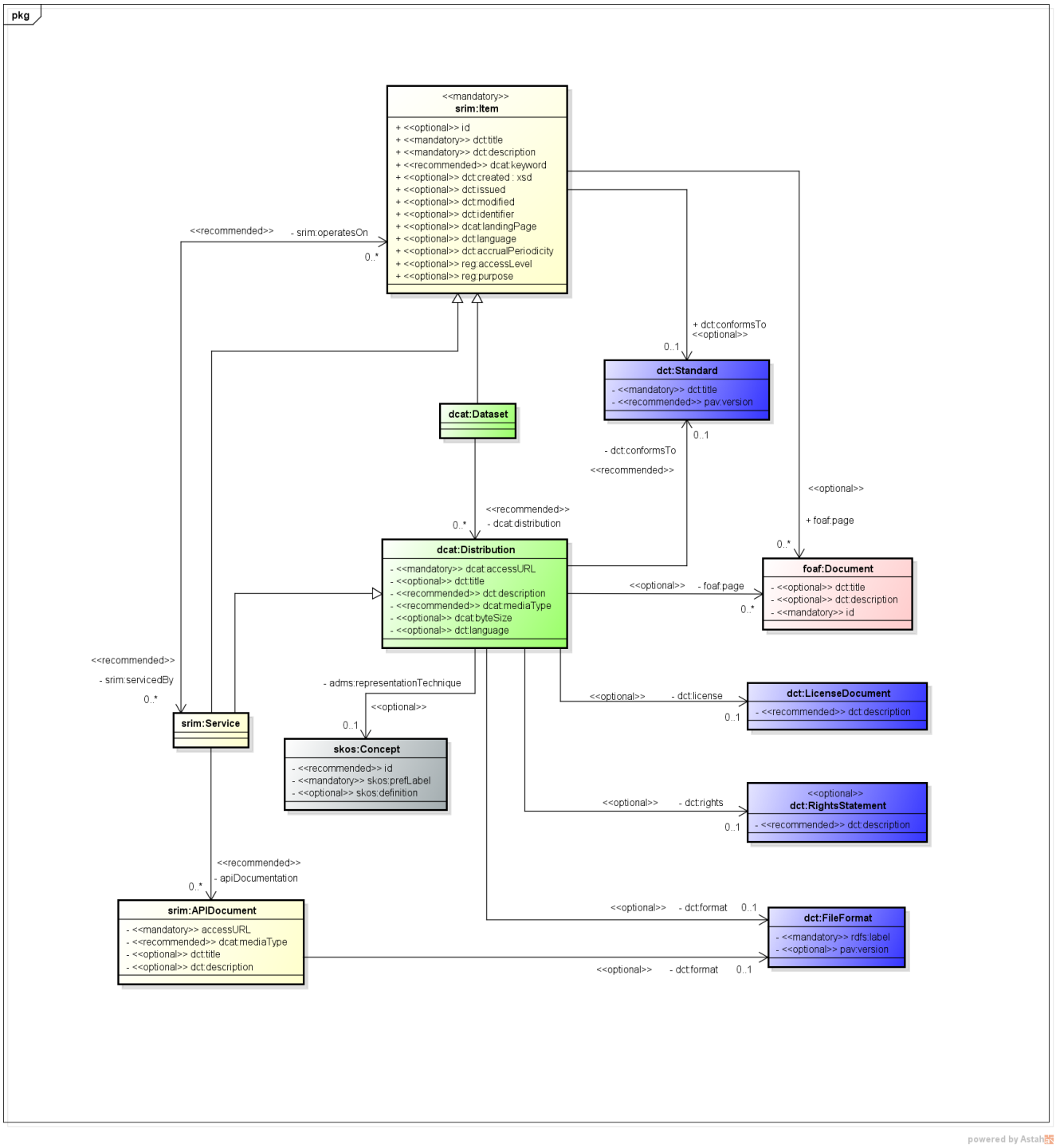

Figure 18. SRIM Dataset-Service-Profile

During Testbed-12, a Semantic Registry Service based on the SRIM model was designed and implemented. The service used a REST API using JSON-LD and HAL format (which provides hypermedia links). The Semantic Registry provides a pluggable harvesting capability demonstrated on CSW and Electronic Business Registry Information Model (ebRIM) catalogs. The semantic registry was also used to manage schema, schema mapping and portrayal information to support semantic mediation services and portrayal services. These different scenarios demonstrated that the SRIM model was extensible enough to accommodate a variety of domains, services and applications. More details can be found in Testbed-12 ER 16-056 and ER for the Task D001 of this Testbed-14.

8.3. Approaches

To restore the balance between service and dataset description in the SWIM, it is recommended to define an application profile of SRIM (for services) and DCAT (for datasets).

8.3.1. Semantic Mapping of SDCM Service to SRIM

The SDCM ontology describes a service from different viewpoints. To support the search and discovery of relevant services, the SDCM Service Profile provides metadata information that are relevant for indexing in a catalog.

One of the main focus areas of SRIM Registry is to enable search and discovery of registry items (datasets, services, layers, maps, etc..). For this reason, the SRIM model for Service focuses on descriptive metadata, classification of services, its functions and quality of service (which is mostly addressed in the SDCM Profile description). However, the details of how to access the service mostly rely on existing well-defined standards such as OGC WFS, WMS or service API specification such as WSDL, Open API, Swagger, OWL-S, RAML etc. For the first case, an SRIM Service uses the Dublin Core property *dct:conformsTo* to refer to a well-defined *dct:Standard* which refers to well defined URI. To accommodate the second case, the SRIM Service model introduces the concept of an API Document (*srim:APIDocument*), which provides an access URL (*dcat:accessURL*) to this service description document and format information (OWL-S, OpenAPI etc). The description of the Service API is mostly useful for developers and software agents that need to access the service, but rarely plays a role in the discovery of the services in the registry.

The following table defines the mapping from SDCM to the relevant classes and properties used by SRIM.

SDCM Class	SDCM Property	Description	SRIM Class	SRIM Property	SRIM Range
Profile	service name	The full name (and acronym, if any) of the service.	srim:Service	dct:title	xsd:string
Profile	service id	The identifier by which the service is uniquely referenced.	srim:Service	dct:identifier	xsd:string
Profile	service description	A description of the service.	srim:Service	dct:description	xsd:string
Profile	service version	The current version or revision level of the service.	srim:Service	pav:version	xsd:string
Profile	service category	category classifying the service	srim:Service	dct:type	skos:Concept

Profile	provider	provider of the service	srim:Service	dcat:publisher	org:Organization
Profile	consumer	consumer of the service	srim:Service	dct:audience	skos:Concept
SDCM Model	model	Service model	srim:Service	srim:apiDocument	srim:APIDocument
SDCM Grounding	grounding	Service Grounding	srim:Service	srim:apiDocument	srim:APIDocument

For future testbeds, an SRIM Service Model should be investigated to identify additional properties that better align with the SDCM model. The results of this effort will be an application profile for SWIM. Harvesting of the services from the current SWIM registry could be done and converted to the application profile, so services can be searched for and discovered, as well as be linked to datasets and other assets managed by the SRIM registry. The current SWIM taxonomies would be used to classify services and search for services.

8.3.2. Dataset metadata

The current SWIM architecture is mostly focused on describing services but there is no relationship to the datasets a service operates on. To close this gap, the metadata about datasets should be fully formalized using GeoDCAT-AP (subset of SRIM Model) vocabulary. SRIM model provides relationships to link services and datasets using the property srim:operatesOn and its inverse property srim:usedBy. An analysis of the gaps between the current SRIM Dataset core profile and the SWIM datasets should be performed and addressed by defining an application profile specific for SWIM. The metadata documents should be made available in linked data formats (RDF, JSON-LD, Turtle) in a RESTful way so they can be harvested by a semantic registry to be indexed, to allow search and discovery.

8.3.3. SRIM Registry

The SRIM Registry developed during Testbed-12 and 13 provides a service to manage semantic metadata about the different assets used in SWIM. The service has been used successfully to manage metadata about datasets, services, map layers, portrayal information, schemas and schema mappings. For future testbeds, a gap analysis should be performed to accommodate the specificities of the SWIM data model. The SRIM core model can be extended by defining application profiles that capture the specific information about the SWIM services such as Quality-of-Service, Functions, API specifications. The use of controlled vocabularies specific for aviation domain could be used for classifying, enriching, reasoning and searching the assets managed by the registry.

Chapter 9. Data Silos Semantic Enablement

Information integration is one the greatest challenges currently faced by the industry. One of the biggest obstacles is data silos. Most of the data is generated and stored in different systems throughout an organization, and these systems are separate, isolated programs. Data silos are a problem for both simple and complicated analyses:

- When data sources are siloed, it is impossible to conduct a simple enterprise-wide search of content. If you do not know exactly which system has the information you need, or it is stored in multiple systems, you must manually search multiple systems to find the answers you are looking for, wasting valuable time. If you forget to check a key system, or you don't find every silo with relevant data, you will not have all the information you need to make the best possible decisions.
- Siloed data often contains duplicate information stored in individual siloes. When the same content is stored in disparate and unconnected systems, it's difficult to know which source to trust. Which version is the correct version? How do you choose? What are the consequences of making the wrong choice?
- You cannot gain a complete view of a situation unless you combine information from different systems to understand all the factors impacting your business. Only by generating a complete view of the situation can you make the best decisions.

These data-silos are built mostly using a syntactic-based (non-semantic), protocol-centric and often document-centric approach. Information between system cannot be easily interconnected due to the lack of a common framework to represent semantic links between data. Linked Data standards provide a common and unified framework to represent any types of data structures, connect them in unified way (using RDF Model), query and reason automatically using off-the-shelf tools using Linked Data Standards (e.g. SPARQL engines,DL Reasoners). The following discusses the different approaches to semantically-enable data silos to remove the barrier of integration.

9.1. Approach for Semantic-Enablement

9.1.1. Extension of existing RESTful services protocol

It is possible to extend existing OGC services to support Linked Data representation, including if the services are RESTful. For example, the SensorThings API standard provides a RESTful API to access observations and sensor information. It is relatively easy to extend the API by introducing an additional RDF representation for each resource endpoint without breaking the existing APIs. If the JSON model of the service is compatible with the ontology representing the data model, it is possible to use JSON-LD by adding a JSON-LD context to existing JSON response. This allows clients to consume JSON-LD documents and convert the information to RDF representation for further processing. Providing a SPARQL endpoint of these services will facilitate the query of information without requiring the harvesting of information.

Applicability	• Full access to the service and backend store

Pros	• Facilitate integration with other Linked Data sources • Easier integration with Semantic Registry
Cons	• Requires access to codebase of the service. • May require RDF mapping techniques or RDF backend.

9.1.2. Semantic mapping

Until today, data integration has been accomplished using a single layer approach by writing a data product translator from one format to another. For example, it is common practice today to use XSLT to transform one XML document to another XML format. The problem with this approach is that it mixes the structural and semantic transformation together. Further, it does not scale, because it is based on an N-to-N mapping approach, and is error-prone due to reliance on human interpretation of data products.

The rules, which carry out the complete transformation process in one shot, have proven to be very complex. This causes serious problems in implementing and maintaining the rules of transformation. These problems arise due to the mixture of several different aspects of the overall transformation process, such terminology, granularity representation and structural and syntactic alignment. For this reason, any re-use of such rules is practically impossible.

To overcome this bottleneck a multi-layered framework should be used, which separates different aspects of the transformation process. The approach used in Image Matters Knowledge Mapping Service (KMS) in Testbed-10 [22] is able to transform a complex programming task into a simple plug-and-play process where straightforward rule patterns are selected, instantiated, and combined. KMS uses a methodology for data integration based on a three-layer model, as presented in the Figure 19. The model contains a Data Product layer, a Data Model layer, and an Ontology layer.

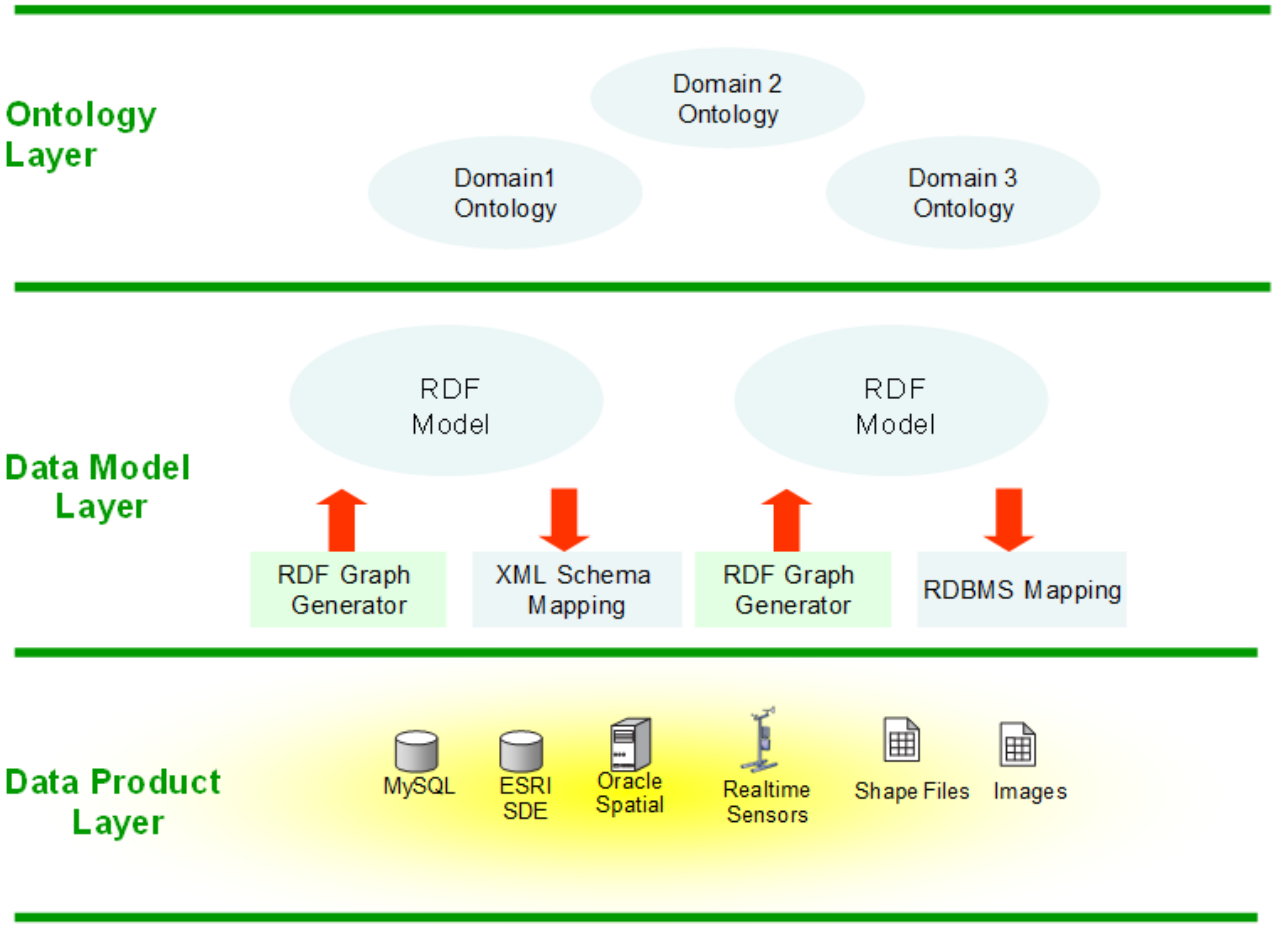

Figure 19. Three layer approach for data integration

KMS provides the ability to map 'legacy' (geospatial or not) data stores and formats to a RDF knowledge representation using a unified declarative mapping expressed in RDF. KMS uses this mapping to translate semantic query (graph query, SPARQL,...) to native query language (such as Spatial SQL, XPath/XQuery, OGC Filter) or API calls. This framework allows the virtualization of the data into a semantic graph representation and provides real-time access to data into a unified semantic representation, which could be leveraged by other knowledge-centric service components (reasoners, query engine, (Geo)SPARQL endpoints, semantic mediation, visualizations)

In Testbed-10 [23], the semantic mapping of the open source Geonames.org database, USGS GNIS gazetteers and NGA Geonames was investigated. The semantic mapping component was used to offer virtual GeoSPARQL endpoints over the mapped database and services. This approach provided a unified knowledge representation, query language and protocol to access existing gazetteer data infrastructure as illustrated in Figure 20. The semantic mapping was used to integrate WFS-G serving NGA Geonames (Interactive Instrument) and USGS GNIS (Compusult). However, the semantic mapping was limited due to some limitations in OGC Filter and issues related to usage of XLink for complex features, which tremendously impacted performance (issues are explained more in details in the OGC Engineering Report OGC 14-029).

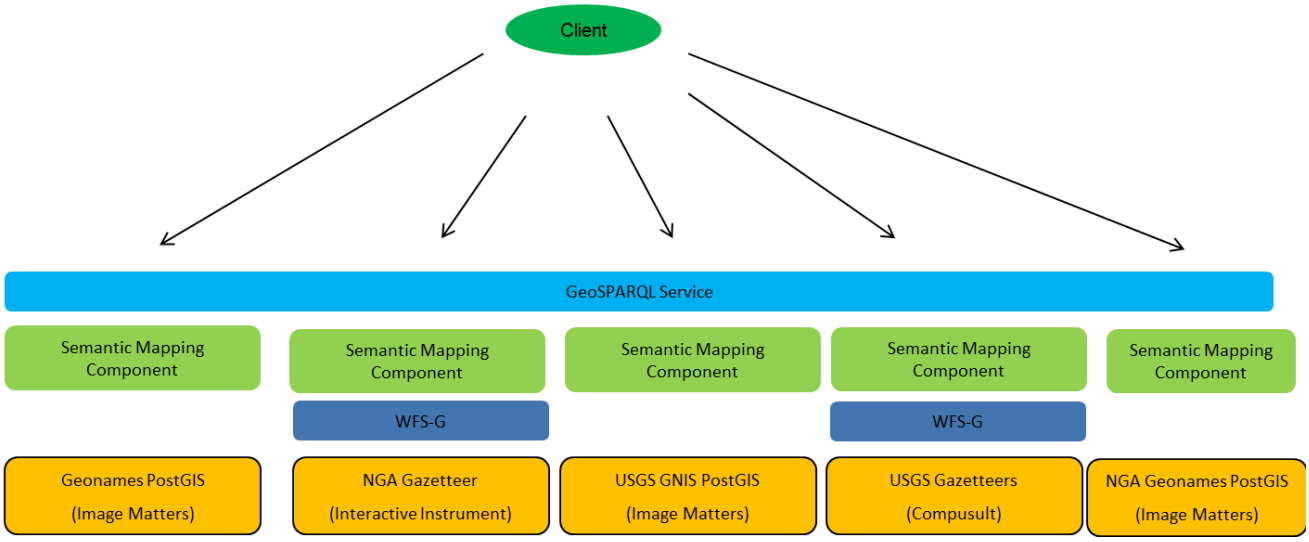

Figure 20. Example of Semantic Mapping for gazetteer sources (Testbed 10)

Applicability	• Access to Datastore API or endpoint.
Pros	• Leverage native performance of data stores • Different mappings can be used for the same stores • Does not require migration of data to RDF
Cons	• Requires access to the datastore API.

9.1.3. Web API Semantic Wrapper

A large numbers of Web applications have started to make their data available on the Web through Web APIs. Examples of data sources providing such APIs include Social Media APIs (Twitter, YouTube, Flickr), OGC Services such as WFS, WMS, WCS, and Sensor Observation Service (SOS). Different APIs provide diverse query and retrieval interfaces and return results using a number of different formats such as XML, JSON or ATOM. This leads to three general limitations of Web APIs:

- their content cannot be crawled by search engines
- Web APIs cannot be accessed using generic data browsers
- Mashups are implemented against a fixed number of data sources and cannot take advantage of new data sources that appear on the Web.

These limitations can be overcome by implementing Linked Data wrappers around APIs. In general, Linked Data wrappers do the following:

- They assign HTTP URIs to the non-information resources about which the API provides data.
- When one of these URIs is dereferenced asking for application/rdf+xml, the wrapper rewrites the client's request into a request against the underlying API.
- The results of the API request are transformed to RDF and sent back to the client.

This approach has been used in Testbed-11 to semantic-enabled Social Media APIs by using a

semantic wrapper around APIs. This approach for semantically enabled WFS-G (i.e. Gazetteer profile of WFS) by transforming GeoSPARQL queries to OGC filter queries on the fly and convert the GML response to Linked Data representation.

Applicability	• Service codebase is not accessible but API is well documented.
Pros	• Relatively easy to implement
Cons	• Requires RDF storage to save results of scraping • API is not queryable using SPARQL.

9.1.4. RDFa

RDFa (or Resource Description Framework in Attributes) is a W3C Recommendation that adds a set of attribute-level extensions to HTML, XHTML and various XML-based document types for embedding rich metadata within Web documents. The RDF data-model mapping enables its use for embedding RDF subject-predicate-object expressions within XHTML documents. It also enables the extraction of RDF model triples by compliant user agents. There is a wide variety of tools that can be used to generate or process RDFa data. Google leverages RDFa annotation in web pages to build its Knowledge Graph. By using RDFa in web pages, they will be displayed in an enhanced format on all major search engines. Facebook's Open Graph Protocol, which is based on RDFa, is used express concepts that are contained in web pages, like people, places, events, movies and recipes. Major search and social companies are supporting indexing RDFa content to improve search experience.

```
<html>
<head>
  ...
</head>
<body>
  ...
  <h2 property="http://purl.org/dc/terms/title">Flight Plan Service</h2>
  <p>Date: <span property="http://purl.org/dc/terms/created">2017-09-10</span></p>
  ...
</body>
```

It is worth emphasizing that RDFa uses URLs to identify just about everything. This is why, instead of just using properties like title or created, we use http://purl.org/dc/terms/title and http://purl.org/dc/terms/created. The reason behind this design decision is rooted in data portability, consistency, and information sharing. Using URLs removes the possibility for ambiguities in terminology. Without ensuring that there is no ambiguity, the term "title" might mean "the title of a work", "a job title", or "the deed for real-estate property". When each vocabulary term is a URL, a detailed explanation for the vocabulary term is just one click away. It allows anything, humans or machines, to follow the link to find out what a particular vocabulary term means. By using a URL to identify a particular creation time, for example http://purl.org/dc/terms/created, both humans and machines can understand that the URL unambiguously refers to the "Date of creating the resource", such as a web page or dataset.

Applicability	• Mostly HTML resources (possibly XML but not found evidence of usage except in SVG)
Pros	• Can describe any web page using any ontology
Cons	• Annotation are tedious to capture

9.1.5. SAWSDL

The Semantic Annotations for WSDL and XML Schema (SAWSDL) specification [24] defines how to add semantic annotations to various parts of a WSDL document such as input and output message structures, interfaces and operations. The extension attributes defined in this specification fit within the WSDL 2.0, WSDL 1.1 and XML Schema extensibility frameworks. For example, this specification defines a way to annotate WSDL interfaces and operations with categorization information that can be used to publish a Web service in a registry. The annotations on schema types can be used during Web service discovery and composition. In addition, SAWSDL defines an annotation mechanism for specifying the data mapping of XML Schema types to and from an ontology; such mappings could be used during invocation, particularly when mediation is required. To accomplish semantic annotation, SAWSDL defines extension attributes that can be applied both to WSDL elements and to XML Schema elements.

The semantic annotations reference a concept in an ontology or a mapping document. The annotation mechanism is independent of the ontology expression language and this specification requires and enforces no particular ontology language. It is also independent of mapping languages and does not restrict the possible choices of such languages. One of the main motivations for SAWSDL specification is to provide mechanisms using which semantic annotations can be added to WSDL documents so that these semantics can be used to help automate the matching and composition of Web services.

SAWSDL has been suggested as a possible solution to integrate semantics within XML schemata describing geospatial data (OGC 08-167r2). Examples of SAWSDL annotations of ATM service XML schema are presented in OGC 18-022.

Applicability	• web services are using WSDL document to describe the API • Content of services in XML using XML Schema
Pros	• Works well for services adhering to the WSDL and XML Schema
Cons	• Does not work with web services that are not based on WSDL (REST API using OpenAPI for example). • Does not work with services using JSON or other formats not based on XML schema.

9.1.6. GRDDL

GRDDL is a mechanism for **Gleaning Resource Descriptions from Dialects of Languages**. The W3C GRDDL specification introduces markup based on existing standards for declaring that an XML document includes data compatible with the Resource Description Framework (RDF) and for linking to algorithms (typically represented in XSLT), for extracting this data from the document. The results of the transformations will usually be RDF/XML documents, although other RDF syntaxes may be used.

The markup includes a namespace-qualified attribute for use in general-purpose XML documents and a profile-qualified link relationship for use in valid XHTML documents. The GRDDL mechanism also allows an XML namespace document (or XHTML profile document) to declare that every document associated with that namespace (or profile) includes gleanable data and for linking to an algorithm for gleaning the data.

Individual publishers of data using popular vocabularies can also give users their data transformed into RDF **without having to even add any new markup to individual documents**. This is done by referencing GRDDL transformations in a profile document referenced in the head of the HTML. Other XML vocabularies may use their namespace documents for the same purpose. This method requires no work from the content author of individual documents but requires that the profile document contain a reference to a GRDDL transformation and be accessible to the GRDDL client, and so may require work from the creator and maintainer of the dialect. Yet this is a good use of time, since once the transformation has been linked to the profile document, all the users of the dialect get the added value of RDF.

`Applicability`	• web services using WSDL document to describe the API • Content of services in XML using XML Schema
`Pros`	• Works well for HTML pages that are rendered from databases
`Cons`	• Does not work with web services that are not based on WS-* stack (WSDL) (REST API using OpenAPI for example). • Mapping of XML Schema to RDF seems to be focused only on API messages. Not sure if applicable for any XML schema. • Does not work with services using JSON or other formats not based on XML schema.

9.1.7. W3C Annotation

The Web Annotation Working Group has published recommendations for capturing annotations:

- **Web Annotation Data Model**: This specification describes a structured model and format, in JSON-LD, to enable annotations to be shared and reused across different hardware and software platforms. Common use cases can be modeled in a manner that is simple and convenient, while at the same time enabling more complex requirements, including linking arbitrary content to a particular data point or to segments of timed multimedia resources.

- **Web Annotation Vocabulary**: This specifies the set of RDF classes, predicates and named entities that are used by the Web Annotation Data Model. It also lists recommended terms from other ontologies that are used in the model, as well as providing the JSON-LD Context and profile definitions needed to use the Web Annotation JSON serialization in a Linked Data context.
- **Web Annotation Protocol**: The Web Annotation Protocol describes the transport mechanisms for creating and managing annotations in a method that is consistent with the Web Architecture and REST best practices.

JSON-LD is the serialization format used in the Web Annotation Data Model. HTML can accommodate this serialization format directly via the use of the HTML <script> element with its type attribute assigned the media type for a Web Annotation: application/ld+json;profile="http://www.w3.org/ns/anno.jsonld" (see section about Embedded JSON-LD below)

Another approach of embedding annotations into HTML is to use RDFa. The advantage of using RDFa is that the annotation terms are mixed with the core HTML content, meaning that, for example, the text in the HTML source can be also re-used as an annotation textual body. In other words, a single resource may become both human visible as well as machine-readable. On the other hand, RDFa is an RDF serialization syntax: the RDF vocabulary described in Annotation Vocabulary specification must therefore be used instead of the JSON-LD encoding used in the Annotation Data Model document.

In OGC 18-022, the web annotation data model is proposed as a solution to associate domain specific semantic tags with ATM service descriptions as well as whole or parts of the related text documents and XML schemata. More information and concrete examples can be found in OGC 18-022.

Applicability	• Any web resources
Pros	• W3C annotations can work on any document (XML, JSON, Video, Image) • Annotations are not modifying existing documents.
Cons	• Annotation are tedious to capture and needs to be done on a document-basis

9.1.8. Microdata

Microformats are small patterns of HTML to represent commonly published things like people, events, blog posts, reviews and tags in web pages. Microformats are the quickest & simplest way to provide an API to the information on your website. Microdata is an open-community HTML specification used to nest structured data within HTML content. Like RDFa, it uses HTML tag attributes to name the properties you want to expose as structured data. It is typically used in the page body, but can be used in the head. However the number of microdata implementations is limited (h-card, XFN, h-calendar, h-review, rel-tag, rel-license,..) and format such as RDFa and JSON-LD have gain marketshare due to their extensibility to accommodate any domain.

Applicability	• HTML resources
Pros	• Quickest & simplest way to provide an API to the information on your website
Cons	• Limited on small general domains

9.1.9. Embedded JSON-LD

JSON-LD is a lightweight Linked Data format. It is easy for humans to read and write. It is based on the already successful JSON format and provides a way to help JSON data interoperate at Web-scale. JSON-LD is an ideal data format for programming environments, REST Web services, and unstructured databases such as CouchDB and MongoDB. A JSON-LD document can be embedded in a <script> tag in the page head or body. The markup is not interleaved with the user-visible text, which makes nested data items easier to express, such as the Dataset, Service description in the context of NAS. Also, Google can read JSON-LD data when it is dynamically injected into the page's contents, such as by JavaScript code or embedded widgets in a content management system.

For example, Google uses structured data that it finds on the web to understand the content of the page, as well as to gather information about the web and the world in general. For example, here is a JSON-LD structured data snippet that might appear on the contact page of the FAA s corporation, describing their contact information:

```
<script type="application/ld+json">
{
  "@context": "http://schema.org",
  "@type": "Organization",
  "url": "http://www.faa.gov",
  "name": "Federal Aviation Agency",
  "contactPoint": {
    "@type": "ContactPoint",
    "telephone": "+1-401-555-1212",
    "contactType": "Customer service"
  }
}
</script>
```

Google Search also uses structured data to enable special search result features and enhancements. For example, a recipe page with valid structured data is eligible to appear in a graphical search result, as shown here:

Google recently recommended the use of JSON-LD as the preferred approach to capture structured

data in web pages. Also note that the W3C recommended this technique to embed W3C Annotation in HTML (https://www.w3.org/TR/annotation-html/)

Applicability	• Any HTML resources
Pros	• Can describe any web page against any ontology. • Easy to add in HTML file and cleanly separated from HTML content.
Cons	• Works only on HTML page.

9.2. Pure RDF approach

Using a pure RDF approach requires the backend to be implemented using an RDF store. RDF stores such as RDF4J, Blazegraph, AWS Neptune, OntoText GraphDB, AllegroGraph comes with built-in reasoners, SPARQL API support.

Applicability	Information can be stored in RDF directly
Pros	• Easy to implement • Leverage reasoning and standard SPARQL query • Integration with other Linked Data sources is easier.
Cons	• Does not leverage existing legacy systems • Performance may be an issue if not supported by additional indices (spatial index for example). • Does not easily integrate with web-based clients

Chapter 10. The role of Controlled Vocabularies

Controlled Vocabularies play a central role in the search and discovery of digital assets (services, datasets, maps, layers,etc.). The purpose of controlled vocabularies is to organize information and to provide terminology to annotate, catalog and retrieve information. While capturing the richness of variant terms, controlled vocabularies also promote consistency in preferred terms and the assignment of the same terms to similar content.

Controlled vocabularies could be used for the following use cases:

- Classification of different dimensions of an entity such as places (using gazetteer for example), temporal periods, subjects (GCMD for example), themes, topics, functions, audience, communities.
- Semantic Tagging Recommendation: concepts are suggested when typing a keyword enabling the tagging of information without ambiguities. For example, a user may search for New York, the semantic recommender may suggest New York (the city) or New York (the state).
- Semantic Enrichment/Inferencing by use of reasoners (partonomic relationships or broader/narrower relationships)
- Navigation of information by traversing conceptual spaces linking to resources
- Query enrichment by injecting alternative labels or synonyms or more specialized terms.
- Conceptual search instead of keyword search

One of the biggest obstacles today into enabling semantic interoperability of different systems is the lack of policies for usage and governance of controlled vocabularies. The problem is not a technical one, it is the lack of enforcement of policies for usage terminology that are semantically grounded in the different standards.

> **RECOMMENDATION**: Define usage and governance policies for controlled vocabularies within FAA.

The **formal encoding, management and governance of controlled vocabularies is essential to support semantic annotations and improved discovery and search** based on the semantic of the concepts of the vocabularies. There are already a number of well-defined semantic controlled vocabularies used by the different communities (GCMD, AgroVoc, Eurovoc,...) that are used to classify digital assets managed in different registry such as DCAT Catalog or CKAN, however few policies are put in place by agencies to enforce their usage.

FAA has currently deployed a web service providing access to controlled vocabularies encoded in SKOS at https://semantics.aero/. While this is a huge step toward the right direction, formal policies for management and governance of vocabularies needs to be put in place, as well as APIs to discover and search terms.

RECOMMENDATION: Explore in future testbeds an API for a Vocabulary Management Service that is needed to manage, index, query vocabularies terms and define best practices for governance of the lifecycle of controlled vocabularies.

The following subsections provide more details about the definition of controlled vocabularies, their different classifications, different ways of encoding and the requirements for controlled vocabulary service that can support the different use cases described above.

10.1. Controlled Vocabulary Definition

This subsection and the next are adapted from Neiswender [25]. A **vocabulary** is a set of terms (words, codes, etc.) that are used in a specific community. Vocabularies provide a mechanism for communication- be it written, oral or electronic- because the meaning of the terms are known and agreed upon by the community members. When a vocabulary is formally managed, it becomes a **controlled vocabulary**. In this case, "managed" means the terms are stored and maintained using agreed-upon procedures. Procedures should exist for adding terms, modifying terms and, more rarely, deprecating terms from a controlled vocabulary.

A controlled vocabulary is a collection of terms that are:

- **Accepted**: The term must adhere to community practices.
- **Defined**: The terms are precisely characterized. Typically, this means the terms have rigorous definitions.
- **Managed**: In general, there will be a body of experts that create and maintain the controlled vocabulary. The controlled vocabulary maintenance will involve periodic review, addition of new terms, modification of terms, and occasionally deprecation of terms.

Notice, this definition does not specify a particular scope of usage. Controlled vocabularies could be developed for a local project, a broader community (e.g. GeoINT), or as a part of a widely used standard or tool (ISO 19115).

10.2. Vocabulary Classification

To many people, the English language contains a well-known vocabulary. There are have many ways of representing the terms in the English language. For example, if we want to figure out what a specific word means we might consult a glossary; if we want to know the origin of the term, we might consult a dictionary; and if we want to know how the term relates to other terms, we might consult a thesaurus. We also need to recognize that the meaning of terms may change through time. Generations use terms in different ways (cool in one generation means a low temperature, while cool in another is a positive adjective).

To enable formal management, a controlled vocabulary can be organized in several ways. There are three broad categories of controlled vocabularies: flat, multi-level and relational.

- **Flat controlled vocabularies** provide a set of used terms. Some flat controlled vocabularies will provide additional information about each term.

- **Multi-level controlled vocabularies** build upon a flat controlled vocabulary by assigning each term to a category.
- **Relational controlled vocabularies** provide a set of terms, and capture how they are associated with each other.**

Figure 21 shows a classification of vocabulary types.

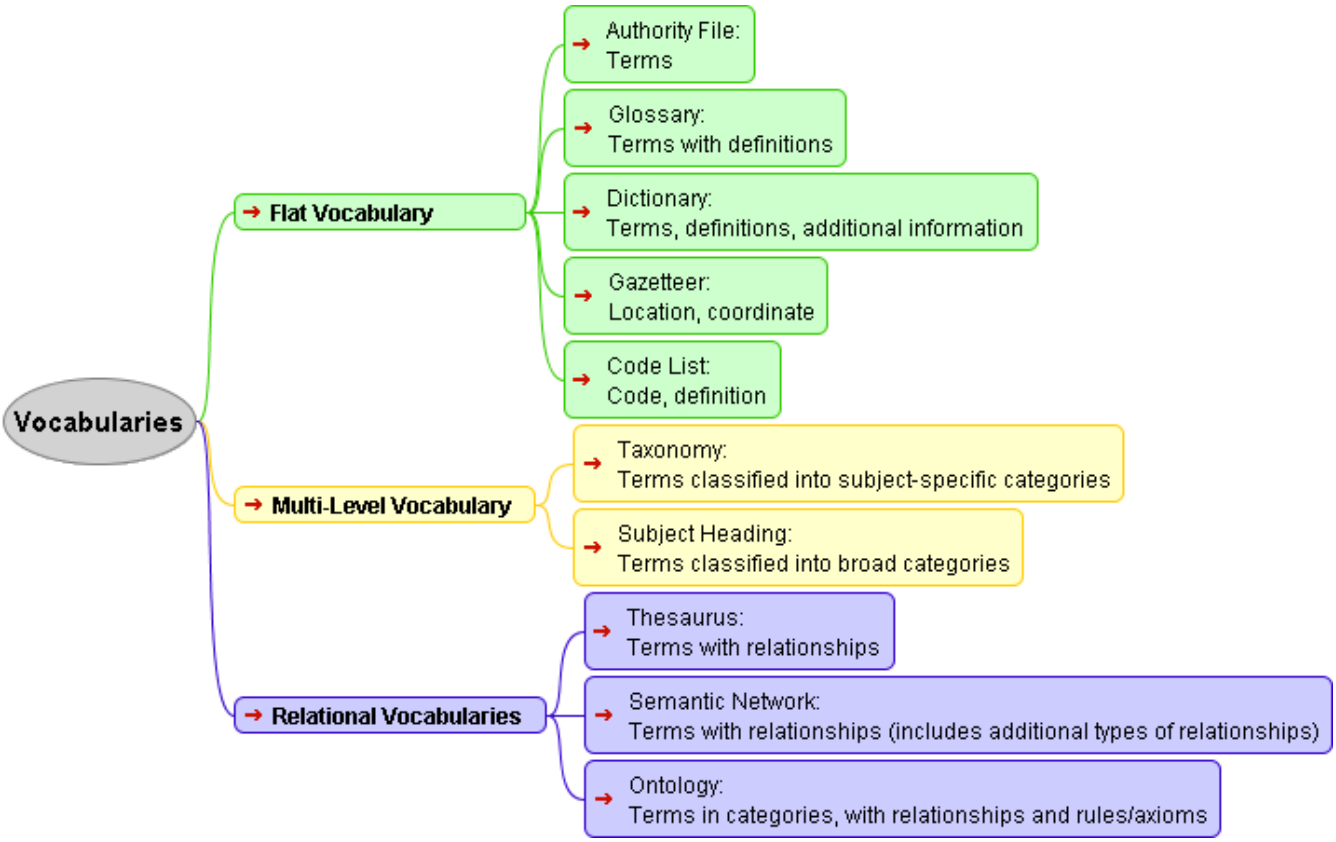

Figure 21. Vocabulary types

10.3. Encoding of Controlled Vocabularies

To make the use of controlled vocabularies more widespread, controlled vocabularies need to be encoded using a standard that captures semantic relationships between the different terms. The W3C Simple Knowledge Organization System (SKOS), RDF Schema and Web Ontology Language (OWL) are widely used to capture the different expressiveness of controlled vocabularies.

10.3.1. SKOS

SKOS provides a standard vocabulary for representing thesauri, subject headings, taxonomies, flat vocabularies, using RDF. SKOS has a simple model with few key constructs, focusing on labeling and basic hierarchies. While it lacks the expressivity and rigor of languages such as OWL, its simplicity allows a broad range of vocabularies and classifiers to be ported from a diverse set of formats to RDF, promoting ease of sharing and cross-linking between vocabularies. Many existing vocabularies have been ported to SKOS including large vocabularies such as AGROVOC and the Library of Congress Subject Headers (LCSH). SKOS is now one of the most commonly used vocabularies for structured data on the web. Figure 22 shows a sample of SKOS concept.

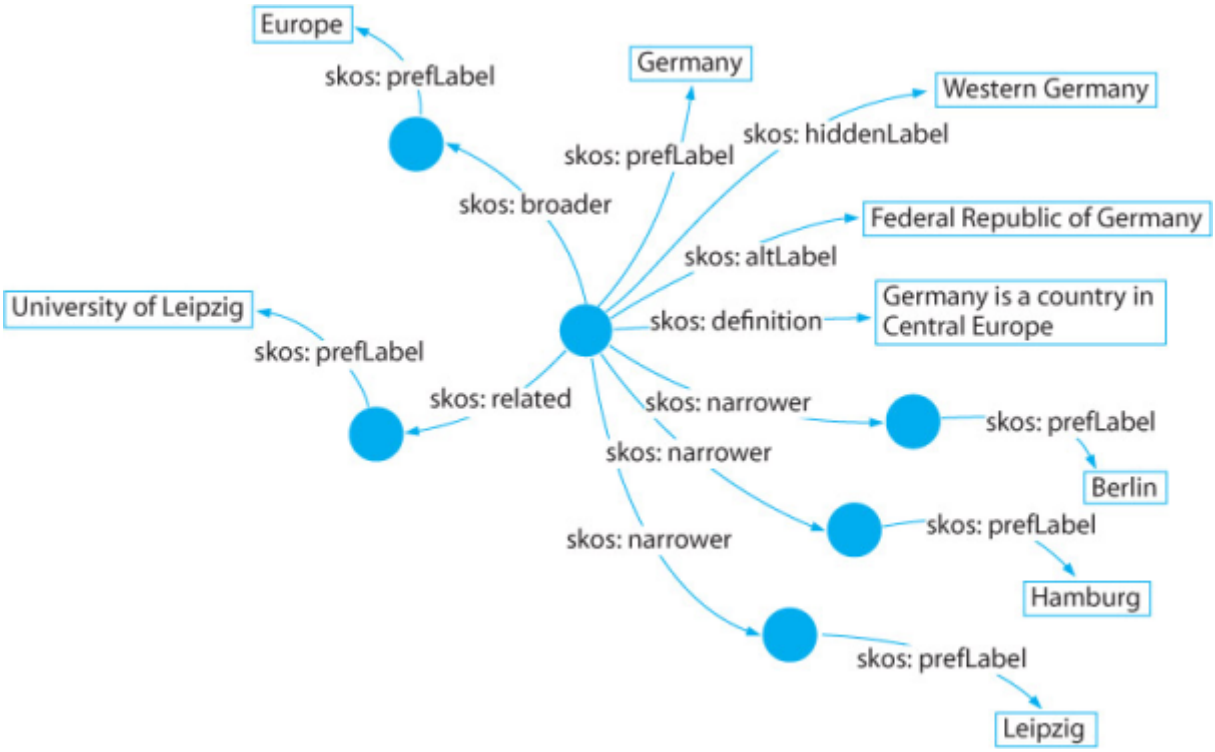

Figure 22. SKOS Concept Example

10.3.2. Ontology languages

An **ontology** is a formal unambiguous naming and definition of the types, properties, and interrelationships of the entities that really or fundamentally exist for a particular domain of discourse (as illustrated below). There are two main standards defined by W3C and used to encode ontologies:

- **RDF Schema (RDFS)** is the simplest RDF vocabulary description language. It provides constructs to define class and property hierarchies as well as domain and range of properties.
- **Web Ontology language (OWL)** is an extension of RDFS and is more expressive. It is based on construct from Description Logic and provides ability to provide rich semantic description of classes and properties. Figure 23 shows an example of ontology (SIOC Ontology).

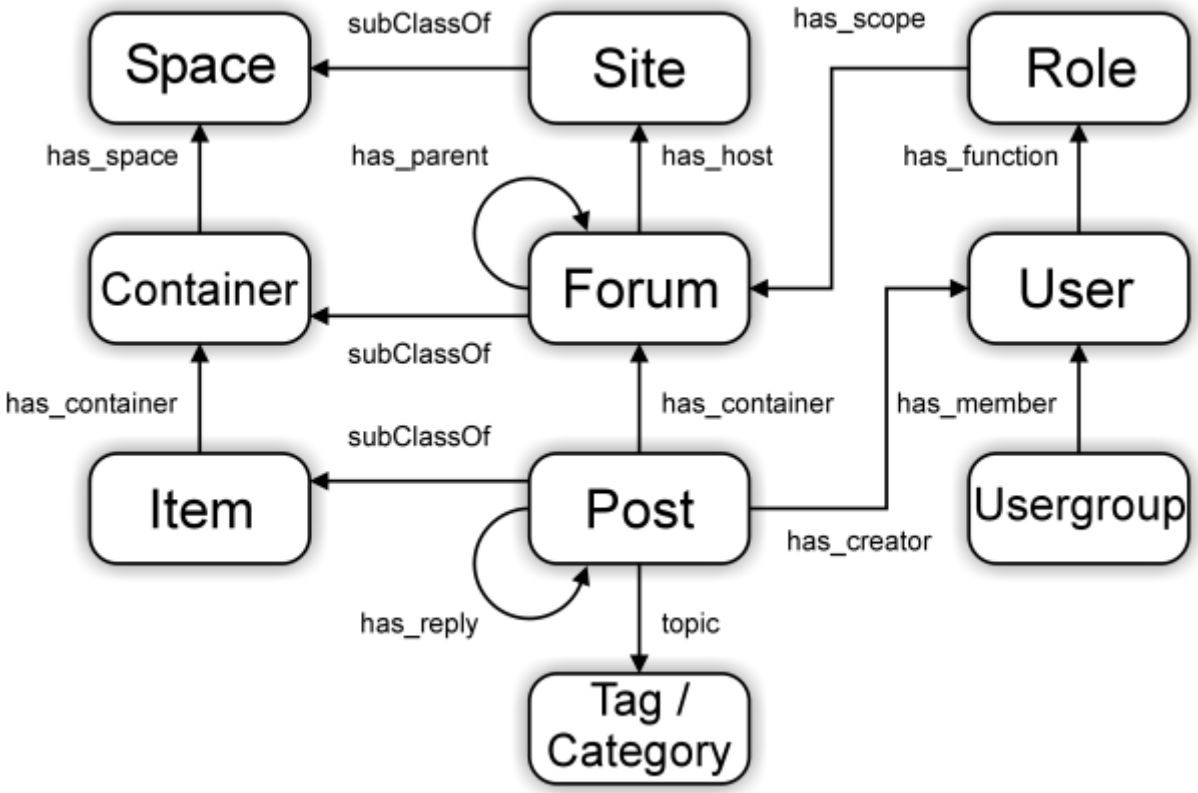

Figure 23. SIOC Ontology example

10.4. Improvements on existing vocabularies

10.4.1. CodeList, Taxonomy and Thesauri conversion to OWL

The current taxonomies used in SWIM and hosted in semantics.aero are encoded in SKOS. While this is step toward the right direction, there is no ability to distinguish the categories of concepts (ServiceProduct, ICAORegion) without knowing a well-known concept scheme. This is problematic, if different taxonomies for the same concept categories exists. The ISO 19150-2 standard defines the rules of conversion of UML CodeList and enumeration to OWL to address this issue. This rule can also be used for taxonomies and thesauri.

The ISO 19150-2 specification defines the rules of conversion of UML codelist to OWL as:

> A CODELIST shall correspond to a Class <OWL>, a ConceptScheme <SKOS>, and a Collection <SKOS>. The Class <OWL> shall be a subclass of skos:Concept. The SKOS concept scheme shall be related to the Class <OWL> using a dct:isFormatOf property. Each member of the CODELIST shall correspond to an individual whose type is the Class <OWL> corresponding to the CODELIST, and with a skos:inScheme property whose value is the ConceptScheme <SKOS> corresponding to the CODELIST. Additionally, each member of the CODELIST shall also be member of the Collection <SKOS> using a skos:members declaration. Each of the resources shall be annotated with the following:
>
> - a label, using rdfs:label,
> - a source for the definition, using rdfs:isDefinedBy for the IRI of the resource.

Figure 24 shows the conversion of code list ClassA.

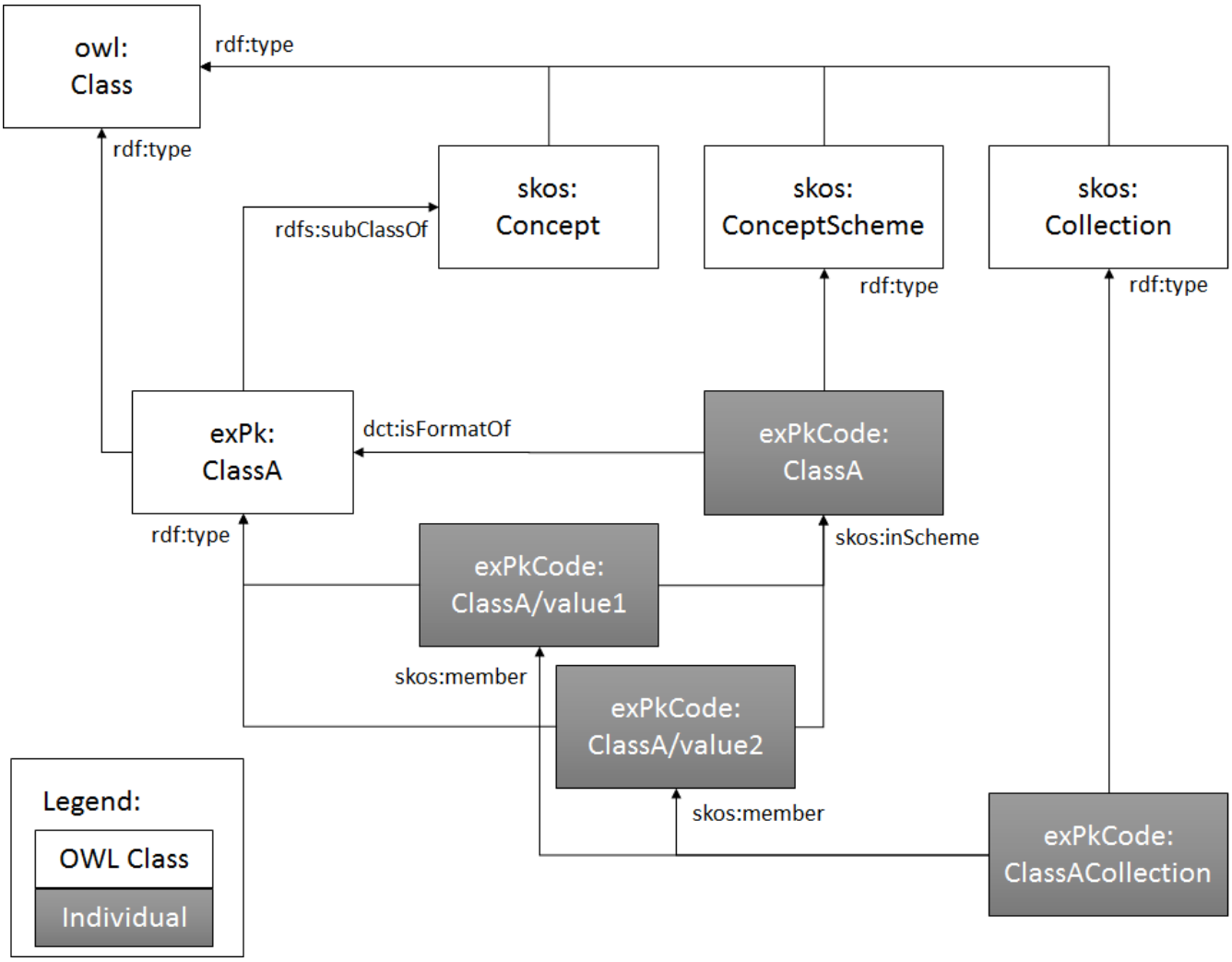

Figure 24. CodeList conversion to OWL

10.4.2. Considerations on Ontology Lifecycle and Code List Conversion

An application schema can evolve, meaning that multiple versions of the schema can be published. An automatically derived ontology can then also have multiple versions. ISO 19150-2 defines rules for the conversion of code lists to OWL and SKOS. This section discusses potential issues that can arise when considering multiple versions of an application schema and the derived ontology, specifically in these cases:

- The ontology name changes between subsequent versions of the ontology
- The code lists defined by the application schema are externalized, meaning they are converted to OWL and SKOS only once; afterwards, the OWL and SKOS representation is managed outside of the application schema, which subsequently only refers to this representation.

NOTE If ISO 19150-2 is strictly followed, the ontology name can change if the URIbase changes or if the names of UML packages are changed. The conversion rules described in the section on package name and namespaces can also lead to changes in the names of ontologies derived from different versions of the application schema namespaces.

Assume that an application schema defines a code list ClassA that contains a collection of codes. The intent is to derive an ontology from this schema, thereby deriving an OWL and SKOS representation of the code list. Also assume that this representation will subsequently be managed outside of the application schema. For example, addition of new codes as well as deprecation of old codes may be performed in a registry, without reflecting those changes in the application schema. To acknowledge the fact that the code list is managed externally, subsequent versions of the application schema only contain a stub for the code list. To manage externally managed code list 'ClassA', the OWL 'ClassA' should have been created in a namespace that is different to that of the application schema as illustrated in Figure 25.This would requires to import the external ontology in the new application profile.

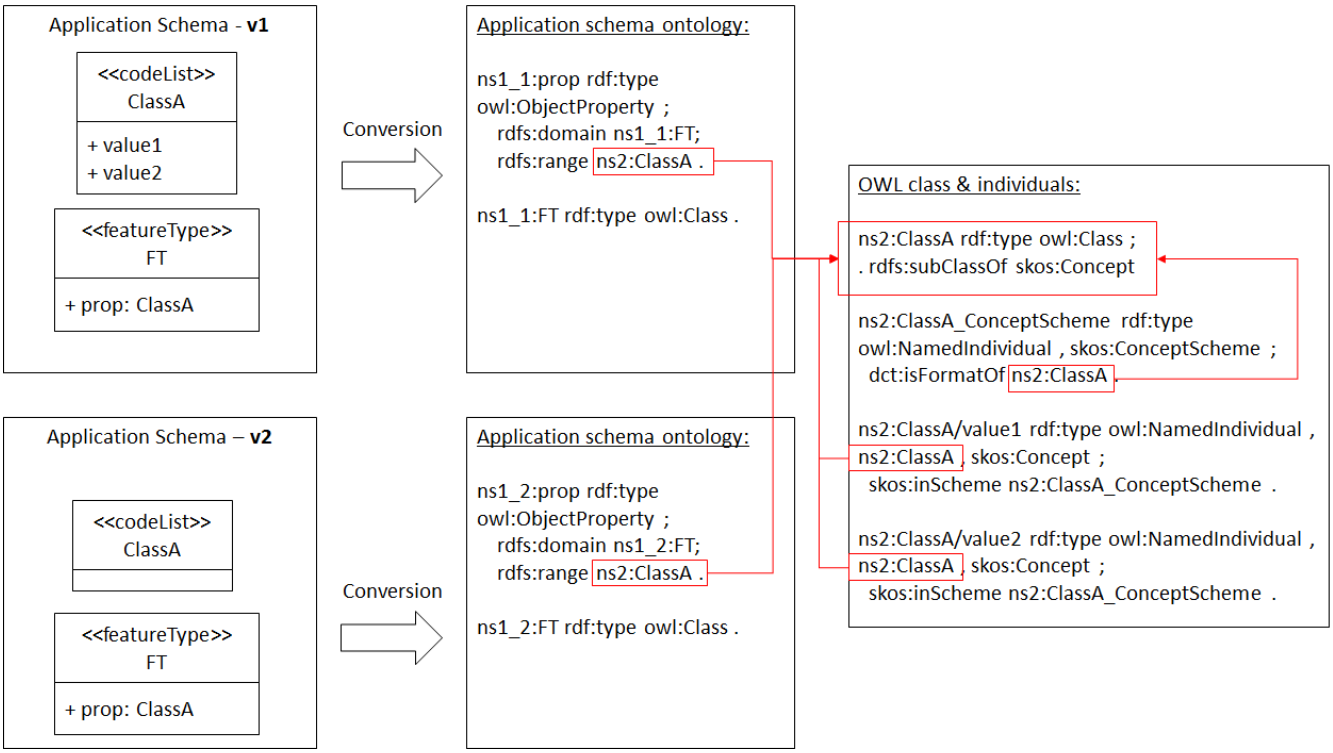

Figure 25. CodeList conversion to OWL using external taxonomies

RECOMMENDATION: Migrate existing SWIM taxonomies by applying rules from ISO 19150-2

10.4.3. Gazetteer for Aviation

A gazetteer provides information about place names (toponyms), location types, partonomy and geometric information (typically point, but could also be polygon or lines). Gazetteers are useful for classifying information spatially using place names. By leveraging partonomy information, partonomic search can be performed using transitive reasoner and geometric coordinate can be used to perform spatial operations. A gazetteer place could be seen as a specialization of skos:Concept by adding toponyms are specialization of skos-xl:Label. Partonomic relations can be seen as specialization of skos:broader (partOf) and skos:narrower (hasPart). Testbed-10 explored the use of semantic for gazetteer using this approach using semantic mapping components [OGC 14-029r2].

RECOMMENDATION: Deploy a semantic gazetteer service that could be used to tag FAA assets and enable spatial semantic search.

10.4.4. Best practices for domain ontology design

This section describes best practices to design ontologies for aviation data models. NASA ATM ontologies is a good example of modular ontologies following these design principles.

Minimal ontological commitment

Modular design of the ontologies should follow the principle of making a minimal ontological commitment to the nature of concepts and of relationships between concepts. As explained by Thomas Gruber [26], an ontology should require the minimal ontological commitment sufficient to support the intended knowledge sharing activities. An ontology should make as few claims as possible about the world being modeled, allowing the parties committed to the ontology freedom to specialize and instantiate the ontology as needed (which is often called ontology application profile).

Opting for such a minimal approach is made dramatically easier by the vocabulary extension mechanisms offered natively by Semantic Web technology. Applications that require more constrained behavior may define compatible extensions to OWL or SKOS. For example, modelers may coin sub-classes and sub-properties of OWL or SKOS properties, or associate those properties with specific formal axioms. By making a minimal ontological commitment, the ontologies can be applied and reused across multiple Communities of Interests (COIs), thus increasing the rate of wide-spread adoption.

Modularization of ontologies

Quoting Stuckenschmidt and Klein [27], “ontologies that contain thousands of concepts cannot be created and maintained by a single person”. Modularization helps designers manage complexity by reducing the size of the design problem [28]. Ideally, designers should design modules of a size that they can apprehend, and later either integrate these modules into a final repository or build the relationships among modules that support interoperability. This is a typical application of the divide-and-conquer principle.

Modularization also provides a way to keep performance of ontology services at an acceptable level. Performance concerns may be related to query processing techniques, reasoning engines and ontology modeling and visualization tools. Reasoners currently available are performing well on small-scale ontologies, with performance degrading rapidly as the size of the ontology increases. Keeping ontologies small is one way to avoid the performance loss, and modularization is a way to replace an ontology that tends to become oversized by smaller subsets. Modularization fulfills the performance goal if, whenever a query has to be evaluated or an inference to performed, this can be done by looking at just few modules, rather than exploring the whole ontology [28].

Reusability of ontologies

Reusability is a well-known goal in software engineering. Reuse is most naturally seen as an essential motivation for approaches aiming at building a broader, more generic repository from

existing, more specialized repositories. However, it may also apply to the inverse approaches aiming at splitting an ontology into smaller modules. In this case, the decomposition criterion should be based on the expected reusability of a module, i.e. how well can a module fill the purpose of various applications. Reusability emphasizes the need for rich mechanisms to describe modules, in a way that maximizes the chances for the modules to be understood, selected and used by other services and applications.

Understandability

An obvious prerequisite to the ability to use ontology to understand its content. Whether the content is shown in visual or textual format, understanding is easier if the ontology is small (a module). Small ontologies are undoubtedly preferable if the user is a human being. Size, however is not the only criterion that influences understandability. The way it is structured contributes to improving or decreasing understandability.

Appendix A: Sample datasets

A.1Airport Dataset Metadata example

The following example describes the metadata about a dataset describing airports, which is serviced by a GEOSPARQL service endpoint and two distribution downloads in RDF/XML and Turtle format.

```
@prefix : <http://www.imagemattersllc.com/atmonto/datasets#> .
@prefix acManuInst: <https://data.nasa.gov/ontologies/atmonto/acManufInst#> .
@prefix airlineInst: <https://data.nasa.gov/ontologies/atmonto/airlineInst#> .
@prefix dcat: <http://www.w3.org/ns/dcat#> .
@prefix dcterms: <http://purl.org/dc/terms/> .
@prefix eqp: <https://data.nasa.gov/ontologies/atmonto/equipment#> .
@prefix nas: <https://data.nasa.gov/ontologies/atmonto/NAS#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix service-product: <http://semantics.aero/service-product#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

<https://data.nasa.gov/ontologies/atmonto/airportInst>
  rdf:type dcat:Dataset ;
  dcterms:conformsTo <https://data.nasa.gov/ontologies/atmonto/NAS> ;
  dcterms:description "airports, runways, terminals, gates" ;
  dcterms:subject nas:Airport, nas:InternationalAirport, nas:CanadianAirport,
nas:CONUSairport ;
  dcterms:title "Airport Instances" ;
  dct:publisher <http://www.nasa.gov>,
  dcat:keyword "airports" ;
  dcat:keyword "gates" ;
  dcat:keyword "runways" ;
  dcat:keyword "terminals" ;
  dcat:theme service-product:infrastructure ;
  srim:servicedBy :InfrastructureGeoSPARQLService
  srim:credits "This work was funded by the National Aeronautics and Space
Administration under the Aviation Operations and Safety Program.";
  extent:geographicBoundingBox [
        a extent:GeographicBoundingBox;
        extent:northBoundLatitude 51.29148;
        extent:southBoundLatitude 49.6959;
        extent:westBoundLongitude -77.53714;
        extent:eastBoundLongitude -74.96695
  ];
  dcat:distribution [
      rdf:type dcat:Distribution ;
      dcterms:title "RDF Distribution of Airport Instances" ;
      dcat:downloadURL <https://data.nasa.gov/ontologies/atmonto/airportInst.rdf> ;
      dcat:mediaType "application/rdf+xml" ;
    ] ;
  dcat:distribution [
      rdf:type dcat:Distribution ;
      dcterms:title "TTL Distribution of Airport Instances" ;
      dcat:downloadURL <https://data.nasa.gov/ontologies/atmonto/airportInst.ttl> ;
      dcat:mediaType "text/turtle" ;
    ] ;
.
```

A.2 NAS Facilities dataset metadata

The following example describes the metadata about a dataset describing NAS Facilities, which is serviced by a GEOSPARQL service endpoint and two distribution downloads in RDF/XML and Turtle format.

```
<https://data.nasa.gov/ontologies/atmonto/NASinst>
  rdf:type dcat:Dataset ;
  dcterms:conformsTo <https://data.nasa.gov/ontologies/atmonto/NAS> ;
  dcterms:description "US Air Traffic Control Command Center (ATCCC) and Air Route
Traffic Control Center (ARTCC) en-route facility instances" ;
  dcterms:spatial <http://www.geonames.org/6252001/united-states.html> ;
  dcterms:subject nas:NASfacility ;
  dcterms:title "NAS Facility Instance" ;
  dct:publisher <http://www.faa.gov>,
  dcat:keyword "US" ;
  dcat:keyword "United States" ;
  dcat:keyword "en-route" ;
  dcat:keyword "facility" ;
  dcat:keyword "infrastructure" ;
  dcat:theme service-product:infrastructure ;
  srim:servicedBy :InfrastructureGeoSPARQLService
  extent:geographicBoundingBox [
        a extent:GeographicBoundingBox;
        extent:northBoundLatitude 51.29148;
        extent:southBoundLatitude 49.6959;
        extent:westBoundLongitude -77.53714;
        extent:eastBoundLongitude -74.96695
  ];
  dcat:distribution [
      rdf:type dcat:Distribution ;
      dcterms:title "RDF Distribution of NAS Facility Instances" ;
      dcat:downloadURL <https://data.nasa.gov/ontologies/atmonto/NASinst.rdf> ;
      dcat:mediaType "application/rdf+xml" ;
    ] ;
  dcat:distribution [
      rdf:type dcat:Distribution ;
      dcterms:title "TTL Distribution of NAS Facility Instances" ;
      dcat:downloadURL <https://data.nasa.gov/ontologies/atmonto/NASinst.ttl> ;
      dcat:mediaType "text/ttl" ;
  ] ;
.
```

A.3 GeoSPARQL Service instance example

The following example describes an instance of GeoSPARQL Service

```
:InfrastructureGeoSPARQLService
    a srim:Service;
    dcterms:title "GeoSPARQL Service for Aviation Infrastructure datasets" ;
    dcterms:description "GeoSPARQL Service for Aviation Infrastructure datasets" ;
    dcterms:spatial <http://www.geonames.org/6252001/united-states.html> ;
    dcterms:subject nas:NASfacility, nas:Airport, nas:InternationalAirport,
nas:CanadianAirport, nas:CONUSairport ;
    dct:publisher <http://www.faa.gov>,
    dcat:keyword "US" ;
    dcat:keyword "United States" ;
    dcat:keyword "en-route" ;
    dcat:keyword "facility" ;
    dcat:keyword "infrastructure" ;
    dcat:theme service-product:infrastructure ;
    srim:operatesOn <https://data.nasa.gov/ontologies/atmonto/NASinst>,
                    <https://data.nasa.gov/ontologies/atmonto/airportInst>;
    dcterms:conformsTo <https://www.opengis.org/standards/GeoSPARQL> ;
    dcat:accessURL <http://www.faa.gov/services/infrastructure/sparql>.
```

A.4 Example of Airport description using ATM NASA Ontology

The following example describes the Dulles Airport using the ATM NASA Ontology.

```
@prefix atm: <https://data.nasa.gov/ontologies/atmonto/ATM#> .
@prefix data: <https://data.nasa.gov/ontologies/atmonto/data#> .
@prefix doc: <https://data.nasa.gov/ontologies/atmonto/documentation#> .
@prefix eqp: <https://data.nasa.gov/ontologies/atmonto/equipment#> .
@prefix gen: <https://data.nasa.gov/ontologies/atmonto/general#> .
@prefix nas: <https://data.nasa.gov/ontologies/atmonto/NAS#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix xml: <http://www.w3.org/XML/1998/namespace> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

nas:KIADairport a nas:CONUSairport ;
    rdfs:label "KIAD Airport"^^xsd:string ;
    nas:airportLocation nas:KIADcoordinates ;
    nas:airportName "WASHINGTON DULLES INTL"^^xsd:string ;
    nas:faaAirportCode "IAD"^^xsd:string ;
    nas:hasRunway nas:KIADphysicalRunway01C_19C,
        nas:KIADphysicalRunway01L_19R,
        nas:KIADphysicalRunway01R_19L,
        nas:KIADphysicalRunway12_30 ;
    nas:hoursOffsetFromUTC -5 ;
    nas:iataAirportCode "IAD"^^xsd:string ;
    nas:icaoAirportCode "KIAD"^^xsd:string ;
    nas:locatedInState "VA"^^xsd:string ;
    nas:withinARTCC nas:ZDCcenter ;
    nas:withinTimezone "America/New_York"^^xsd:string .
```

```
nas:KIADphysicalRunway01C_19C a nas:KIADphysicalRunway ;
    rdfs:label "Airport KIAD Runway 01C/19C"^^xsd:string ;
    nas:associatedAirport nas:KIADairport ;
    nas:associatedOpRunway nas:KIADoperationalRunway01C,
        nas:KIADoperationalRunway19C ;
    nas:runwayID "01C/19C"^^xsd:string ;
    nas:runwayLengthInFeet "11500"^^xsd:string ;
    nas:runwayWidthInFeet "150"^^xsd:string .

nas:KIADphysicalRunway01L_19R a nas:KIADphysicalRunway ;
    rdfs:label "Airport KIAD Runway 01L/19R"^^xsd:string ;
    nas:associatedAirport nas:KIADairport ;
    nas:associatedOpRunway nas:KIADoperationalRunway01L,
        nas:KIADoperationalRunway19R ;
    nas:runwayID "01L/19R"^^xsd:string ;
    nas:runwayLengthInFeet "9400"^^xsd:string ;
    nas:runwayWidthInFeet "150"^^xsd:string .

nas:KIADphysicalRunway01R_19L a nas:KIADphysicalRunway ;
    rdfs:label "Airport KIAD Runway 01R/19L"^^xsd:string ;
    nas:associatedAirport nas:KIADairport ;
    nas:associatedOpRunway nas:KIADoperationalRunway01R,
        nas:KIADoperationalRunway19L ;
    nas:runwayID "01R/19L"^^xsd:string ;
    nas:runwayLengthInFeet "11500"^^xsd:string ;
    nas:runwayWidthInFeet "150"^^xsd:string .

nas:KIADphysicalRunway12_30 a nas:KIADphysicalRunway ;
    rdfs:label "Airport KIAD Runway 12/30"^^xsd:string ;
    nas:associatedAirport nas:KIADairport ;
    nas:associatedOpRunway nas:KIADoperationalRunway12,
        nas:KIADoperationalRunway30 ;
    nas:runwayID "12/30"^^xsd:string ;
    nas:runwayLengthInFeet "10501"^^xsd:string ;
    nas:runwayWidthInFeet "150"^^xsd:string .


nas:KIADcoordinates a gen:PointLocation ;
    rdfs:label "KIAD coordinates"^^xsd:string ;
    gen:altitude "313"^^xsd:float ;
    gen:latitude "38.947444"^^xsd:float ;
    gen:longitude "-77.459944"^^xsd:float .

nas:KIADoperationalRunway01C a nas:KIADoperationalRunway ;
    rdfs:label "Airport KIAD Runway 01C"^^xsd:string ;
    nas:associatedAirport nas:KIADairport ;
    nas:runwayID "01C"^^xsd:string ;
    nas:touchdownPoint nas:KIADrunway01Ccoordinates .

nas:KIADoperationalRunway01L a nas:KIADoperationalRunway ;
```

```
    rdfs:label "Airport KIAD Runway 01L"^^xsd:string ;
    nas:associatedAirport nas:KIADairport ;
    nas:runwayID "01L"^^xsd:string ;
    nas:touchdownPoint nas:KIADrunway01Lcoordinates .

nas:KIADoperationalRunway01R a nas:KIADoperationalRunway ;
    rdfs:label "Airport KIAD Runway 01R"^^xsd:string ;
    nas:associatedAirport nas:KIADairport ;
    nas:runwayID "01R"^^xsd:string ;
    nas:touchdownPoint nas:KIADrunway01Rcoordinates .

nas:KIADoperationalRunway12 a nas:KIADoperationalRunway ;
    rdfs:label "Airport KIAD Runway 12"^^xsd:string ;
    nas:associatedAirport nas:KIADairport ;
    nas:runwayID "12"^^xsd:string ;
    nas:touchdownPoint nas:KIADrunway12coordinates .

nas:KIADoperationalRunway19C a nas:KIADoperationalRunway ;
    rdfs:label "Airport KIAD Runway 19C"^^xsd:string ;
    nas:associatedAirport nas:KIADairport ;
    nas:runwayID "19C"^^xsd:string ;
    nas:touchdownPoint nas:KIADrunway19Ccoordinates .

nas:KIADoperationalRunway19L a nas:KIADoperationalRunway ;
    rdfs:label "Airport KIAD Runway 19L"^^xsd:string ;
    nas:associatedAirport nas:KIADairport ;
    nas:runwayID "19L"^^xsd:string ;
    nas:touchdownPoint nas:KIADrunway19Lcoordinates .

nas:KIADoperationalRunway19R a nas:KIADoperationalRunway ;
    rdfs:label "Airport KIAD Runway 19R"^^xsd:string ;
    nas:associatedAirport nas:KIADairport ;
    nas:runwayID "19R"^^xsd:string ;
    nas:touchdownPoint nas:KIADrunway19Rcoordinates .

nas:KIADoperationalRunway30 a nas:KIADoperationalRunway ;
    rdfs:label "Airport KIAD Runway 30"^^xsd:string ;
    nas:associatedAirport nas:KIADairport ;
    nas:runwayID "30"^^xsd:string ;
    nas:touchdownPoint nas:KIADrunway30coordinates .

nas:KIADrunway01Ccoordinates a gen:PointLocation ;
    rdfs:label "KIAD runway 01C coordinates"^^xsd:string ;
    gen:altitude "286"^^xsd:float ;
    gen:latitude "38.939066"^^xsd:float ;
    gen:longitude "-77.459780"^^xsd:float .

nas:KIADrunway01Lcoordinates a gen:PointLocation ;
    rdfs:label "KIAD runway 01L coordinates"^^xsd:string ;
    gen:altitude "286"^^xsd:float ;
    gen:latitude "38.944967"^^xsd:float ;
```

```
    gen:longitude "-77.474810"^^xsd:float .

nas:KIADrunway01Rcoordinates a gen:PointLocation ;
    rdfs:label "KIAD runway 01R coordinates"^^xsd:string ;
    gen:altitude "312"^^xsd:float ;
    gen:latitude "38.923757"^^xsd:float ;
    gen:longitude "-77.436451"^^xsd:float .

nas:KIADrunway12coordinates a gen:PointLocation ;
    rdfs:label "KIAD runway 12 coordinates"^^xsd:string ;
    gen:altitude "310"^^xsd:float ;
    gen:latitude "38.943772"^^xsd:float ;
    gen:longitude "-77.490444"^^xsd:float .

nas:KIADrunway19Ccoordinates a gen:PointLocation ;
    rdfs:label "KIAD runway 19C coordinates"^^xsd:string ;
    gen:altitude "269"^^xsd:float ;
    gen:latitude "38.970639"^^xsd:float ;
    gen:longitude "-77.459324"^^xsd:float .

nas:KIADrunway19Lcoordinates a gen:PointLocation ;
    rdfs:label "KIAD runway 19L coordinates"^^xsd:string ;
    gen:altitude "293"^^xsd:float ;
    gen:latitude "38.955329"^^xsd:float ;
    gen:longitude "-77.435979"^^xsd:float .

nas:KIADrunway19Rcoordinates a gen:PointLocation ;
    rdfs:label "KIAD runway 19R coordinates"^^xsd:string ;
    gen:altitude "269"^^xsd:float ;
    gen:latitude "38.970773"^^xsd:float ;
    gen:longitude "-77.474440"^^xsd:float .

nas:KIADrunway30coordinates a gen:PointLocation ;
    rdfs:label "KIAD runway 30 coordinates"^^xsd:string ;
    gen:altitude "288"^^xsd:float ;
    gen:latitude "38.933610"^^xsd:float ;
    gen:longitude "-77.455898"^^xsd:float .
```

A.5 Example of Flight description using ATM NASA Ontology

The following example describes the details of a FLight using the ATM ontology.

```
atm:CPZ5839-201407150054 a atm:Flight ;
    rdfs:label "Flight CPZ5839 on 2014-07-15 00:54:00"^^xsd:string ;
    atm:actualArrivalDay nas:Day20140715 ;
    atm:actualArrivalTime "2014-07-15T04:21:00"^^xsd:dateTime ;
    atm:actualDepartureDay nas:Day20140715 ;
    atm:actualDepartureTime "2014-07-15T00:54:00"^^xsd:dateTime ;
    atm:aircraftTypeFlown eqp:E170AircraftType ;
    atm:arrivalAirport nas:KIAHairport ;
    atm:callSign "CPZ5839"^^xsd:string ;
    atm:departureAirport nas:KLGAairport ;
    atm:hasActualRoute atm:ActualRouteCPZ5839-201407150054 ;
    atm:hasPlannedRoute atm:PlannedRouteCPZ5839-201407150054 ;
    atm:operatedBy nas:CPZairline ;
    atm:userCategory "COMMERCIAL"^^xsd:string .
```

A.6 Example of Flight description using ATM NASA Ontology

The following example describes the details of a FLight using the ATM ontology.

```
atm:CPZ5839-201407150054 a atm:Flight ;
    rdfs:label "Flight CPZ5839 on 2014-07-15 00:54:00"^^xsd:string ;
    atm:actualArrivalDay nas:Day20140715 ;
    atm:actualArrivalTime "2014-07-15T04:21:00"^^xsd:dateTime ;
    atm:actualDepartureDay nas:Day20140715 ;
    atm:actualDepartureTime "2014-07-15T00:54:00"^^xsd:dateTime ;
    atm:aircraftTypeFlown eqp:E170AircraftType ;
    atm:arrivalAirport nas:KIAHairport ;
    atm:callSign "CPZ5839"^^xsd:string ;
    atm:departureAirport nas:KLGAairport ;
    atm:hasActualRoute atm:ActualRouteCPZ5839-201407150054 ;
    atm:hasPlannedRoute atm:PlannedRouteCPZ5839-201407150054 ;
    atm:operatedBy nas:CPZairline ;
    atm:userCategory "COMMERCIAL"^^xsd:string .
```

A.6 Example of Hourly Airport Condition using ATM NASA Ontology

The following describes Hourly Airport Condition in JFK Airport in NYC.

```
data:KJFKairportData20140729000000 a data:AirportData ;
    rdfs:label "Airport Data for KJFK @00:00:00Z on 20140729"^^xsd:string ;
    data:airportArrivalRate 44 ;
    data:airportDepartureRate 42 ;
    data:arrivalDemand 75 ;
    data:aspmFlightRules "V"^^xsd:string ;
    data:dataIntervalEndDay nas:Day20140729 ;
    data:dataIntervalEndTime "2014-07-29T01:00:00"^^xsd:datetime ;
    data:dataIntervalStartDay nas:Day20140729 ;
    data:dataIntervalStartTime "2014-07-29T00:00:00"^^xsd:datetime ;
    data:departureDemand 70 ;
```

```
    data:edctArrivalHold "449.0"^^xsd:float ;
    data:edctDepartureHold "0.0"^^xsd:float ;
    data:etmsArrivals 34 ;
    data:etmsDepartures 34 ;
    data:hasASPMmetCondition data:KJFKaspmHourlyMetCond20140729000000 ;
    data:highWindWITIdaily "104"^^xsd:float ;
    data:highWindWITIhourly "33"^^xsd:float ;
    data:lowCeilingWITIdaily "0"^^xsd:float ;
    data:lowCeilingWITIhourly "0"^^xsd:float ;
    data:lowVisibilityWITIdaily "0"^^xsd:float ;
    data:lowVisibilityWITIhourly "0"^^xsd:float ;
    data:oagArrivalDelay 526 ;
    data:oagGateDepartureDelay 1464 ;
    data:scheduledArrivals 33 ;
    data:scheduledDepartures 41 ;
    data:totalAirborneDelay "379.0"^^xsd:float .


data:KJFKaspmHourlyMetCond20140729000000 a data:ASPMmeteorologicalCondition ;
    rdfs:label "ASPM Meteorological Condition for KJFK @00:00:00Z on
20140729"^^xsd:string ;
    data:dataIntervalEndDay nas:Day20140729 ;
    data:dataIntervalEndTime "2014-07-29T01:00:00"^^xsd:datetime ;
    data:dataIntervalStartDay nas:Day20140729 ;
    data:dataIntervalStartTime "2014-07-29T00:00:00"^^xsd:datetime ;
    data:hasSkyCondition data:KJFKaspmHourlySkyCond20140729000000 ;
    data:hasSurfaceWindCondition data:KJFKaspmHourlyWindCond20140729000000 ;
    data:hasVisibilityCondition data:KJFKaspmHourlyVisCond20140729000000 ;
    data:surfaceTemperature "78.0"^^xsd:float .


data:KJFKaspmHourlySkyCond20140729000000 a data:SkyCondition ;
    rdfs:label "ASPM Sky Condition for KJFK @00:00:00Z on 20140729"^^xsd:string ;
    data:ceiling 85 .

data:KJFKaspmHourlyWindCond20140729000000 a data:SurfaceWindCondition ;
    rdfs:label "ASPM Wind Condition for KJFK @00:00:00Z on 20140729"^^xsd:string ;
    data:surfaceWindDirectionStatus "fixed" ;
    data:surfaceWindSpeed "18.0"^^xsd:float ;
    data:windDirectionFixed "300.0"^^xsd:float .

data:KJFKaspmHourlyVisCond20140729000000 a data:VisibilityCondition ;
    rdfs:label "ASPM Visibility Condition for KJFK @00:00:00Z on 20140729"^^xsd:string
;
    data:limitedVisibilityDistance "10.0"^^xsd:float .
```

Appendix B: Revision History

Table 2. Revision History

Date	Editor	Release	Primary clauses modified	Descriptions
May 05, 2018	S. Fellah	.1	all	initial version
August 08, 2018	S. Fellah	.2	all	Aviation Model review
August 15, 2018	S. Fellah	.3	all	Semantic Enablement Approaches
August 15, 2018	S. Fellah	.4	all	Controlled Vocabularies section
Sept 26, 2018	S. Fellah	.5	all	SRIM Model
Oct 26, 2018	S. Fellah	.6	all	All
Nov. 23, 2018	S. Fellah	.7	all	Integrated edits from Gobe.
Nov. 26, 2018	S. Fellah	.8	all	Clean up bibTex and references

Appendix C: Bibliography

1. EUROCONTROL, Federal Aviation Administration: AAIXM 5.1 Specification, http://aixm.aero/page/aixm-51-specification.
2. EUROCONTROL, Federal Aviation Administration: WXXM 2.0, http://wxxm.aero/news/wxxm-20.
3. Federal Aviation Administration: FIXM 4.1 Core, https://www.fixm.aero/fixm_410.pl.
4. Kaplun, M., Blomqvist, P., Uri, C., Haggstrom, N.: Service Description Conceptual Model (SDCM) 1.0. (2014).
5. Martin, D., Burstein, M., Mcdermott, D., Mcilraith, S., Paolucci, M., Sycara, K., Mcguinness, D.L., Sirin, E., Srinivasan, N.: OWL-S: Semantic Markup for Web Services - W3C Member Submission 22 November 2004. (2004).
6. Chinnici, R., Moreau, J.-J., Ryman, A., Weerawarana, S.: Web services description language (wsdl) version 2.0 part 1: Core language. W3C recommendation. 26, 19 (2007).
7. Balabanh, A.: OGC Testbed-12 Aviation Semantics Engineering Report. OGC 16-039,Open Geospatial Consortium, http://docs.opengeospatial.org/per/16-039.html (2016).
8. Chen, C.: OGC Testbed-13 Geospatial Taxonomies ER. OGC 17-036,Open Geospatial Consortium, http://docs.opengeospatial.org/per/17-036.html (2018).
9. FAA: NAS Service Registry and Repository, https://nsrr.faa.gov/, (2018).
10. Miles, A., Bechhofer, S.: SKOS simple knowledge organization system reference. W3C recommendation. 18, W3C (2009).
11. Fellah, S.: OGC Testbed-12 Semantic Portrayal, Registry and Mediation. OGC 16-059,Open Geospatial Consortium, http://docs.opengeospatial.org/per/16-059.html (2017).
12. Maali, F., Erickson, J., Archer, P.: Data catalog vocabulary (DCAT). W3C Recommendation. 16, (2014).
13. GeoDCAT-AP v1.0, https://joinup.ec.europa.eu/release/geodcat-ap-v10 , (2016).
14. Ciccarese, P., Soiland-Reyes, S., Belhajjame, K., Gray, A.J.G., Goble, C., Clark, T.: PAV ontology: provenance, authoring and versioning. Journal of biomedical semantics. 4, 37 (2013).
15. Group, O.W.L.W., others: OWL 2 Web Ontology Language Document Overview: W3C Recommendation 27 October 2009. W3C recommendation. (2009).
16. Brickley, D.: RDF vocabulary description language 1.0: RDF schema. http://www.w3.org/TR/rdf-schema/. (2004).
17. Sancho, P.F., Kaplun, M., Uri, C., Zhy, W., Pola, T., Schrempf, O.: Service Description Conceptual Model (SDCM), Version 2.0. (2016).
18. Kaplun, M.: FAA Web Service Description Ontological Model (WSDOM) - an Introduction. (2015).
19. FAA: FAA Semantics.aero, https://semantics.aero/ , (2018).
20. FAA: Registry Integration Module, http://swim.aero/rim/, (2018).
21. Project Open Data, https://project-open-data.cio.gov/ , (2016).
22. Fellah, S.: OGC Testbed-11 Implementing Linked Data and Semantically Enabling OGC Services Engineering Report . OGC 15-054,Open Geospatial Consortium, https://portal.opengeospatial.org/

files/?artifact_id=64405 (2018).

23. Fellah, S.: OGC Testbed 10 Virtual Global Gazetteer Engineering Report . OGC 14-029r2,Open Geospatial Consortium, https://portal.opengeospatial.org/files/?artifact_id=61108 (2015).
24. Kopeckỳ, J., Vitvar, T., Bournez, C., Farrell, J.: Sawsdl: Semantic annotations for wsdl and xml schema. IEEE Internet Computing. 60–67 (2007).
25. Neiswender, C.: What is a Controlled Vocabulary? In: In The MMI Guides: Navigating the World of Marine Metadata. Marine Metadata Interoperability, http://uop.whoi.edu/techdocs/presentations/MMI_Guides.pdf (2009).
26. Gruber, T.R.: A translation approach to portable ontology specifications. Knowledge acquisition. 5, 199–220 (1993).
27. Stuckenschmidt, H., Klein, M.: Structure-based partitioning of large concept hierarchies. In: International semantic web conference. pp. 289–303. Springer (2004).
28. Stuckenschmidt, H., Parent, C., Spaccapietra, S.: Modular ontologies: concepts, theories and techniques for knowledge modularization. Springer (2009).