OGC Testbed-14

SWIM Information Registry Engineering Report

Table of Contents

Publication Date: 2019-02-11

Approval Date: 2018-12-13

Submission Date: 2018-11-13

Reference number of this document: OGC 18-022r1

Reference URL for this document: http://www.opengis.net/doc/PER/t14-D001

Category: Public Engineering Report

Editor: Yann Le Franc

Title: OGC Testbed-14: SWIM Information Registry 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 findings and recommendations for building an information registry working together with the existing Federal Aviation Administration (FAA) System Wide Information Management (SWIM) aviation service registries, the National Airspace System Service Registry and Repository (NSRR). This information registry should allow the different Air Traffic Management (ATM) stakeholders to retrieve the appropriate service registered in the NSRR using the semantic representation of real-life entities represented by the data served by the services (e.g. "estimated departure time", "estimated time of arrival", ”runway true bearing”…). To support the integration of this domain-specific information, the ER proposes different strategies based on the semantic annotation proposal made in OGC 08-167r2 [1] extended with a recent World Wide Web Consortium (W3C) recommendation, the Web Annotation data model [1: https://www.w3.org/TR/annotation-model/]. In particular, the ER focuses on a solution using the W3C web annotation data model which adds semantics to the NSRR without changing the content of the database. This solution provides a low-cost, flexible and efficient alternative to add domain-specific semantics to NSRR content. The ER concludes with remarks on the elements necessary for implementing the information registry as a web annotation store as well as the necessity to build domain-specific knowledge models to support further interoperability and further service discoverability and the added-values of using the Data Catalog (DCAT) or Semantic Registry Information Model (SRIM) to better describe and retrieve ATM services.

1.1. Requirements & Research Motivation

SWIM is an interoperability framework for publishing, exchanging and manipulating ATM data based on Service Oriented Architecture (SOA) principles. One of the key elements of SOA is the service registry which provides a central access point to distributed and heterogeneous web services publishing ATM data. FAA developed such a registry, the NSRR, to provide access to the existing US ATM data service for the various ATM stakeholders. However, the discoverability of the services is limited to service-centric information. Therefore, finding services publishing specific ATM data such as airport runway information, *en-route* flight localization, estimated time of arrival, airport security measures, alternative arrival airport, weather information from a specific airport is cumbersome. This discoverability issue is due to the lack of accessibility to the domain-specific semantics used by the various experts for their search. This information is either buried within the XML schemas, documents such as a Web Service Deployment Descriptor (WSDD) or other documents hidden behind the service Application Programming Interface (API). The aim of this ER is to investigate possible solutions for the development of an "information registry" to extend the discoverability capabilities of NSRR with the following requirements:

- the information registry should be a searchable catalog of information about the data served by the various web services.
- it should ease the discoverability process for agents (both humans and machines) by providing access to domain-specific semantics hidden within the various elements of the service description.
- link service descriptions in NSRR with domain semantics from external semantic models (i.e. controlled vocabularies, thesauri and formal ontologies).

This ER proposes a semantic annotation approach to associate service records in the NSRR with domain-specific semantics from existing external ATM semantic models. The ER does not discuss how to enable semantic information and build domain-specific models but rather to integrate semantic information with the existing descriptions. These two topics are covered in detail in the companion deliverable OGC Testbed-14 Semantically Enabled Aviation Data Models Engineering Report (OGC 18-035) [2].

1.2. Prior-After Comparison

Semantic enablement and enrichment of ATM data has been the focus of several ERs and Components in Testbeds 12 and 13. This work focused on the semantic modeling of service description, the integration of semantic geospatial information (GeoSPARQL) with the service description (OGC 16-039r2) [3], the development of semantic based business rules (SBVR, (OGC 16-061) [4]), semantic models and requirements for semantic service registries (SRIM) (OGC 16-046) [5] as well as a prototype implementation of these models (OGC 16-059) [6] and the DCAT/SRIM extension with a prototype of a semantic registry (OGC 17-040) [7]. Despite all these efforts, little has been done on the definition of ATM specific semantic models and how to associate this domain-specific information with service registries content to enrich the user experience and enhance the quality and ease to retrieve services. In this ER, we are proposing a unique solution to integrate both domain-specific semantics and geospatial semantics together with the content of NSRR without changing both the records and the database.

1.3. Recommendations for Future Work

The following future tasks are recommended:

1.3.1. Recommended Future Tasks

- Develop domain-specific ontologies as common reference.
- Transformation of the NSRR repository to become Linked Data conformant
- Transform the NSRR into a semantic registry
- Use DCAT-SRIM for structuring NSRR content
- Enforce the use of DCAT-SRIM and existing ontologies by service providers
- Develop an ATM Vocabulary registry to publish the various ATM semantic resources.

1.3.2. Recommended Future Deliverables

Recommended Future Components

- Implementation of an annotation-based Information Registry for NSRR.
- Creation of an ATM ontology repository and definition of the requirements, design and implementation of an API for controlled vocabulary services that manage ontologies and taxonomies.

Recommended Future Engineering Reports (ER)

- Design of modular domain-specific ontologies covering at least aeronautical, flight and weather information.
- Gap analysis to accommodate the specificities of the SWIM data model (SDCM) within the context of the SRIM model [8].

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

This ER describes a simple and efficient solution to associate a structured ATM service description with domain-specific semantics using the W3C Web Annotation data model. This work falls in the scope of both the Geosemantic Domain Working Group (DWG) for the semantic aspect and the Aviation DWG for the thematic aspects. As the solution presented in the ER can be used to integrate semantics (including geospatial semantics) with various types of data coming for various domains of application (intelligence, weather,geology,...), it should be reviewed mainly by the GeoSemantics DWG.

1.5. Document contributor contact points

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

Contacts

Name	Organization
Yann Le Franc (ed.)	e-Science Data Factory
Stephane Fellah	ImageMatters, LLC

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

The following normative documents are referenced in this document.

NOTE: Only normative standards are referenced here, e.g. OGC, ISO or other SDO standards. All other references are listed in the bibliography. Example:

- OGC: OGC 06-121r9, OGC® Web Services Common Standard [https://portal.opengeospatial.org/files/?artifact_id=38867&version=2]
- W3C: Web Annotation Data Model, 2017 [https://www.w3.org/TR/annotation-model]
- W3C: Data Catalog Vocabulary (DCAT), 2014 [https://www.w3.org/TR/vocab-dcat/]
- W3C: RDF Concepts and Abstract Syntax, 2014 [https://www.w3.org/TR/rdf11-concepts/]
- W3C: OWL 2 Web Ontology Language, 2012 [https://www.w3.org/TR/owl2-overview/]
- W3C: OWL-S: Semantic Markup for Web Services (OWL-S) Submission, 2004 [https://www.w3.org/Submission/OWL-S/]
- W3C: JSON-LD 1.0: A JSON-based Serialization for Linked Data, 2014 [https://www.w3.org/TR/json-ld/]
- W3C: SKOS Simple Knowledge Organization System Reference, 2009 [https://www.w3.org/TR/skos-reference/]

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.

- Concept

  ```
  A concept is an element of a semantic model. This specification makes no
  assumptions about the nature of concepts, except that they must be identifiable by
  URIs. A concept can for example be a classifier in some language, a predicate logic
  relation, the value of the property of an ontology instance, some object instance
  or set of related instances, an axiom, etc.
  SOURCE: https://www.w3.org/TR/sawsdl/
  ```

- Data

  ```
  Re-interpretable representation of information in a formalized manner suitable for
  communication, interpretation, or processing [NOTE Data can be processed by humans
  or by automatic means.]
  SOURCE: ISO/IEC 2382-1:1993, 01.01.02
  ```

- Information

  ```
  Knowledge concerning objects, such as facts, events, things, processes, or ideas,
  including concepts, that within a certain context has a particular meaning
  SOURCE: ISO/IEC 2382-1:1993, 01.01.01
  ```

- Information registry

  ```
  An enabling infrastructure that stores, catalogs and manages information describing
  services and the data types they serve.
  ```

- Interoperability

  ```
  Capability to communicate, execute programs, or transfer data among various
  functional units in a manner that requires the user to have little or no knowledge
  of the unique characteristics of those units.
  SOURCE: OGC 17-040
  ```

- Metadata

```
Data about data.
SOURCE OGC 17-040
```

- Ontology

```
A formal specification of concrete or abstract things, and the relationships among
them, in a prescribed domain of knowledge [ISO/IEC 19763]
SOURCE: OGC 17-040
```

- Service description

```
The information needed in order to use, or consider using, a service.
SOURCE OGC 16-039r2
```

- Service Description Conceptual Model

```
Graphical and lexical representation of the properties, structure, and
interrelationships of all service metadata elements.
SOURCE: OGC 16-024r2
```

- Semantic Model

```
A semantic model is a set of machine-interpretable representations used to model an
area of knowledge or some part of the world, including software. Examples of such
models are ontologies that embody some community agreement, logic-based
representations, etc. Depending upon the framework or language used for modelling,
different terminologies exist for denoting the building blocks of semantic models.
SOURCE: https://www.w3.org/TR/sawsdl/
```

- Service 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 the resources, as well as query capabilities.
SOURCE OGC 16-039r2
```

- Semantics

```
Semantics in the scope of this specification refers to sets of concepts identified
by annotations. SOURCE: https://www.w3.org/TR/sawsdl/
```

- Semantic Annotation

```
A semantic annotation in a document is additional information that identifies or
defines a concept in a semantic model in order to describe part of that document.
SOURCE: https://www.w3.org/TR/sawsdl/
```

- Semantic Interoperability

```
The aspect of interoperability that assures that the content is understood in the
same way in both systems, including by those humans interacting with the systems in
a given context.
SOURCE: OGC 15-054
```

- Syntactic interoperability

```
The aspect of interoperability that assures that there is technical connection,
i.e. that the data can be transferred between systems.
SOURCE: OGC 15-054
```

- System Wide Information Management

```
Global Air Traffic Management (ATM) industry initiative to harmonize the exchange
of Aeronautical, Weather and Flight information for all Airspace Users and
Stakeholders.
SOURCE: https://en.wikipedia.org/wiki/System_Wide_Information_Management
```

3.1. Abbreviated terms

NOTE: The abbreviated terms clause gives a list of the abbreviated terms and the symbols necessary for understanding this document. All symbols should be listed in alphabetical order. Some more frequently used abbreviated terms are provided below as examples.

- AIXM The Aeronautical Information Exchange Model
- API Application Program Interface
- ATM Air Traffic Management
- DCAT Data Catalogue
- FAA Federal Aviation Administration
- FIXM The Flight Information Exchange Model
- GML Geography Markup Language
- ICAO International Civil Aviation Organization
- IR Information Registry
- NAS National Airspace System
- NSRR National Airspace System (NAS) Service Registry and Repository

- OGC Open Geospatial Consortium
- OWL Web Ontology Language
- OWL-S Ontology Web Language (OWL)-based Web Service Ontology
- RDF Resource Description Framework
- RIM Registry Integration Module
- SAWSDL Semantic Annotation for Web Service Description Language (WSDL)
- SCR SWIM Common Registry
- SDCM Service Description Conceptual Model
- SKOS Simple Knowledge Organization System
- SOA Service Oriented Architecture
- SPARQL SPARQL Protocol and RDF Query Language
- SRIM Semantic Registry Information Model
- SWIM System Wide Information Management
- W3C World Wide Web Consortium
- WSDD Web Service Description Document
- WSDL Web Service Description Language
- WSDOM Web Service Description Ontological Model
- WXXM The Weather Information Exchange Model
- XML eXtensible Markup Language

Chapter 4. Overview

Section 5 provides an introduction to the current state of SWIM service registries and standardized service description documents/models. It also provides an overview of the current challenges and issues, a user perspective and the ground information regarding semantic annotation and semantic resources for aviation.

Section 6 discusses in more detail the meaning of information in context of SWIM and provides an initial description of the different types of information, as well as how to relate the types both together and with the services. Based on this initial information model, the testbed participants investigated information gaps and biases in the current NSRR information model and by analyzing the information coverage of each of the information sources.

Section 7 describes different technical solutions to integrate the missing information using semantic resources.

Section 8 investigates a solution for extending NSRR search capabilities based on the World Wide Web Consortium (W3C) Web Annotation model. It presents the various annotation models for each of the information sources in the NSRR, a short description of the necessary architecture to implement this approach and propose various usage scenarios.

Section 9 provides a short summary and recommendations for future work.

Chapter 5. Background and Status Quo

SWIM is an interoperability framework to support data exchange and retrieval between distributed, loosely-coupled regional and local ATM services. This framework has been developed in the context of an international collaboration involving FAA and EUROCONTROL (SESAR) to support ATM data access and exchange across national and legal borders. Based on SOA principles, the interoperability framework provides standards for information exchange, infrastructure and governance for ATM data.

A key element of SOA is the service registry which provides a central point of access to the various services that are part of the SWIM framework. In this context, FAA developed the SWIM-enabled NSRR.

Based on Drupal, the NSRR provides the following features:

- A curated service registration process with lifecycle management
- A database of service description in standardized format based on the Service Description Conceptual Model (SDCM) data model
- A programmatic access to service descriptions via a Registry Integration Module (RIM), a dedicated web API [2: https://nsrr.faa.gov/sites/default/files/library/rim-spec-1.0.0-draft-20171020.pdf] [3: https://nsrr.faa.gov/sites/default/files/library/Testing%20the%20RIM%20Interface.pdf]
- A document repository storing the various documents describing the service (pdfs, xml schemas,...)
- A search interface to support service retrieval for human operators using hierarchical and faceted search strategies [9] or using document types.

The general architecture is shown in Figure 1

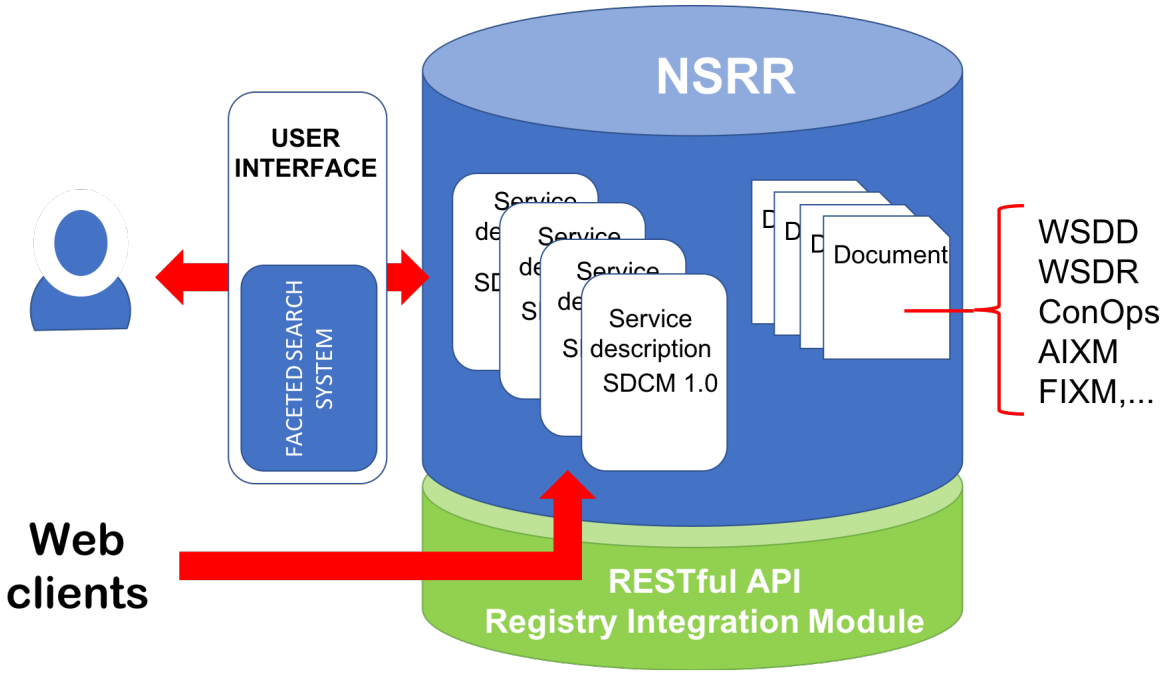


Figure 1. schematic representation of the NSRR service

Driven by the effort initiated by NextGen (the FAA program) and SESAR (the European program) several other efforts have emerged with various levels of maturity such as the Brazilian SWIM registry [4: http://clima.icea.gov.br/pesquisa/I-workshopcyberswim/downloads/I_WorkshopCyberSWIM_UNB_SWIM%20Registry.pdf] [10]. In order to foster interoperability between the increasing number of service registries, FAA defined a vision of SWIM Inter-Registry Framework [5: https://nsrr.faa.gov/nsrr-library-document/9040]. To support interoperability of distributed service registries, a common data model for describing service metadata i.e. information about the service, the SDCM data model has been proposed [6: http://swim.aero/sdcm/2.0.0/sdcm-2.0.0.html#OASIS-RO].

The SDCM data model provides a common structure and well-defined semantics for describing ATM services. The current version of the standard is SDCM version 2.0. SDCM2.0 is aligned with the Semantic Markup for Web Services (OWL-S), which is a W3C specification model for describing services shown in Figure 2. Note that although OWL-S was originally submitted to W3C in 2004, it has not progressed into a W3C Recommendation (i.e. W3C standard). OWL-S is still labelled as a W3C Submission (i.e. W3C candidate standard).

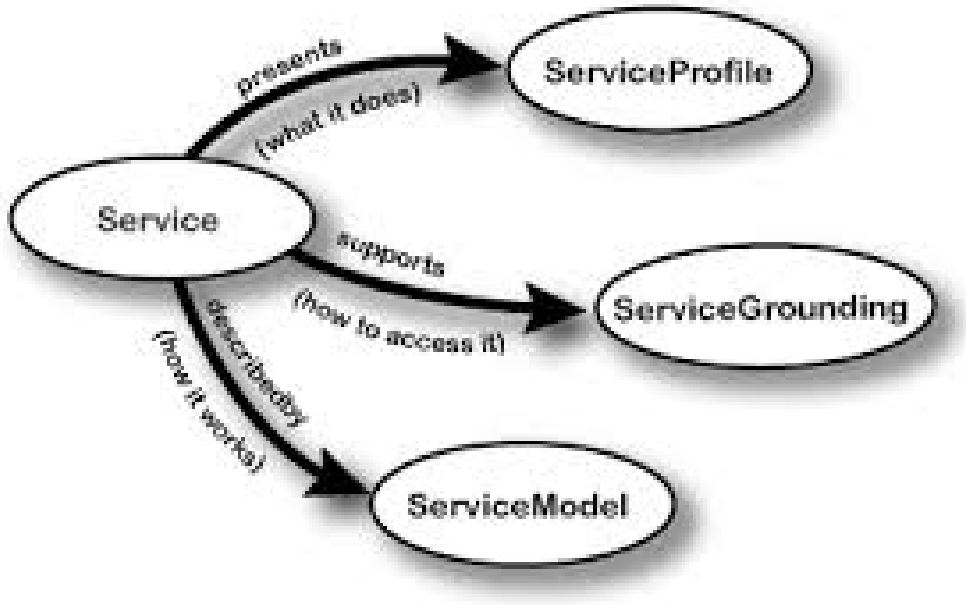


Figure 2. OWL-S data model

The OWL-S model provides a unique semantic to describe what the service does, how to access it and how it works.

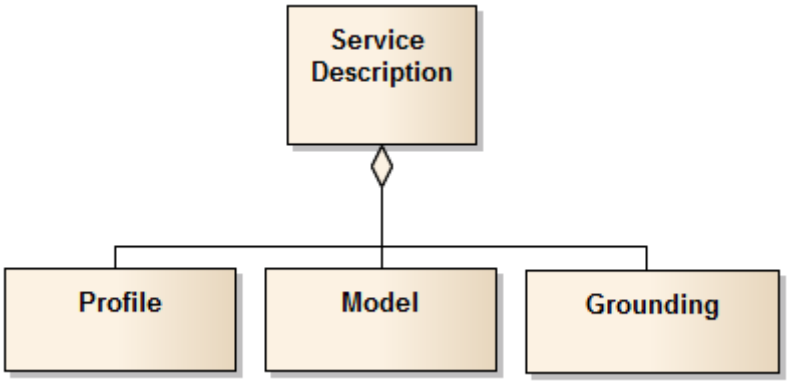


Figure 3. SDCM 2.0 Service Description

The SDCM Service Description is composed of 3 classes corresponding to the main OWL-S classes, as shown in Figure 3:

- Service Profile
- Service Model
- Service Grounding

Additional documents are associated with the standardized service descriptions. Text-documents such as WSDD and Web Service Description Requirements (WSDR) have been defined by FAA to support the registration process. These documents act as structured and documented text templates for providing information about the service to be registered. These documents are used by service providers to enter information into the fields corresponding to the SDCM elements. This textual service description could then be converted into a machine-processable description. The other additional sources of information contained in NSRR are the XML schema used by the different services. These models are used to provide syntactic and structural interoperability and therefore contain information regarding the data served by the service and how to structure the messages.

In addition to service-centric search capabilities, the NSRR has text-based search capabilities building upon the indexation of the additional text documents mentioned previously and tag-based search to categorize services as business products [9]. However, the NSRR has not been designed to leverage search using semantics and external semantic models. Our objective in this ER is to investigate how to use the existing available resources to extend service discovery by human and software agents with semantics. The next chapter discusses the different levels of information for describing a service and investigates existing gaps in the information model within the current NSRR implementation. Thereafter, in the following chapters, the ER investigates various integration models to integrate the missing information.

Chapter 6. Information Model and Coverage

This section defines a generic information model for describing ATM web services. Using this model, the coverage of the NSRR content is evaluated and the information gaps that prevent a more efficient discoverability are identified.

First, this section starts by defining what is meant by 'information'. In the context of this ER, we consider the ISO definition of data and information i.e. data is a re-interpretable formalized representation of information and information is knowledge concerning objects that within a certain context have a particular meaning.

These definitions can be contrasted with the more recent definitions provided by the Knowledge Modeling community [11, 12]. In this context, data is considered as a meaningless symbol (i.e. a string, a number, a map, an image,...). It can be transformed into information by adding syntactic, structural and semantic elements. The created information is used to provide answers to questions such as Who? What? Where? When? How? [11].

Although different, both sets of definitions underline the contribution of a formalized representation of semantics to provide the meaning of the data element/information.

6.1. An information model to describe ATM web services

In this work, a web service is considered to present 5 main types of information:

- The **administrative information** which addresses questions such as who is responsible for the service, what its lifecycle stage, what is the version of the service, which licenses,...
- The **technical information** which addresses questions about how to access the data and how the service works (processes)
- The **domain-specific information** which addresses questions related to the real-life entities associated with the served data or process. This type of information is composed of three subtypes:
  - The **data entity information** which addresses questions regarding the quality of the data, the acquisition methods and rates,... and provides also the semantic mapping with real-life entities.
  - The **data model information** which addresses questions regarding the type of data served by the service i.e. how the different data entities articulate together? This information describes the domain-specific semantic model (i.e. relations between the different data entities).
- The **pragmatic information** which addresses questions relating to the usage of the service, its purpose, the type of product,...

This simple information model can be used to categorize the different metadata elements available within NSRR service descriptions and documents. It can be either integrated directly within the SDCM model or used as additional external facets. The next section will use it to identify

information gaps and bias within the NSRR information model.

6.2. Information gap analysis

Based on this simple theoretical model, this ER now investigates where the different types of information can be found in NSRR. This analysis will allow us to identify gaps and biases in information coverage within the current NSRR information model.

The **administrative information** corresponds to the metadata or "data about data" describing the service and how to access it, published by service registries. This information can be accessed as structured information with a common semantic model (SDCM). However, the semantics are embedded within the service description and cannot be leveraged for inference or reasoning. Service descriptions are curated, stored and maintained by the Service Registry provider. This service-centric information is found in the Service Profile section of the SDCM 1.0 and 2.0 data model. It can be extracted from documents such as WSDD and WSDR used in NSRR. Some additional information regarding the data, i.e. the service record, is available as associated keywords within documents such as the OGC common-WS or schemata based on the Geography Markup Language (GML).

The **technical information** corresponds to the description of the different methods and protocols to access data, messaging system and structure, security handling, Quality-of-Service (QoS), ... This information can be found within the Service Descriptions provided by NSRR as instances of the classes "Service Interface" and "Service Implementation" of the SDCM 1.0 data model. In the SDCM 2.0 these classes correspond to the "Model" and "Grounding" classes, in accordance to the OWL-S data model. This information describes the various processes to get the data from the service and/or the various operations that can be performed by services on specific data (e.g. geospatial mapping, overlay, messaging,...).

The **domain-specific information** corresponds to the description of the data provided by the service i.e. what kind of real-world measurement is provided by the service. This information relies on the semantic representation of ATM concepts used by the different stakeholder. This information is not integrated directly within the SDCM model which covers the service-centric information. The following chapter investigates different solutions to associate this information with the service-centric information.

In the NSRR, the domain-specific information provided by the Service Provider is contained within the interchange data models i.e. the specific XML schemata for example the Aeronautical Information Exchange Model (AIXM), Flight Information Exchange Model (FIXM), and the Weather Information Exchange Model (WXXM). These schemata are used for syntactic and structural interoperability and the information they contain is stored in NSSR as downloadable documents, which prevents the access to the integrated information. Two main categories of information available in these documents can be identified:

1. **data entity information** describing the type of data served by the service i.e. the semantic of the data or the link to real-world entities (Estimated Time of Arrival, Flight plan, route, ...) as well as information regarding the syntax of data entities i.e. data type (string, int,...) and
2. **data model information** regarding the data itself such as which service are consuming and/or updating the data or the relation with other data entities.

These two types of information are related to the domain reference described in OGC 08-167r2 [1]. These domain references or domain vocabularies are semantic models within which complex relations can be expressed (i.e. using OWL-DL and n-ary relations) about the datum and the relations with other data. These models should be defined by the expert community and used as a semantic reference for the different stakeholders. As proposed in OGC 08-167r2 [1], these common models should be used as base elements for defining application specific models. Service providers should be responsible for providing such information to enhance the discoverability of their service.

Finally, the **pragmatic information** about the usage and the purpose of the data should describe in which case to use this service e.g. a service could have the activity of Flight Planning as its main purpose. This information is provided in the NSRR as a facet i.e. a categorization facet [9]. It should describe the particular business process for which the agent uses the service registry. By adding such a facet, it is possible for the user to find services for a specific business process.

In Table 1, the findings are summarized. This ER proposes association of the type of information with the information source in the NSRR, the coverage of the existing description (complete, partial or none), accessibility of the information (direct access, indirect access, hidden) and the responsible stakeholder.

Table 1. NSSR Information Coverage

Information type	**Source**	**Coverage**	**Accessibility**	**Responsible stakeholder**
Administrative information	SDCM representation – Service Profile WSDD document	Complete	Direct access (API and search interface)	NSRR
Technical information	SDCM representation – Service Model and Service Grounding	Complete	Direct and indirect access	NSRR
Data model information	AIXM, Web Feature Service (WFS), Web Map Service (WMS), FIXM, WXXM,... - Service developer	Partial	Indirect access and hidden	NSRR
Data entity information	AIXM, WFS, WMS, FIXM, WXXM,... - Service developer	Partial	Indirect access and hidden	Service provider
Pragmatic information	NSRR index	Partial	Direct access via the search interface	NSRR curators

As expected, the information currently available within the NSRR fully covers the functional information of web services i.e. administrative and technical information. The domain-specific information regarding the data entities and models served by the various services is currently within the database but is not accessible for extending the service discoverability. In the current

situation, users can efficiently retrieve services using queries based on the administrative and technical information such as follows:

- Which service provides gridded weather information?
- Which service provides data validation for the flight plan?
- What is the version of the service?
- Which validated service provides Notice-to-Airmen (NOTAM) data?
- Which service is provided by FAA?
- What is the lifecycle stage of a particular service?
- Which service provide Java Messaging Interface?

Despite the importance of this information for the usage of the service, access to data or operations on the data, the different stakeholders will first search for services matching their business need based on domain concepts representing real-world measurements or information. To illustrate this, the following potential queries formulated by service registry users are considered:

- Which service provides runway true bearing information from East Cost airports ?
- Which service provides wind measurement at Washington Dulles Airport (FAA airport code: IAD)?
- Which service provides aeronautical information for flight planning and uses JMS?

A quick analysis of the queries shows that the different of types of information defined in our initial information model are aggregated (domain-specific information, administrative and technical information) as shown in Figure 4'.

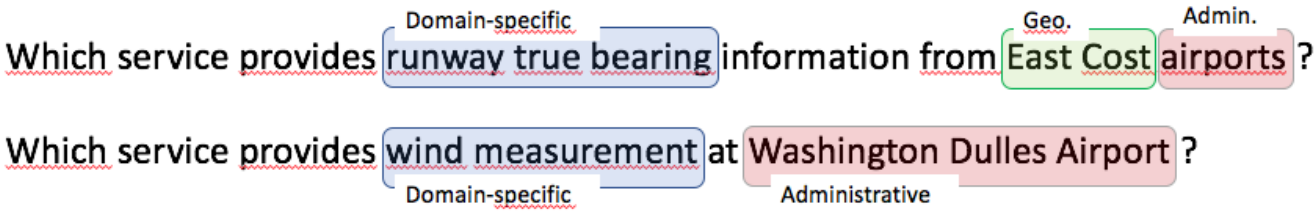


Figure 4. The different information types within queries

By extending and refining the various types of queries that a user would need in order to achieve some specific activity (e.g. pre-flight bulletin publication, flight planning, flight plan validation, flight plan update,...), it should be able to refine the information content of the registry.

The main information gap identified in the current registry is related to the thematic or domain-specific information. This information includes the semantics describing the specific domain information e.g. departure airport, alternate airport, runway true bearing,*en-route* flight information, weather information, ... but also a semantic model relating these different types of information together. This information relates to the domain expertise of the user to retrieve services offering the needed domain-specific information. The next chapter presents the different possible solutions to associate these different levels of information together to enhance the user experience by enabling queries such as the examples provided above.

Chapter 7. Integrating domain-specific information

The various documents and existing data models used to describe ATM web services provide contextual information about the service, as well as information about how to access and involve the service in retrieving ATM data. They provide an interoperability layer describing the operation and the format to encode the data i.e. covering only the functional dimension of the Web services. But they lack a methodology to describe the thematic dimension of the web services i.e. what the served data or process represent.

To support our research work, the fictitious Flight Planning Service provided by the NSRR was used to build our examples. This fictitious service has been designed for training and demonstration purposes. It is one of the services registered in the NSRR (see Figure 5).

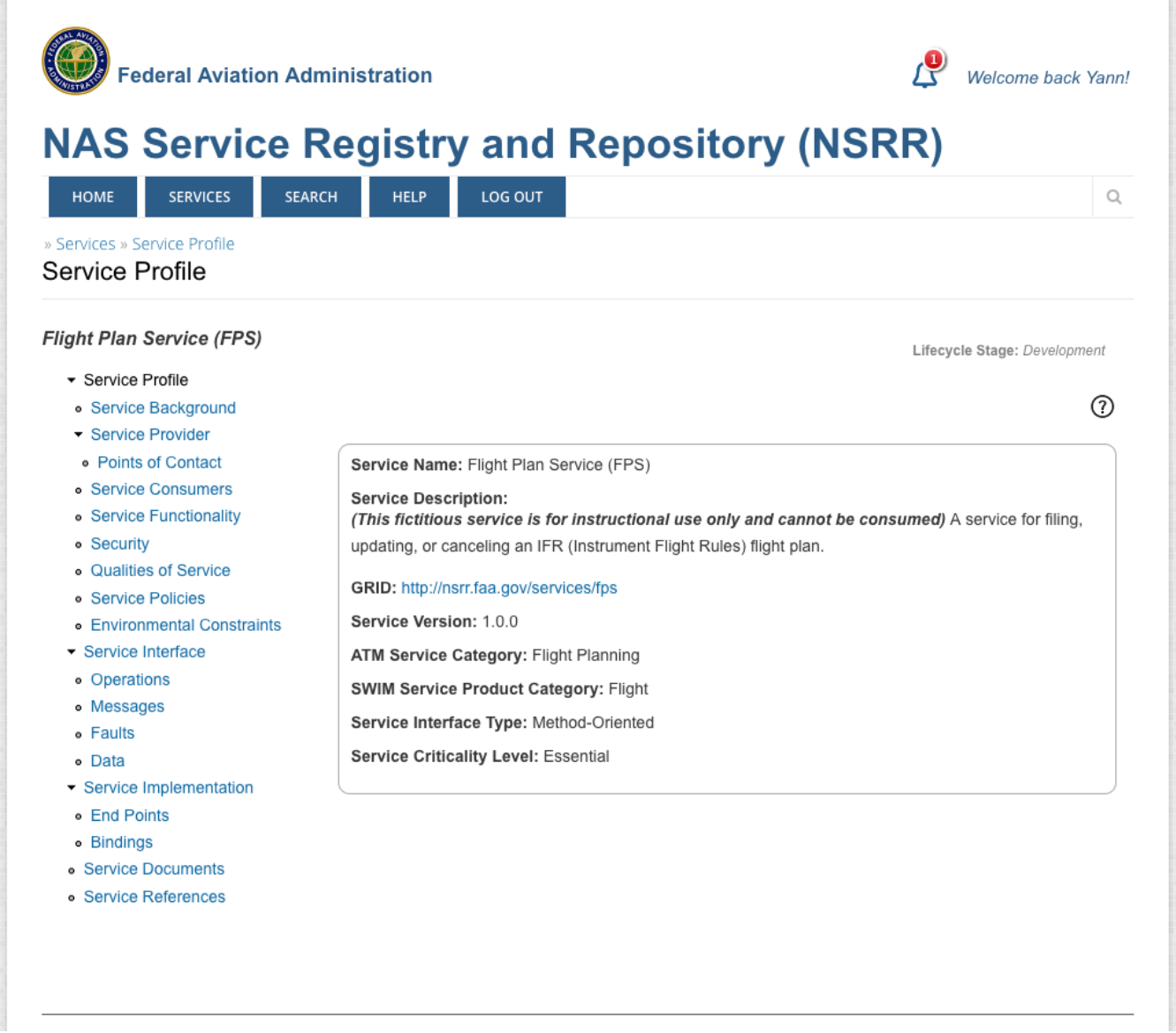


Figure 5. FPS Service Profile in NSRR

In the NSRR database, the service is registered and identified with a GRID : http://nsrr.faa.gov/services/fps. Service record information are identified by a specific Uniform Resource Identifier (URI) pointing to the service profile resource.

From this landing page, it is possible for a human agent to access a variety of information regarding the service, supported by the SDCM model briefly described in chapter 5 and more extensively described in (OGC 18-035) [2].

The FPS service provides several types of documents extending the initial service description as shown in Figure 6.

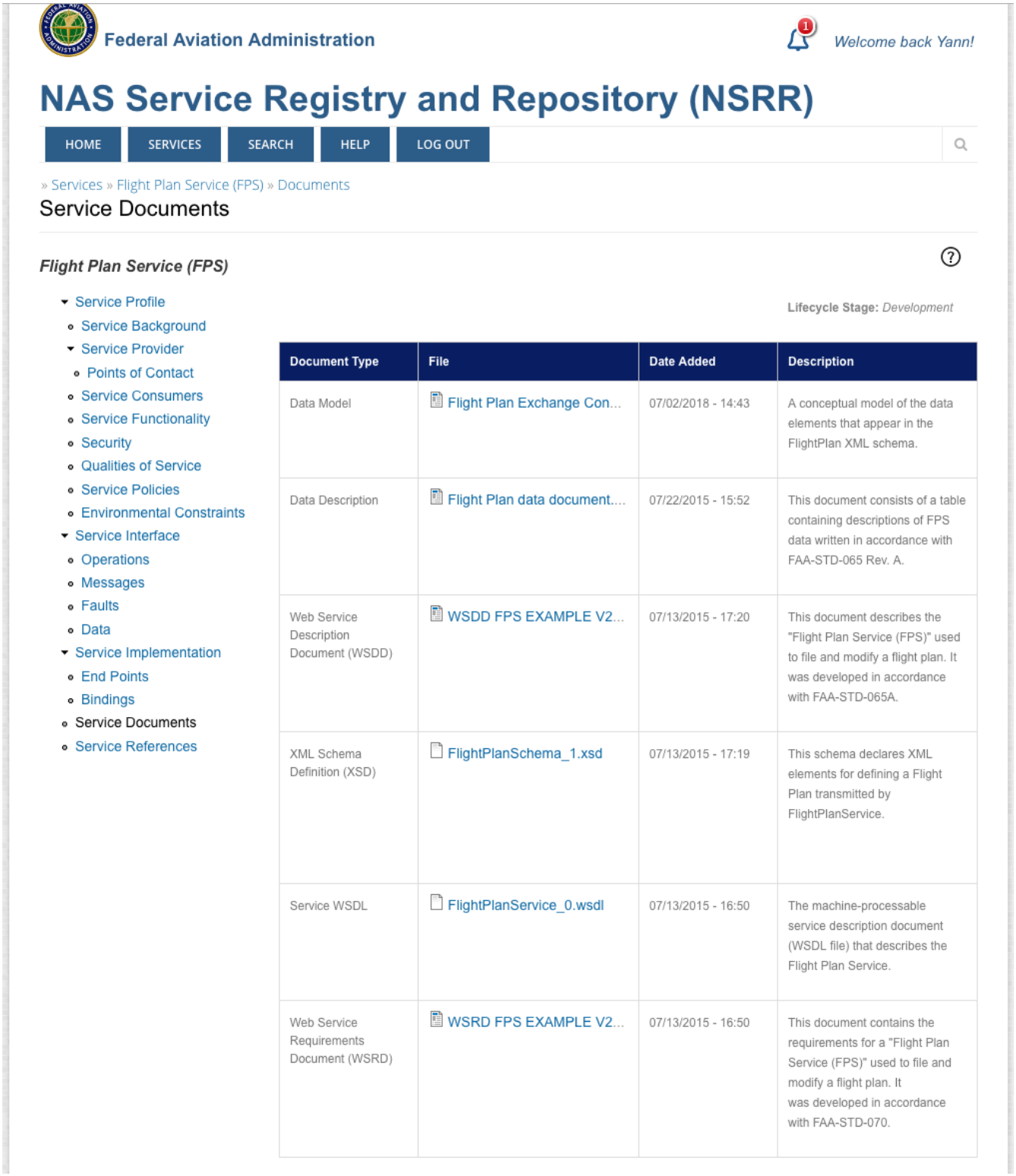


Figure 6. FPS service additional documentation

These documents include the FAA WSDD and WSDR documents, a word document including the UML diagram of the data model underlying the service, a word document containing a table describing a subset of the data elements provided by the service and 2 XML documents, the WSDL

description of the service and the XML schema defining the data elements provided by the service.

In addition to these sources of information, the FPS service profile can be accessed programmatically through the RIM API as shown by the Service Profile snippet provided below in section 7.2.1. The complete service description can be found in Annex A.

7.1. Using Semantic Annotations

7.1.1. General principles

To extend the existing descriptions with a thematic dimension which will be used by users to discover the services, it is necessary to consider how to link the service descriptions with external models. These external models comprise conceptualized knowledge about the represented ATM operations or information.

Linking the service description with these ATM knowledge models will allow reasoning algorithms to be able to infer if the web services match the user information need.

A proposal for such integration has already been made in the “Semantic Annotation in OGC standards” document (OGC 08-167r2) [1].

In this proposal, the authors have identified three levels of annotations for OGC compliant web services:

- Level 1: Service metadata contained in WFS, WMS, and Capabilities documents of other OGC Web Services (OWS)
- Level 2: Data model provided by GML-based schemata
- Level 3: Data entity

In order to link the different levels of information together with external Knowledge Models (i.e. thesauri, ontologies, controlled vocabularies, code lists, …), they are proposing different mechanisms of integration that we will be considering in the next sections.

In particular they introduce the concept of resource ontology as a means to describe the inter-relationship between the different elements specific to a particular service i.e. an application specific ontology describing the served data model but building upon common elements coming from global and accessible domain ontologies.

7.1.2. Why using thesauri, controlled vocabulary or ontologies?

Controlled vocabularies, thesauri, and ontologies are conceptual models of the real-world defined by humans in a machine processable format. These semantic models are used to build a common understanding between humans and machines and also reach consensus between domain experts. These models need to address the various semantic heterogeneities such as synonymy, homonymy and polysemy to allow machines to disambiguate the data, provide hierarchical information to organize the various concepts and support multi-lingual communities. Such resources can be built using W3C standards such as RDF, OWL or SKOS depending on the complexity of the model. For an overview of these standards and their usage refer to [OGC 18-035] [2]. These models enable

machines to make inferences on the data thereby retrieving information more relevant to humans. As we argued in the previous chapter, the domain-specific information is the missing element to achieve semantic interoperability and improve service discovery. The following section investigates the existing domain-specific semantic resource which could be used to extend the NSRR's Semantic interoperability.

7.1.3. Existing ATM semantic resources

In order to foster semantic interoperability of the service descriptions and of service categorization, the FAA has developed several semantic resources including various SKOS based thesauri describing generic types such as the type of SWIM Service products, the Service Availability Status, Service Interface types, Flight Phase, and various classifications of US ATM information (ICAO regions, US airways, US Flight Information Region and Airspace Classes). These semantic resources are available on the semantic aero website (https://semantics.aero/). In addition to these thesauri, FAA also developed different semantic resources to describe the registered services in NSRR i.e. the SWIM Controlled Vocabulary and Web Service Description Ontological Model (WSDOM). The latter corresponds to an OWL serialization of the SDCM data model. The current version of WSDOM is aligned with the SDCM1.0 data model. Efforts to update the WSDOM ontology to the new SDCM version are under way.

Unfortunately, these resources offer little coverage for domain-specific concepts such as airport runways, runway true bearing, and so on. Currently little effort has been dedicated to building domain-specific ontologies for ATM concepts except for the proof-of-concept ATM ontology proposed by NASA (NASA Atmonto, https://data.nasa.gov/ontologies/atmonto/index.html) and mainly based on concepts from AIXM. As previously discussed, these domain-specific ontologies will deeply contribute to the improvement of service discoverability in NSRR.

This ER focuses on usage of these resources to enrich the current NSRR content. Further details regarding these different resources are provided in [OGC 18-035] [2]. The following section proposes practical solutions for integrating domain-specific semantics with little modifications to the current NSRR registry.

7.2. Semantic Annotation and the SDCM model

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). The Service Profile mainly contains administrative information (provider's information, function, security, …) as shown in Figure 7.

The Service Model (Figure 8) and the Service Grounding classes provide technical information about the interfaces and the connection protocols to the service.

7.2.1. Integration within the Service Profile class

As in OWL-S, SDCM data model provides ways to associate the service description with categorization information defined in taxonomies (Figure 7).

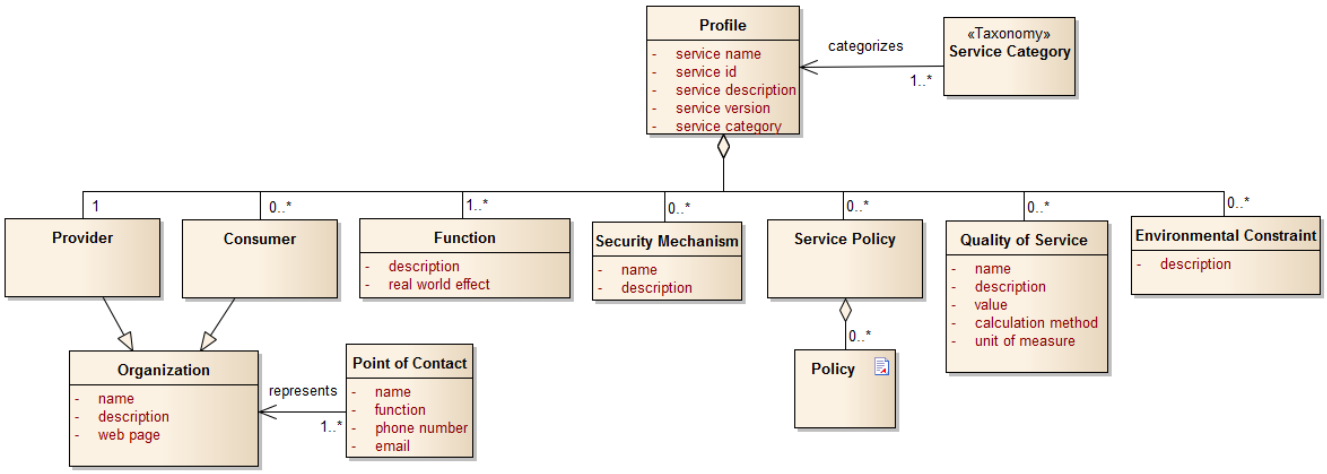


Figure 7. SDCM2.0 Service Profile

Domain-specific information describing the type of data served by the services could be integrated within the service profile class which provides generic information. For this purpose, the information could be provided within the “Service Category” attribute associated with the service profile. This attribute would thus contain references to concepts from external ontologies or thesauri developed by FAA and other ATM stakeholders. The reference to the model should be a resolvable URI allowing access to the full knowledge model and specific information about the concept within the context of the knowledge model i.e. subclass, superclass or relations.

In OWL-S, specific placeholders have been defined to add categorization facets to service description corresponding to generic facets or to product types. These placeholders are of two types:

- attributes of the profile i.e. in this context datatype properties or
- dedicated class named ServiceCategory.

Attribute of the profile are described in Table 2.

Table 2. Profile attribute in OWL-S (from OWL-S specification)

Attribute name	description
`serviceClassification`	defines a mapping from a Profile to an OWL ontology of services, such as an OWL specification of NAICS.
`serviceProduct`	defines a mapping from a Profile to an OWL ontology of products, such as an OWL specification of UNSPCS.

These properties are used to link the service description with external knowledge models (i.e. ontologies, taxonomies, controlled vocabularies, thesauri,...) describing the type of service and the type of products served by the service. The type of value for these properties are either XML Schema Definitions (XSD) links or URIs corresponding to instances of classes specified in OWL ontologies of services and products. These two properties are similar to serviceCategory described above, but they differ in that the values of the properties are OWL instances rather than strings referring to non-OWL business taxonomies.

The ServiceCategory class describes categories of services on the basis of external semantic models.

The ServiceCategory class has 4 different attributes described in Table 3.

Table 3. OWL-S ServiceCategory (from OWL-S specification)

Attribute name	Description
`categoryName`	is the name of the actual category, which could be just a literal, or perhaps the URI of the process parameter (a property).
`taxonomy`	stores a reference to the taxonomy scheme. It can be either a URI of the taxonomy, or a URL where the taxonomy resides, or the name of the taxonomy or anything else.
`value`	points to the value in a specific taxonomy There may be more than one value for each taxonomy, so no restriction is added here.
`code`	to each type of service stores the code associated to a taxonomy.

In SDCM2.0, a similar structure has been considered in which the Service profile has a specific attribute pointing to a taxonomy/ontology element and an abstract class. The descriptions of these elements in the SDCM specification are listed in Table 4 and Table 5.

Table 4. Profile Attributes in SDCM2.0 (from SDCM 2.0 specification)

Name	Definition	Notes
service category	A classification of the service that could be based on various criteria; e.g., type of service provided, technological or architectural solutions, etc. (see Service Category class)	Service categories are described as taxonomies, usually established by the implementing organization

Table 5. ServiceCategory in SDCM2.0 (from SDCM 2.0 specification)

Definition	Notes	Prototype
A taxonomy used to classify a service by the type of service provided or by some other technological or architectural solution.	The Service Category class is a placeholder for one or more taxonomies. Because of ongoing efforts to either update existing or create new service categorization taxonomies, this class is defined as an abstract class.	Taxonomy

This approach has already been used to integrate categorization based on SWIM service product and service category, as well as referring to the following taxonomies created by FAA as shown in the snippet of the RIM service description for the Flight Plan Service.

FPS Service Profile snippet

```
<ServiceDescription xmlns="http://swim.aero/rim/1.0.0">
    <Profile>
        <ServiceName>Flight Plan Service (FPS)</ServiceName>
        <ServiceVersion>1.0.0</ServiceVersion>
        <ServiceDescription>(This fictitious service is for instructional use only and
cannot be consumed).
        A service for filing, updating, or canceling an IFR (Instrument Flight Rules)
flight plan.
        </ServiceDescription>
        <sd:Categories xmlns:sd="http://swim.aero/rim/1.0.0">
            <sd:Category>
                <sd:Title>ATM Service Category</sd:Title>
                <sd:Value>Flight Planning</sd:Value>
            </sd:Category>
            <sd:Category>
                <sd:Title>SWIM Service Product Category</sd:Title>
                <sd:Value>Flight</sd:Value>
            </sd:Category>
            <sd:Category>
                <sd:Title>Lifecycle Status</sd:Title>
                <rim:Value xmlns:rim="http://swim.aero/rim/1.0.0">
Development</rim:Value>
            </sd:Category>
        </sd:Categories>
        <Provider>
       ...
```

In this example, three different facets are used to categorize the service: the ATM service Category, the SWIM Service Product Category and the LifeCycle Status. Two of these categories refer to taxonomies published in http://semantic.aero although the reference in this example is not direct.

It is important to note here that there is still a discrepancy between the categories format presented here and the SDCM2.0 schema used to structure the category complex type. In the SDCM2.0 schema, Service Category is aligned with the OWL-S ServiceCategory class and includes the 4 properties described in Table 3. While in the example used here, the ServiceCategory is typed with a static element (<title>) and points to a value which is a human readable label (similar to the value of <rdfs:label/>) within an ontology. Work for further alignment with the SDCM2.0 model is in progress and should include a URI as pointer to the element of the semantic resource.

The scope of the ServiceCategory class could be extended by the integration of the semantic representation of the data elements published by the service i.e. domain-specific semantics. In this case, the value definition should be extended to allow holding a list of URIs. The following example could be considered, where elements of the FPS schema are aligned with concepts coming from the NASA ontology and added as a list of values. In this case, the complexType “AircraftType” and “WakeTurbulenceCategory” are being considered. The integration of these concepts is shown in the following code snippet.

FPS Service Profile annotated: Adding domain-specific information

```
<ServiceDescription xmlns="http://swim.aero/rim/1.0.0">
    <Profile>
        <ServiceName>Flight Plan Service (FPS)</ServiceName>
        <ServiceVersion>1.0.0</ServiceVersion>
        <ServiceDescription>(This fictitious service is for instructional use only and
cannot be consumed) A service for filing, updating, or canceling an IFR (Instrument
Flight Rules) flight plan.
</ServiceDescription>
        <sd:Categories xmlns:sd="http://swim.aero/rim/1.0.0">
            <sd:Category>
                <sd:Title>ATM Service Category</sd:Title>
                <sd:Value>Flight Planning</sd:Value>
            </sd:Category>
            <sd:Category>
                <sd:Title>SWIM Service Product Category</sd:Title>
                <sd:Value>Flight</sd:Value>
            </sd:Category>
            <sd:Category>
                <sd:Title>Lifecycle Status</sd:Title>
                <rim:Value xmlns:rim="http://swim.aero/rim/1.0.0">
Development</rim:Value>
            </sd:Category>
            <sd:Category>
                <sd:Title>Domain specific information</sd:Title>
                <sd:Value>
https://data.nasa.gov/ontologies/atmonto/equipment#AircraftType </sd:Value> ①
                <sd:Value>
https://data.nasa.gov/ontologies/atmonto/equipment#AircraftWakeCategory </sd:Value> ②
            </sd:Category>
        </sd:Categories>
        <Provider>
       ...
```

① Annotation with the concept **"AircraftType"** defined in the atmonto ontology developed by NASA.

② Annotation with the concept **"AircraftWakeCategory"** defined in the atmonto ontology developed by NASA.

It is important to note here that the scope of the NASA ATM ontology covers mostly concepts from AIXM. In order to fully describe Flight Plans, it is necessary to aggregate concepts from aeronautical information (AIXM) but also flight information (FIXM) and weather information (WXXM).

The integration of URIs from external knowledge models such as ontologies, controlled vocabularies or thesauri reduces the human readability of the service description but enables a reasoning agent to better discover the information and make inferences on the service. For this, the knowledge model element should correspond to resolvable URIs providing access to the complete model. In the previous example, concept URIs present the label as part of the identifier. Although this method reduces the number of calls to web service serving the semantic resource by

downloading the whole knowledge model, it requires the agent to build the corresponding graph internally in order to be able to make simple inferences such as subsumption for instance. This can become computationally expensive for the agent. Furthermore, this approach reduces the possibility to encode homonymy and to differentiate the two homonyms. This ER advises using numbered unique IDs such as Universally Unique Identifier (UUID) or even a Persistent Identifier (PID), and attach the human readable label as <rdfs:label>. In the latter, the rdfs:label should be included along with the URI in order to provide both information for the machine and humans.

One clear limitation of this approach is that the hierarchy/categories are defined in the "Service Profile" data model (i.e. ATM Service type, …). The extension of this hierarchy will require modifications to the data model. It would therefore be more efficient to define the hierarchy as an external model. The hierarchy can then be inferred from the corresponding external data model.

Finally, the approach presented here includes a lot of unnecessary information which could be accessed automatically if the URIs used to build taxonomies and ontologies are resolvable and would provide back an answer either as rdf/xml or json-ld. To address this issue, taxonomies, controlled vocabularies, ontologies should be published within a vocabulary management system which provides direct access to the content of the semantic resources through simple API calls (e.g. get narrow, get superclass, get subclass,…) instead of downloading the file to extract this information (see chapter 9 for more details).

With such a system, the last example will become as follows:

```
<ServiceDescription xmlns="http://swim.aero/rim/1.0.0">
    <Profile>
        <ServiceName>Flight Plan Service (FPS)</ServiceName>
        <ServiceVersion>1.0.0</ServiceVersion>
        <ServiceDescription>(This fictitious service is for instructional use only and
cannot be consumed) A service for filing, updating, or canceling an IFR (Instrument
Flight Rules) flight plan.
</ServiceDescription>
        <sd:Categories xmlns:sd="http://swim.aero/rim/1.0.0">
            <sd:Category>
                <sd:Title>ATM Service Category</sd:Title>
                <sd:Value>http://semantics.aero/flight-phase#flight-
planning</sd:Value> ①
            </sd:Category>
            <sd:Category>
                <sd:Title>SWIM Service Product Category</sd:Title>
                <sd:Value>http://semantics.aero/service-product#flight</sd:Value> ②
            </sd:Category>
            <sd:Category>
                <sd:Title>Lifecycle Status</sd:Title>
                <rim:Value xmlns:rim="http://swim.aero/rim/1.0.0">
Development</rim:Value>
            </sd:Category>
            <sd:Category>
                <sd:Title>Domain specific information</sd:Title>
                <sd:Value>
https://data.nasa.gov/ontologies/atmonto/equipment#AircraftType </sd:Value>
                <sd:Value>
https://data.nasa.gov/ontologies/atmonto/equipment#AircraftWakeCategory </sd:Value>
            </sd:Category>
        </sd:Categories>
        <Provider>
       ...
```

① Annotation with **"flight planning"** defined in semantics.aero flight phase thesaurus

② Annotation with **"flight"** defined in semantics.aero service product thesaurus

This process requires one to extend the categories based on the available information in the XML schemata or other service description documents. The process would be tedious as it will have to be done manually and would require updating of the XML description of each service.

However, the clear advantage is the low-cost of this solution as it just extends the data model and will require minimal changes.

This annotation method corresponds to the level 1 annotation described in OGC 08-167r2 [1]

7.2.2. Integration within the Service Model class

Another possible way to link domain-specific information would be to associate it with the data

entities served by a particular service. The SDCM2.0 model includes two placeholders which could be used for such purpose: “data entity” and “data description”.

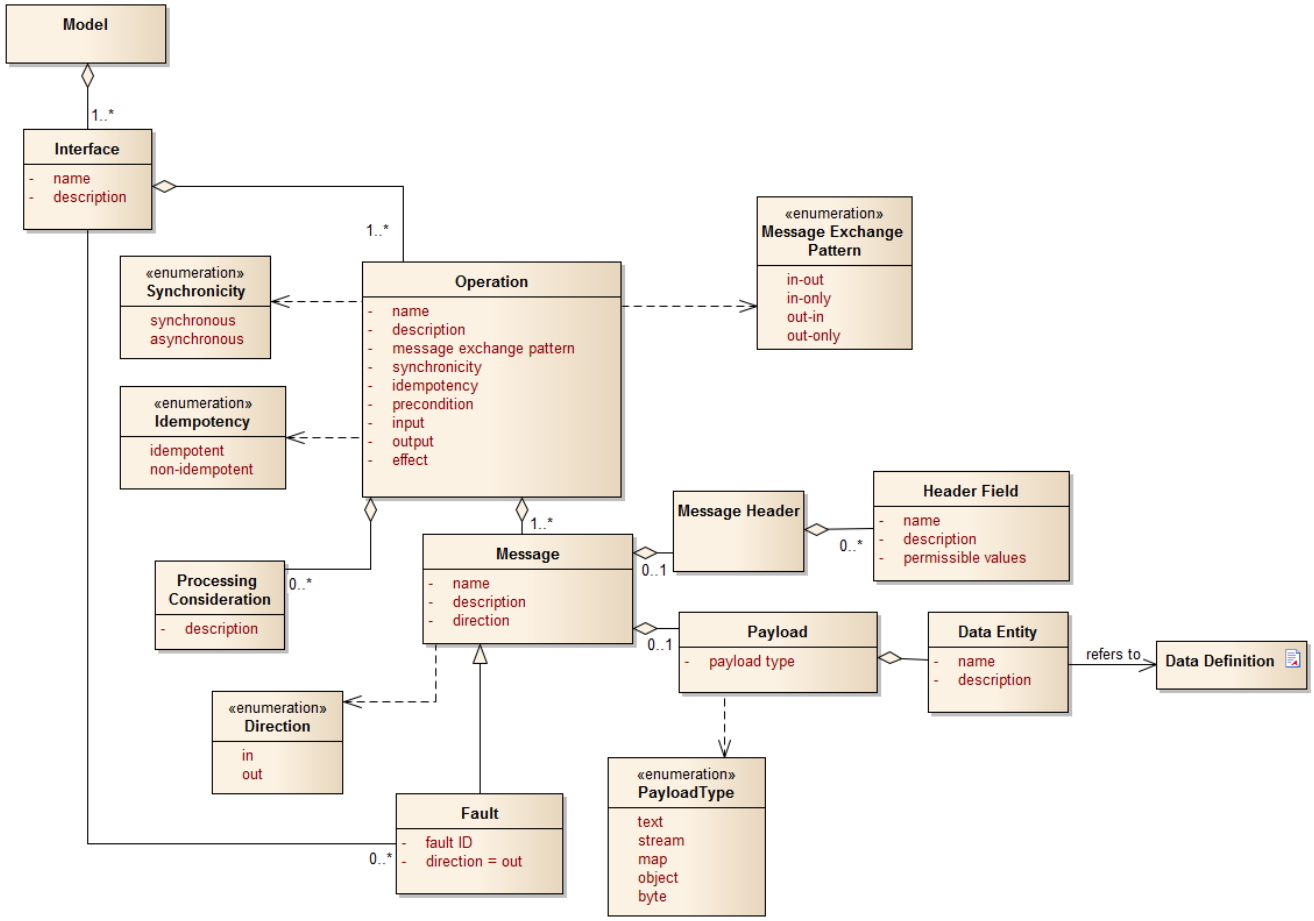


Figure 8. SDCM2.0 Service Model

These two placeholders are described in Table 6 and Table 7.

Table 6. Data Entity (from SDCM2.0 specification)

Name	Definition	Notes
name	the name of the data entity	relates to rdfs:label and concept URI
description	A description of the data entity	corresponds to concept definition provided within the associated semantic resource

Table 7. Data Definition (from SDCM2.0 specification)

Definition	Notes	Prototype
A resource that defines and describes the meaning, structure, inter-relationships, permissible values, format and other aspects of a data entity	Possible examples are an XML schema file, a data model diagram, a common information exchange model such as AIXM, or some other resource that describes the data entity.	Document

The data definition class provides the means to link a data entity with various kind of documents describing it. Although these documents provide information regarding the data entity, the format used does not provide direct access to the semantics of this information. This class could thus be used to link the data entity to a dedicated ontology i.e. a resource ontology as described in OGC 08-167r2 [1].

In the following code snippet, a way to integrate the reference to a semantic resource within the "data entity" class is proposed. For this example and the following others, we have defined an ontology describing the concepts used in the FPS XML schema. This ontology is available in Annex B.

Integrating semantic in the Service Model class

```
<ServiceDescription xmlns="http://swim.aero/rim/1.0.0">
    <Profile>
        <ServiceName>Flight Plan Service (FPS)</ServiceName>
        <ServiceVersion>1.0.0</ServiceVersion>
        <ServiceDescription>(This fictitious service is for instructional use only and
cannot be consumed) A service for filing, updating, or canceling an IFR (Instrument
Flight Rules) flight plan.
        </ServiceDescription>
        <sd:Categories xmlns:sd="http://swim.aero/rim/1.0.0">
            <sd:Category>
                <sd:Title>ATM Service Category</sd:Title>
                <sd:Value>Flight Planning</sd:Value>
            </sd:Category>
            <sd:Category>
                <sd:Title>SWIM Service Product Category</sd:Title>
                <sd:Value>Flight</sd:Value>
            </sd:Category>
            <sd:Category>
                <sd:Title>Lifecycle Status</sd:Title>
                <rim:Value xmlns:rim="http://swim.aero/rim/1.0.0">
Development</rim:Value>
            </sd:Category>
        </sd:Categories>
        <Provider>
            <Name>FAA En Route Services Modernization Group (ESMG)</Name>
            <Description>A program within the FAA Air Traffic Organization responsible
for developing Web services.
            </Description>
            <WebPage>http://www.faa.gov/air_traffic/flight_info</WebPage>
            <PointsOfContact>
                <POC>
                    <Name>John  D. Doe</Name>
                    <Function>ATO-X ESMG Manager</Function>
                    <Phone>(609) 444-5555</Phone>
                    <Email>Joe.doe@faa.gov</Email>
                </POC>
                <POC>
                    <Name>Mark Kaplun</Name>
```

```
                    <Function>Governance Lead</Function>
                    <Phone>23423423423</Phone>
                    <Email>mark.kaplun@faa.gov</Email>
                </POC>
                <POC>
                    <Name>Carol Uri</Name>
                    <Function>SWIM PO Contact</Function>
                    <Phone>555-555-5555</Phone>
                    <Email>curi@faa.gov</Email>
                </POC>
            </PointsOfContact>
        </Provider>
        ...

    </Profile>
    <Model>
        <Interface>
            <Operation>
                <Message>
                    <MessageHeader>
                    </MessageHeader>
                    <Payload>
                        <DataEntity>
                            <name> departure aerodrome </name> ①
                            <description>http://example.org/testbed-
14/flightplan#departure_aerodrome </description> ②
                            <DataDefinition>http://example.org/testbed-
14/flightplan.xsd//xs:element[name="FlightPlan"</DataDefinition> ③
                        </DataEntity>
                    </Payload>
                </Message>
            </Operation>
        </Interface>
    </Model>
    <Grounding/>
</ServiceDescription>
```

① Data entity name corresponds here to the rdfs:label related to the concept "departure_aerodrome"

② Link to the concept providing access to additional information such as synonyms, definition,...

③ Link to the syntactic information i.e. the corresponding XML element within the FPS XML schema.

In this example, the attribute "name" corresponds to the rdfs:value of the corresponding concept in the external semantic resource. Using such a solution would require one to automatically extract the value of the rdfs:label corresponding to the concept URI within the considered ontology.

7.3. Annotation of the XML schema with SAWSDL

The domain-specific information is contained in the XML schema defining the structure of the data provided by the service. This information is included in GML like schema such AIXM, FIXM and WXXM. The XML schemata use specific name values with definitions of the various elements but this information is not usable by software agent. To extend the scope of this information to software agents, it is necessary to integrate semantics within the XML schema. The use of semantics provides a common reference model to align the different XML schemata and a description of the relation between the different elements. In the context of complex fields such ATM, such unique common reference model will rapidly grow and become hard to maintain. To address this issue, (OGC 08-167r2) proposed to design resource ontologies describing the data model specific to the service based on core elements of the common reference model.

To support this testbed, the participants considered annotating the FPS XML schema shown below:

FPS XML Schema snippet

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
 xmlns="urn:us:gov:dot:faa:example:atm:enroute:fps:entities" targetNamespace=
"urn:us:gov:dot:faa:example:atm:enroute:fps:entities" elementFormDefault="qualified"
attributeFormDefault="unqualified">
  <xs:annotation>
    <xs:documentation xml:lang="en">
     Title: FlightPlan schema for WSDD Example.
     Description: This schema declares XML elements for defining
     a Flight Plan transmitted by FlightPlanService
     Creator: Mark Kaplun (mark.kaplun@faa.gov)
     Date:  2010-01-21
     </xs:documentation>
  </xs:annotation>

  <!-- ////////////////////////////////////////////////////////////////
                  Global types
       //////////////////////////////////////////////////////////////// -->

  <xs:element name="FlightPlan">
    <xs:complexType>
      <xs:sequence>
        <!-- "FlightPlanId" is always required.
        When flight plan is filed and the "FlightPlanId" element has no content
        - the content is nil. -->
        <xs:element name="FlightPlanId" type="FlightPlanIdType" nillable="true"/>
        <xs:element name="Originator" type="OriginatorType"/>
        <xs:element ref="Aircraft"/>
        <xs:element ref="Route"/>
      </xs:sequence>
      <xs:attribute name="filingTime" type="xs:dateTime" use="required"/>
      <xs:attribute name="flightRule" type="FlightRuleType" use="required"/>
      <xs:attribute name="numberOfAircraft" type="xs:positiveInteger" default="1"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="Aircraft" type="AircraftType"/>
  <xs:element name="Route" type="RouteType"/>
  <!-- ////////////////////////////////////////////////////////////////
                  Types definitions
   //////////////////////////////////////////////////////////////// -->
  <xs:simpleType name="FlightPlanIdType">
```

One of the solutions proposed in OGC 08-167r2 [1], is to integrate semantics within XML schemata and WSDL documents using the W3C recommendation SAWSDL (for more information see, OGC 08-167r2 [1] and OGC 18-035 [2]). SAWSDL defines a specific class called model reference which points to an entry of a particular knowledge model and associates it with elements of the XML schema as a special attribute. For the sake of the current exercise, a flight plan ontology which encodes the concepts listed in the schema as OWL Classes was developed. The resulting ontology can be used to

annotate flight plan using the following URL base http://www.example.org/testbed-14/flightplan (see annex B).

The annotated FPS schema would be as follows:

Annotated FPS Schema with SAWSDL

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
 xmlns="urn:us:gov:dot:faa:example:atm:enroute:fps:entities" *xmlns:sawsdl=
"http://www.w3.org/ns/sawsdl"*  targetNamespace=
"urn:us:gov:dot:faa:example:atm:enroute:fps:entities" elementFormDefault="qualified"
attributeFormDefault="unqualified"> ①
  <xs:annotation>
    <xs:documentation xml:lang="en">
    Title: FlightPlan schema for WSDD Example.
     Description: This schema declares XML elements for defining
     a Flight Plan transmitted by FlightPlanService
     Creator: Mark Kaplun (mark.kaplun@faa.gov)
     Date:  2010-01-21
     </xs:documentation>
  </xs:annotation>
  <!-- ////////////////////////////////////////////////////////////////
                  Global types
        //////////////////////////////////////////////////////////////// -->
  <xs:element name="FlightPlan" sawsdl:modelReference="http://www.example.org/testbed-
14/flightplan#Flight Plan"
/>
    <xs:complexType>
      <xs:sequence>
        <!-- "FlightPlanId" is always required.
        When flight plan is filed and the "FlightPlanId" element has no content
        - the content is nil. -->
        <xs:element name="FlightPlanId" type="FlightPlanIdType" nillable="true"
sawsdl:modelReference="http://www.example.org/testbed-14/flightplan#FlightPlanId" />
②
        <xs:element name="Originator" type="OriginatorType"
sawsdl:modelReference="http://www.example.org/testbed-14/flightplan
#FlightPlanOriginator" /> ③
        <xs:element ref="Aircraft"/>
        <xs:element ref="Route"/>
      </xs:sequence>
      <xs:attribute name="filingTime" type="xs:dateTime" use="required"
sawsdl:modelReference="http://www.example.org/testbed-14/flightplan#FillingTime" /> ④
      <xs:attribute name="flightRule" type="FlightRuleType" use="required"
sawsdl:modelReference="http://www.example.org/testbed-14/flightplan#FlightRules" /> ⑤
      <xs:attribute name="numberOfAircraft" type="xs:positiveInteger" default="1"
sawsdl:modelReference="http://www.example.org/testbed-14/flightplan#NumberOfAircraft"
/> ⑥
    </xs:complexType>
  </xs:element>
```

```
  <xs:element name="Aircraft" type="AircraftType"
sawsdl:modelReference="http://www.example.org/testbed-14/flightplan#Aircraft" /> ⑦
  <xs:element name="Route" type="RouteType"
sawsdl:modelReference="http://www.example.org/testbed-14/flightplan#route" /> ⑧
  <!-- ////////////////////////////////////////////////////////////////
                  Types definitions
   //////////////////////////////////////////////////////////////// -->
  <xs:simpleType name="FlightPlanIdType">
```

① Import sawsdl namespace

② Annotate with Flight Plan Ontology concept

③ Annotate with Flight Plan Ontology concept

④ Annotate with Flight Plan Ontology concept

⑤ Annotate with Flight Plan Ontology concept

⑥ Annotate with Flight Plan Ontology concept

⑦ Annotate with Flight Plan Ontology concept

⑧ Annotate with Flight Plan Ontology concept

This example extends the initial specification by associating the model reference to attributes of a particular element. These attributes define properties of the flight plan. The SAWSDL information can also be extracted from the XML schema and converted to direct annotations within SDCM either using the service category attribute of the Service Profile or using the data entity/data description classes within the Service Model. However, this solution is not cheap as it requires tedious manual work to align the schemata with the corresponding external model.

7.4. Lightweight semantic integration using web annotation

The previous sections investigated semantic annotation solutions that require direct modification of XML schemata or SDCM based service descriptions with NSRR. These two approaches require modification or extension of all of the service descriptions stored in NSRR and do not allow an easy link between the service description and the document describing the service operations and data model.

To address this issue, this ER considers the use of the W3C Web Annotation (WA) data model [7: https://www.w3.org/TR/annotation-model/]. This recent data model allows linking to the service, the service description document or even a part of the document and a related semantic concept. The following section describes the current WA data model and associated standards such as the WA vocabulary [8: https://www.w3.org/TR/annotation-vocab/] and the WA protocol [9: https://www.w3.org/TR/annotation-protocol/]. The next chapter investigates the possibility to extend the current WA data model to use a structured representation of the different elements used to describe a service (i.e. XML schemas, WSDD, WFS, ...). The precursor of WA has been used in Testbed-10 for image integrated annotations (OGC 14-002) [13].

7.4.1. The Web Annotation data model

The WA data model is derived from the OpenAnnotation data model [10: http://www.openannotation.org/spec/core/core.html]. This initial model relies on the association of three core elements as shown in Figure 9:

- the **annotation** class which makes the link between the two other class i.e. the target and the body. This link establishes an implicit "related to" or "about" relations between these two class. The annotation class also supports the integration of annotation provenance information as well as a link to describe the motivation of the annotation (see more details below)
- the **target** relation which represents the datum to be annotated. The target can be any document as long as the document can be uniquely identified i.e. data element within a database, an excel spreadsheet, an image, a text document, a web page,.... The target can point to a whole document or a part of the document as long as it can be uniquely identified. There are no restrictions on the type of target which can be annotated.
- the **body** relation which corresponds to the content of the annotation i.e. a keyword, a semantic tag, a comment, a link to another file. There are currently no restrictions on the type of body that can be used.

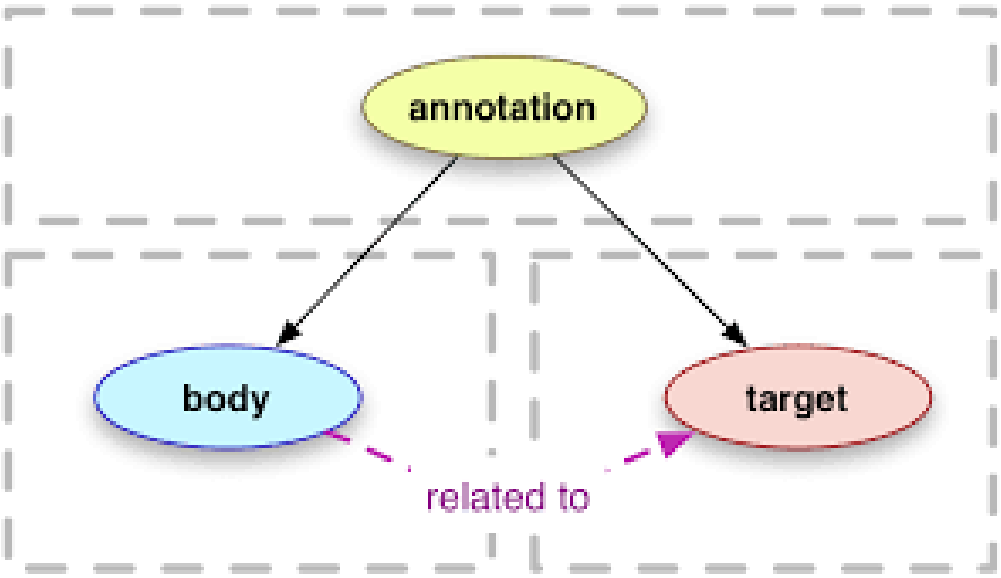


Figure 9. the Web Annotation data model

In our case, the target class represents the service record within the registry or could also be a document associated with service description as shown in Figure 10.

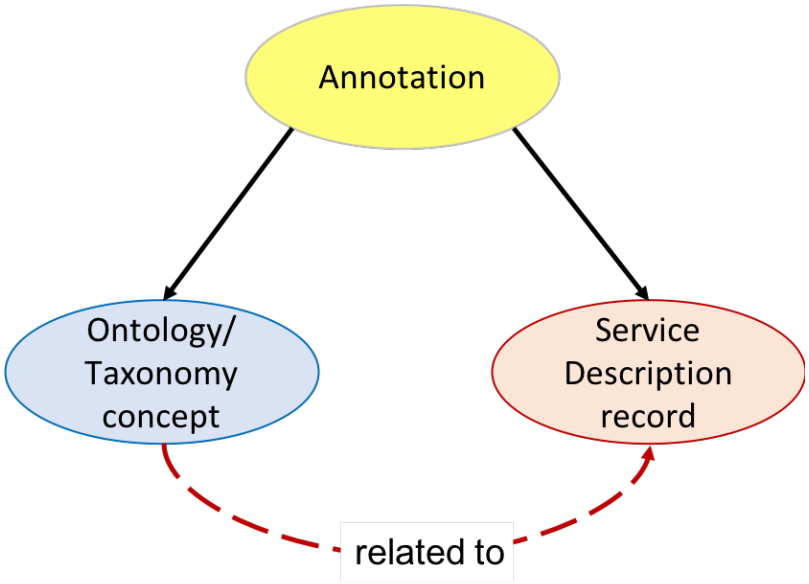


Figure 10. Annotation as a link between services (description and documents) and semantics.

This simple schema allows one to create new facets describing a service record within NSRR without modifying the underlying service data model or the resulting XML description.

Annotations can also be used to link two files/documents together or to link a service record in NSRR with a document (e.g. WSDD, GML based schemata, WSDR, ...) describing the service as shown in Figure 11.

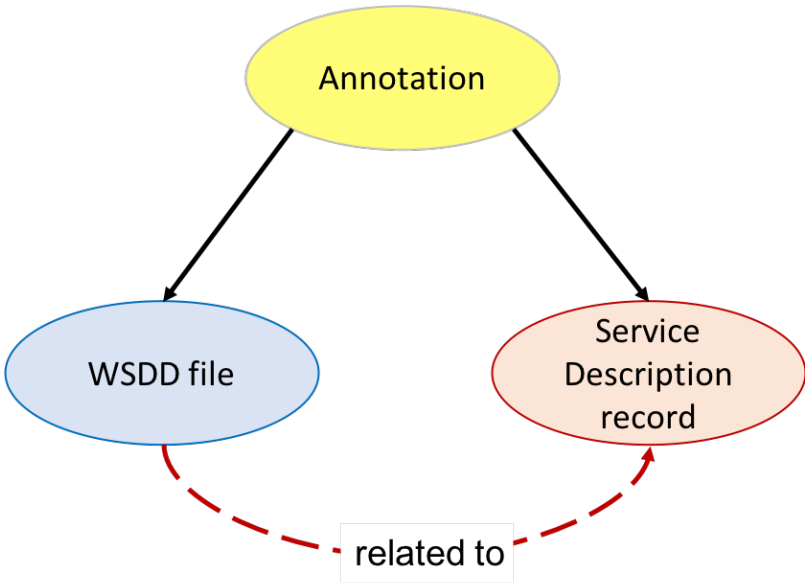


Figure 11. Annotation as a link between a service record and the associated documents.

The WA data model is based on Linked Data (LD) principles. Therefore, the core requirement for using this model is to guarantee that the service records and the documents are uniquely identified with a unique URI within NSRR. For instance, this is not the case for XML descriptions which are accessible as a zip file which aggregate multiple files. This structure will prevent the annotation of individual XML files. Annotations are serialized as JSON-LD, a Linked Data compliant version of JSON proposed as a W3C standard (https://www.w3.org/TR/json-ld/). JSON-LD is an extension of JSON integrating semantic information through the use of a context document that includes namespaces of the ontologies used for a particular JSON-LD document.

Example of annotation serialized as JSON-LD

```
{
  "@context": "http://www.w3.org/ns/anno.jsonld",
  "id": "http://example.org/testbed-14/annotations/anno12",
  "type": "Annotation",
  "body": {
    "type" : "TextualBody",
    "value" : "This service rocks !!",
    "format" : "text/html",
    "language" : "en"
  },
  "target": "http://nsrr.faa.gov/services/fps"
}
```

7.4.2. Web Annotation Vocabulary

The WA Vocabulary comprises a list of RDF classes, named entities and predicates used to describe the WA data model as well as reference to other ontologies/vocabularies terms and relations used by the WA data model. This vocabulary is used to create the JSON-LD context shown in the following code snippet.

Web Annotation context

```
{
 "@context": {
    "oa":      "http://www.w3.org/ns/oa#",
    "dc":      "http://purl.org/dc/elements/1.1/",
    "dcterms": "http://purl.org/dc/terms/",
    "dctypes": "http://purl.org/dc/dcmitype/",
    "foaf":    "http://xmlns.com/foaf/0.1/",
    "rdf":     "http://www.w3.org/1999/02/22-rdf-syntax-ns#",
    "rdfs":    "http://www.w3.org/2000/01/rdf-schema#",
    "skos":    "http://www.w3.org/2004/02/skos/core#",
    "xsd":     "http://www.w3.org/2001/XMLSchema#",
    "iana":    "http://www.iana.org/assignments/relation/",
    "owl":     "http://www.w3.org/2002/07/owl#",
    "as":      "http://www.w3.org/ns/activitystreams#",
    "schema":  "http://schema.org/",

    "id":      {"@type": "@id", "@id": "@id"},
    "type":    {"@type": "@id", "@id": "@type"},

    "Annotation":           "oa:Annotation",
    "Dataset":              "dctypes:Dataset",
    "Image":                "dctypes:StillImage",
    "Video":                "dctypes:MovingImage",
    "Audio":                "dctypes:Sound",
    "Text":                 "dctypes:Text",
    "TextualBody":          "oa:TextualBody",
```

```
    "ResourceSelection":    "oa:ResourceSelection",
    "SpecificResource":     "oa:SpecificResource",
    "FragmentSelector":     "oa:FragmentSelector",
    "CssSelector":          "oa:CssSelector",
    "XPathSelector":        "oa:XPathSelector",
    "TextQuoteSelector":    "oa:TextQuoteSelector",
    "TextPositionSelector": "oa:TextPositionSelector",
    "DataPositionSelector": "oa:DataPositionSelector",
    "SvgSelector":          "oa:SvgSelector",
    "RangeSelector":        "oa:RangeSelector",
    "TimeState":            "oa:TimeState",
    "HttpRequestState":     "oa:HttpRequestState",
    "CssStylesheet":        "oa:CssStyle",
    "Choice":               "oa:Choice",
    "Person":               "foaf:Person",
    "Software":             "as:Application",
    "Organization":         "foaf:Organization",
    "AnnotationCollection": "as:OrderedCollection",
    "AnnotationPage":       "as:OrderedCollectionPage",
    "Audience":             "schema:Audience",

    "Motivation":    "oa:Motivation",
    "bookmarking":   "oa:bookmarking",
    "classifying":   "oa:classifying",
    "commenting":    "oa:commenting",
    "describing":    "oa:describing",
    "editing":       "oa:editing",
    "highlighting":  "oa:highlighting",
    "identifying":   "oa:identifying",
    "linking":       "oa:linking",
    "moderating":    "oa:moderating",
    "questioning":   "oa:questioning",
    "replying":      "oa:replying",
    "reviewing":     "oa:reviewing",
    "tagging":       "oa:tagging",

    "auto":          "oa:autoDirection",
    "ltr":           "oa:ltrDirection",
    "rtl":           "oa:rtlDirection",

    "body":          {"@type": "@id", "@id": "oa:hasBody"},
    "target":        {"@type": "@id", "@id": "oa:hasTarget"},
    "source":        {"@type": "@id", "@id": "oa:hasSource"},
    "selector":      {"@type": "@id", "@id": "oa:hasSelector"},
    "state":         {"@type": "@id", "@id": "oa:hasState"},
    "scope":         {"@type": "@id", "@id": "oa:hasScope"},
    "refinedBy":     {"@type": "@id", "@id": "oa:refinedBy"},
    "startSelector": {"@type": "@id", "@id": "oa:hasStartSelector"},
    "endSelector":   {"@type": "@id", "@id": "oa:hasEndSelector"},
    "renderedVia":   {"@type": "@id", "@id": "oa:renderedVia"},
    "creator":       {"@type": "@id", "@id": "dcterms:creator"},
```

```
    "generator":     {"@type": "@id", "@id": "as:generator"},
    "rights":        {"@type": "@id", "@id": "dcterms:rights"},
    "homepage":      {"@type": "@id", "@id": "foaf:homepage"},
    "via":           {"@type": "@id", "@id": "oa:via"},
    "canonical":     {"@type": "@id", "@id": "oa:canonical"},
    "stylesheet":    {"@type": "@id", "@id": "oa:styledBy"},
    "cached":        {"@type": "@id", "@id": "oa:cachedSource"},
    "conformsTo":    {"@type": "@id", "@id": "dcterms:conformsTo"},
    "items":         {"@type": "@id", "@id": "as:items", "@container": "@list"},
    "partOf":        {"@type": "@id", "@id": "as:partOf"},
    "first":         {"@type": "@id", "@id": "as:first"},
    "last":          {"@type": "@id", "@id": "as:last"},
    "next":          {"@type": "@id", "@id": "as:next"},
    "prev":          {"@type": "@id", "@id": "as:prev"},
    "audience":      {"@type": "@id", "@id": "schema:audience"},
    "motivation":    {"@type": "@vocab", "@id": "oa:motivatedBy"},
    "purpose":       {"@type": "@vocab", "@id": "oa:hasPurpose"},
    "textDirection": {"@type": "@vocab", "@id": "oa:textDirection"},

    "accessibility": "schema:accessibilityFeature",
    "bodyValue":     "oa:bodyValue",
    "format":        "dc:format",
    "language":      "dc:language",
    "processingLanguage": "oa:processingLanguage",
    "value":         "rdf:value",
    "exact":         "oa:exact",
    "prefix":        "oa:prefix",
    "suffix":        "oa:suffix",
    "styleClass":    "oa:styleClass",
    "name":          "foaf:name",
    "email":         "foaf:mbox",
    "email_sha1":    "foaf:mbox_sha1sum",
    "nickname":      "foaf:nick",
    "label":         "rdfs:label",

    "created":       {"@id": "dcterms:created", "@type": "xsd:dateTime"},
    "modified":      {"@id": "dcterms:modified", "@type": "xsd:dateTime"},
    "generated":     {"@id": "dcterms:issued", "@type": "xsd:dateTime"},
    "sourceDate":    {"@id": "oa:sourceDate", "@type": "xsd:dateTime"},
    "sourceDateStart": {"@id": "oa:sourceDateStart", "@type": "xsd:dateTime"},
    "sourceDateEnd": {"@id": "oa:sourceDateEnd", "@type": "xsd:dateTime"},

    "start":         {"@id": "oa:start", "@type": "xsd:nonNegativeInteger"},
    "end":           {"@id": "oa:end", "@type": "xsd:nonNegativeInteger"},
    "total":         {"@id": "as:totalItems", "@type": "xsd:nonNegativeInteger"},
    "startIndex":    {"@id": "as:startIndex", "@type": "xsd:nonNegativeInteger"}
  }
}
```

7.4.3. Web Annotation protocol

The WA protocol defines an interoperability layer between annotation clients and server to improve the creation, management and access to annotations. These specifications are based on REST principles and elements of the Linked Data platform [11: https://www.w3.org/TR/ldp/]. The W3C recommendation proposes specification for establishing a standardized annotation API on top of annotation web servers.

These specifications provide information regarding the type of HTTP requests to provide access to annotations, defines an annotation container. The annotation container is at the same time a Container as defined in the W3C Linked Data Platform specification (i.e. a service managing annotations) and an ordered collection of annotations. The retrieval, representation of the Annotation Containers, paging constrains and CRUD operations for annotations are defined in these specifications.

The next section argues that a solution based on the web annotation data model provides an easy-to-link service description and service documents within the NSRR, offers the means to insert domain-specific semantics and use inference. Such a solution would be potentially easy to deploy in order to aggregate service descriptions across multiple service registries.

Chapter 8. Integrating Semantics with Web Annotation

The previous section presented various solutions to integrate semantics with service description hosted in NSRR. Most of the proposed solutions require one to modify the content of the NSRR database, except for the Web Annotation approach.

Annotations are separate information entities linking any accessible NSRR elements (i.e. elements uniquely identified by URIs) either together or with external source of information such as semantic concepts from external ontologies or controlled vocabularies. In the context of this ER, annotations could be used to enrich NSRR content at different levels as shown by the following non-exhaustive list:

- associating elements of the XML service descriptions with concepts from service description ontologies such as WSDOM or OWL-S,
- associating part of a WSDD or WSDR document with concepts from WSDOM and ATM ontologies,
- associating service description with domain-specific concepts from ATM ontologies such as the NASA ATM ontology,
- associating elements of XML schema describing the data served by the services (e.g. AIXM, FIXM, WXXM,...) with domain-specific concepts from ATM ontologies,
- associating geospatial information with service description,
- associating service descriptions with related documents i.e. XML schemata, WSDD, WSDR,
- associating service description with curation/lifecycle information, and
- associating service description with functional facets.

Annotations can be used to answer complex queries aggregating various type of information such as the examples provided in Section 6. For example, retrieving services that deliver "runway bearing" information for the "west coast" would require to find all the services which are annotated with the "runway true bearing" concept AND "west coast" geospatial concept. Interestingly, the query presented here is rather simple and does not require the use of SPARQL. However, it is possible to refine this query by retrieving information about the concepts within their semantic models (i.e. synonyms, relations,...) using either SPARQL or by interacting with a vocabulary management system which will provide direct access to the semantic models.

This section focuses first on describing annotation models to support some of the annotations listed above. The second part of this section describes an architecture to use annotations with NSRR.

8.1. The ATM Web Annotation Model

The Web Annotation model offers multiple classes and constructs to accomodate a large range of annotations. In this section, we use these various constructs to build specific annotations for extending NSRR discoverability.

8.1.1. Annotation metadata

The first level of information that should be considered in the context of annotations is the metadata concerning the annotation itself i.e. provenance information.

Provenance information

To associate the provenance information with the annotation, two relationships and three properties are considered. They are summarized in Table 8.

Table 8. Annotation metadata and provenance information

Term	Type	Description
creator	relationship	The agent responsible for creating the resource. This may be either a human, an organization or a software agent.
created	property	The time at which the annotation was created
generator	relationship	The agent responsible for generating the serialization of the annotation
generated	property	The time at which the annotation serialization was generated.
modified	property	The time at which the annotation was modified, after creation.

The creator and generator relationship can be extended with more details by providing a unique identifier that unambiguously identifies the agent, and the type of the agent which can be an instance of the classes Person, Organization or Software. These classes have associated properties such as name, nickname, email, email_sha1 and homepage. This list of property can be extended by using specific agent descriptions from FAA.

```
{
  "@context": "http://www.w3.org/ns/anno.jsonld"
  "id": "http://example.org/testbed-14/annotation/anno_1"
  "type":"Annotation"
  "creator": {
    "id": "http://example.org/testbed-14/annotation/user1",
    "type": "Person",
    "name": "Yann Le Franc",
    "nickname": "testman",
  },
  "generator": {
    "id": "http://example.org/testbed-14/annotation/client1",
    "type": "Software",
    "name": "Code v3.6",
    "homepage": "http://example.org/testbed-14/annotation/client1/homepage1"
  },
  "body": "http://example.net/review1",
  "target": "http://example.com/restaurant1"

}
```

These different elements are reused from existing standards such as Dublin Core (http://dublincore.org/), FOAF (http://xmlns.com/foaf/spec/). In order to make provenance information more interoperable, the integration of the existing W3C standard for provenance i.e. PROV (https://www.w3.org/TR/prov-primer/) should be integrated together with the web annotation model.

Annotation motivation and purpose: pragmatic information

Within the WA data model an additional class has been proposed to represent the motivation of the annotation. This class refers to the motivation of the annotation but could also be used to specify the purpose of the body class. Several instance of the Motivation class have been defined and are listed in Table 9:

Table 9. list of motivations proposed in W3C WA data model

Motivation	Description
assessing	The motivation for when the user intends to assess the target resource in some way, rather than simply make a comment about it. For example, to write a review or assessment of a book, assess the quality of a dataset, or provide an assessment of a student's work.
bookmarking	The motivation for when the user intends to create a bookmark to the Target or part thereof. For example, an Annotation that bookmarks the point in a text where the reader finished reading.

Motivation	Description
`classifying`	The motivation for when the user intends to classify the Target as something. For example, to classify an image as a portrait.
`commenting`	The motivation for when the user intends to comment about the Target. For example, to provide a commentary about a particular PDF document.
`describing`	The motivation for when the user intends to describe the Target, as opposed to (for example) a comment about it. For example, describing the above PDF's contents, rather than commenting on their accuracy.
`editing`	The motivation for when the user intends to request a change or edit to the Target resource. For example, an Annotation that requests a typo to be corrected.
`highligthing`	The motivation for when the user intends to highlight the Target resource or segment of it. For example, to draw attention to the selected text that the annotator disagrees with.
`identifying`	The motivation for when the user intends to assign an identity to the Target. For example, to associate the IRI that identifies a city with a mention of the city in a web page.
`linking`	The motivation for when the user intends to link to a resource related to the Target.
`moderating`	The motivation for when the user intends to assign some value or quality to the Target. For example, annotating an Annotation to moderate it up in a trust network or threaded discussion.
`questioning`	The motivation for when the user intends to ask a question about the Target. For example, to ask for assistance with a particular section of text, or question its veracity.
`replying`	The motivation for when the user intends to reply to a previous statement, either an Annotation or another resource. For example, providing the assistance requested in the above.
`tagging`	The motivation for when the user intends to associate a tag with the Target.

This list can be easily extended by simply adding new instances to the Motivation class. Some of these motivations are related to collaborative curation processes (e.g. replying, questioning, moderating,...) and therefore could be used in the context of the curation usage scenario described in section 8.3.2.

8.1.2. Annotating service records in NSRR

As shown in Figure 10, the web annotation model allows the linking of a service record in the NSRR identified by its unique id (GRID) and various types of body (a concept from any external knowledge model, a textual description, a keyword, another file, …)

In the context of this work, this section considers 4 different types of annotations for enriching the service description:

- A semantic tag linking a service to a domain-specific concept from an external model.
- A pragmatic annotation linking a service with a concept from FAA thesauri describing SWIM products.
- Free-text keyword to compensate any gaps in existing semantic resources.
- Textual comments to add extended description to the users.

These different types of annotations correspond to different structures of the WA body Class.

Free-text keyword and Comments

This particular type of annotation associates a textual description with a service record in NSRR. A specific type of body defined by the class oa:TextualBody has been proposed to represent embedded textual annotation. The resulting instance of the class can then be linked to information regarding the format of the annotation (using dc:format), the content of the text as value and the language used for the text (using the property dc:language).

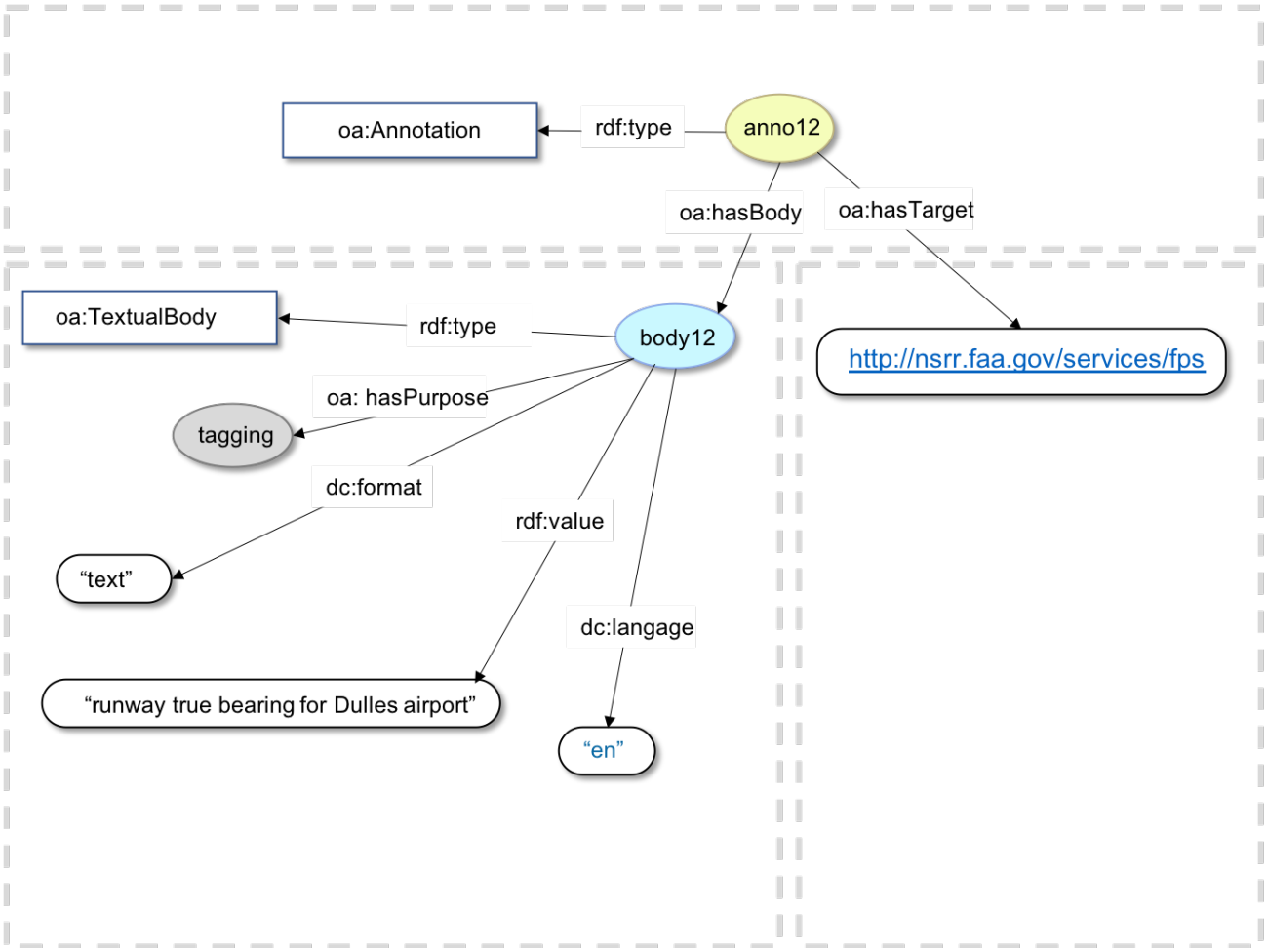


Figure 12. Textual tag as instance of oa:TextualBody

In Figure 12 is presented a diagrammatic representation of an annotation example of the FPS service associated with a long free text keyword here "runway true bearing for Dulles airport". In this context the annotation would allow the addition of user-defined information regarding the variables served by the service.

This construct can be used to capture short free-text keywords but also longer textual descriptions. To differentiate between the two types of annotations, the use of the body's purpose (using the relation hasPurpose) is proposed. Free-text keywords are used mostly for **tagging** data elements while large textual descriptions are used for **commenting** data elements. To illustrate this, this ER provides a diagrammatic representation of a comment annotation with the highlighted corresponding purpose.

With the same oa:TextualBody construct it is possible to implement two types of annotations (see Figure 12 and Figure 13). The distinction can be made by setting up a threshold of characters and associate two different purpose for short and long text.

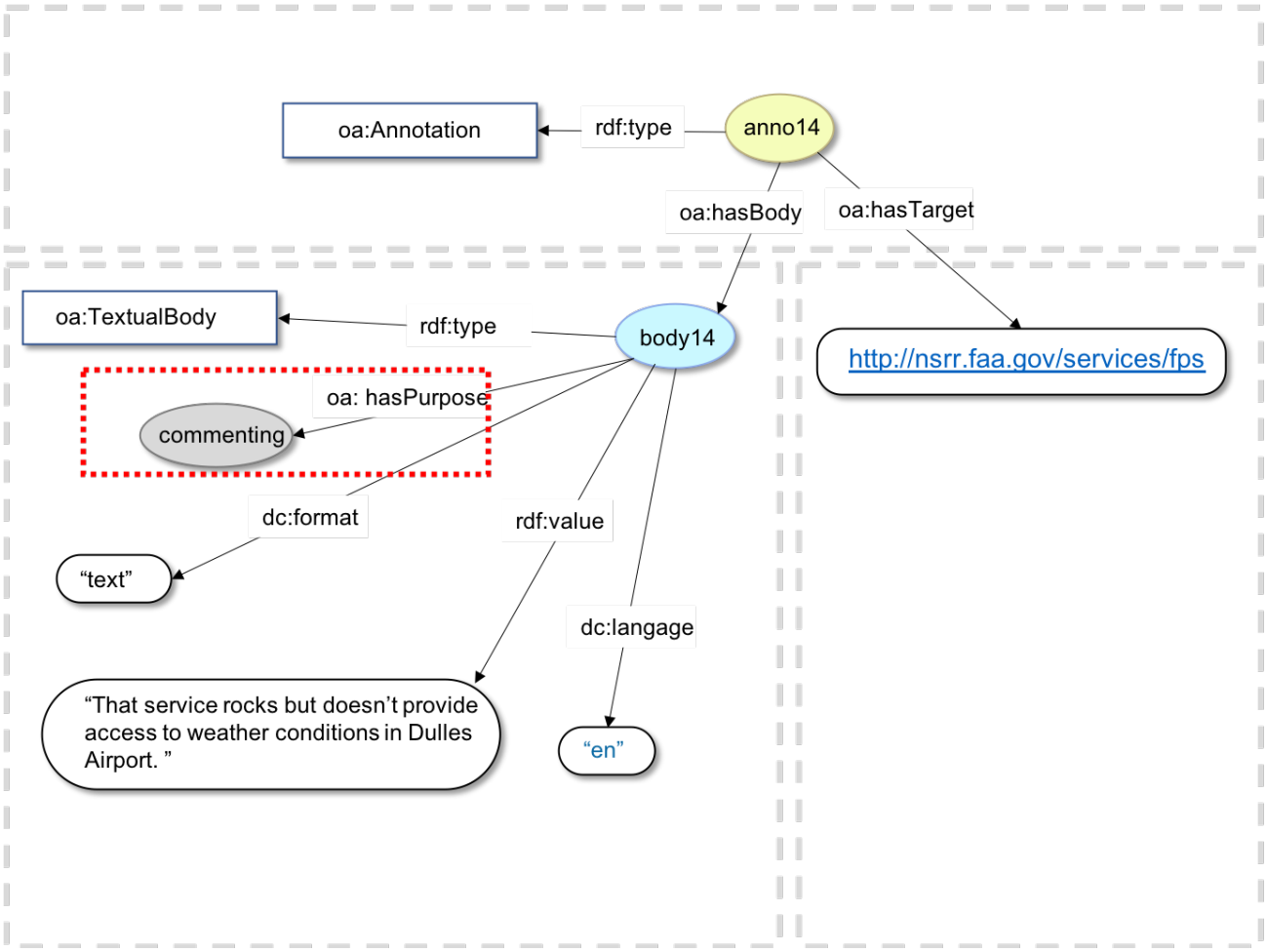


Figure 13. Comments: an oa:TextualBody which has purpose commenting

Semantic Tag data model

In this particular case, the body is an external web resource, i.e. an ontology or a thesaurus. To describe the semantic tag, the URI of the concept within the ontology has to be first considered, along with the label associated with that concept. The latter will provide a human readable information for the user.

To create semantic annotations, two different models of the body, both using the class oa:SpecificResource, are considered. The first one is simple and considers the concept URI (e.g. http://semantics.aero/service-product#flight) as the source of the semantic information and links it with the associated rdfs:label which should be fetched by resolving the concept URI (see Figure 14). As the body is an external web resource, the body is an instance of the class oa:SpecificResource and associate the concept URI as source and the rdf:label extracted from the ontology source by resolving the concept URI. This model is compact but is not necessarily convenient as it does not separate the concept from the ontology source. This will prevent users from retrieving annotations made with specific models without any processing to extract the ontology URI from the concept URIs.

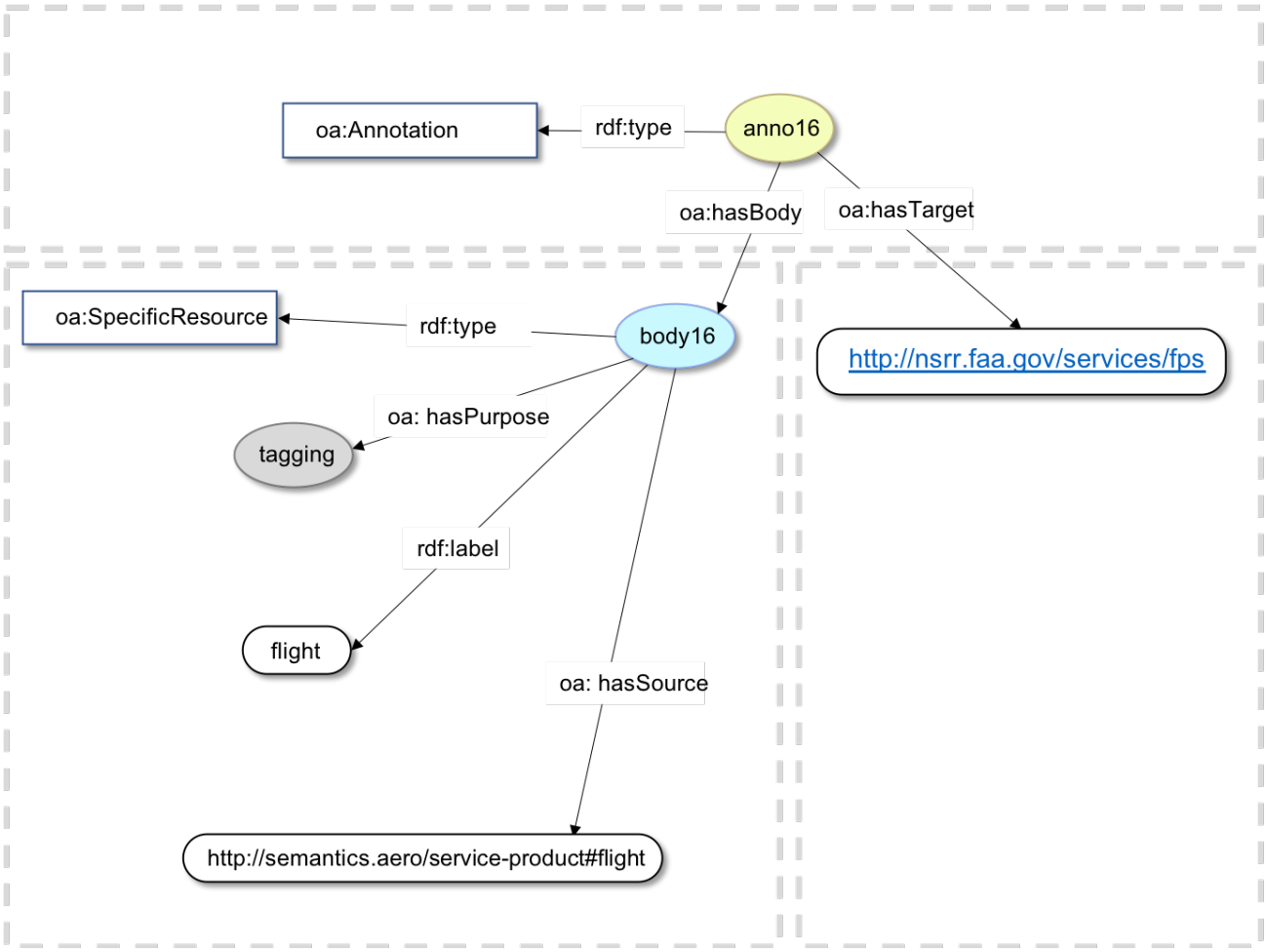


Figure 14. Semantic Tag as SpecificResource

The second option is to consider that the concept is a part of an ontology/controlled vocabulary. In this case, the class oa:FragmentSelector is instantiated. An example of this construct is shown in Figure 15. In this example, the instance "Selector1" (of type oa:FragmentSelector) links the label of the concept using the property rdf:value (here "Flight") with the source ontology (here http://semantics.aero/service-product) and the format of the data to which the value conforms to i.e. in this case rdf/xml. The label of the concept can be either extracted from the concept URI http://semantics.aero/service-product#flight or by resolving the URI and retrieving the value of the associated rdfs:label. The class oa:FragmentSelector supports several other data formats. For a list of the supported data formats, see https://www.w3.org/TR/annotation-model/#fragment-selector.

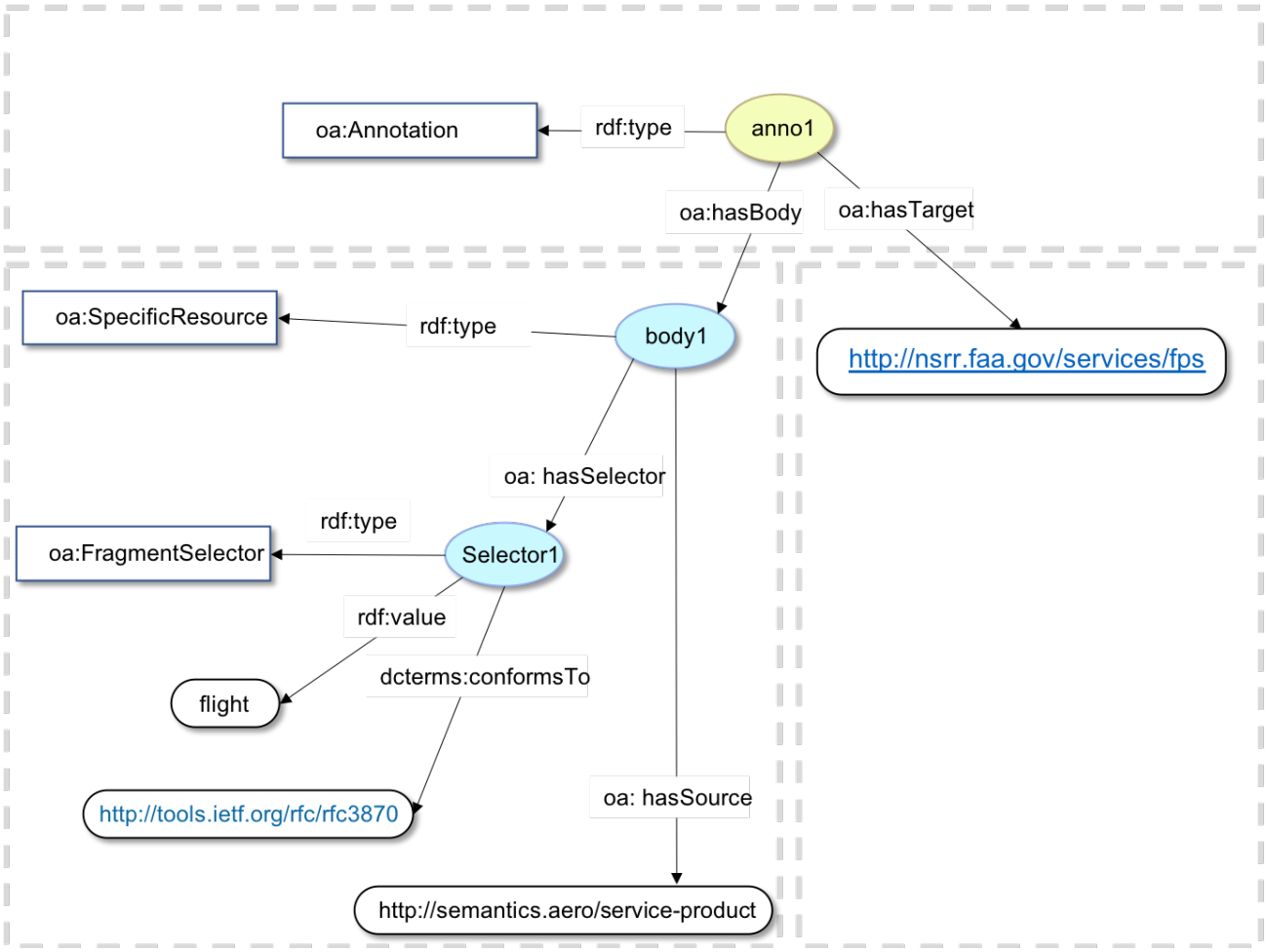


Figure 15. Semantic Tag as FragmentSelector

In the example presented in Figure 15, the semantic annotation is used to create a functional facet associated with the FPS service description. The annotation here directly links the service description with a concept defined in the service product FAA taxonomy. This annotation corresponds to the pragmatic information mentioned previously. The same model can be used to associate domain-specific concepts from ATM ontologies with NSRR service descriptions. The distinction between these two types of annotations could be made using the purpose of the annotation as proposed previously to discriminate between text keywords and comments. Pragmatic annotations would correspond to a classifying purpose while the domain-specific annotation would correspond to a tagging purpose.

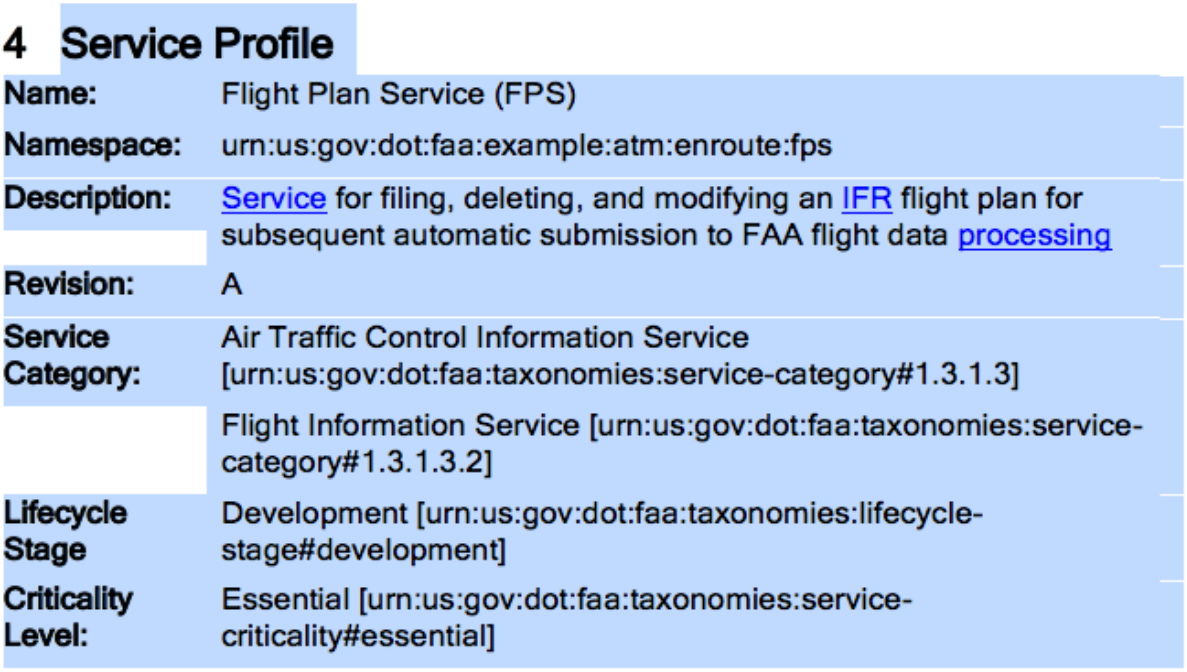
8.1.3. Annotating Service documents

As seen in Chapter 6, service descriptions published by the NSRR are linked with additional description documents in different formats (mainly text and XML files). This section aims to associate part of these documents with semantic resources. For this, the ER needs to consider models of target which will allow description of sections of the documents.

The W3C WA model proposes different models of targets to annotate various file contents. This is possible by defining dedicated structures of the target class and creating a reference to a specific position within a file using the selector class. The annotation can be done on multiple file types ranging from text to images. In the case of XML files, it is possible to use XPATH links to relate an element in the XML document with the annotation.

Annotating Text documents

This section considers the annotation of text documents such as WSDD or WSDR usually associated with the service description. Here, the ER investigates the annotation of the WSDD file provided for the fictitious FPS service. WSDD files are text templates including all the elements of the SDCM description. The annotation should associate a specific part of the document content with a concept here coming from WSDOM 1.1. This section associates the paragraph shown in Figure 16 with the concept Service Profile defined in WSDOM. The resulting annotation is represented in Figure 17.



4 Service Profile

Name:	Flight Plan Service (FPS)
Namespace:	urn:us:gov:dot:faa:example:atm:enroute:fps
Description:	Service for filing, deleting, and modifying an IFR flight plan for subsequent automatic submission to FAA flight data processing
Revision:	A
Service Category:	Air Traffic Control Information Service [urn:us:gov:dot:faa:taxonomies:service-category#1.3.1.3]
	Flight Information Service [urn:us:gov:dot:faa:taxonomies:service-category#1.3.1.3.2]
Lifecycle Stage	Development [urn:us:gov:dot:faa:taxonomies:lifecycle-stage#development]
Criticality Level:	Essential [urn:us:gov:dot:faa:taxonomies:service-criticality#essential]

Figure 16. text selection for annotation

This example used the model of semantic body described in Figure 15. The references (URL and label) have been updated to point to the WSDOM ontology.

The target description is extended compared to the previous example. The target is now linked with a specific selector for text documents, oa:TextPositionSelector and a pointer to the source. This selector has by definition two main properties oa:start and oa:ends pointing to the starting position of the annotated text and the final position of the annotated text. In this example, the position of text within the document is measured as follows: the position 0 is immediately before the first character of the text stream, the position 1 is immediately after the first character and before the second and so on and so forth. In the example here, the testbed participants used text statistics tools to estimate the position of the paragraph within the document. The extraction of the positions (start and end) should be automated by the annotation client.

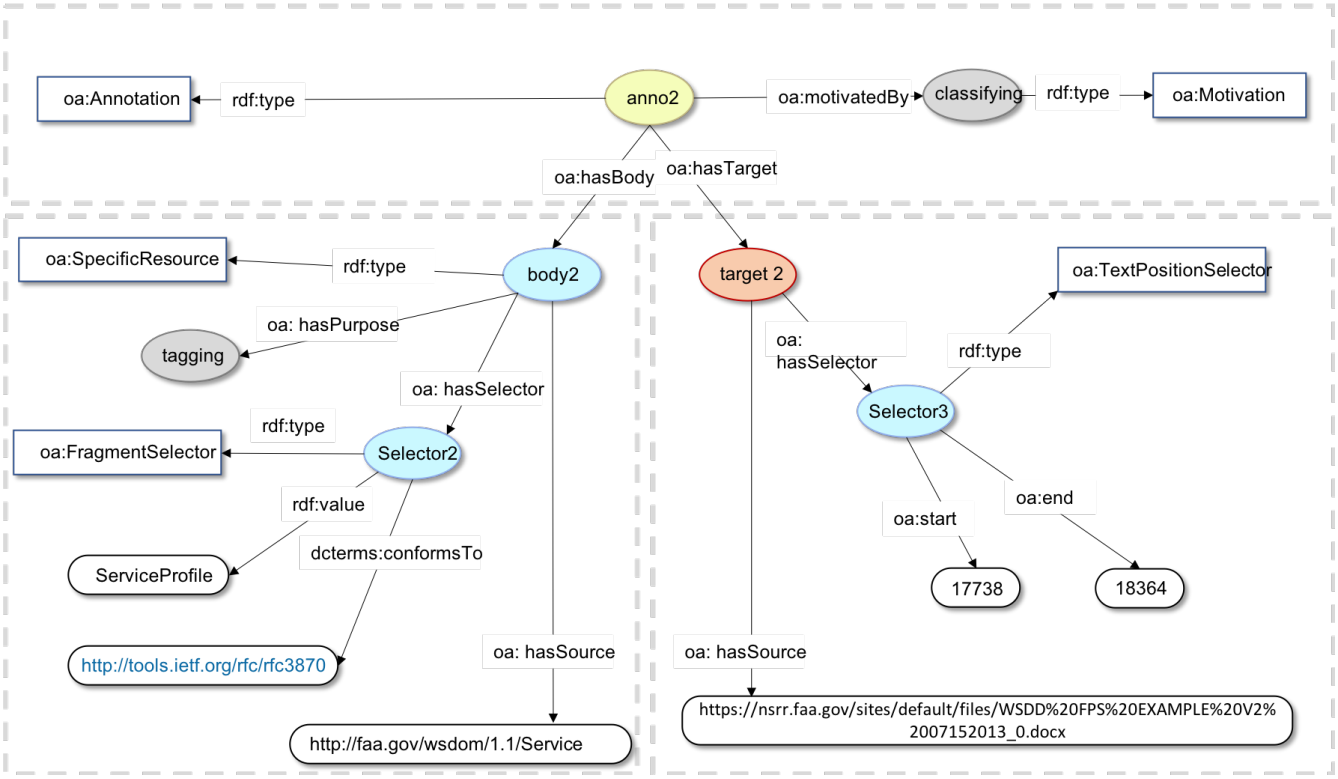


Figure 17. text content annotation

To fetch the text content associated with the annotation, a client should be able to open the text document and extract the text using the start and end position values.

Annotating XML Schemata

In this case, the ER considers the annotation of another key type of documents provided by the service provider i.e. XML schemata defining the message and data structure proposed by the service. Within the W3C standard, a specific type of "selector", the "XPathSelector" has been proposed to link a specific element of a webpage or an XML document together with an annotation. We propose, here, to link XML schemata elements with concepts defined in an ontology (here the FPS Ontology, see Annex B). The figure below shows the annotation model associating the element "aircraft" of the XML schema with the corresponding concept within the FPS ontology (see Figure 18).

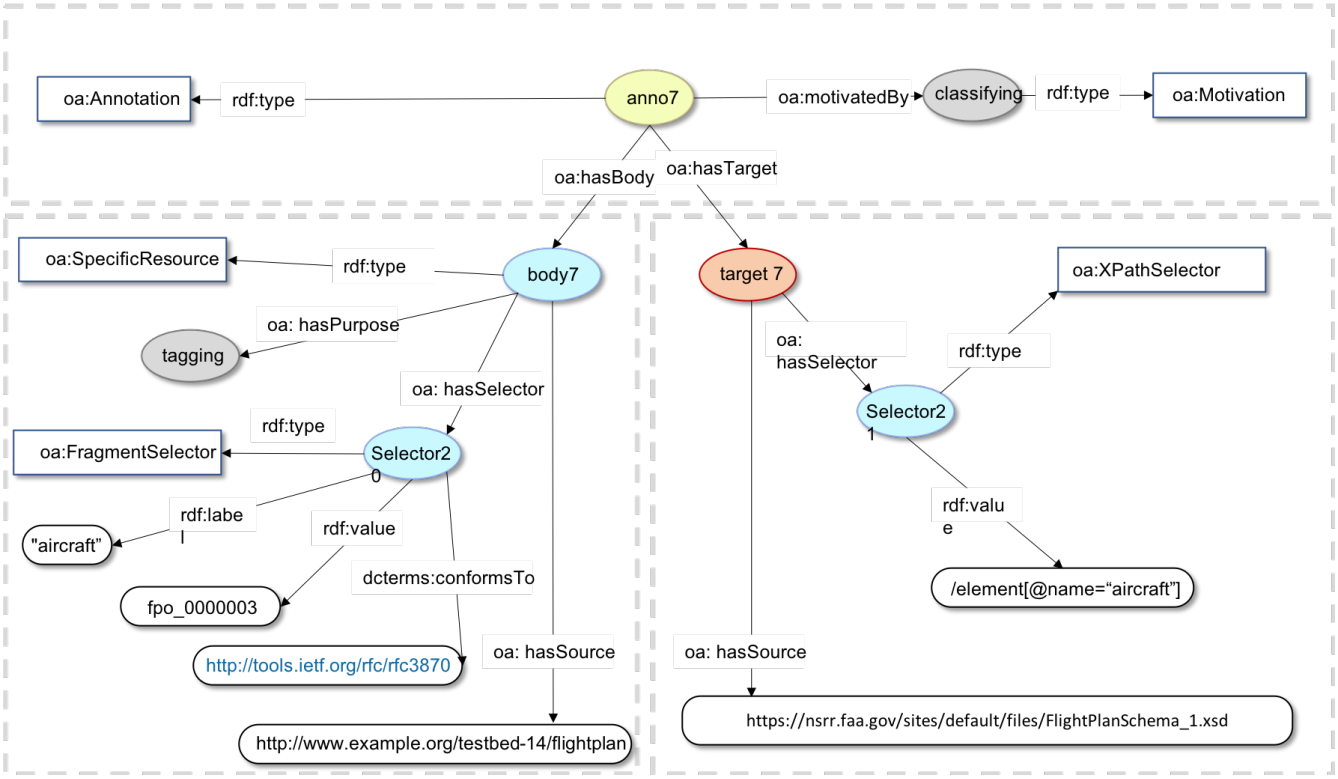


Figure 18. XML schema annotation

As for the text annotation, the target of the annotation is composed of the source pointing to the URL of the file, the selector which provide the XPath pointer within the document using the document root class as reference point.

To support our tests, the FPS Ontology has been built using Protégé 5.0.0 (see Annex B). For this example, the testbed participants designed a second version of the ontology following some of the Open Biological and Biomedical Ontology (OBO) Foundry principles (http://www.obofoundry.org/). In particular, these principles requires that each ontology elements (i.e. classes, instances and properties) should be uniquely identified (http://www.obofoundry.org/principles/fp-003-uris.html). In the current example, the concept describing an aircraft has the following unique URL: http://www.example.org/testbed-14/flightplan#fpo_0000003. This URL is associated with a human readable label associated to the concept as an rdfs:label. It is also recommended to avoid camelCase convention for the rdfs:label (http://www.obofoundry.org/principles/fp-012-naming-conventions.html). The labels should be lower case and composed words should be separated with a space (e.g. "runway true bearing" instead of "runwayTrueBearing" or "runway_true_bearing"). The second version of the FPS ontology is available in Annex B.

In order to provide both machine readable and human readable information linked to the annotation, this ER proposes here to integrate within the "FragmentSelector" the property rdfs:label associated with the concept URL. This can be done by adding a dedicated functionality to the annotation service which will either load the ontology and extract the rdfs:label associated with the URI within the document or retrieve this information through the resolution of the URI. For the latter, it is required that the ontology should be published with a triple store or within a vocabulary management service publishing the ontology.

8.1.4. Linking Service Description with Geospatial information.

A proposal has been made to add geospatial information such as bounding boxes within service

description in Testbed-12. The proposed approach was to integrate a new class within the SDCM data model which would contain geospatial information (OGC 16-039r2) [3] as shown in Figure 19.

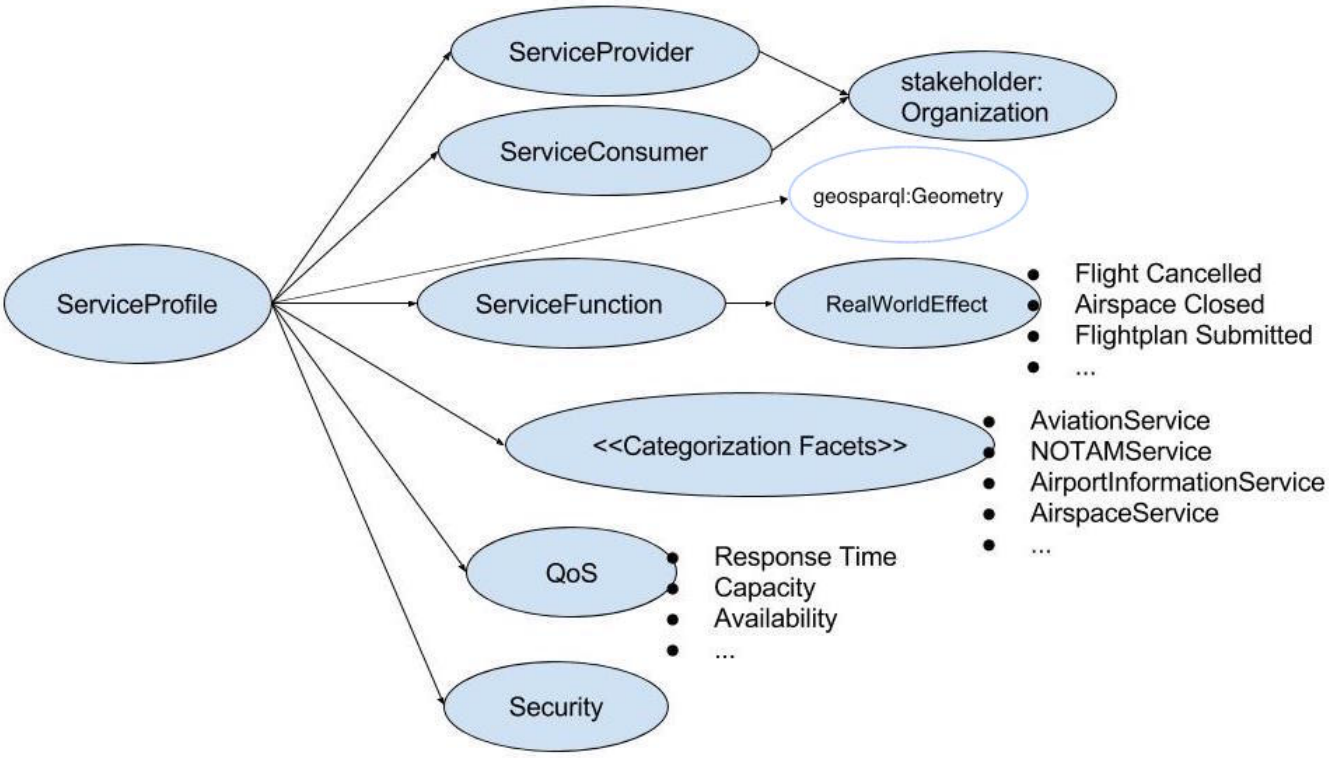


Figure 19. WSDOM extension to include geospatial information related to ATM services (from OGC 16-039r2).

We are considering here the possibility to use the WA model to associate the geospatial information as body with the service description as target.

To enable this, the body of the annotation can be used to include elements of the GeoSPARQL ontology [OGC 11-052r4]. It could thus use the same structure as the semantic tag i.e. using the SpecificResource construct. This construct ties together the source of the information and the selector which aggregates the value and format information. In this particular case, the source would be the URL of the GeoSPARQL ontology. The value could be a point or a polygon class. However, two information are missing in this model: the coordinates and the features the class represents i.e. as in the examples provided in OGC 16-039r2 an airport localized by a set of coordinates.

To support the aggregation of this information, it is proposed to create a specific selector for geospatial information which would aggregate the 2 main elements composing a spatial objects: the geometry and the feature (see Figure 20).

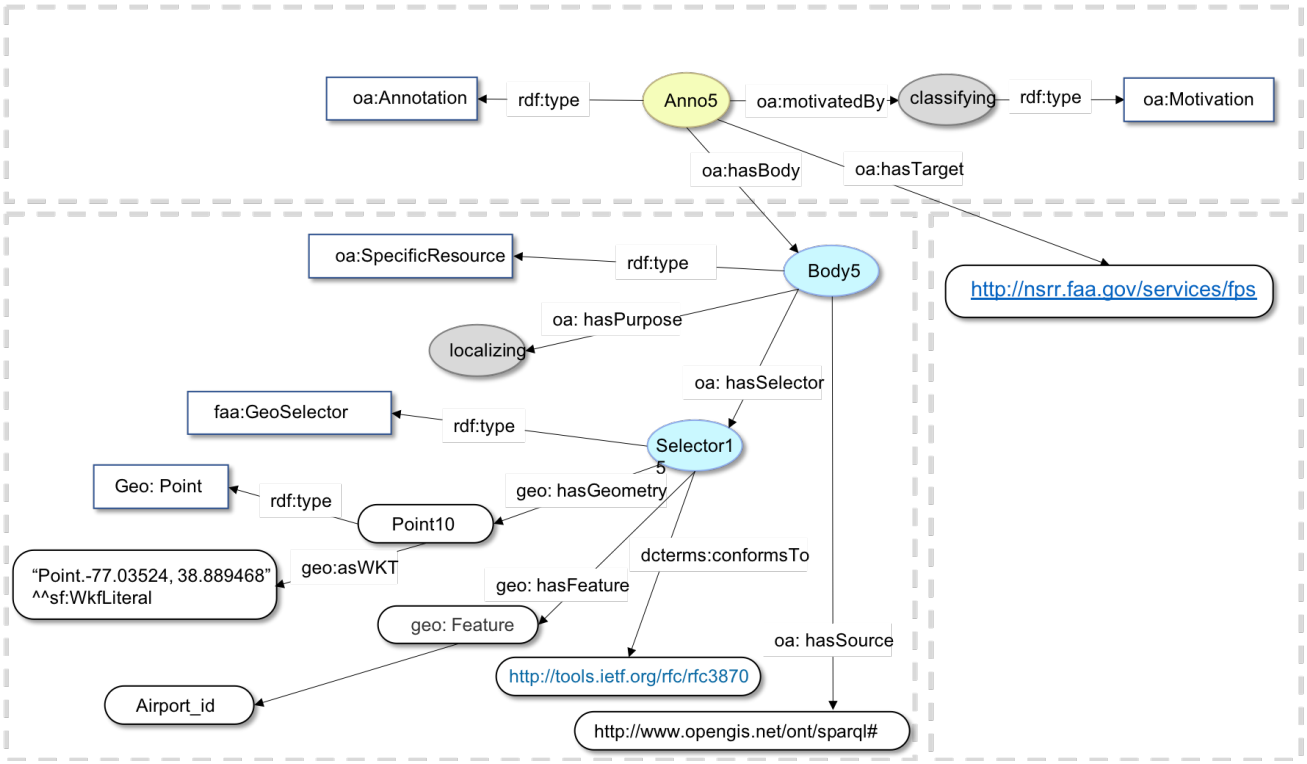


Figure 20. Annotation model for integrating GeoSPARQL information (from OGC 16-039r2).

8.1.5. Linking Service Description and Document together

The previous sections investigated different types of annotations and their specific data models based on the current W3C WA specifications. In all the cases, documents or service records within the NSRR have been linked with textual and semantic information. As mentioned in the specification, both the target and the body can refer to external web resources and documents.

As previously discussed, there are two main types of information describing the services stored in the NSRR: the SDCM based service description (NSRR service record) and additional documents (text, XML,...). In order to build a coherent information system, two problems are faced: how do we link together the annotations of these two main sources of information? Can we retrieve service records based on document annotations alone?

In order to allow clients to understand the data format of the resources linked as target and body, the specification proposes to use the rdf:type construct to link the resource with the type of file defined in the dcmi ontology i.e. dcmi:Datasets, dcmi:MovingImage, dcmi: StillImage,...

This specific structure would allow linking the service record and the associated documents by extending the resource types with FAA specific types. These resource types will describe the various types of documents provided by NSRR to describe an ATM service. The table below proposes a short list of such types which could be extended depending on the use-cases. These resources could be described within a dedicated thesaurus represented in Table 10 by the "faa:" prefix.

Table 10. FAA-NSRR Information Source Type

Type	Description
`faa: NSRR Service Record`	NSRR database entry for an ATM service

Type	Description
faa: WSDD document	FAA text template to capture service description
faa: XML Schema	XML schema describing
faa: AIXM Schema	Interchange XML schema for Aeronautic data
faa: FIXM Schema	Interchange XML schema for Flight data
faa: WXXM Schema	Interchange XML schema for Weather data
faa: WSDR document	Document describing the service requirements
faa: Data Model document	Document describing the data model of the service

Using this construct, it is now possible to enrich the annotation with FAA specific document types. As shown in Figure 21, an annotation which links a service record with a WSDD document can be generated. In this case, the body is composed of the URL of the WSDD document and the target is composed of the URL of the service record within NSRR (i.e. http://nsrr.faa.gov/services/fps in our initial use-case). The document can be subsequently annotated as shown in Figure 17 (WSDD annotation) with the addition of the target type faa:WSDD document.

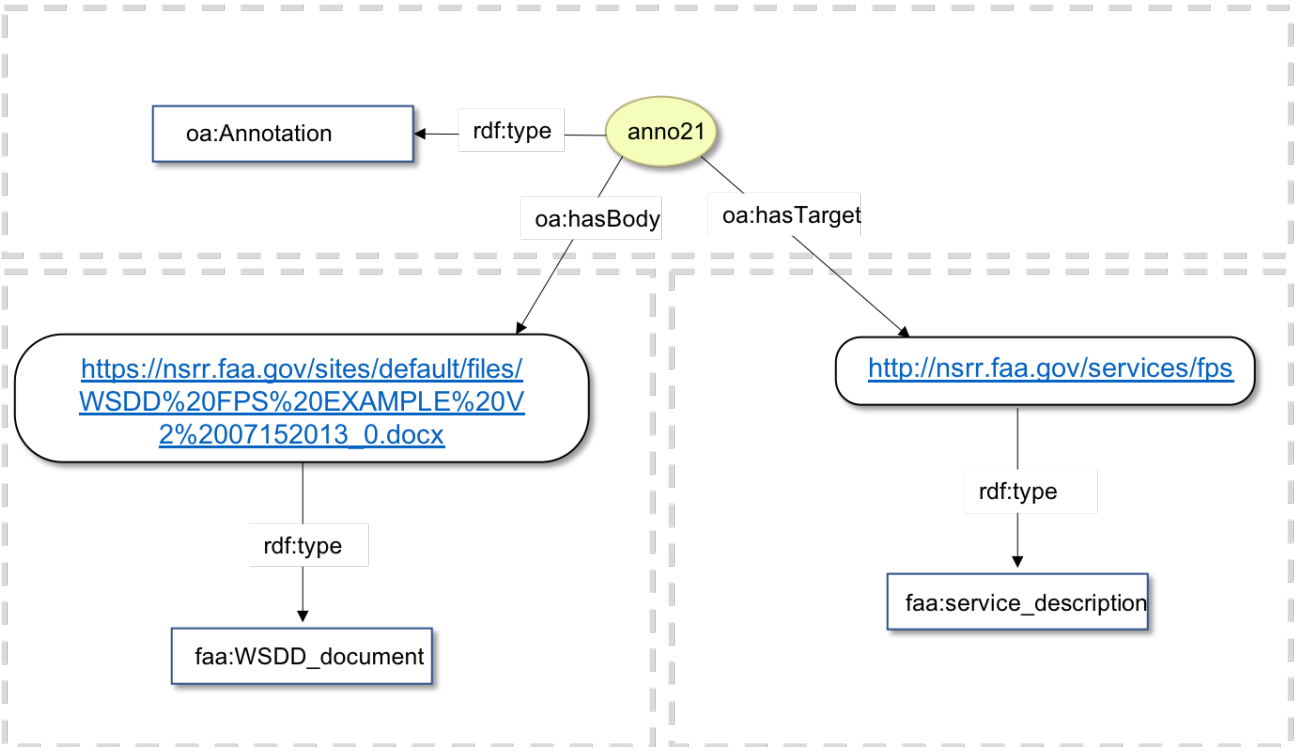


Figure 21. Annotation model for associating a service description with a WSDD document.

With this approach, queries such as follows can be answered: "give me the target of type "faa:NSRR service record" where the body is of type "faa:WSDD document" and this faa:WSDD document has body semantic tag value “alternative airport”". This type of query would provide means to retrieve service record based on the sole annotations of the associated documents.

8.2. Building an information registry based on annotation

This section proposes to build an annotation store which will act as an information registry working on top of the NSRR service database. To build such a store, several components need to be considered:

- an annotation storage system
- an annotation client
- a vocabulary management system
- a semantic index
- a search system

The following subsection describes the requirements and potential technological solutions.

8.2.1. Storing annotations

Web Annotations are serialized as JSON-LD. A simple solution would be to use off-the-shelf existing document databases such as MongoDB to store JSON-LD files. This solution would provide a structured storage allowing to make simple queries on the annotations and retrieve services accordingly. However, it does not allow to make SPARQL based queries. To harness the semantic added value, it would be required to convert the JSON-LD documents into RDF graphs and store these graphs in existing triple stores or graph databases providing a SPARQL interface such as Apache Jena Fuseki or RDF4J Server. To provide programmatic access to the annotation, the annotation store should be connected to a REST API following the W3C Web Annotation protocol described in chapter 7 and described using the OpenAPI specification for machine discovery (https://swagger.io/specification/).

8.2.2. Creating and managing annotations

The client should provide a simple user interface to support the creation and management of annotations, the visualization of the existing annotations and a search engine to retrieve services based on annotations. This client should be integrated with the NSRR web interface to provide annotation capabilities to users and NSRR curators (see section usage below). The process for creating annotation should be simple and support the different types of annotation. Once the body is defined, the client should serialize the annotation into a JSON-LD structure and store it within the annotation database. For semantic tagging, it is necessary to provide a fast access to the related ontologies, controlled vocabularies and thesauri through an auto-completion function. Finally, the client should allow to graphically annotate file content (mostly text and XML documents).

8.2.3. Vocabulary management system and semantic index: supporting semantic annotations

As discussed in the previous section, users of the annotation service should be able to have fast and easy access to the existing semantic resources through an auto-completion feature. Indeed, it is not possible for a user, even an expert user, to know all the existing ontologies, controlled vocabularies

and thesauri available to describe ATM data. To build such key functionality, semantic resources (i.e. ontologies, controlled vocabularies, thesauri,…) should be published and available through a unique central repository. This repository should provide both an interface for human to navigate and search through the resources and a programmatic interface to access the content and metadata of these resources i.e. the different types of relations and properties linking concepts together (hierarchical relation, object properties,…) as well as the different metadata fields describing the semantic resource and its content (e.g. rdfs: label, skos: definition,…). Such vocabulary management systems are key components of the semantic ecosystems in various scientific fields such as the biomedical domain for instance. In this field, most of the ontologies and thesauri are available through the Bioportal semantic repository (https://bioportal.bioontology.org/) or the Ontology Look Up Service proposed by the European Bioinformatics Institute (EBI, https://www.ebi.ac.uk/ols/index) (for a recent short review on existing vocabulary management systems, see Goldfarb and Le Franc [14].

In order to provide fast access to the semantic content, the vocabulary management system should be integrated with a semantic index. This component should index the minimal necessary information to annotate documents i.e. the human readable label, the concept URI, the definition, synonyms and provenance.

8.2.4. Automating annotations

Annotations can be automated by linking text mining tools together with the annotation service. This link should be done at two different levels: the REST API for automatically serializing and storing the annotations and accessing the semantic index, in order to provide the text mining tools access to the domain-specific semantic reference.

Although the automation of annotation is possible, the results of this process should be validated by human intervention to remove any ambiguities. Finally, to further extend the efficiency of the automated annotation, the use of Machine Learning algorithms working on the semantics could be considered.

8.2.5. B2NOTE: a semantic annotation service for scientific datasets

To support scientists to publish, aggregate and reuse scientific data, pan-European data infrastructures have been created through the funding support of the European Union Horizon 2020 program. In the context of the EUDAT infrastructure (http://eudat.eu), a semantic annotation service, called B2NOTE, has been developed (http://b2note.eudat.eu). This service integrates with other services as a widget (using the HTML iFrame construct) to annotate datasets and data elements. Users can create three main types of annotations: semantic tags, free text keywords and comments.

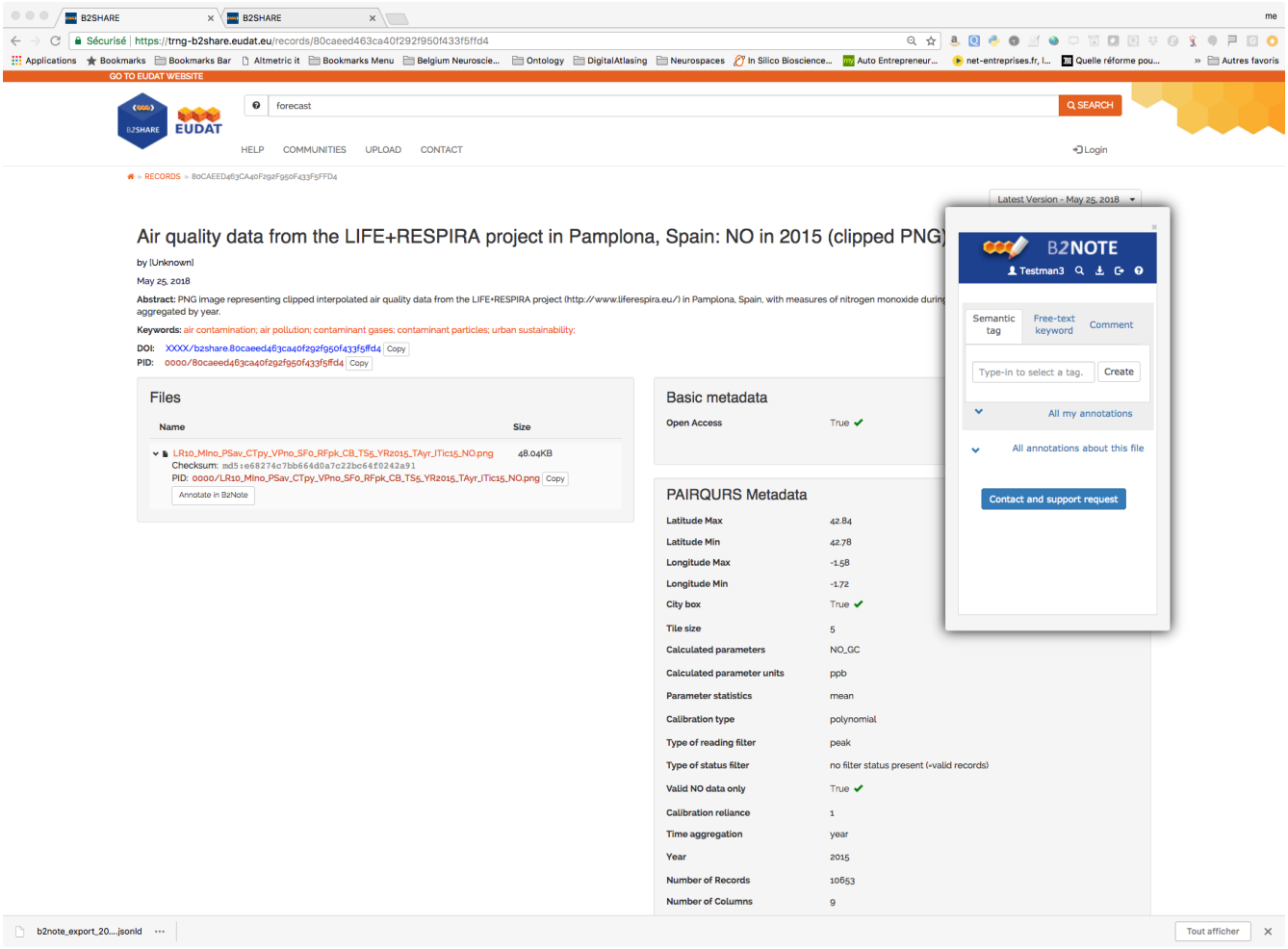


Figure 22. B2NOTE integrated as a widget with the B2SHARE data repository

B2NOTE is an Open Source solution (https://github.com/EUDAT-B2NOTE/b2note) deployed in a production environment for scientists to annotate data stored and published in the B2SHARE repository (https://b2share.eudat.eu). The service stores annotations in a centralized MongoDB database and provides a search engine for retrieving datasets based on the existing annotations as well as a REST API to programmatically retrieve annotations. This service is now part of the European Open Science Cloud Hub (EOSC-Hub) service portfolio and will be integrated with the other data services of the EOSC infrastructure. Figure 22 shows a screen capture of the annotation service associated with the B2SHARE data repository.

To support the semantic annotation, a dedicated semantic index has been created aggregating semantic concepts from the different biomedical semantic resource repositories i.e. Bioportal and EBI-OLS. The current index provides access to more than 5 million concepts. As described in Goldfarb and Le Franc [14], the challenge is to build a pluri-disciplinary semantic index which will provide access to semantic resources from as many domains as possible. ATM semantic information could be part of this index to support research work using such information.

8.3. Using Web Annotations in different contexts

This section investigates the various processes which could be supported by the use of web annotation as a facilitator to capture the information at different level of the business chain.

8.3.1. Service provider annotation at service registration time

Information about the type of data should be provided by the service provider when he register his service. This information should then be verified and validated and can be used to make queries to retrieve services. In this scenario, the service provider could directly embed the semantics within the documents describing the services by annotating manually or semi-automatically the various documents describing the service (i.e. WSDD, WSDR and XML schemata). As discussed in the previous section, document annotation could be then used to retrieve services without annotating the service record within NSRR.

8.3.2. Expert curation of the registry content

The service provider does not provide this information. It should be added during the curation/verification/validation process by NSRR curators. In this scenario, web annotation could support the addition of semantic information without changing the documents or even help the curator during the process. It could also be used to support the interaction with the service provider by providing comments on the current descriptions (change request, curation step,...) or even annotating with defined curation step for tracking the progress in the curation process using the motivation and purpose proposed by the WA standard. For the latter, the process could be supported by extending this list of motivation and purpose with specific terms.

8.3.3. User based search

When users are searching for services, they could add up information using web annotation to have a more personalized experience. They could rank the search results or simply add a missing facet to aggregate specific services. User comments could also be added and published to provide user feedbacks on the service for specific usages. These comments could be then used by the service provider to improve their service or even for providing missing information to the users.

8.4. From Web Annotation to RDF graphs

This part considers how annotations encoded using the web annotation model could be transformed either directly as RDF to be accessible through a SPARQL query interface or even used to build a more formal knowledge graph.

JSON-LD has been designed to be directly converted into RDF. Dedicated libraries exist in various programming languages and provide a one-to-one mapping between the JSON-LD structure and the corresponding RDF structure. RDF annotations should then be stored within a dedicated triple store or a graph database providing a SPARQL interface.

In this case, we are just considering the direct conversion of the annotations. As we mentioned in the introductory section on the web annotation model, the web annotation data model provides a way to create an implicit relation of "aboutness" between two resources be it two files or a file and semantic concept within an ontology. In some cases, this implicit relation will not be enough to represent the complexity of the underlying structure associating the different data elements. As web annotations can be serialized as RDF, it is therefore possible to use the annotations as a basis to create more detailed knowledge models. For this purpose, annotations could be transformed into a more detailed set of triples describing the relation between the service and the concept.

8.5. Business value

This ER describes the use of a web annotation framework as an information registry for the NSRR registry. This usage can be extended to the context of the SWIM common registry service. The annotation system creates a semantic interoperability layer on top of the existing registries allowing to align the content and formats with the same semantic resources. The annotation system B2NOTE has been developed with the aim of allowing scientists to aggregate datasets from multiple distributed and heterogeneous data repositories using common sets of keywords. The use of such a system for annotating the various service registries would allow the creation of a semantic interoperability layer across distributed service registries. Any service description from any registry could be annotated as long as the registry provides unique URL to identify without ambiguity the service description record and the associated documents. This platform-independent system would be an ideal candidate to create an information registry for the CSR, allowing to federate search across distributed and heterogeneous service registries as illustrated in Annotating distributed and heterogeneous ATM service registries.

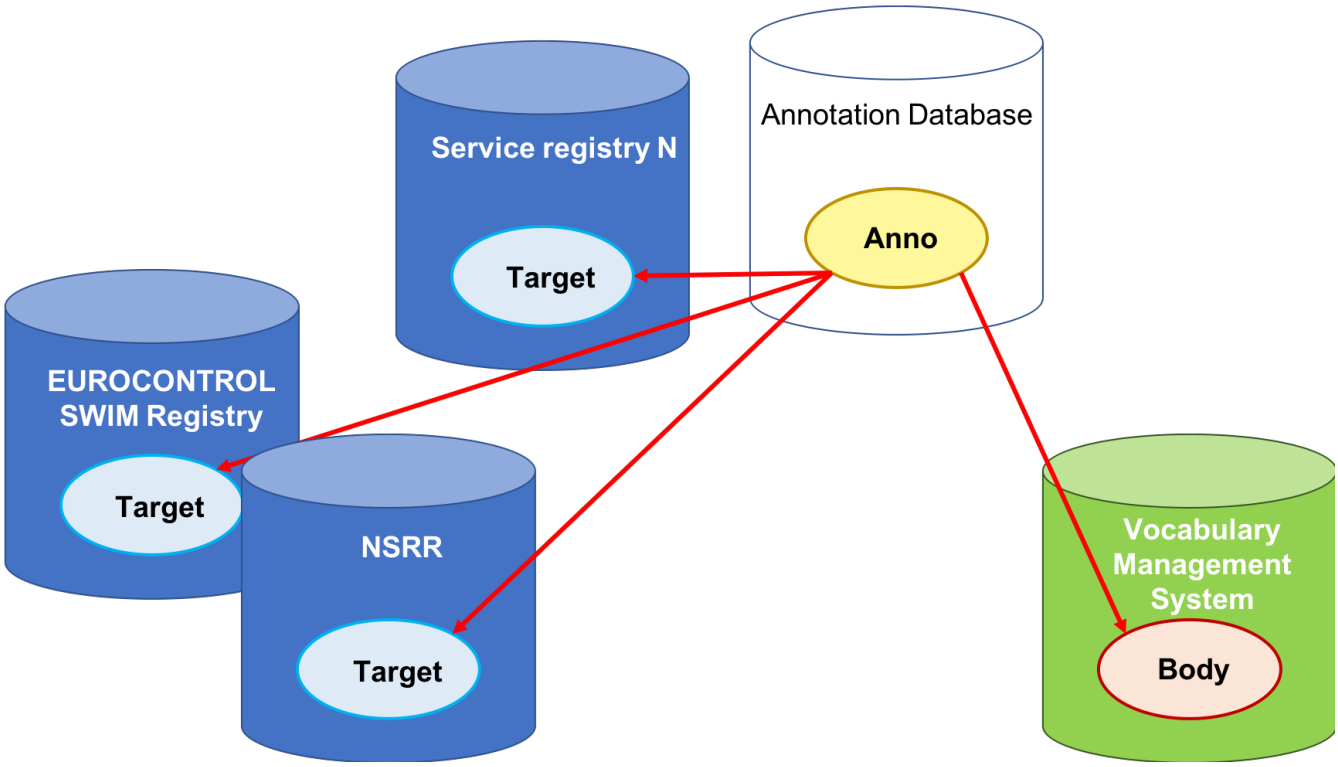


Figure 23. Annotating distributed and heterogeneous ATM service registries

The technical solution presented in this ER represents a cheap and easy way to build a solution to rapidly and efficiently address service discoverability problems in the NSRR. A web annotation based-service provides a unique framework for adding semantics without modifying the resource. Therefore, reducing the entry cost to add domain-specific semantics to the NSRR registry. Using existing annotation services such as B2NOTE would reduce the cost of development, deployment and maintenance.

Finally, as we discussed previously, annotations can be used to establish a collaborative service registration and used to extend the user experience.

Chapter 9. Summary and Recommendations

This ER investigated various practical solutions to integrate domain-specific semantics with the existing data describing ATM services in NSRR (Service description and associated documents). During this work, the testbed participants identified different barriers and necessary elements of the architecture for creating a semantic information registry on top of the NSRR. As a cheap and easy-to-use alternative, the testbed participants proposed to build the information registry based on semantic annotations. These annotations are following the W3C Web Annotation data model and are serialized in JSON-LD. While this ER was more focused on the practical aspects of building an information registry, the associated Testbed-14 Semantically Enabled Aviation Data Models Engineering Report [OGC 18-035] [2] has been reviewing the various semantic enablement techniques supporting the description of semantics of services in the NSRR and of the NSRR data set. In addition, it does provide recommendations for designing domain-specific ontologies.

In this final section, we discuss the different changes we consider necessary to build an information registry using the different pragmatic semantic-enablement approaches outlined in this ER.

9.1. Implementing Linked Data principles

To enable the integration of the data silos currently existing in the NAS, a common framework to represent and link information is needed. To address this challenge, the current infrastructure should be semantic-enabled by using the different techniques suggested in this ER and the Testbed-14 Semantically Enabled Aviation Data Models Engineering Report [OGC 18-035]. One of the key pillars to build a semantically enabled discovery system is to adhere to Linked Data principles. These principles support the creation of an interoperability layer among the different types of data. The information about ATM services should be made accessible as Linked Data and using best practices as defined by the W3C Spatial Data on the Web Best Practices [12: https://www.w3.org/TR/sdw-bp/] and 5-star Linked Data principles [13: https://www.w3.org/DesignIssues/LinkedData.html]. By adhering to the Linked Data principles, machines will be able to access and mine more efficiently the NSRR content. As an example, services are identified with a URL pointing to a landing page. Unless a client has the capability to parse this landing page to identify the links to follow to fetch more information, the landing page represents a dead end for any client. If the LD principles are applied, the URL identifying the service should be the entry point for a client to access the necessary information such as a Service Model. In several service descriptions, XML schemata are packaged into a zip file which prevents any regular client from accessing these XML schemata individually. To leverage semantics for service discovery and retrieval, future tasks should involve the transformation of the NSRR repository to become LD compliant.

9.2. Developing ATM knowledge models

The work presented here focuses on the integration of domain-specific semantic to support more efficient user search. Several existing thesauri have been developed by the FAA to provide additional facets to the service discovery. Unfortunately, none of these thesauri cover the semantics of aeronautical data, flight data and weather data. As far as we know, only one ontology exists describing aeronautical data, the NASA ATM ontology. The lack of common, well-defined, modular ontologies for aviation is one of main limitations to this work. Future testbeds should investigate the creation of modular domain-specific ontologies using the NASA ATM ontology as a starting

point. The various modules should cover the gaps with the current existing standards (e.g. AIXM, WXXM,or FIXM).

9.3. Vocabulary management service and semantic index

ATM ontologies and thesauri are currently available for download and therefore cannot be queried to retrieve information encoded within the ontology such hierarchical relations, logical relations or even synonym information. To allow clients to access such information and make inferences with the annotations, these semantic resources should be published by a vocabulary management system which will present the semantic content through an API for clients to use. In the biomedical domain, vocabulary management systems such as Bioportal (https://bioportal.bioontology.org/) or EBI Ontology Look Up Service (EBI-OLS, https://www.ebi.ac.uk/ols/index) provide access to more than 600 ontologies and controlled vocabularies relevant for the field. Concepts and relations within these ontologies can be queried and retrieved from the dedicated API. For a non-exhaustive review of the existing vocabulary management systems see [14]. These systems provide services for the management and governance of controlled vocabularies. As we have seen in this work, these semantic resources play a crucial role in the semantic "tagging" of resources managed by National Airspace System (NAS) (Datasets, Services, Maps, Layers, Documents,etc..) and in the search and discovery of these resources. In the context of the annotation system, the semantic content should be made available to the user at the time of the annotation creation. Indeed, it is not possible to require any user to know all the semantic concepts from the different thesauri and ontologies. Semantic annotation should be supported by a semantic index providing fast access through an auto-completion function to existing concepts available for annotation. The Vocabulary Management System is a central element of the semantic infrastructure. For future testbeds, the creation of an ATM Vocabulary Management System and the requirements, design and implementation of an API for a controlled vocabulary services that manage ontologies and taxonomies (encoded in SKOS) should be investigated.

9.4. SRIM Semantic Registry

As has been discussed in this document, semantic annotations represent an easy way to integrate domain-specific semantics with the NSRR content without changing the service record. It can also be used to associate semantics of the service description. However, it is not necessarily the best solution for adding semantics to the service metadata. 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 QoS, 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.

9.5. SWIM Application Profile based on SRIM

For future testbeds, an SRIM Service Model should be investigated to identify additional properties that align with the SDCM model better. 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 search and discovered and 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.

9.6. 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. The 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 semantic registry to be indexed so can be searched and discovered.

Appendix A: RIM API - FPS Service Description

```
<ServiceDescription xmlns="http://swim.aero/rim/1.0.0">
    <Profile>
        <ServiceName>Flight Plan Service (FPS)</ServiceName>
        <ServiceVersion>1.0.0</ServiceVersion>
        <ServiceDescription>(This fictitious service is for instructional use only and
cannot be consumed) A service for filing, updating, or canceling an IFR (Instrument
Flight Rules) flight plan.
</ServiceDescription>
        <sd:Categories xmlns:sd="http://swim.aero/rim/1.0.0">
            <sd:Category>
                <sd:Title>ATM Service Category</sd:Title>
                <sd:Value>Flight Planning</sd:Value>
            </sd:Category>
            <sd:Category>
                <sd:Title>SWIM Service Product Category</sd:Title>
                <sd:Value>Flight</sd:Value>
            </sd:Category>
            <sd:Category>
                <sd:Title>Lifecycle Status</sd:Title>
                <rim:Value xmlns:rim="http://swim.aero/rim/1.0.0">
Development</rim:Value>
            </sd:Category>
        </sd:Categories>
        <Provider>
            <Name>FAA En Route Services Modernization Group (ESMG)</Name>
            <Description>A program within the FAA Air Traffic Organization responsible
for developing Web services.
</Description>
            <WebPage>http://www.faa.gov/air_traffic/flight_info</WebPage>
            <PointsOfContact>
                <POC>
                    <Name>John  D. Doe</Name>
                    <Function>ATO-X ESMG Manager</Function>
                    <Phone>(609) 444-5555</Phone>
                    <Email>Joe.doe@faa.gov</Email>
                </POC>
                <POC>
                    <Name>Mark Kaplun</Name>
                    <Function>Governance Lead</Function>
                    <Phone>23423423423</Phone>
                    <Email>mark.kaplun@faa.gov</Email>
                </POC>
                <POC>
                    <Name>Carol Uri</Name>
                    <Function>SWIM PO Contact</Function>
```

```
                    <Phone>555-555-5555</Phone>
                    <Email>curi@faa.gov</Email>
                </POC>
            </PointsOfContact>
        </Provider>
        <Functions>
            <Function>
                <Description>File a flight plan.</Description>
                <RealWorldEffect>A flight plan has been filed and persists in the FAA
Web server for distribution to the FAA flight data processing application within some
parameter time of the estimated departure time.
</RealWorldEffect>
            </Function>
            <Function>
                <Description>Cancel a flight plan.</Description>
                <RealWorldEffect>A previously filed flight plan has been retracted
before being submitted to FAA Air Traffic Services, thereby reducing the flight plan
processing load and systemic workload of the FAA air traffic planning system.
</RealWorldEffect>
            </Function>
            <Function>
                <Description>Change destination aerodrome of a flight
plan.</Description>
                <RealWorldEffect>The destination aerodrome of a filed flight plan has
been changed.
</RealWorldEffect>
            </Function>
        </Functions>
        <SecurityMechanisms>
            <SecurityMechanism>
                <Name>Authorization</Name>
                <Description>The FPS deploys role-based access control (RBAC) for
implementing authorization in accordance with ANSI/INCITS 359-2004.  Two roles are
defined, “Reader” (a user who only has permission to view a filed flight plan) and
“Originator” (a user, generally a pilot or operator, who submits a flight plan and has
permission to file and subsequently modify or cancel the filed flight plan).
</Description>
                <RegulatingProtocol>
                    <Title></Title>

<Location>http://www.cs.purdue.edu/homes/ninghui/readings/AccessControl/ANSI+INCITS+35
9-2004.pdf</Location>
                </RegulatingProtocol>
            </SecurityMechanism>
            <SecurityMechanism>
                <Name>Integrity</Name>
                <Description>The FPS deploys Transport Layer Security (TLS) Version
1.2, RFC 5246; therefore data is checked for possible corruption.
</Description>
                <RegulatingProtocol>
                    <Title></Title>
```

```
                    <Location>http://tools.ietf.org/html/rfc5246</Location>
                </RegulatingProtocol>
            </SecurityMechanism>
            <SecurityMechanism>
                <Name>Non-Repudiation</Name>
                <Description>The FPS complies with W3C's Recommendation for XML
Signature Syntax and Processing to ensure that each user's message is digitally signed.
</Description>
                <RegulatingProtocol>
                    <Title></Title>
                    <Location>http://www.w3.org/TR/xmldsig-core</Location>
                </RegulatingProtocol>
            </SecurityMechanism>
            <SecurityMechanism>
                <Name>Authentication</Name>
                <Description>The FPS requires each service consumer to authenticate
itself to the FPS at the transport level by deploying a Username/Token credential in
accordance with the Web Services Security UsernameToken Profile 1.0, OASIS Standard
200401.
</Description>
                <RegulatingProtocol>
                    <Title></Title>
                    <Location>http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
username-token-profile-1.0.pdf</Location>
                </RegulatingProtocol>
            </SecurityMechanism>
        </SecurityMechanisms>
        <Policies>
            <Policy>
                <Title>Flight Plan Service (FPS) Policy Document Version 1</Title>
                <Location>https://www.faa.gov/atm/policies/fps-policy.xml</Location>
            </Policy>
            <Policy>
                <Title>Runtime Policy</Title>

<Location>http://www.faa.gov/air_traffic/flight_info/operation_policy.pdf</Location>
            </Policy>
            <Policy>
                <Title>FAA Order 1370.92A, Password and PIN Management Policy</Title>
                <Location>
http://www.faa.gov/documentLibrary/media/Order/1370.92A.pdf</Location>
            </Policy>
            <Policy>
                <Title>NIST FIPS Publication 200, Minimum Security Requirements for
Federal Information and Information Systems</Title>
                <Location>http://csrc.nist.gov/publications/fips/fips200/FIPS-200-
final-march.pdf</Location>
            </Policy>
        </Policies>
        <QualityOfService>
            <QualityOfServiceParameter>
```

```
                <Name>Response Time</Name>
                <Value>3</Value>
                <UnitOfMeasure>Seconds</UnitOfMeasure>
            </QualityOfServiceParameter>
            <QualityOfServiceParameter>
                <Name>Capacity</Name>
                <Value>20 per minute</Value>
                <Definition>Number of service requests that the service can
accommodate within a given time period.
</Definition>
                <CalculationMethod>Simple count.
</CalculationMethod>
                <UnitOfMeasure>Whole positive number, per period of
time</UnitOfMeasure>
            </QualityOfServiceParameter>
            <QualityOfServiceParameter>
                <Name>Availability</Name>
                <Value>≥ 99.900</Value>
                <Definition>Probability that the service is present or ready for
immediate use.
</Definition>
                <CalculationMethod>100 * ((24 – Total Outage Time) / 24). Measurements
are taken daily and apply to the preceding 24-hour period.
</CalculationMethod>
                <UnitOfMeasure>Percentage, accurate to 3 decimal
places</UnitOfMeasure>
            </QualityOfServiceParameter>
        </QualityOfService>
        <EnvironmentalConstraints>
            <Constraint>
                <Description>The FPS operates within the FAA Telecommunications
Infrastructure (FTI) and is subject to its performance constraints.  All FPS requests
and responses are brokered via the NAS Enterprise Messaging Service (NEMS).
</Description>
            </Constraint>
        </EnvironmentalConstraints>
    </Profile>
    <Model/>
    <Grounding/>
</ServiceDescription>
```

Appendix B: Flight Plan Service Ontology

This ontology has been created using the Ontology building software Protégé v5.0.0. This annex proposes two different implementations of the ontology.

In the first code snippet, the concept URIs contain the label which is used as a unique key to identify the concept e.g. http://www.example.org/testbed-14/flightplan#flight

In the second code snippet, the ontology design is aligned with the OBO Foundry principles (http://www.obofoundry.org/):

- Unique identifier for each ontology element: the identifier is build using the three-letter acronym of the ontology name followed by an underscore and a 7-digit number.
- Human readable labels do not follow the usual CamelCase convention. All the words composing the label are in low case letters except when a Capital letter is required (i.e. name of a person, a city,...). The labels are associated with the ID using the rdfs:label construct.

FlightPlanOntology.owl

```
<?xml version="1.0"?>
<rdf:RDF xmlns="http://www.example.org/testbed-14/flightplan#"
     xml:base="http://www.example.org/testbed-14/flightplan"
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:owl="http://www.w3.org/2002/07/owl#"
     xmlns:xml="http://www.w3.org/XML/1998/namespace"
     xmlns:flightplan="http://www.example.org/testbed-14/flightplan#"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
     xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
     xmlns:dc="http://purl.org/dc/elements/1.1/">
     <owl:Ontology rdf:about="http://www.example.org/testbed-14/flightplan">
        <owl:imports rdf:resource="http://protege.stanford.edu/plugins/owl/dc/protege-
dc.owl"/>
        <dc:creator xml:lang="en">Yann Le Franc</dc:creator>
        <dc:description xml:lang="en">An ontology describing the domain specific
concepts used in the
        Fligh Plan Service XML schema. This ontology is just an exemple of structured
vocabulary
        supporting the annotation of the flight plan XML schema.

        This ontology has been built using the OBO foundry principles
        (http://www.obofoundry.org/principles/fp-000-summary.html) i.e. a unique
identifier for each
        element of the ontology composed of the 3 letter acronym of the ontology name,
an underscore
        and a 7 digit number; a human readable label separating the different words
used in the label
        with a space and without using capital unless it requires it. The human
readable label is
        associated to the concept ID using the
        rdfs:label element.
```

```
        Ontology metadata is based on Dublin Core elements</dc:description>

    <owl:versionInfo rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
1.0</owl:versionInfo>
    <dc:source rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
>https://nsrr.faa.gov/sites/default/files/FlightPlanSchema_1.xsd</dc:source>
    </owl:Ontology>



    <!--

///////////////////////////////////////////////////////////////////////////////////
/
    //
    // Annotation properties
    //

///////////////////////////////////////////////////////////////////////////////////
/
     -->




    <!-- http://www.example.org/testbed-14/flightplan#alternative_label -->

    <owl:AnnotationProperty rdf:about="http://www.example.org/testbed-
14/flightplan#alternative_label">
        <rdfs:label xml:lang="fr">alternative label</rdfs:label>
    </owl:AnnotationProperty>



    <!--

///////////////////////////////////////////////////////////////////////////////////
/
    //
    // Classes
    //

///////////////////////////////////////////////////////////////////////////////////
/
     -->




    <!-- http://www.example.org/testbed-14/flightplan#flight_plan -->
```

```
    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#flight_plan">
        <rdfs:label xml:lang="en">flight plan</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#flight_plan_ID -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#flight_plan_ID
">
        <rdfs:subClassOf rdf:resource="http://www.example.org/testbed-
14/flightplan#identifier"/>
        <rdfs:label xml:lang="en">flight plan ID</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#aircraft -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#aircraft">
        <rdfs:label xml:lang="en">aircraft</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#flight_route -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#flight_route">
        <rdfs:label xml:lang="en">flight route</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#originator -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#originator">
        <rdfs:label xml:lang="en">originator</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#filling_time -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#filling_time">
        <rdfs:label xml:lang="fr">filling time</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#flight_rule -->
```

```
    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#flight_rule">
        <rdfs:label xml:lang="fr">flight rule</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#number_of_aircraft -->

    <owl:Class rdf:about="http://www.example.org/testbed-
14/flightplan#number_of_aircraft">
        <rdfs:label xml:lang="fr">number of aircraft</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#aircraft_equipage -->

    <owl:Class rdf:about="http://www.example.org/testbed-
14/flightplan#aircraft_equipage">
        <rdfs:label xml:lang="en">aircraft equipage</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#aircraft_communication_system
-->

    <owl:Class rdf:about="http://www.example.org/testbed-
14/flightplan#aircraft_communication_system">
        <rdfs:label xml:lang="en">aircraft communication system</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#aircraft_navigation_system -->

    <owl:Class rdf:about="http://www.example.org/testbed-
14/flightplan#aircraft_navigation_system">
        <rdfs:label xml:lang="fr">aircraft navigation system</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#aircraft_surveillance_system -->

    <owl:Class rdf:about="http://www.example.org/testbed-
14/flightplan#aircraft_surveillance_system">
        <rdfs:label xml:lang="fr">aircraft surveillance system</rdfs:label>
    </owl:Class>
```

```
    <!-- http://www.example.org/testbed-
14/flightplan#aircraft_wake_turbulence_category -->

    <owl:Class rdf:about="http://www.example.org/testbed-
14/flightplan#aircraft_wake_turbulence_category">
        <rdfs:label xml:lang="en">aircraft wake turbulence category</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#flight_rule_type -->

    <owl:Class rdf:about="http://www.example.org/testbed-
14/flightplan#flight_rule_type">
        <rdfs:label xml:lang="fr">flight rule type</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#aircraft_ID -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#aircraft_ID">
        <rdfs:subClassOf rdf:resource="http://www.example.org/testbed-
14/flightplan#identifier"/>
        <rdfs:label xml:lang="fr">aircraft ID</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#aircraft_type -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#aircraft_type">
        <rdfs:label xml:lang="fr">aircraft type</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#originator_name -->

    <owl:Class rdf:about="http://www.example.org/testbed-
14/flightplan#originator_name">
        <rdfs:label xml:lang="fr">originator name</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#airman_ID -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#airman_ID">
```

```
        <rdfs:subClassOf rdf:resource="http://www.example.org/testbed-
14/flightplan#identifier"/>
        <rdfs:label xml:lang="fr">airman ID</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#route_type -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#route_type">
        <rdfs:label xml:lang="fr">route type</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#altitude -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#altitude">
        <rdfs:label xml:lang="fr">altitude</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#estimated_time -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#estimated_time
">
        <rdfs:label xml:lang="fr">estimated time</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#estimated_departure_time -->

    <owl:Class rdf:about="http://www.example.org/testbed-
14/flightplan#estimated_departure_time">
        <rdfs:subClassOf rdf:resource="http://www.example.org/testbed-
14/flightplan#estimated_time"/>
        <rdfs:label xml:lang="fr">estimated departure time</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#estimated_en_route_time -->

    <owl:Class rdf:about="http://www.example.org/testbed-
14/flightplan#estimated_en_route_time">
        <rdfs:subClassOf rdf:resource="http://www.example.org/testbed-
14/flightplan#estimated_time"/>
        <rdfs:label xml:lang="fr">estimated en route time</rdfs:label>
    </owl:Class>
```

```
    <!-- http://www.example.org/testbed-14/flightplan#air_speed -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#air_speed">
        <rdfs:label xml:lang="fr">air speed</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#departure_aerodrome -->

    <owl:Class rdf:about="http://www.example.org/testbed-
14/flightplan#departure_aerodrome">
        <rdfs:label xml:lang="fr">departure aerodrome</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#destination_aerodrome -->

    <owl:Class rdf:about="http://www.example.org/testbed-
14/flightplan#destination_aerodrome">
        <rdfs:label xml:lang="fr">destination aerodrome</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#alternate_aerodrome -->

    <owl:Class rdf:about="http://www.example.org/testbed-
14/flightplan#alternate_aerodrome">
        <rdfs:label xml:lang="fr">alternate aerodrome</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#aerodrome -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#aerodrome">
        <rdfs:label xml:lang="fr">aerodrome</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#aerodrome_ID -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#aerodrome_ID">
        <rdfs:subClassOf rdf:resource="http://www.example.org/testbed-
14/flightplan#identifier"/>
```

```
        <rdfs:label xml:lang="fr">aerodrome ID</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#aerodrome_name -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#aerodrome_name
">
        <rdfs:label xml:lang="fr">aerodrome name</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#speed_unit -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#speed_unit">
        <rdfs:subClassOf rdf:resource="http://www.example.org/testbed-
14/flightplan#unit"/>
        <rdfs:label xml:lang="en">speed unit</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#true_speed -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#true_speed">
        <rdfs:subClassOf rdf:resource="http://www.example.org/testbed-
14/flightplan#air_speed"/>
        <rdfs:label xml:lang="fr">true speed</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#mach number -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#mach number">
        <rdfs:subClassOf rdf:resource="http://www.example.org/testbed-
14/flightplan#air_speed"/>
        <rdfs:label xml:lang="fr">mach number</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#altitude_reference -->

    <owl:Class rdf:about="http://www.example.org/testbed-
14/flightplan#altitude_reference">
        <rdfs:label xml:lang="fr">altitude reference</rdfs:label>
    </owl:Class>
```

```
    <!-- http://www.example.org/testbed-14/flightplan#altitude_unit -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#altitude_unit">
        <rdfs:subClassOf rdf:resource="http://www.example.org/testbed-
14/flightplan#unit"/>
        <rdfs:label xml:lang="en">altitude unit</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#unit -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#unit">
        <rdfs:label xml:lang="fr">unit</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#local_altitude_reference -->

    <owl:Class rdf:about="http://www.example.org/testbed-
14/flightplan#local_altitude_reference">
        <rdfs:subClassOf rdf:resource="http://www.example.org/testbed-
14/flightplan#altitude_reference"/>
        <rdfs:label xml:lang="fr">local altitude reference</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#standard_altitude_reference -->

    <owl:Class rdf:about="http://www.example.org/testbed-
14/flightplan#standard_altitude_reference">
        <rdfs:subClassOf rdf:resource="http://www.example.org/testbed-
14/flightplan#altitude_reference"/>
        <rdfs:label xml:lang="fr">standard altitude reference</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#measurement -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#measurement">
        <rdfs:label xml:lang="fr">measurement</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#speed_measure -->
```

```
    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#speed_measure">
        <rdfs:subClassOf rdf:resource="http://www.example.org/testbed-
14/flightplan#measurement"/>
        <rdfs:label xml:lang="fr">speed measure</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#altitude_measure -->

    <owl:Class rdf:about="http://www.example.org/testbed-
14/flightplan#altitude_measure">
        <rdfs:subClassOf rdf:resource="http://www.example.org/testbed-
14/flightplan#measurement"/>
        <rdfs:label xml:lang="fr">altitude measure</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#identifier -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#identifier">
        <alternative_label xml:lang="en">ID</alternative_label>
        <rdfs:label xml:lang="en">identifier</rdfs:label>
    </owl:Class>



    <!--

///////////////////////////////////////////////////////////////////////////////////
/
    //
    // Individuals
    //

///////////////////////////////////////////////////////////////////////////////////
/
     -->




    <!-- http://www.example.org/testbed-14/flightplan#heavy -->

    <owl:NamedIndividual rdf:about="http://www.example.org/testbed-
14/flightplan#heavy">
        <rdf:type rdf:resource="http://www.example.org/testbed-
14/flightplan#aircraft_wake_turbulence_category"/>
        <alternative_label xml:lang="en">H</alternative_label>
```

```
        <rdfs:label xml:lang="en">heavy</rdfs:label>
    </owl:NamedIndividual>



    <!-- http://www.example.org/testbed-14/flightplan#medium -->

    <owl:NamedIndividual rdf:about="http://www.example.org/testbed-
14/flightplan#medium">
        <rdf:type rdf:resource="http://www.example.org/testbed-
14/flightplan#aircraft_wake_turbulence_category"/>
        <alternative_label xml:lang="en">M</alternative_label>
        <rdfs:label xml:lang="fr">medium</rdfs:label>
    </owl:NamedIndividual>



    <!-- http://www.example.org/testbed-14/flightplan#light -->

    <owl:NamedIndividual rdf:about="http://www.example.org/testbed-
14/flightplan#light">
        <rdf:type rdf:resource="http://www.example.org/testbed-
14/flightplan#aircraft_wake_turbulence_category"/>
        <alternative_label xml:lang="en">L</alternative_label>
        <rdfs:label xml:lang="fr">light</rdfs:label>
    </owl:NamedIndividual>



    <!-- http://www.example.org/testbed-14/flightplan#I -->

    <owl:NamedIndividual rdf:about="http://www.example.org/testbed-14/flightplan#I">
        <rdf:type rdf:resource="http://www.example.org/testbed-
14/flightplan#flight_rule_type"/>
        <rdfs:label xml:lang="fr">I</rdfs:label>
    </owl:NamedIndividual>



    <!-- http://www.example.org/testbed-14/flightplan#V -->

    <owl:NamedIndividual rdf:about="http://www.example.org/testbed-14/flightplan#V">
        <rdf:type rdf:resource="http://www.example.org/testbed-
14/flightplan#flight_rule_type"/>
        <rdfs:label xml:lang="fr">V</rdfs:label>
    </owl:NamedIndividual>



    <!-- http://www.example.org/testbed-14/flightplan#Y -->
```

```
    <owl:NamedIndividual rdf:about="http://www.example.org/testbed-14/flightplan#Y">
        <rdf:type rdf:resource="http://www.example.org/testbed-
14/flightplan#flight_rule_type"/>
        <rdfs:label xml:lang="fr">Y</rdfs:label>
    </owl:NamedIndividual>



    <!-- http://www.example.org/testbed-14/flightplan#Z -->

    <owl:NamedIndividual rdf:about="http://www.example.org/testbed-14/flightplan#Z">
        <rdf:type rdf:resource="http://www.example.org/testbed-
14/flightplan#flight_rule_type"/>
        <rdfs:label xml:lang="fr">Z</rdfs:label>
    </owl:NamedIndividual>
</rdf:RDF>



<!-- Generated by the OWL API (version 4.1.3.20151118-2017)
https://github.com/owlcs/owlapi -->
```

FlightPlanOntology.owl aligned with OBO Foundry principles

```
<?xml version="1.0"?>
<rdf:RDF xmlns="http://www.example.org/testbed-14/flightplan#"
     xml:base="http://www.example.org/testbed-14/flightplan"
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:owl="http://www.w3.org/2002/07/owl#"
     xmlns:xml="http://www.w3.org/XML/1998/namespace"
     xmlns:flightplan="http://www.example.org/testbed-14/flightplan#"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
     xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
     xmlns:dc="http://purl.org/dc/elements/1.1/">
     <owl:Ontology rdf:about="http://www.example.org/testbed-14/flightplan">
        <owl:imports rdf:resource="http://protege.stanford.edu/plugins/owl/dc/protege-
dc.owl"/>
        <dc:creator xml:lang="en">Yann Le Franc</dc:creator>
        <dc:description xml:lang="en">An ontology describing the domain specific
concepts used in the
        Fligh Plan Service XML schema. This ontology is just an exemple of structured
vocabulary
        supporting the annotation of the flight plan XML schema.

        This ontology has been built using the OBO foundry principles
        (http://www.obofoundry.org/principles/fp-000-summary.html) i.e. a unique
identifier for each
        element of the ontology composed of the 3 letter acronym of the ontology name,
an underscore
        and a 7 digit number; a human readable label separating the different words
used in the label
```

```
        with a space and without using capital unless it requires it. The human
readable label is
        associated to the concept ID using the
        rdfs:label element.

        Ontology metadata is based on Dublin Core elements</dc:description>

    <owl:versionInfo rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
1.0</owl:versionInfo>
    <dc:source rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
>https://nsrr.faa.gov/sites/default/files/FlightPlanSchema_1.xsd</dc:source>
    </owl:Ontology>



    <!--

///////////////////////////////////////////////////////////////////////////////////
/
    //
    // Annotation properties
    //

///////////////////////////////////////////////////////////////////////////////////
/
     -->




    <!-- http://www.example.org/testbed-14/flightplan#fpo_0000015 -->

    <owl:AnnotationProperty rdf:about="http://www.example.org/testbed-
14/flightplan#fpo_0000015">
        <rdfs:label xml:lang="fr">alternative label</rdfs:label>
    </owl:AnnotationProperty>



    <!--

///////////////////////////////////////////////////////////////////////////////////
/
    //
    // Classes
    //

///////////////////////////////////////////////////////////////////////////////////
/
     -->
```

```
    <!-- http://www.example.org/testbed-14/flightplan#fpo_0000001 -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#fpo_0000001">
        <rdfs:label xml:lang="en">flight plan</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#fpo_0000002 -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#fpo_0000002">
        <rdfs:subClassOf rdf:resource="http://www.example.org/testbed-
14/flightplan#fpo_0000050"/>
        <rdfs:label xml:lang="en">flight plan ID</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#fpo_0000003 -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#fpo_0000003">
        <rdfs:label xml:lang="en">aircraft</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#fpo_0000004 -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#fpo_0000004">
        <rdfs:label xml:lang="en">flight route</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#fpo_0000005 -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#fpo_0000005">
        <rdfs:label xml:lang="en">originator</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#fpo_0000006 -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#fpo_0000006">
        <rdfs:label xml:lang="fr">filling time</rdfs:label>
    </owl:Class>
```

```
<!-- http://www.example.org/testbed-14/flightplan#fpo_0000007 -->

<owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#fpo_0000007">
    <rdfs:label xml:lang="fr">flight rule</rdfs:label>
</owl:Class>



<!-- http://www.example.org/testbed-14/flightplan#fpo_0000008 -->

<owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#fpo_0000008">
    <rdfs:label xml:lang="fr">number of aircraft</rdfs:label>
</owl:Class>



<!-- http://www.example.org/testbed-14/flightplan#fpo_0000009 -->

<owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#fpo_0000009">
    <rdfs:label xml:lang="en">aircraft equipage</rdfs:label>
</owl:Class>



<!-- http://www.example.org/testbed-14/flightplan#fpo_0000010 -->

<owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#fpo_0000010">
    <rdfs:label xml:lang="en">aircraft communication system</rdfs:label>
</owl:Class>



<!-- http://www.example.org/testbed-14/flightplan#fpo_0000011 -->

<owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#fpo_0000011">
    <rdfs:label xml:lang="fr">aircraft navigation system</rdfs:label>
</owl:Class>



<!-- http://www.example.org/testbed-14/flightplan#fpo_0000012 -->

<owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#fpo_0000012">
    <rdfs:label xml:lang="fr">aircraft surveillance system</rdfs:label>
</owl:Class>



<!-- http://www.example.org/testbed-14/flightplan#fpo_0000013 -->
```

```
    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#fpo_0000013">
        <rdfs:label xml:lang="en">aircraft wake turbulence category</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#fpo_0000018 -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#fpo_0000018">
        <rdfs:label xml:lang="fr">flight rule type</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#fpo_0000023 -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#fpo_0000023">
        <rdfs:subClassOf rdf:resource="http://www.example.org/testbed-
14/flightplan#fpo_0000050"/>
        <rdfs:label xml:lang="fr">aircraft ID</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#fpo_0000024 -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#fpo_0000024">
        <rdfs:label xml:lang="fr">aircraft type</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#fpo_0000025 -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#fpo_0000025">
        <rdfs:label xml:lang="fr">originator name</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#fpo_0000026 -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#fpo_0000026">
        <rdfs:subClassOf rdf:resource="http://www.example.org/testbed-
14/flightplan#fpo_0000050"/>
        <rdfs:label xml:lang="fr">airman ID</rdfs:label>
    </owl:Class>
```

```
    <!-- http://www.example.org/testbed-14/flightplan#fpo_0000027 -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#fpo_0000027">
        <rdfs:label xml:lang="fr">route type</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#fpo_0000028 -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#fpo_0000028">
        <rdfs:label xml:lang="fr">altitude</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#fpo_0000029 -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#fpo_0000029">
        <rdfs:label xml:lang="fr">estimated time</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#fpo_0000030 -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#fpo_0000030">
        <rdfs:subClassOf rdf:resource="http://www.example.org/testbed-
14/flightplan#fpo_0000029"/>
        <rdfs:label xml:lang="fr">estimated departure time</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#fpo_0000031 -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#fpo_0000031">
        <rdfs:subClassOf rdf:resource="http://www.example.org/testbed-
14/flightplan#fpo_0000029"/>
        <rdfs:label xml:lang="fr">estimated en route time</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#fpo_0000032 -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#fpo_0000032">
        <rdfs:label xml:lang="fr">air speed</rdfs:label>
    </owl:Class>
```

```
    <!-- http://www.example.org/testbed-14/flightplan#fpo_0000033 -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#fpo_0000033">
        <rdfs:label xml:lang="fr">departure aerodrome</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#fpo_0000034 -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#fpo_0000034">
        <rdfs:label xml:lang="fr">destination aerodrome</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#fpo_0000035 -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#fpo_0000035">
        <rdfs:label xml:lang="fr">alternate aerodrome</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#fpo_0000036 -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#fpo_0000036">
        <rdfs:label xml:lang="fr">aerodrome</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#fpo_0000037 -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#fpo_0000037">
        <rdfs:subClassOf rdf:resource="http://www.example.org/testbed-
14/flightplan#fpo_0000050"/>
        <rdfs:label xml:lang="fr">aerodrome ID</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#fpo_0000038 -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#fpo_0000038">
        <rdfs:label xml:lang="fr">aerodrome name</rdfs:label>
    </owl:Class>
```

```
    <!-- http://www.example.org/testbed-14/flightplan#fpo_0000039 -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#fpo_0000039">
        <rdfs:subClassOf rdf:resource="http://www.example.org/testbed-
14/flightplan#fpo_0000044"/>
        <rdfs:label xml:lang="en">speed unit</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#fpo_0000040 -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#fpo_0000040">
        <rdfs:subClassOf rdf:resource="http://www.example.org/testbed-
14/flightplan#fpo_0000032"/>
        <rdfs:label xml:lang="fr">true speed</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#fpo_0000041 -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#fpo_0000041">
        <rdfs:subClassOf rdf:resource="http://www.example.org/testbed-
14/flightplan#fpo_0000032"/>
        <rdfs:label xml:lang="fr">mach number</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#fpo_0000042 -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#fpo_0000042">
        <rdfs:label xml:lang="fr">altitude reference</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#fpo_0000043 -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#fpo_0000043">
        <rdfs:subClassOf rdf:resource="http://www.example.org/testbed-
14/flightplan#fpo_0000044"/>
        <rdfs:label xml:lang="en">altitude unit</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#fpo_0000044 -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#fpo_0000044">
```

```
        <rdfs:label xml:lang="fr">unit</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#fpo_0000045 -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#fpo_0000045">
        <rdfs:subClassOf rdf:resource="http://www.example.org/testbed-
14/flightplan#fpo_0000042"/>
        <rdfs:label xml:lang="fr">local altitude reference</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#fpo_0000046 -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#fpo_0000046">
        <rdfs:subClassOf rdf:resource="http://www.example.org/testbed-
14/flightplan#fpo_0000042"/>
        <rdfs:label xml:lang="fr">standard altitude reference</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#fpo_0000047 -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#fpo_0000047">
        <rdfs:label xml:lang="fr">measurement</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#fpo_0000048 -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#fpo_0000048">
        <rdfs:subClassOf rdf:resource="http://www.example.org/testbed-
14/flightplan#fpo_0000047"/>
        <rdfs:label xml:lang="fr">speed measure</rdfs:label>
    </owl:Class>



    <!-- http://www.example.org/testbed-14/flightplan#fpo_0000049 -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#fpo_0000049">
        <rdfs:subClassOf rdf:resource="http://www.example.org/testbed-
14/flightplan#fpo_0000047"/>
        <rdfs:label xml:lang="fr">altitude measure</rdfs:label>
    </owl:Class>
```

```
    <!-- http://www.example.org/testbed-14/flightplan#fpo_0000050 -->

    <owl:Class rdf:about="http://www.example.org/testbed-14/flightplan#fpo_0000050">
        <fpo_0000015 xml:lang="en">ID</fpo_0000015>
        <rdfs:label xml:lang="en">identifier</rdfs:label>
    </owl:Class>



    <!--

///////////////////////////////////////////////////////////////////////////////////////
/
    //
    // Individuals
    //

///////////////////////////////////////////////////////////////////////////////////////
/
     -->




    <!-- http://www.example.org/testbed-14/flightplan#fpo_0000014 -->

    <owl:NamedIndividual rdf:about="http://www.example.org/testbed-
14/flightplan#fpo_0000014">
        <rdf:type rdf:resource="http://www.example.org/testbed-
14/flightplan#fpo_0000013"/>
        <fpo_0000015 xml:lang="en">H</fpo_0000015>
        <rdfs:label xml:lang="en">heavy</rdfs:label>
    </owl:NamedIndividual>



    <!-- http://www.example.org/testbed-14/flightplan#fpo_0000016 -->

    <owl:NamedIndividual rdf:about="http://www.example.org/testbed-
14/flightplan#fpo_0000016">
        <rdf:type rdf:resource="http://www.example.org/testbed-
14/flightplan#fpo_0000013"/>
        <fpo_0000015 xml:lang="en">M</fpo_0000015>
        <rdfs:label xml:lang="fr">medium</rdfs:label>
    </owl:NamedIndividual>



    <!-- http://www.example.org/testbed-14/flightplan#fpo_0000017 -->
```

```
    <owl:NamedIndividual rdf:about="http://www.example.org/testbed-
14/flightplan#fpo_0000017">
        <rdf:type rdf:resource="http://www.example.org/testbed-
14/flightplan#fpo_0000013"/>
        <fpo_0000015 xml:lang="en">L</fpo_0000015>
        <rdfs:label xml:lang="fr">light</rdfs:label>
    </owl:NamedIndividual>



    <!-- http://www.example.org/testbed-14/flightplan#fpo_0000019 -->

    <owl:NamedIndividual rdf:about="http://www.example.org/testbed-
14/flightplan#fpo_0000019">
        <rdf:type rdf:resource="http://www.example.org/testbed-
14/flightplan#fpo_0000018"/>
        <rdfs:label xml:lang="fr">I</rdfs:label>
    </owl:NamedIndividual>



    <!-- http://www.example.org/testbed-14/flightplan#fpo_0000020 -->

    <owl:NamedIndividual rdf:about="http://www.example.org/testbed-
14/flightplan#fpo_0000020">
        <rdf:type rdf:resource="http://www.example.org/testbed-
14/flightplan#fpo_0000018"/>
        <rdfs:label xml:lang="fr">V</rdfs:label>
    </owl:NamedIndividual>



    <!-- http://www.example.org/testbed-14/flightplan#fpo_0000021 -->

    <owl:NamedIndividual rdf:about="http://www.example.org/testbed-
14/flightplan#fpo_0000021">
        <rdf:type rdf:resource="http://www.example.org/testbed-
14/flightplan#fpo_0000018"/>
        <rdfs:label xml:lang="fr">Y</rdfs:label>
    </owl:NamedIndividual>



    <!-- http://www.example.org/testbed-14/flightplan#fpo_0000022 -->

    <owl:NamedIndividual rdf:about="http://www.example.org/testbed-
14/flightplan#fpo_0000022">
        <rdf:type rdf:resource="http://www.example.org/testbed-
14/flightplan#fpo_0000018"/>
        <rdfs:label xml:lang="fr">Z</rdfs:label>
```

```
    </owl:NamedIndividual>
</rdf:RDF>



<!-- Generated by the OWL API (version 4.1.3.20151118-2017)
https://github.com/owlcs/owlapi -->
```

Appendix C: Revision History

Table 11. Revision History

Date	Editor	Release	Descriptions
November 6, 2018	Yann Le Franc	1.0	draft for review
November 21, 2018	Yann Le Franc	1.1	corrected draft: address sponsors' comments and comments/edits suggested by Gobe Hokona

Appendix D: Bibliography

1. Houbie, F.: Semantic Annotations in OGC standards. OGC 08-167r2,Open Geospatial Consortium, https://portal.opengeospatial.org/files/?artifact_id=47857 (2012).
2. Fellah, S.: OGC Testbed-14: Semantically Enabled Aviation Data Models Engineering Report. OGC 18-035,Open Geospatial Consortium, http://www.opengeospatial.org/docs/er (2019).
3. Balaban, A.: OGC Testbed-12 Aviation Semantics Engineering Report. OGC 16-039r2,Open Geospatial Consortium, http://docs.opengeospatial.org/per/16-039r2.html (2017).
4. Thomas, T.: OGC Testbed-12 Aviation SBVR. OGC 16-061,Open Geospatial Consortium, http://docs.opengeospatial.org/per/16-061.html (2017).
5. Fellah, S.: OGC Testbed-12 Semantic Portrayal, Registry and Mediation Engineering Report. OGC 16-059,Open Geospatial Consortium, http://docs.opengeospatial.org/per/16-059.html (2016).
6. Fellah, S.: OGC Testbed-12 Semantic Protrayal, Registry and Mediation. OGC 16-059,Open Geospatial Consortium, http://docs.opengeospatial.org/per/16-059.html (2017).
7. McCann, S.: OGC Testbed-13 DCAT/SRIM. OGC 17-040,Open Geospatial Consortium, http://docs.opengeospatial.org/per/17-040.html (2018).
8. Martell, R.: OGC Testbed-12 Catalog Services for Aviation. OGC 16-024r2,Open Geospatial Consortium, http://docs.opengeospatial.org/per/16-024r2.html (2017).
9. Kaplun, M., Fernandez-Sancho, P., Roelants, E., Uri, C.: Utilization of Faceted Classification in the Context of the SWIM Service Registry. Presented at the October (2014).
10. Borges Monteiro, L., Souza Ramos Barbosa, I., Weigang, L., Fregagni, J.A., Oliveira, I. Romani de, Balvedi, G.: A System Wide Information Management Architecture Proposal for Brazilian Scenarios. In: Proceedings of the 2017 SITRAER Conference (SITRAER2017) (2017).
11. Beynon-Davies, P.: Information systems : an introduction to informatics in organisations. Palgrave, Basingstoke (2002).
12. Rowley, J.: The wisdom hierarchy: representations of the DIKW hierarchy. Journal of Information Science. 33, 163–180 (2007).
13. Masó, J., Singh, R.: OGC Testbed 10: Annotations Engineering Report. OGC 14-002,Open Geospatial Consortium, https://portal.opengeospatial.org/files/?artifact_id=58965 (2014).
14. Goldfarb, D., Le Franc, Y.: Enhancing the Discoverability and Interoperability of Multi-disciplinary Semantic Repositories. In: Proceedings of the 2nd International Workshop on Semantics for Biodiversity co-located with 16th International Semantic Web Conference (ISWC 2017) (2017).