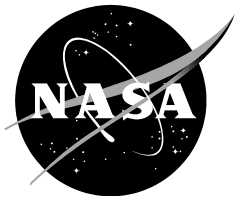
NASA/TM—2017-219526



The NASA Air Traffic Management Ontology

Technical Documentation

Richard M. Keller
Ames Research Center, Moffett Field, California

June 2017

Acknowledgments

This work was funded by the National Aviation and Space Administration under the Aviation Operations and Safety Program. My sincere thanks to Mei Wei and Shubha Ranjan, who contributed long hours to software development for the ATM Ontology, and to Michelle Eshow, the lead for the Sherlock Aviation Data Warehouse team, who generously supported, encouraged, and contributed effort toward this activity from its inception.

Abstract

This document is intended to serve as comprehensive documentation for the NASA Air Traffic Management (ATM) Ontology. The ATM Ontology is a conceptual model that defines key classes of entities and relationships pertaining to the US National Airspace System (NAS) and the management of air traffic through that system. A wide variety of classes are represented in the ATM Ontology, including classes corresponding to flights, aircraft, manufacturers, airports, airlines, air routes, NAS facilities, air traffic control advisories, weather phenomena, and many others. The Ontology can be useful in the context of a variety of information management tasks relevant to NAS, including information exchange, data query and search, information organization, information integration, and terminology standardization.

Table of Contents

1 Introduction

This document describes the NASA Air Traffic Management (ATM) Ontology. The ATM Ontology defines key classes of entities pertaining to the US National Airspace System (NAS) and the management of air traffic through that system. A wide variety of classes are represented in the ATM Ontology, including classes corresponding to flights, aircraft, manufacturers, airports, airlines, air routes, NAS facilities, air traffic control advisories, weather phenomena, and many others.

The motivation for developing this ontology stems from NASA'S need to integrate heterogeneous forms of aviation data for use in aeronautics research. To accomplish data integration, we applied semantic integration techniques [1, 2] that depend upon existence of common ontology to serve as an integrative data model. Data from multiple aviation sources were transformed into ATM Ontology instances and loaded into a triple store. Queries against the triple store could then be executed across the integrated data and results could be delivered to users. This document is intended to serve as comprehensive documentation for the ATM Ontology; other documents are more appropriate for a high-level overview of the Ontology and its context of its use within NASA [3-6] .

The Ontology is formatted as a set of OWL (Web Ontology Language) files [7], and basic knowledge of OWL, RDF [8] (Resource Description Framework), and RDFS [9] (RDF Schema) is presumed in this document.

1.1 Document Structure

Most of this document is devoted to describing classes defined in the ATM Ontology, along with their object properties (i.e., links to other classes) and datatype properties (i.e., class attributes). The classes have been organized into eight major sections describing thematically-related sets of classes; each of these classes is described in its own subsection of the document:

- Airspace Structures and Facilities
- Navigation: Routes, Fixes, Arrival and Departure Procedures
- Traffic Management Initiatives
- Operations: Flight, Carrier, and Aircraft
- Airport and Surface Operations
- Weather
- Sequences, Subsequences, Sequenced Items
- Temporal/Spatial

Note that the organization of this document into sections and subsections is based on pedagogical considerations and on maximizing understandability; this structure does not necessarily match the ATM Ontology's subsumption (i.e. subclass) hierarchy detailed in Appendix C or the partitioning of classes defined by the ontology namespace structure detailed in Appendix B. At the end of each major section, there is a subsection devoted to illustrative figures showing how the classes work together and are used in practice.

1.2 Document Terminology and Conventions

In general, we follow OWL terminology and use the term 'class' rather than 'entity', 'resource', 'object', or 'concept'. Similarly, we use the terms 'object property' and 'datatype property' rather than the terms 'relation' and 'attribute'.

Class descriptions in this document are formatted as follows:

CLASSNAME

- **DESCRIPTION:** *CLASS DESCRIPTION TEXT*
- **SUPERCLASSES:**
  - **SUPERCLASS1**
  - **SUPERCLASS2**
  - **…**
- **SUBCLASSES:**
  - **SUBCLASS1**
  - **SUBCLASS2**
  - **…**
- **OBJECT PROPERTIES**
  - **PROPERTYNAME1 [RANGE1]:** *PROPERTY DESCRIPTION TEXT*
  - **PROPERTYNAME2 [RANGE2]:** *PROPERTY DESCRIPTION TEXT*
  - **…**
- **DATATYPE PROPERTIES**
  - **PROPERTYNAME1 [DATATYPE1]:** *PROPERTY DESCRIPTION TEXT*
  - **PROPERTYNAME2 [DATATYPE2]:** *PROPERTY DESCRIPTION TEXT*
  - **…**

WHERE:

- **CLASSNAME** REPRESENTS THE DOMAIN OF PROPERTYNAME$_i$
- **RANGE$_i$** REPRESENTS THE RANGE OF PROPERTYNAME$_i$
- **DATATYPE$_i$** = ONE OF {'INTEGER', 'FLOAT', 'DATETIME', 'STRING'}, REPRESENTING THE DATATYPE OF PROPERTYNAME$_i$

  NOTE: IF PRESENT, THE SPECIAL SYNTAX *'STRING: "VAL1", "VAL2",…, "VAL3N" '* INDICATES A STRING WHOSE VALUE IS CONSTRAINED TO TAKE ONE OF THE N LISTED VALUES; OTHERWISE THE STRING VALUE IS UNCONSTRAINED

ALL BULLETED ITEMS ARE OPTIONAL EXCEPT THE DESCRIPTION.

2 Airspace Structures and Facilities

The classes in this section define various structures and facilities that constitute key components of the NAS. Also included are classes to measure the aircraft flow and capacity passing through these structures.

2.1 nas:NASfacility

- **Description**: The set of FAA facilities involved in operational air traffic management. The object property *nas:hasLOAwith* signifies that a facility has a Letter of Agreement (LOA) with another facility; this property is used to connect facilities together into a topology that reflects the FAA operations hierarchy. The FAA ATSCC (System Command Center) is at the top of this hierarchy; the ATSCC has agreements with the ARTCCs; the ARTCCs have agreements with the TRACONS; the ARTCCs and TRACONs have agreements with the airports.
- **Subclasses**:
  - nas:Airport
  - nas:ARTCC
  - nas:ATCSCC
  - nas:ATCT
  - nas:TRACON
- **Object properties**:
  - **nas:hasLOAwith [nas:NASfacility]**: This property links a NAS facility to another NAS facility when there is a letter of agreement (LOA) in place between them. Typically, a LOA is needed when two or more FAA facilities cooperate to accomplish flight operations. The agreement indicates each party's responsibilities.

2.1.1 nas:ATCSCC

- **Description**: Air Traffic Control System Command Center (ATCSCC) facility is in charge of overall NAS operations. There is one instance of the ATCSCC class: nas:UScommandCenter.
- **Superclasses:**
  - nas:NASfacility

2.1.2 nas:ATCT

- **Description**: Air Traffic Control Tower (ATCT) controls airport departures and arrivals.
- **Superclasses:**
  - nas:NASfacility

2.1.3 nas:ARTCC

- **Description:** An Air Route Traffic Control Center has traffic management responsibility for high-altitude enroute aircraft transiting through the NAS. The adjacency structure of the ARTCCs is represented by nas:ARTCCtier, which captures Centers that are proximate to a given center.
- **Superclasses:**
  - nas:NASfacility
- **Object properties:**
  - nas:hasCenterGeometry [gen:Polygonal2DRegion]: the 2-dimensional bounding region of the ARTCC

- nas:hasTier [nas:ARTCCtier]: Associates an ARTCC with the set of ARTCCs in its nth level tier. Tier 1 includes all ARTCCs immediately adjoining the central ARTCC; tier 2 includes tier 1 plus all adjacent ARTCCs two steps away from the central ARTCC. And so on.

- **Datatype properties:**
  - nas:artccID [string]: The FAA-assigned 3-letter code for the ARTCC

2.1.3.1 nas:ARTCCtier

- **Description:** A donut-shaped geographical area surrounding a designated central ARTCC. The first tier includes all the ARTCCs that share a boundary with the designated ARTCC. The tier two ARTCCs includes those adjacent to the first tier ARCCCs. And so on.
- **Superclasses:**
  - nas:AirspaceInfrastructureComponent
- **Object properties:**
  - nas:includesARTCC [nas:ARTCC]: Links the tier with the included ARTCCs. The ARTCC at the center of the donut is not linked.
- **Datatype properties:**
  - nas:tierLevel [integer]: The tier level of this tier. The degree number of the ARTCC tier. Tier 1 includes all ARTCCs immediately adjoining the central ARTCC; tier 2 includes all adjacent ARTCCs two steps away from the central ARTCC. And so on.

2.1.4 nas:TRACON

- **Description**: The Terminal Radar Approach Control (TRACON) facility manages aircraft transiting between the airport and the enroute Center, handling both airport arrivals and departures.
- **Superclasses:**
  - nas:NASfacility
- **Object properties:**
  - nas:hasTRACONlayer [nas:AirspaceLayer]: A link between the TRACON and the layers that represent its geographic extent, represented as a stack of single shear-sided polygonal volumes.
- **Datatype properties**:
  - nas:hasTRACONcity [string]: The city name where the TRACON command center is located.
  - nas:hasTRACONid [string]: The FAA alphanumeric code for the TRACON.
  - nas:hasTRACONname [string]: The common name for the TRACON.
  - nas:hasTRACONstate [string]: The name of the US state in which the TRACON command center is located.
  - nas:hasTRACONtype [string: “TRACON”,”TOWER”]: The type of TRACON: Tower-collocated TRACON or standalone TRACON facility.

2.2 nas:AirspaceInfrastructureComponent

- **Description:** This class consists of airspace structures defined to control, monitor, or manage air traffic.
- **Superclasses:**
  - nas:TFMcontrolElement
- **Subclasses:**
  - nas:AirspaceLayer

- nas:AirspaceRoute
- nas:ARTCC
- nas:ARTCCtier
- nas:Sector
- nas:SIDSTAR
  - nas:SID
  - nas:STAR
- nas:TRACON

2.3 nas:Sector

- **Description:** A defined volume in the airspace of an ARTCC whose flight traffic is typically controlled by a single controller. A sector contains multiple layers, each of which is a polygonal volume defined by an identical surface boundary polygon on the top and bottom, and vertical sides (called a gen:ShearSidedPolygonalVolume). The sector is modeled as a vertical stack of these volumes forming a type of jagged 'layer cake' structure. Each layer is linked to the sector via the property nas:hasSectorLayer. Any sector immediately adjacent to (i.e., touching) the sector is linked via the property nas:adjacentSector, and the ARTCC in which the sector is located is linked via the property nas:locatedInCenter.
- **Superclasses:**
  - nas:AirspaceInfrastructureComponent
- **Object properties**:
  - nas:adjacentSector [nas:Sector]: Links to any immediately proximate sectors with boundaries that touch the current sector
  - nas:hasSectorLayer [nas:AirspaceLayer]: Links to all of the layers within the sector
  - nas:locatedInCenter [nas:ARTCC]: Links to the unique ARTCC in which this sector is located
- **Datatype properties:**
  - nas:hasSectorID [string]: The FAA-assigned sector identifier that consists of the ARTCC identifier concatenated with a three-digit number. This identifier is in general different than the sector name.
  - nas:hasSectorName [string]: The FAA-assigned sector name, an alphanumeric sector name that is in general different than the sector ID.
  - nas:tfmsMonitorAlertParameter [integer]: The maximum number of aircraft permitted in a given sector at any time, per agreement between FAA and Air Traffic Controller union. MAP (Monitor Alert Parameter) is a TFMS input parameter.

2.3.1 nas:AirspaceLayer

- **Description:** Represents a horizontal layer within the airspace, defined by a shear-sided polygon with specified low and high altitudes. The property nas:hasAirspaceLayerGeometry links the layer to a representation of the volume as a gen:ShearSidedPolygonalVolume, while the property nas:airspaceLayerLowAltitude and nas:airspaceLayerHighAltitude give the bounding altitudes of the top and bottom of the layer in feet. (The difference between the top and bottom altitude should be equal to the height of the shear-sided polygonal volume.)
- **Superclasses:**
  - nas:AirspaceInfrastructureComponent

- **Object properties**:
  - nas:hasAirspaceLayerGeometry [gen:ShearSidedPolygonalVolume]: the geometric volume representing the extent of the sector layer
- **Datatype properties**:
  - nas:airspaceLayerHighAltitude [integer]: the upper altitude of the sector layer
  - nas:airspaceLayerLowAltitude [integer]: the lower altitude of the sector layer

2.4 atm:AircraftFlowCapacity

- **Description**: A superclass of data relevant to measuring the actual and theoretical maximum number of aircraft that flow through an airspace structure during some interval of time, given weather and other air traffic control constraints.
- **Superclasses:**
  - data:IntervalData
- **Subclasses**:
  - atm:AircraftCapacity
  - atm:AircraftFlow
- **Object properties**:
  - atm:flowMeasurementRegion [atm:TFMcontrolElement]: Links to the traffic management structure through which flow or capacity is being measured or analyzed.

2.4.1 atm:AircraftCapacity

- **Description**: A class representing the capacity of an airspace region to safely contain aircraft simultaneously traversing the region during a specified period of time.
- **Superclasses**:
  - atm:AircraftFlowCapacity
- **Subclasses**:
  - data:FixCapacity
  - data:SectorCapacity
- **Datatype properties**:
  - atm:flowCapacity [float]: The theoretical maximum number of aircraft that can be safely controlled in an airspace region during some interval of time.

2.4.1.1 data:FixCapacity

- **Description**: A class representing the capacity of a fix to safely handle aircraft traversing during a specified period of time.
- **Superclasses**:
  - atm:AircraftCapacity
- **Note**: This class constrains atm:flowMeasurementRegion to be atm:NavigationFix using an OWL restriction.

2.4.1.2 data:SectorCapacity

- **Description**: A class representing the capacity of a sector to safely contain aircraft simultaneously traversing the sector during a specified period of time.

- **Superclasses**:
  - atm:AircraftCapacity
- **Note**: This class constrains atm:flowMeasurementRegion to be nas:Sector using an OWL restriction.

2.4.2 *atm:AircraftFlow*

- **Description**: A class representing the actual flow of aircraft through an airspace region over some interval of time. For example, an instance of this class could represent the flow of flights from SFO to ORD on a specific date, passing through a specific sector.
- **Superclasses**: atm:AircraftFlowCapacity
- **Object properties**:
  - atm:includedFlights [atm:FlightSpec]: Links an aircraft flow to a specification of the flights included in the flow.
- **Datatype properties**:
  - atm:aircraftCount [integer]: The number of aircraft flowing through an airspace region per some interval of time.

2.4.2.1 *data:FixFlow*

- **Description**: A subclass of data:AircraftFlow that measures aircraft flow at a fix.
- **Superclasses**:
  - atm:AircraftFlow
- **Note**: This subclass is specified using an OWL restriction on the atm:flowMeasurementRegion property that constrains its value to be an instance of atm:NavigationFix.

2.4.2.2 *data:SectorFlow*

- **Description**: A subclass of data:AircraftFlow that measures aircraft flow at a sector.
- **Superclasses**:
  - atm:AircraftFlow
- **Note**: This subclass is specified using an OWL restriction on the atm:flowMeasurementRegion property that constrains its value to be an instance of nas:Sector.

2.5 Illustrative Figures

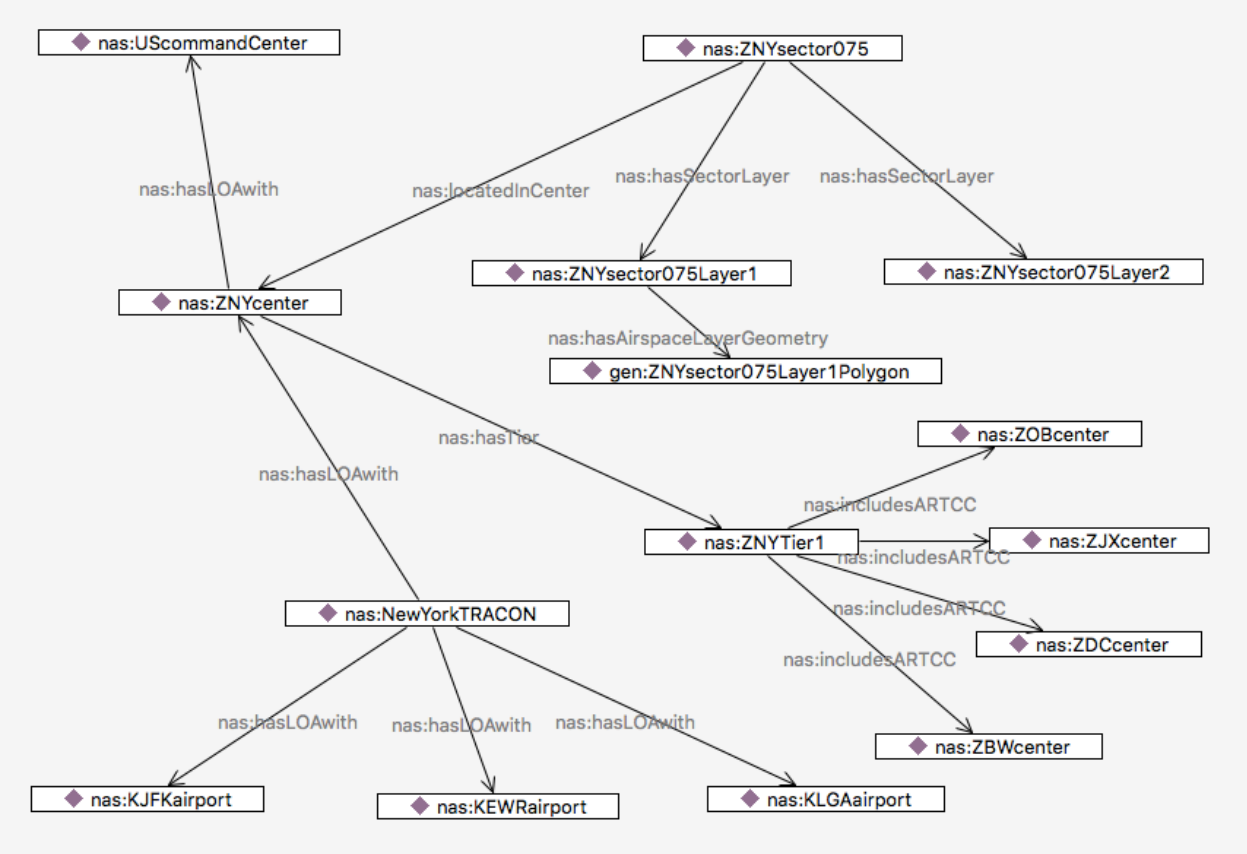


Figure 1: Illustration of key relationships among selected airspace structure and facility instances. Sector nas:ZNYsector075 is one of the sectors located in the New York ARTCC (nas:ZNYcenter). Sector 075 is composed of two stacked horizontal layers of airspace, each represented by a shear-sided polygon of a certain height (only one polygon is depicted in the figure). See also Figure 17 for the expanded sector layer representation. The ZNY ARTCC has agreements with the FAA command center and the New York TRACON, which in turn has agreements with each of the airports in its territory. The ZNY Tier 1 structure contains all ARTCCs that directly neighbor the ZNY ARTCC. (Note that only a small subset of instances is illustrated in order to keep the figure uncluttered and readable.)

3 Navigation: Routes, Fixes, Arrival and Departure Procedures

The classes in this section define the routes, airways, fixes, and arrival/departure procedures by which aircraft navigate through the NAS.

3.1 atm:NavigationElement

- **Description**: This class represents the set of components that can be used to specify a path through the airspace. These compositional elements include fixes (points in the airspace), routes (pre-specified paths through the airspace defined by the FAA), airports, SID/STAR traverses (specific paths through airport departure/arrival networks), etc. The navigational elements are assembled into paths using a sequence (gen:Sequence) – in particular, using the sequence subclass called atm:NavigationPath. (Note that atm:NavigationPath is also a type of navigation element because predefined paths can be placed in sequence preceding or following other elements to form a path.)
- **Superclasses**:
  - atm:TFMcontrolElement
- **Subclasses:**
  - atm:NavigationFix
  - atm:NavigationPath
  - atm:NavigationSubPath
    - atm:AirspaceRouteSegment
    - atm:FlightPlanSegment
  - atm:SIDSTARtraverse
  - atm:Airport

3.2 atm:NavElementContainer

- **Description:** A navigation element container is a wrapper around an existing navigation element (atm:NavigationElement), such as a fix or a route. These containers can be sequenced together to represent flight paths (see gen:Sequence, gen:SequencedElement).
- **Note:** A container construct is used so that the same navigation element can appear in multiple sequences. If the elements themselves were sequenced, each element could have only a single preceding or following element in a sequence. This would preclude an element from being sequenced in different ways in different sequences.
- **Superclasses**:
  - gen:SequencedItem
- **Object properties:**
  - atm:hasNavElement [atm:NavigationElement]: Links a container for a navigational element to the element contained.

3.3 atm:NavigationPath

- **Description**: An ordered sequence of navigation element containers (atm:NavElementContainer) representing an arbitrary path through the airspace.
- **Superclasses**:
  - atm:NavigationElement
  - gen:Sequence

- **Subclasses**:
  - atm:PlannedFlightRoute
  - nas:AirspaceRoute

3.4 atm:PlannedFlightRoute

- **Description**: A planned flight route is an ordered sequence of navigation element containers (atm:NavElementContainer) that contain navigation elements (atm:NavigationElement) the pilot intends to traverse en route from origin to destination airport.
- **Superclasses**:
  - atm:FlightSequence
  - atm:NavigationPath
- **Object properties**:
  - atm:planFilingDay [nas:NASday]: Links to the day when the flight plan was filed with the FAA.
- **Datatype properties**:
  - atm:flightPlanETA [dateTime]: The estimated time of arrival specified in the flight plan.
  - atm:flightRouteString [string]: A string that provides the route of flight proposed in a flight plan. Syntactically, the string shows a path of fixes, airways, SIDs, and STARs, separated by either one or two dots. This string is parsed to create an equivalent ontology representation: an ordered sequence of navigation elements that constitute the planned flight route.
  - atm:planFilingTime [dateTime]: The time that the flight plan was filed with the FAA.
  - atm:planGapAfterSequenceElement [integer]: Indicates the location of a gap in the flight plan sequence due to non-parsable/uninterpretable elements in the flight route string.

3.5 nas:AirspaceRoute

- **Description**: An FAA-defined sequence of navigation elements specifying a path through the airspace. An airspace route is a subclass of atm:NavigationPath, which is a more general class of arbitrary airspace paths; nas:AirspaceRoute contains only the routes defined by FAA, including the specific subclasses of routes defined below.
- **Superclasses**:
  - atm:NavigationPath
  - nas:AirspaceInfrastructureComponent
- **Subclasses**:
  - nas:FederalAirway
    - nas:RNAVroute
      - nas:QRoute
      - nas:TRoute
    - nas:VORroute
      - nas:JetRoute
      - nas:VictorRoute
  - nas:RadialRoute
  - nas:SIDSTARroute
    - nas:AirportRoute
    - nas:CommonRoute

        - nas:TransitionRoute
- **Datatype properties**:
    - nas:routeID [string]: FAA alphanumeric identifier assigned to the route/airway.

3.5.1 nas:FederalAirway

- **Definition**: An airspace route that is defined by FAA and can be filed as part of a flight plan: a jet route, a Q-route, a T-Route, a Victor route.
- **Superclasses**:
    - nas:AirspaceRoute
- **Subclasses**:
    - nas:RNAVroute
    - nas:VORroute

3.5.1.1 nas:RNAVroute

- **Definition**: An RNAV (Random Navigation) route is defined relative to a network of existing ground-based navigation beacons. RNAV allows an aircraft to choose any course within the network, rather than navigate using only point-to-point routes defined by the beacons.
- **Superclasses**:
    - nas:FederalAirway
- **Subclasses**:
    - nas:QRoute
    - nas:TRoute

3.5.1.1.1 nas:QRoute

- **Definition**: A high altitude RNAV airway route. (RNAV routes allow an aircraft to choose any course within a network of navigation beacons, rather than navigate directly to and from the beacons.)
- **Superclasses**:
    - nas:RNAVroute

3.5.1.1.2 nas:TRoute

- **Definition**: A low altitude RNAV airway route. (RNAV routes allow an aircraft to choose any course within a network of navigation beacons, rather than navigate directly to and from the beacons.)
- **Superclasses**:
    - nas:RNAVroute

3.5.1.2 nas:VORroute

- **Definition**: VOR route is defined relative to a network of existing ground-based VOR navigation beacons. VOR routes consist of point-to-point segments, where the points are defined by the beacons. More modern RNAV routes allow any path within the network, not limited to VOR-to-VOR segments.
- **Superclasses**:
    - nas:FederalAirway
- **Subclasses**:
    - nas:Jetroute

- nas:VictorRoute

3.5.1.2.1 nas:Jetroute

- **Definition**: A high altitude airway based on ground-based VOR sensor locations.
- **Superclasses**:
  - nas:VORroute

3.5.1.2.2 nas:VictorRoute

- **Definition**: A low altitude airway based on ground-based VOR sensor locations.
- **Superclasses**:
  - nas:VORroute

3.5.2 nas:RadialRoute

- **Definition**: A route that follows a specified radial path emanating from a given fix.
- **Superclasses**:
  - nas:AirspaceRoute
- **Object properties**:
  - atm:radialFix [atm:NavigationFix]: Links to the fix through which the radial route passes.
- **Datatype properties**:
  - atm:radialAngle [integer]: The angle (in degrees) that defines the radial route's angular position with respect to the defined fix.

3.5.3 nas:SIDSTARroute

- **Definition**: A route that defines a SID or STAR.
- **Superclasses**:
  - nas:AirspaceRoute
- **Subclasses**:
  - nas:AirportRoute
  - nas:CommonRoute
  - nas:TransitionRoute

3.5.3.1 nas:AirportRoute

- **Definition**: A route within a SID or STAR that connects the common route to one of multiple airports that use the SID/STAR.
- **Superclasses**:
  - nas:SIDSTARroute
- **Object properties**:
  - nas:arrivalRouteAirport [nas:Airport]: Links to the STAR airport for which this route is destined.
  - nas:departureRouteAirport [nas:Airport]: Links to the SID airport from which this route originated.

3.5.3.2 nas:CommonRoute

- **Definition**: The backbone portion of a SID or STAR that is flown by all aircraft.

- **Superclasses**:
  - nas:SIDSTARroute

3.5.3.3 nas:TransitionRoute

- **Definition**: A route within a SID or STAR that feeds aircraft into or out from the common route of the SID/STAR. In other words, the transition route is a feeder route from the en route segment of a flight into the heart of a STAR for arrival; or an exit route from a SID funneling flights to their appropriate initial jet routes for the en route segment of flight.
- **Superclasses**:
  - nas:SIDSTARroute

3.6 atm:NavigationSubPath

- **Description**: A contiguous subportion of an existing defined navigation path. See gen:SubSequence.
- **Superclasses**:
  - atm:NavigationElement
  - gen:SubSequence
- **Subclasses**:
  - atm:FlightPlanSegment
  - atm:AirspaceRouteSegment

3.6.1 atm:FlightPlanSegment

- **Description**: A contiguous subportion of an existing defined flight plan.
- **Superclasses**:
  - atm:NavigationSubPath

3.6.2 atm:AirspaceRouteSegment

- **Description**: A contiguous subportion of an existing defined airspace route.
- **Note***:* A flight plan will typically include one or more contiguous subsections of existing airspace routes.
- **Superclasses**:
  - atm:NavigationSubPath

3.7 atm:AircraftTrackPoint

- **Description**: A point during a flight where various flight parameters are captured and sent (via transponder) to FAA computers.
- **Object Properties**:
  - atm:aircraftFix [atm:navigationFix]: Links an aircraft track point with its associated navigation fix.
  - atm:reportingDay [nas:NASday]: A link between an aircraft track point and the day during which an en route aircraft traversed that point and reported its position, speed, and heading.
- **Datatype Properties**:
  - atm:groundspeed [integer]: The reported ground speed at an aircraft track point.
  - atm:heading [float]: The aircraft heading at the track point: a number between 0.01 and 360.0 indicating the angular heading with respect to North.

- atm:reportingTime [dateTime]: The time when an en route aircraft passed through a trackpoint and reported its position, speed, and heading.

3.8 *atm:ActualFlightRoute*

- **Description**: A flight trajectory, i.e., a sequence of track points (atm:AircraftTrackPoint), as determined by flight track surveillance data. See gen:Sequence.

3.9 atm:NavigationFix

- **Description**: A designated point on or above the surface of the earth used for aeronautical navigation. Fixes are split into subclasses of absolute and relative fixes. An absolute fix is defined explicitly in terms of a specified latitude/longitude/altitude. A relative fix is defined in relation to an absolute fix. In general, fixes are either named (by FAA or ICAO) or unnamed. Different subclasses capture different types of fixes (e.g., intersection fixes, navaid fixes, meter fixes).
- **Superclasses:**
  - atm:NavigationElement
- **Subclasses:**
  - atm:AbsoluteFix - A fix based on some established global measuring scheme.
    - atm:intersectionFix
    - atm:LatLonFix
      - atm:GPSfix
      - atm:NRSfix
    - atm:MeterFix
    - atm:NavaidFix
      - atm:NDBfix
      - atm:TACANfix
      - atm:VORfix
        - atm:AirportFix
  - atm:RelativeFix - A fix defined in relation to another fix.
    - atm:DMEfix
      - atm:FRDfix
- **Object properties**:
  - atm:locatedInSector [nas:Sector]: Links to the unique sector in which this fix is located.
- **Datatype properties**:
  - atm:fixId [string]: A non-unique FAA identifier for a fix. When combined with a fix ICAO code, it defines a unique fix worldwide. Often, but not always, the fixId is identical to the fixName.
  - atm:fixName [string]: A unique FAA identifier for a fix that can be filed as part of a flight plan. The fix name conceptually represents the worldwide-unique combination of a FixID and ICAO code. Often, but not always, the fixId is identical to the fixName.

3.9.1 atm:AbsoluteFix

- **Definition**: A fix based on some established global measuring scheme.
- **Superclasses**:
  - atm:NavigationFix

- gen:PointLocation

3.9.1.1 atm:intersectionFix

- **Definition**: A navigation fix defined by the intersection of two airspace routes.
- **Superclasses***:*
  - atm:AbsoluteFix

3.9.1.2 atm:LatLonFix

- **Definition**: A navigation fix based on latitude/longitude coordinates.
- **Superclasses***:*
  - atm:AbsoluteFix

3.9.1.2.1 atm:GPSfix

- **Definition**: A navigation fix defined by GPS coordinates.
- **Superclasses***:*
  - atm:LatLonFix

3.9.1.2.2 atm:NRSfix

- **Definition**: The NRS is a system of waypoints developed for use within the United States for flight planning and navigation without reference to ground based navigational aids. The NRS waypoints are located in a grid pattern along defined latitude and longitude lines. NRS waypoint names are composed of two letters followed by two numbers, followed by a single letter. The first and second characters of NRS waypoints are the FIR identifier for the United States ("K") and the FIR subdivision, or ARTCC center in which the waypoint is located (e.g. "D" for Denver ARTCC). The third and fourth characters are a number group representing the latitude of the waypoint. These numbers begin at the equator with 00 and advances north and south from 01 to 90 and correspond to every 10 minutes of latitude and repeating every 15°. The final character in the NRS waypoint is a letter representing the line of longitude for which the waypoint is located. This identifier starts at the prime meridian moving west to east and uses the letters A to Z while repeating every 26°. To date, the current density of the NRS grid is one waypoint spaced every 30 minutes of latitude and every 2° of longitude.
- **Superclasses***:*
  - atm:LatLonFix

3.9.1.3 atm:MeterFix

- **Definition**: A fix defined as a point in the terminal airspace through which flights are metered by air traffic control on approach.
- **Superclasses***:*
  - atm:AbsoluteFix

3.9.1.4 atm:NavaidFix

- **Definition**: A fix based on the location of a ground-based Navigation Aid (Navaid) installation.
- **Superclasses***:*
  - atm:AbsoluteFix

3.9.1.4.1 atm:NDBfix

- **Definition**: A fix based on the location of a ground-based non-directional radio beacon (NDB) installation. NDB signals follow the curvature of the Earth, so they can be received at much greater distances at lower altitudes, a major advantage over VOR. However, NDB signals are also affected more by atmospheric conditions, mountainous terrain, coastal refraction and electrical storms, particularly at long range.
- **Superclasses***:*
  - atm:NavaidFix

3.9.1.4.2 atm:TACANfix

- **Definition**: A fix based on the location of a ground-based TACAN (TACtical Air Navigation) installation.
- **Superclasses***:*
  - atm:NavaidFix

3.9.1.4.3 atm:VORfix

- **Definition**: A fix based on the location of a ground-based VOR (VHF Omni Directional Radio Range) installation.
- **Superclasses***:*
  - atm:NavaidFix

3.9.1.4.3.1 atm:AirportFix

- **Definition**: A subclass of navigation fix corresponding to fixes associated directly with a ground reference point that is an airport.
- **Superclasses***:*
  - atm:VORFix

3.9.2 atm:RelativeFix

- **Definition**: A fix defined in relation to another fix.
- **Superclasses***:*
  - atm:NavigationFix
- **Subclasses:**
  - atm:FRDfix
- **Object properties**:
  - atm:referenceFix [atm:AbsoluteFix]: Links a relative fix to the absolute fix that serves as the basis for its location. (A relative fix position is defined relative to an absolute fix position, e.g. as a vector offset from the absolute fix.)
- **Datatype properties**:
  - atm:relativeAngle [integer]: The angular direction in degrees (1-360) of a relative fix with respect to its defining absolute fix.
  - atm:relativeDistance [integer]: The distance (in nautical miles) between a relative fix and its defining (absolute) fix.

3.9.2.1 atm:FRDfix

- **Definition:** A Fix Radial Distance (FRD) fix is located a specified distance from a ground-based navaid, a named fix, or an airport, along a given radial vector.
- **Superclasses***:*
    - atm:RelativeFix

3.10 nas:SIDSTAR

- **Description:** This class represents the generalized structure of an airport departure (nas:SID) or arrival route (nas:STAR). SIDs and STARs are sets of predefined macro-routes that aircraft follow in the immediate vicinity of an airport, either on departure or arrival. SIDs and STARs have a similar structure: they are composed of a common route that all aircraft follow, plus two sets of diverging or converging routes (transition routes and airport routes) that direct the aircraft to or from the common route.
- **Superclasses:**
    - nas:AirspaceInfrastructureComponent
- **Subclasses:**
    - nas:SID
    - nas:STAR
- **Object properties:**
    - nas:hasAirportRoute [AirportRoute]: For STARs, the set of routes that aircraft can follow to a destination airport after leaving the common route; for SIDs, the set of routes that aircraft can follow from an originating airport to join the common route.
    - nas:hasCommonRoute [CommonRoute]: A route that is flown by all aircraft navigating the SID or STAR.
    - nas:hasTransitionRoute [TransitionRoute]: For STARs, the set of routes that aircraft can follow before commencing the common route; for SIDs, the set of routes that aircraft can follow after flying the common route.
- **Datatype properties**:
    - nas:sidstarID [string]: The FAA identifier for the SID or STAR. Typically, but not uniformly, 5 alpha characters plus a revision digit.

3.10.1 nas:SID

- **Description:** A SID (Standard Instrument Departure) is an air traffic control coded departure procedure that has been established at certain airports to simplify clearance delivery procedures.
- **Superclasses:**
    - nas:SIDSTAR

3.10.2 nas:STAR

- **Description**: A STAR (Standard Terminal Arrival Route) is an air traffic control coded arrival route established for application to arriving IFR (Instrument Flight Rules) aircraft destined for certain airports.
- **Superclasses:**
    - nas:SIDSTAR

3.10.3 atm:SIDSTARtraverse

- **Description:** A route through a SID (Standard Instrument Departure route) or a STAR (Standard Arrival Route), traversing the common route at minimum and optionally a transition route and/or an airport route. (See nas:SIDSTARroute.)
- **Superclasses:**
  - atm:NavigationElement
- **Object properties**:
  - atm:traversesSIDSTAR [nas:SIDSSTAR]: Links a SIDSTARtraverse (i.e., a specific route through a SID or STAR) to the specific SID or STAR being traversed by a given flight.
  - atm:usesAirportRoute [nas:AirportRoute]: Links a SIDSTARtraverse (a path through a SID/STAR) to the airport route used in making the traverse.
  - atm:usesTransitionRoute [nas:TransitionRoute]: Links a SIDSTARtraverse (a path through a SID/STAR) to the transition route used in making the traverse.

3.11 Illustrative Figures

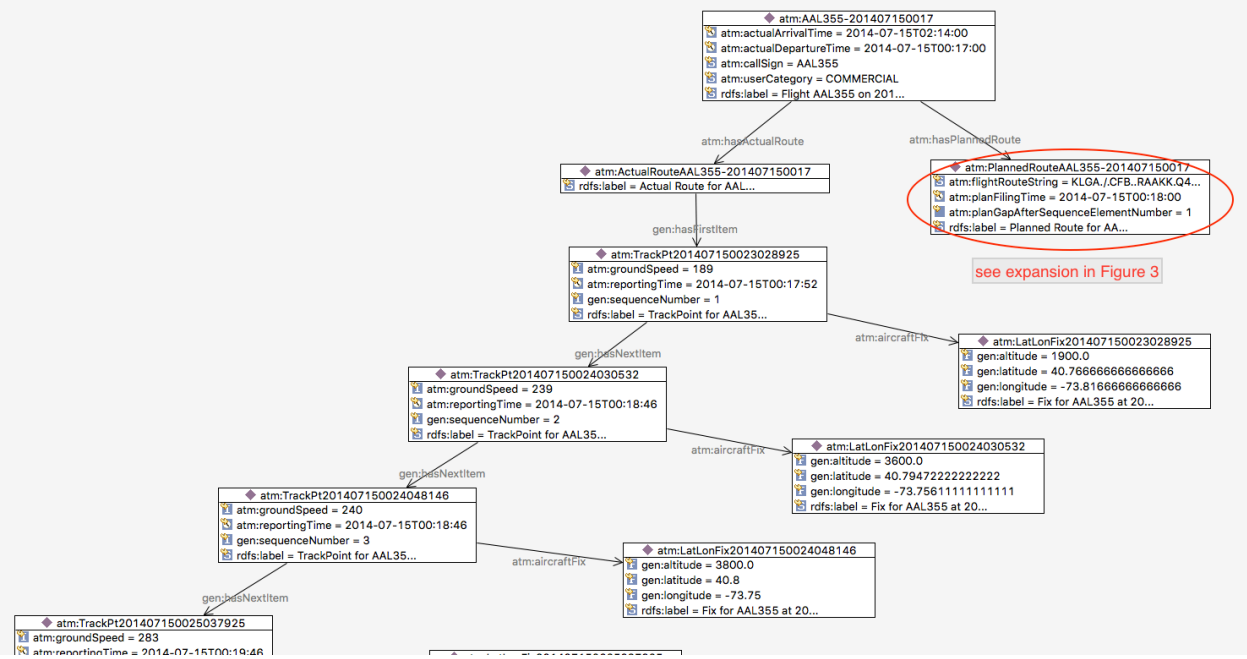


Figure 2: Structure of an Actual Flight Route (Flight Trajectory). This figure illustrates how the actual and planned flight routes for Flight #AAL335 are connected to the flight instance (atm:AAL335-201407150017), which is depicted at the root of the tree structure shown. The actual flight route is represented as a sequence of track points (atm:AircraftTrackPoint). Each track point represents a specific reporting time when the aircraft's fix and speed is captured and relayed to ground systems. The track points are each linked to an instance of atm:LatLonFix, which stores the latitude, longitude, and altitude. The structure of the planned flight route is shown in Figure 3. For a summary of the overall representation of a flight, see Figure 10.

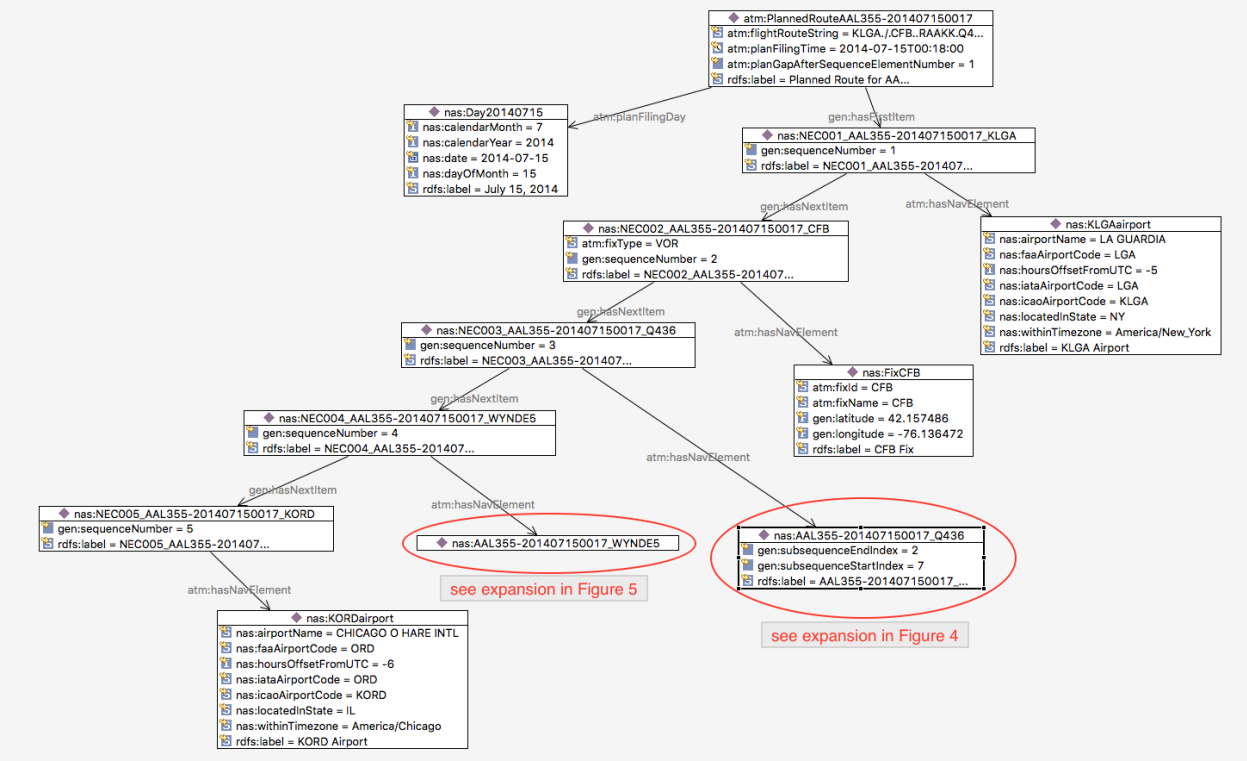


Figure 3: Structure of a Planned Flight Route (Flight Plan). This illustration shows the representation for a flight plan with the flight route string 'KLGA./.CFB..RAAKK.Q436.EMMMA.WYNDE5.KORD'. The flight route string is initially filed by the pilot prior to takeoff and modified as needed en route. The root node (atm:PlannedRouteAAL335-201407150017) contains this route string as a property. The route node is linked to a sequence of 'container' nodes that, in turn, point (via atm:hasNavElement) to the major navigational components through which the flight is planned to progress: the originating airport (KLGA); a VOR fix (CFB); a portion of a high-altitude flight route (Q436); a traverse through a standard terminal arrival route (STAR WYNDE5); and the destination airport (KORD). The route string in this example is truncated, and omits some components early in the flight plan sequence; the gap in the sequence represented by the './.' in the route string. The positional locations of any gaps in the sequence are encoded in the root node as the property atm:planGapAfterSequenceElementNumber. The representation of the Q436 subroute and the traverse of the WYNDE5 STAR is further elaborated in Figure 4 and Figure 5, respectively.

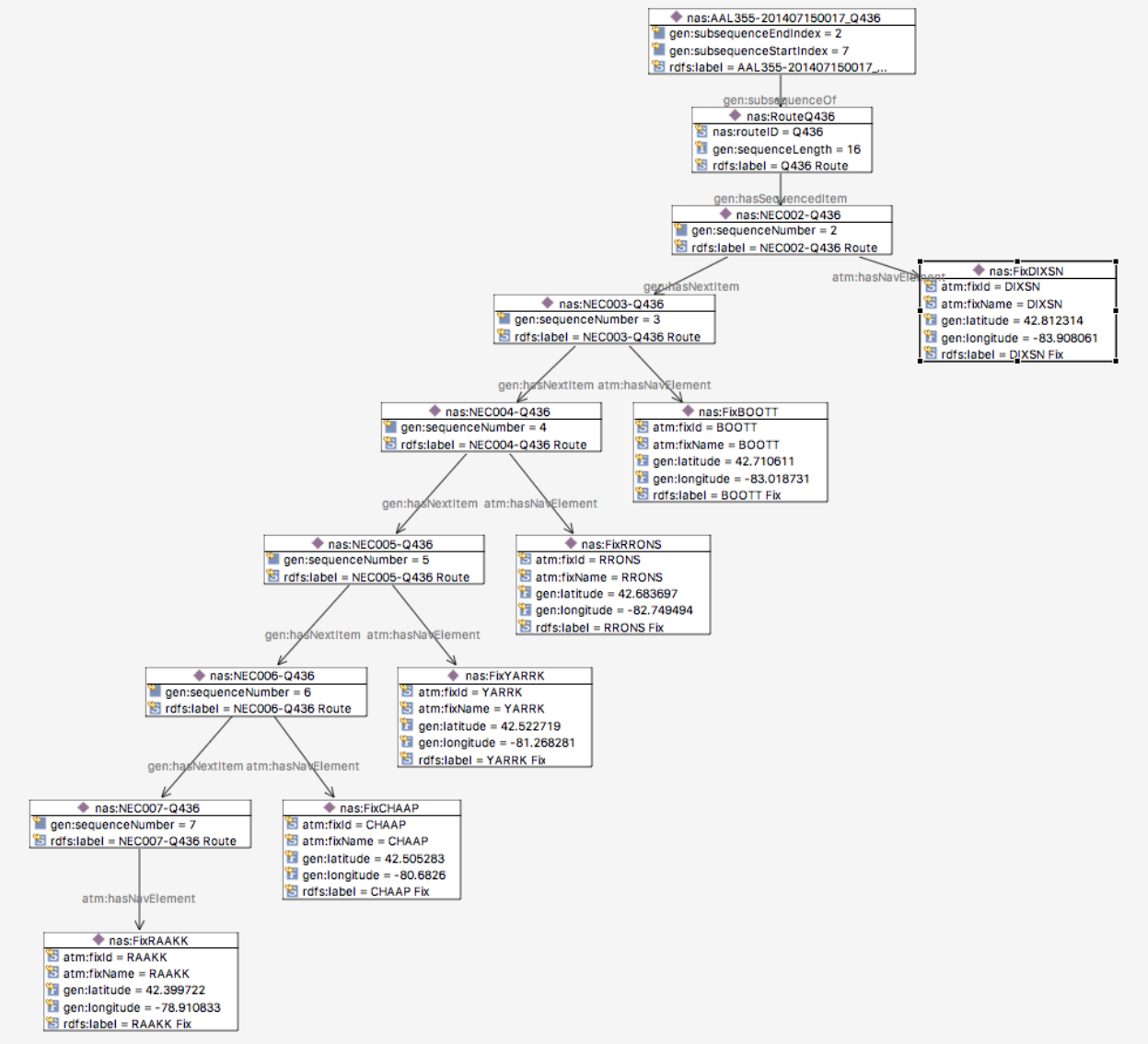


Figure 4: Portion of Route Q436. This figure illustrates how the ontology represents a portion of a predefined sequence (i.e., a subsequence). In this case, the sequence is the high-altitude flight route named Q436. Q436 is defined by the FAA as a predetermined sequence of 16 navigational fixes. In this case, only a portion of the route is to be followed, as specified in the flight plan described in Figure 3 in the node labelled nas:AAL355-201407150017_Q436. The root of this structure indicates that the subsequence starts at position 7 in the Q436 sequence and ends at position 2. Like the flight plan in Figure 3, the flight route itself is represented as a sequence of 'containers' that in turn link to the navigational fixes defined for the route. Note that nodes and links are omitted from this figure to reduce clutter.

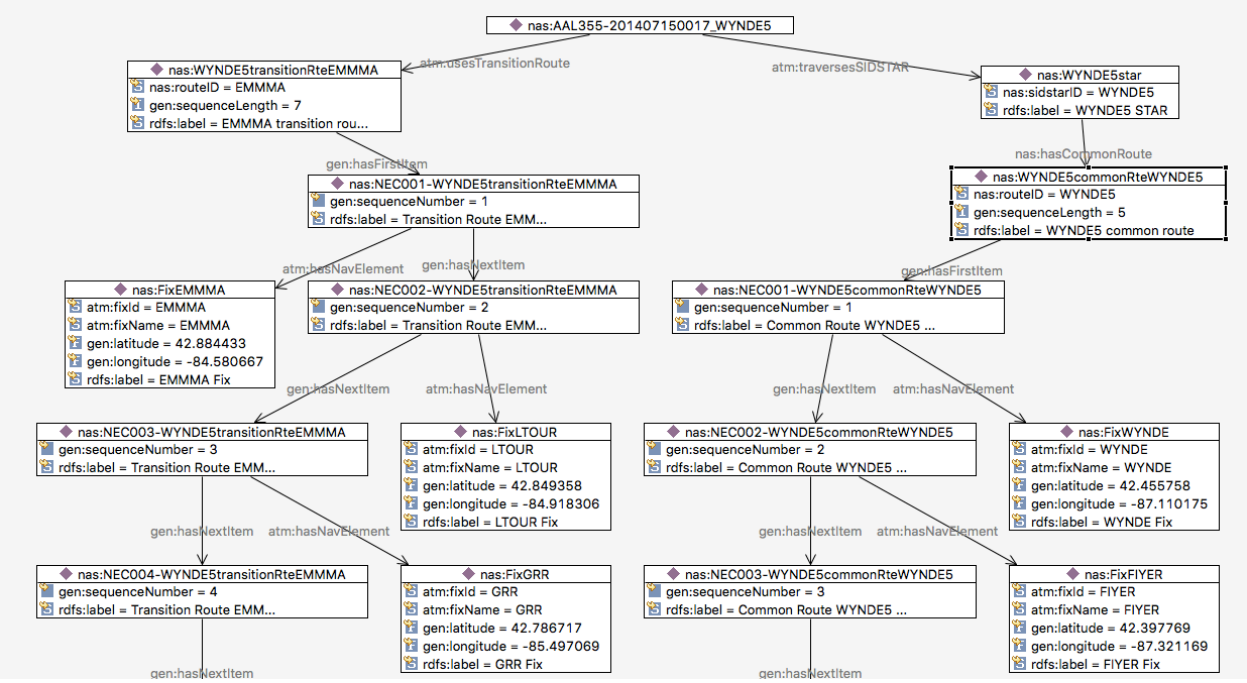


Figure 5: Traverse of STAR WYNDE5. This figure illustrates a portion of the flight plan in Figure 3 that specifies how the aircraft will traverse through the WYNDE5 STAR. A STAR is defined by a set of different flight paths and the flight plan specifies which of the paths will be traversed on this approach into the destination airspace. In this case, the flight will follow the WYNDE5 transition route named EMMA, which merges onto the WYNDE5 common route. Both the transition route and common route are represented as sequences of navigation element containers, where each container points to a navigation fix. This is identical to the representation for route Q436 in Figure 4.

4 Traffic Management Initiatives

The classes in this section define concepts relevant to the representation of Traffic Management Initiatives (TMIs). Each TMI implements a different kind of restriction to aircraft flow, but some aspects are common across all TMIs. All TMIs express a set of conditions under which the TMI is valid, and a set of constraints on aircraft, airports, and/or airspace facilities to which the TMI applies. An attempt has been made to use the same underpinning set of classes across all TMIs where possible, using abstract, reusable classes to describe these constraints on flights, airports, and FAA facilities.

4.1 atm:TrafficManagementInitiative

- **Description:** A Traffic Management Initiative (TMI) is an orchestrated air traffic management procedure implemented by the FAA system command center (ATCSCC) as needed to control the flow of air traffic in the NAS based on capacity and demand. All TMIs have the same basic properties in common; properties more specific to the type of TMI are defined in the various subclasses. Note that both atm:GroundDelayTMI and atm:GroundStopTMI are linked to a class called atm:DelayModel. This class stores the parameters associated with the computational delay model used in determining and assigning delay times to the aircraft involved in the GDP (ground delay) or GS (ground stop). The classes atm:AirportSpec and atm:FlightSpec are used to constrain the set airports or flights to which a given TMI pertains. The class atm:RerouteSegment links a reroute TMI with the set of reroute flight paths that are authorized for this reroute.
- **Subclasses:**
  - atm:AirspaceFlowProgramTMI
  - atm:GroundDelayProgramTMI
  - atm:GroundStopTMI
  - atm:MilesInTrailTMI
  - atm:ReRouteTMI
- **Object properties**:
  - atm:controlledNASelement [atm:TFMcontrolElement]: The ATM element being controlled by the TMI (the airports, sectors, ARTCC tiers, routes, etc.). This element is specified in the traffic management advisory directive.
  - atm:effectiveEndDay [nas:NASday]: The NAS day the TMI was terminated.
  - atm:effectiveStartDay [nas:NASday]: The NAS day the TMI commenced.
  - atm:issuedDay [nas:NASday]: The TMI date of issuance.
- **Datatype properties**:
  - atm:advisoryNumber [integer]: Advisory number as reported from the FAA Command Center database. The number restarts at 001 on every new day UTC.
  - atm:effectiveEndTime [dateTime]: The expected UTC end time of the traffic management initiative (TMI).
  - atm:effectiveStartTime [dateTime]: The expected UTC start time of the traffic management initiative (TMI).
  - atm:extensionProbability [string: “LOW”, “MEDIUM”, “HIGH”]: The probability that this traffic management initiative (TMI) will be extended.
  - atm:initiativeComments [string]: Provides any specific comments on the traffic management initiative (TMI) made by the issuing authority (e.g., ATCSCC, ARTCC).

- atm:issuedTime [dateTime]: The time when the traffic management initiative (TMI) was issued.

4.1.1 atm:AirspaceFlowProgramTMI

- **Definition**: A subclass of Traffic Management Initiative involving control and metering of air traffic through specified airspace volumes.
- **Superclasses**:
  - atm:TrafficManagementInitiative
- **Note**: This is a placeholder. AFPs were not modeled in any detail.

4.1.2 atm:GroundDelayProgramTMI

- **Definition**: A Ground Delay Program (GDP) traffic management initiative (TMI). A GDP is a traffic management procedure where aircraft are delayed at their departure airport in order to manage demand and capacity at their arrival airport.
- **Superclasses**:
  - atm:TrafficManagementInitiative
- **Object properties**:
  - atm:departureScope [atm:AirportSpec]: Links to the specification of the departure airport(s) involved in the GDP TMI.
  - atm:flightInclusionSpec [atm:FlightSpec]: Links to a specification of the set of flights that are included in the GDP restrictions.
  - atm:flightExclusionSpec [atm:FlightSpec]: Links to a specification of the set of flights that are excluded from the GDP restrictions.
  - atm:modeledBy [atm:DelayModel]: Links to a representation of the model used to design the GDP. The model contains all relevant parameters used to create the initiative.
- **Datatype properties**:
  - atm:impactingCondition [string: "weather" , "volume" , "runway" , "equipment" , "other"]: Indicates the reason for initiating the Ground Delay Program.
  - atm:impactingConditionMessage [string]: A free-text description elaborating on the reason for initiating the Ground Delay Program.

4.1.3 atm:GroundStopTMI

- **Definition**: A Ground Stop (GS) traffic management initiative (TMI). A ground stop is a procedure requiring aircraft that meet specific criteria to remain on the ground. The GS may be airport specific, related to a geographical area, or equipment related.
- **Superclasses**:
  - atm:TrafficManagementInitiative
- **Object properties**:
  - atm:departureScope [atm:AirportSpec]: Links to the specification of the departure airport(s) involved in a ground stop TMI. Traffic bound for the destination airport that the GS is intended to control is disallowed.
  - atm:flightInclusionSpec [atm:FlightSpec]: Links to a specification of the set of flights that are included in the GS restrictions.
  - atm:modeledBy [atm:DelayModel]: Links to a representation of the model used to design the GS. The model contains all relevant parameters used to create the initiative.

4.1.4 atm:MilesInTrailTMI

- **Definition**: A Miles-in-Trail (MIT) traffic management initiative (TMI) is used to apportion traffic into a manageable flow, as well as provide space for additional traffic (merging or departing) to enter the flow of traffic. Miles-in-trail describes the number of miles required between aircraft departing an airport, over a fix, at an altitude, through a sector, or on a specific route.
- **Note**: This is a placeholder. MITs were not modeled in any detail.
- **Superclasses**:
  - atm:TrafficManagementInitiative

4.1.5 atm:ReRouteTMI

- **Definition**: A ReRoute is a traffic management initiative (TMI) that mandates a change in the filed flight plan for a set of specified flights. There are multiple factors that might justify a reroute, including weather, traffic congestion, unusual airspace activity, etc. The substitute flight plans reroute air traffic around the airspace problem area.
- **Superclasses**:
  - atm:TrafficManagementInitiative
- **Object properties**:
  - atm:allowedRoute [atm:RerouteSegment]: Links to an approved route between the ReRoute's origin and destination. Multiple approved routes are typically specified in a ReRoute TMI.
  - atm:flightInclusionExclusion [atm:FlightSpec]: Links a ReRoute to a specification of the flight(s) included in or excluded from the ReRoute traffic management initiative.
- **Datatype properties**:
  - atm:implementationStatus [string: "FYI" , "PLN" , "RMD" , "RQD"]: Indicates the enforcement status of the ReRoute advisory: RQD (required), RMD (recommended), PLN (planned for implementation), FYI (informational only).
  - atm:reRouteReason [string: "WEATHER" , "VOLUME" , "EQUIPMENT" , "RUNWAY/TAXIWAY" , "OTHER"]: Provides the reason for initiating a ReRoute.
  - atm:reRouteTimeType [string: "ETD" , "ETA" , "FCA Flight List]: The time period associated with the ReRoute is specified in one of three ways. The ReRoute can apply to flights that: depart specified airports or centers during a certain time window (timeType=ETD); depart specified airports or centers to arrive at their destinations during a certain time window (timeType=ETA); or arrive at the boundary of a flow controlled area (FCA) during a certain time window (timeType=FCA Flight List).
  - atm:reRouteType [string: "ROUTE" , "PLAYBOOK" , "CDR" , "SPECIAL OPERATIONS" , "NRP SUSPENSIONS" , "VS" , "NAT" , "SHUTTLE ACTIVITY" , "FCA" , "FEA" , "INFORMATIONAL" , "MISCELLANEOUS"]: The ReRoute type encodes information about the type of ReRoute being initiated or the reason for the ReRoute.

4.2 atm:TFMcontrolElement

- **Description:** This class represents the diverse set of elements that can be controlled/managed by issuing a Traffic Management Initiative
- **Subclasses:**
  - atm:NavigationElement

    - nas:AirspaceInfrastructureComponent
- **Datatype properties**:
    - atm:maxFlowCapacity [float]: The absolute maximum controllable number of aircraft flowing through an airspace control element per some interval of time under ideal air traffic conditions.

4.3 atm:AirportSpec

- **Description:** This class represents a generalized specification for defining a set of airports. The class can be used, for example, in the definition of a traffic management initiative that is applicable only to a specified set of airports. The *airport set* covered by the specification is determined based on the inclusion or exclusion of airports according to the various properties of atm:AirportSpec listed below.
- **Object properties:**
    - atm:excludesARTCC [nas:ARTCC or nas:ARTCCtier]: any airport not located within the geographical border of an ARTCC or ARTCC tier linked via this property *is* included in the *airport set*
    - atm:excludesAirport [nas:Airport]: any airport linked via this property is *not* included in the *airport set*
    - atm:includesAirport [nas:Airport]: any airport linked via this property *is* included in the *airport set*
    - atm:withinARTCC [nas:ARTCC or nas:ARTCCtier]: any airport located within the geographical border of an ARTCC or ARTCC tier linked via this property *is* included in the *airport set*
    - atm:withinAirportBoundingRegion [gen:CircularRegion]: any airport located within the geographical border of a defined circular region linked via this property *is* included in the *airport set*
- **Datatype properties**:
    - atm:includesAirportType [string: "all","US","CONUS","NonCONUS","International","Canadian"]: all airports of the specified type *are* included in the *airport set*

4.4 atm:FlightSpec

- **Description:** This class represents a generalized specification for defining a set of flights. The class can be used, for example, in the definition of a traffic management initiative (such as a ReRoute) that is only applicable to a specified set of flights. The *flight set* covered by the specification is calculated based on the inclusion or exclusion of flights defined by combining the properties of atm:FlightSpec listed below.
- **Object properties**:
    - atm:excludesAirway [nas:AirspaceRoute]: any flight that is planned to operate along a route linked by this property is *excluded* from the *flight set*
    - atm:excludesFix [atm:NavigationFix]: any flight that is planned to operate through a fix linked by this property is *excluded* from the *flight set*
    - atm:excludesFlight [atm:Flight]: any flight linked by this property is *excluded* from the *flight set*
    - atm:exemptedAFP [atm:AirspaceFlowProgramTMI]: any flight that is included in an airspace flow program linked by this property is *excluded* from the *flight set*
    - atm:includesAirway [nas:AirspaceRoute]: any flight that is planned to operate along a route linked by this property is *included* in the *flight set*

- atm:includesFix [atm:NavigationFix]: any flight that is planned to operate through a fix linked by this property is *included* in the *flight set*
- atm:includesFlight [atm:Flight]: any flight linked by this property is *included* in the *flight set*
- atm:mustHaveDestination [atm:AirportSpec]: any flight whose destination airport is linked via this property is *included* in the *flight set*
- atm:mustHaveOrigin [atm:AirportSpec]: any flight whose origin airport is linked via this property is *included* in the *flight set*
- atm:operatingCarrier [nas:AirCarrier]: any flight operated by a carrier linked via this property is *included* in the *flight set*
- atm:operatingTimeInterval [gen:TimeInterval]: any flight planned to arrive (ETA) – or alternatively depart (ETD) – during the time interval linked via this property is *included* in the *flight set*. The specification of ETA or ETD is made in the property atm:timeConstraintType

- **Datatype properties**:
  - atm:includesAircraftClass [string: "Jet" , "Prop" , "Jet and Prop" , "Turbo" , "All"]: any flight flown using aircraft categorized in the specified class is *included* in the *flight set*
  - atm:aircraftWeightCat [string: "large" , "heavy" , "small"]: any flight operated using an aircraft with a weight in the specified category is *included* in the *flight set*
  - atm:userCategory [string: "GA" , "Air Taxi" , "Cargo" , "Commercial" , "Military"]: any flight that is classified as belonging to the specified category is *included* in the *flight set*
  - atm:timeConstraintType [string: "ETA" or "ETD"]: this property is used in conjunction with specifying atm:operatingTimeInterval

4.5 atm:RerouteSegment

- **Description:** A reroute traffic management initiative provides FAA-approved alternative routings emanating from an origin, converging to a destination, or spanning the entire path between origin and destination. Each instance of the class atm:RerouteSegment specifies one these alternative routings using the defined properties of atm:RerouteSegment.
- **Object properties**:
  - atm:reRouteConstraint [atm:FlightSpec]: links to the flight specification that determines which flights must use this routing
  - atm:reRoutePath [atm:PlannedFlightRoute]: links to the approved flight route for flights satisfying the flight specification
- **Datatype properties**:
  - atm:reRouteSegmentType [string: "origin", "destination", "origin-destination"]: specifies the type of the reroute segment, indicating whether the reRoute path applies to the origin, destination, or entire portion of the flight.

4.6 atm:DelayModel

- **Description:** This class represents a computational delay model used in assigning times to the aircraft involved in the GDP or GS. The properties of this class correspond to parameters that are used by the delay model to compute the delay times.
- **Object properties**:

- atm:adlDay [nas:NASday]: Links a delay model to the day that the ADL (Aggregate Demand List) was generated. Each ADL contains information on the flights arriving and departing from an airport, or entering into and departing from a flow area.
- atm:modeledArrivalRate [atm:ProgramArrivalRateSequence]: Links a delay model used in defining a Ground Delay Program (GDP) or Ground Stop(GS) to a sequence of desired hourly aircraft arrival rates specified for the initiative.
- atm:modeledPopUpFactor [atm:PopupFactorSequence]: Links a delay model used in defining a Ground Delay Program (GDP) or Ground Stop(GS) to a sequence of desired hourly popup factors specified for the initiative. 'Popups' account for late-filing flights (unexpected/unplanned flights) when modeling delays as part of a traffic management initiative (TMI) design process.

- **Datatype properties**:
  - atm:adlTime [dateTime]: The timestamp of the ADL (Aggregate Demand List) that the delay model is based upon. Each ADL contains information on the flights arriving and departing from an airport, or entering into and departing from a flow area.
  - atm:delayAssignmentMode [string: "DAS" , "GAAP" , "UDP"]:The type of delay assignment mode specified for this delay model: Delay Assignment (DAS), General Aviation Airport Program (GAAP), or Unified Delay Program (UDP).
  - atm:modeledAverageDelay [integer]: The average flight delay specified when modeling a Ground Delay Program (GDP) or Ground Stop(GS).
  - atm:modeledMaximumDelay [integer]: The maximum flight delay specified when modeling a Ground Delay Program (GDP) or Ground Stop(GS).
  - atm:modeledTotalDelay [integer]: The total flight delay specified when modeling a Ground Delay Program (GDP) or Ground Stop(GS).
  - atm:PrevAverageDelay [integer]: The average flight delay experienced prior to the start of a Ground Stop(GS).
  - atm:PrevMaximumDelay [integer]: The maximum flight delay experienced prior to the start of a Ground Stop(GS).
  - atm:prevTotalDelay [integer]: The total flight delay experienced prior to the start of a Ground Stop(GS).

4.7 gen:NumericParameter

- **Description**: A class representing numeric parameters explicitly as objects (versus representing them as datatype properties). These parameters can be inserted into containers and sequenced. This enables reuse of the same parameter value in multiple sequences.
- **Subclasses**:
  - gen:FloatParameter
  - gen:IntegerParameter
- **Datatype properties:**
  - gen:parameterValue [*untyped*]: The numeric value of the parameter.

4.7.1 gen:FloatParameter

- **Description**: A subclass of NumericParameter representing parameters with floating-point values. (Note: gen:parameterValue is restricted by this class to *float* only using OWL restrictions.)
- **Superclasses**:

- gen:NumericParameter

4.7.2 gen:IntegerParameter

- **Description**: A subclass of NumericParameter representing parameters with integer values. (Note: gen:parameterValue is restricted by this class to *integer* only using OWL restrictions.)
- **Superclasses**:
  - gen:NumericParameter
- **Subclasses**:
  - atm:PopupFactor
  - atm:ProgramArrivalRate

4.7.2.1 atm:PopupFactor

- **Description**: Factor to account for late-filing flights when modeling delays as part of a traffic management initiative (TMI) design process. This is the number of unanticipated late-filing flights per hour.
- **Superclasses**:
  - gen:IntegerParameter

4.7.2.2 atm:ProgramArrivalRate

- **Description**: The number of aircraft that the Ground Delay Program (GDP) is intended to provide inbound into the airport for a given hour.
- **Superclasses**:
  - gen:IntegerParameter

4.8 atm:NumericParameterContainer

- **Description**: A wrapper around a numeric parameter. Containers can be sequenced to create an ordered list of numeric parameters.
- **Superclasses**:
  - gen:SequencedItem
- Subclasses:
  - atm:PopupFactorContainer
  - atm:ProgramArrivalRateContainer

4.8.1 atm:PopupFactorContainer

- **Description**: A wrapper around a popup factor parameter. Used to sequence popup factors in the specification of a Ground Delay Program (GDP).
- **Superclasses**:
  - atm:NumericParameterContainer

4.8.2 atm:ProgramArrivalRateContainer

- **Description**: A wrapper around a program arrival rate, used to sequence them into an ordered list.
- **Superclasses**:
  - atm:NumericParameterContainer

4.9 atm:PopupFactorSequence

- **Description**: This ordered sequence contains the anticipated hourly popup factor for each hour as the Ground Delay Program (GDP) progresses through its period of implementation.

4.10 atm:ProgramArrivalRateSequence

- **Description**: This ordered sequence contains the planned program arrival rates for each hour as the Ground Delay Program (GDP) progresses through its period of implementation.

4.11 Illustrative Figures

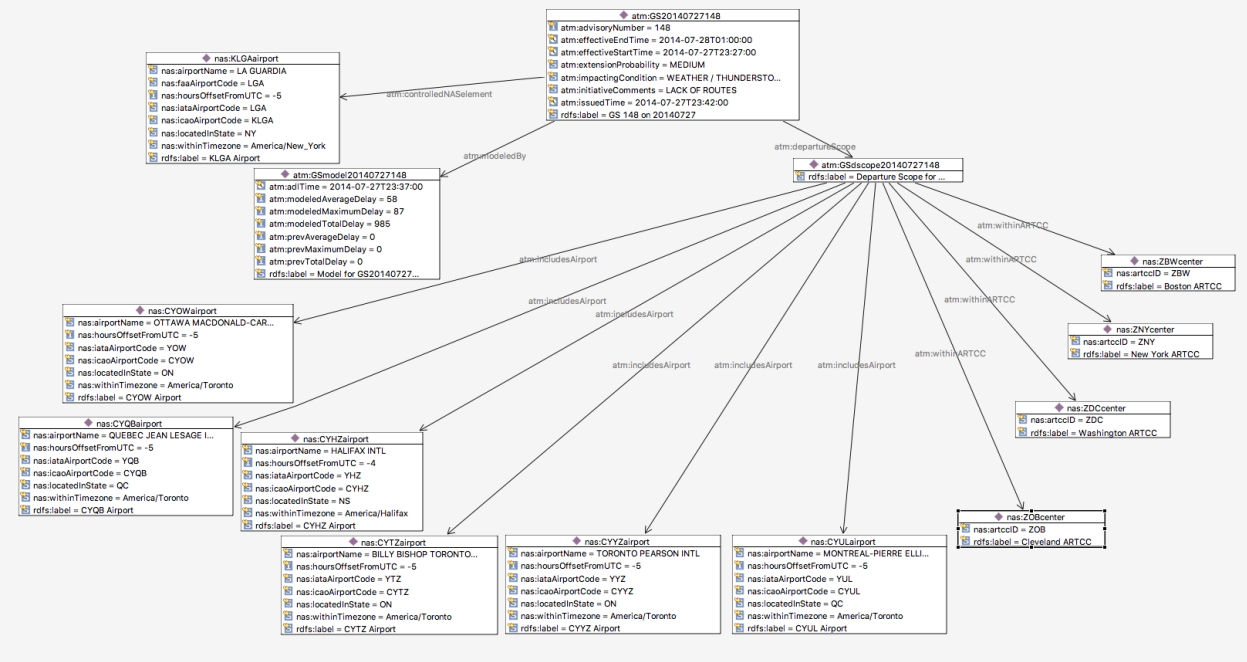


Figure 6: Ground Stop. This figure illustrates the representation for a Ground Stop TMI implemented for flights bound for KLGA on 7/27/14 at 23:27 UTC. The departure scope specifies the airports and Centers from which traffic bound for KLGA is disallowed. The delay model linked by atm:modeledBy indicates the projected average, maximum, and total delay if the ground stop is implemented.

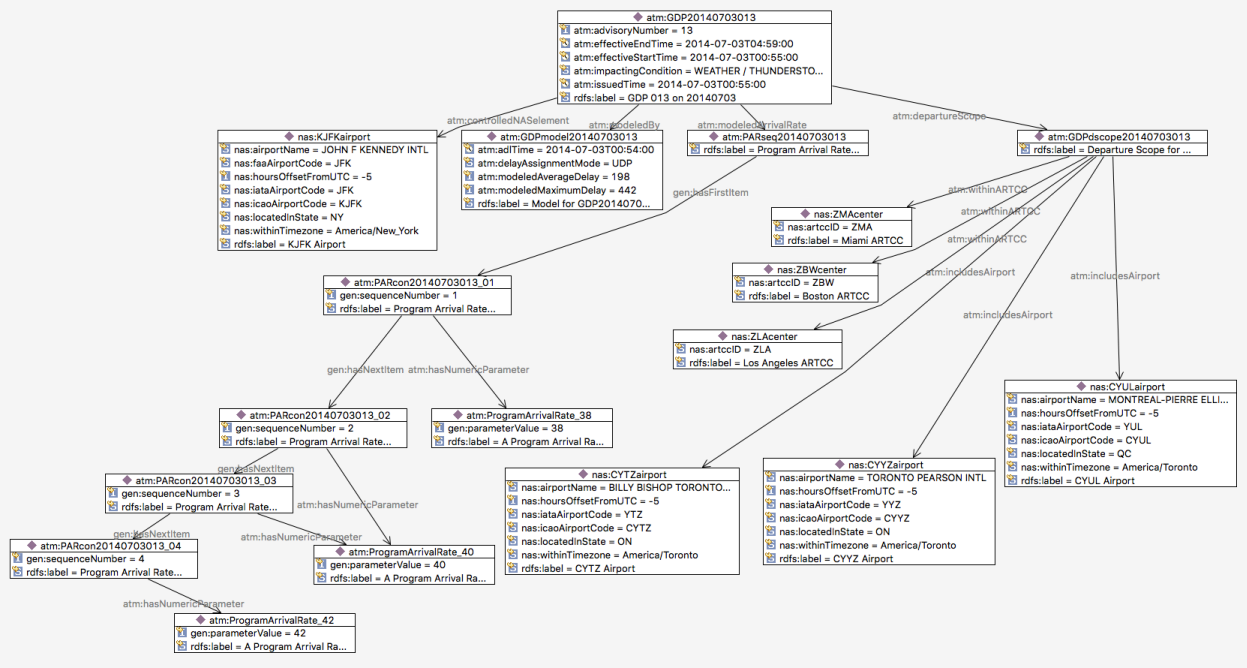


Figure 7: Ground Delay Program. This figure has the same structure as Figure 6, but illustrates how the projected hourly Program Arrival Rates (PARs) are modeled as a sequence of containers that point to program arrival rate values. (The PAR represents the number of desired number of aircraft inbound to a controlled airport in an hour.) In this case, the first hour is projected to have a PAR of 38, the second and third hours have a PAR of 40, and the fourth hour has a PAR of 42. Note that in this figure, the departure scope airports and Centers are not fully represented to avoid cluttering.

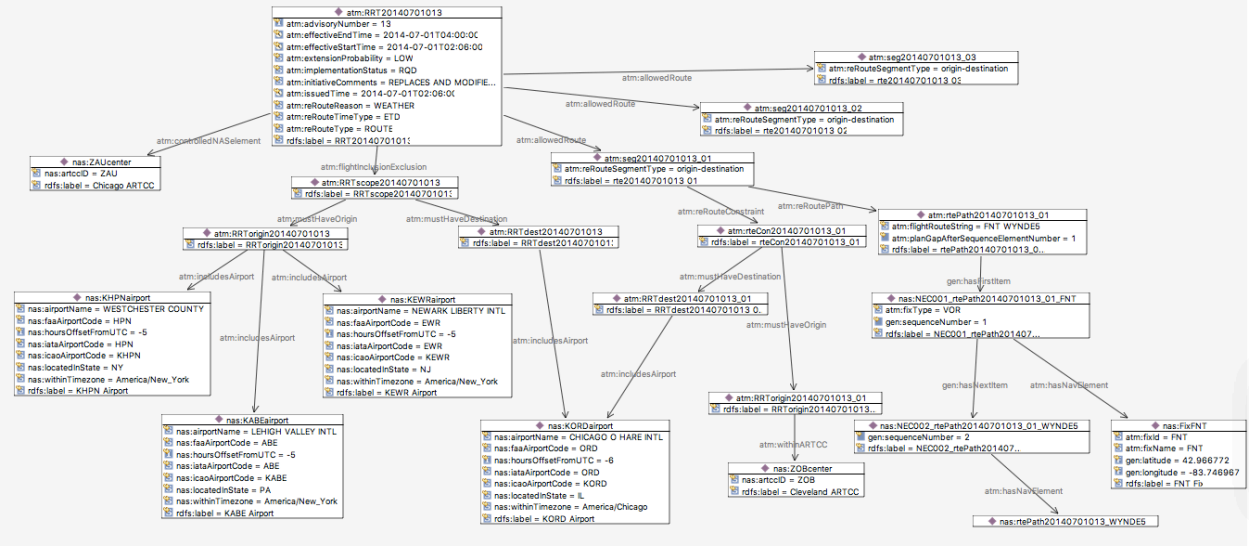


Figure 8: ReRoute TMI. This figure illustrates the ontology structure for the ReRoute TMI shown in Figure 9. (Note however that nodes are omitted to make the graph readable.) Some of the route alternatives specified for this TMI are linked to the main ReRoute TMI node via the atm:allowedRoute link. Each route alterative includes a route (linked via atm:reRoutePath) and a set of constraints under which the route is appropriate (linked via atm:reRouteConstraint). In this example, the first route alternative is a route that follows the flight path designated by the flight route string 'FNT WYNDE5', which corresponds to

flying the FNT transition route into the WYNDE5 STAR on arrival into KORD. That routing is allowed under the constraint that the flight origin must be KORD, and the destination must be an airport within the Cleveland ARTCC (ZOB Center). Aside from constraints on when the specific route alternatives are permitted, there are also overall constraints on the ReRoute TMI, linked with the property atm:flightInclusionExclusion. In this case, the TMI applies to any flight that originates from any one of a set of specified airports and is destined for KORD.

ATCSCC Advisory

ATCSCC ADVZY 013 DCC 07/01/2014 ROUTE RQD /FL

RAW TEXT:

```
ATCSCC ADVZY 013 DCC 07/01/14 ROUTE RQD /FL
NAME: NE_2_ORD
CONSTRAINED AREA: ZAU
REASON: WEATHER
INCLUDE TRAFFIC: ABE/EWR/HPN/JFK/LGA/MDT/PHL/TEB/ZBW/ZDC/ZOB
                 DEPARTURES TO KORD/ORD
FACILITIES INCLUDED: ZAU/ZBW/ZDC/ZNY/ZOB
FLIGHT STATUS: ALL_FLIGHTS
VALID: ETD 010000 TO 010400
PROBABILITY OF EXTENSION: LOW
REMARKS: REPLACES AND MODIFIED ADVZY 001.
ASSOCIATED RESTRICTIONS: SEE NTML
MODIFICATIONS:
ROUTES:

ORIG                DEST                ROUTE
----                ----                -----
ZOB                 ORD                 >FNT WYNDE5<
EWR JFK LGA TEB     ORD                 >COATE Q436 EMMMA WYNDE5<
HPN
PHL ABE MDT         ORD                 >PSB J60 DJB FNT WYNDE5<
ZBW                 ORD                 >FNT WYNDE5<
ZDC                 KORD                >BUFFR J518 DJB FNT WYNDE5<


TMI ID: RRDCC013
010206-010400
14/07/01 02:06  DCCOPS./nfs/lxstn28
```

Figure 9: ReRoute TMI Advisory #013 on 07/01/2014. This is a screenshot from the fly.faa.gov website, which publishes FAA advisories, including TMIs. The ontology representation for this advisory is discussed in Figure 8.

5 Operations: Flight, Carrier, and Aircraft

The classes in this section pertain the definition and operation of a flight, extending to both the operator and the aircraft, including its make, model, and manufacturer.

5.1 atm:Flight

- **Description:** A single flight segment from origin to destination.
- **Object properties:**
  - atm:actualArrivalDay [nas:NASday]: Links a flight with the day on which the flight arrives. This is the day determined by surveillance data sources to be the 'actual' day of arrival, versus the scheduled day or the day specified in the flight plan, etc.
  - atm:actualDepartureDay [nas:NASday]: Links a flight with the day on which the flight departs. This is the day determined by surveillance data sources to be the 'actual' day of departure, versus the scheduled day or the day specified in the flight plan, etc.
  - atm:aircraftFlown [eqp:Aircraft]: Links a flight with the actual aircraft used.
  - atm:aircraftTypeFlown [eqp:AircraftType]: Links a flight with the type of aircraft flown. The aircraft 'type' corresponds to a set of aircraft models with similar characteristics.
  - atm:alternateArrivalAirport [nas:Airport]: Links to the alternate arrival airport to be used in case of weather, traffic, or other unforseen contingency. The alternate airport is specified in the flight plan.
  - atm:arrivalAirport [nas:Airport]: Links to the actual arrival airport for the flight. This is the airport determined by surveillance data sources to be the 'actual' arrival airport, versus the scheduled airport or the airport specified in the flight plan, etc.
  - atm:arrivalRunway [nas:OperationalRunway]: Links to the actual arrival runway for the flight.
  - atm:arrivalTaxiPath [atm:Taxipath]: Links to the taxipath followed by the flight en route to the gate.
  - atm:departureAirport [atm:Airport]: Links to the actual departure airport for the flight. This is the airport determined by surveillance data sources to be the 'actual' departure airport, versus the scheduled airport or the airport specified in the flight plan, etc.
  - atm:departureRunway [nas:OperationalRunway]: Links to the actual arrival airport for the flight.
  - atm:departureTaxiPath [atm:Taxipath]: Links to the taxipath followed by the flight en route to the runway.
  - atm:hasActualRoute [atm:ActualFlightRoute]: Links a flight to its actual trajectory (i.e., a sequence of track points), as determined by flight track surveillance data.
  - atm:hasCrewMember [atm:CrewMember]: Links a flight to its crew members.
  - atm:hasPlannedRoute [atm:PlannedFlightRoute]: Links a flight to a version of its flight plan. (Which version gets stored is application-dependent.) There are multiple versions of the flight plan generated and then amended during the course of the flight, starting in the pre-flight timeframe and extending through to the termination of the flight.
  - atm:operatedBy [nas:AirCarrier]: Links a flight to its carrier airline.
  - atm:plannedArrivalDay [nas:NASday]: Links to the day that the flight is planned to arrive per the initial filed flight plan.
  - atm:plannedDepartureDay [nas:NASday]: Links to the day that the flight is planned to depart per the initial filed flight plan.

- atm:publishedArrivalDay [nas:NASday]: Links to the day that the flight is planned to arrive per the OAG schedule.
- atm:publishedDepartureDay [nas:NASday]: Links to the day that the flight is planned to depart per the OAG schedule.

- **Datatype properties**:
  - atm:actualArrivalTime [dateTime]: The time determined by surveillance data sources to be the 'actual' time of flight arrival, versus the scheduled time or the time specified in the flight plan, etc.
  - atm:actualDepartureTime [dateTime]: The time determined by surveillance data sources to be the 'actual' time of flight departure, versus the scheduled time or the time specified in the flight plan, etc.
  - atm:adsbID [string]: The unique identifier of the ADS-B (Automatic Dependent Surveillance - Broadcast) transponder unit on board the aircraft for this flight.
  - atm:callSign [string]: The flight identifier (3-letter ICAO code plus number).
  - atm:cruisingAltitude [integer]: The cruising altitude (in number of feet) specified in the initial filed flight plan for this flight.
  - atm:fixTrajectoryString [string]: This is an experimental property of a flight that stores an ordered sequence of named fixes corresponding one-to-one to the sequence of trajectory track points recorded for the flight. For a given track point, the corresponding named fix is computed by finding the closest named fix to the track point.
  - atm:plannedArrivalTime [dateTime]: The time that the flight is planned to arrive per the initial filed flight plan.
  - atm:plannedDepartureTime [dateTime]: The time that the flight is planned to depart per the initial filed flight plan.
  - atm:publishedArrivalTime [dateTime]: The time that the flight is planned to arrive per the OAG schedule.
  - atm:publishedDepartureTime [dateTime]: The time that the flight is planned to depart per the OAG schedule.
  - atm:traconID [string]: Three letter FAA TRACON identifier code.
  - atm:trueAirspeed [integer]: The true airspeed of a flight: the speed of the aircraft relative to the airmass in which it is flying.
  - atm:userCategory [string: "GA" , "Air Taxi" , "Cargo" , "Commercial" , "Military"]: The type of airspace user that this flight represents. This property can be used to constrain a flight specification (atm:FlightSpec) to contain only flights representing a specific user category.

5.2 atm:CrewMember

- **Description:** The set of inflight airline personnel operating or performing servicing functions on a flight (pilots and cabin attendants). Note: this is a placeholder class and has not been modeled in any detail.

5.3 eqp:AviationServiceProvider

- **Description:** A superclass encompassing all entities that provide aviation services in the public and private sectors.
- **Subclasses:**
  - nas:AirCarrier

- nas:AviationIndustryManufacturer
- nas:GovernmentAviationServiceProvider

5.3.1 nas:AirCarrier

- **Description:** A commercial entity that is licensed to operate aircraft. The instances of this class were derived from the list published at http://openflights.org/data.html#airline.
- **Superclasses:**
  - nas:AviationServiceProvider

5.3.2 nas:AviationIndustryManufacturer

- **Description:** A superclass encompassing the set of airframe and engine manufacturers.
- **Superclasses:**
  - nas:AviationServiceProvider
- **Subclasses:**
  - nas:AirframeManufacturer
  - nas:AircraftEngineManufacturer

5.3.2.1 nas:AirframeManufacturer

- **Description:** A manufacturer of aircraft.
- **Note:** The instances of this class were derived from the IACIS (International Aircraft Categorization And Identification Standard) Aircraft Taxonomy produced by the CAST/ICAO Common Taxonomy Team (http://www.intlaviationstandards.org).
- **Superclasses:** nas:AviationIndustryManufacturer

5.3.2.2 nas:AircraftEngineManufacturer

- **Description:** A manufacturer of aircraft engines.
- **Superclasses:** nas:AviationIndustryManufacturer

5.3.3 nas:GovernmentAviationServiceProvider

- **Description:** A superclass encompassing the set of airframe and engine manufacturers.
- **Superclasses:**
  - nas:AviationServiceProvider

5.4 eqp:EngineeredSystem

- **Description:** Represents engineered systems as either decomposable or non-decomposable subsystems.
- **Subclasses:**
  - eqp:DecomposableSystem
  - eqp:UnitAssembly
- **Object properties:**
  - eqp:manufacturedBy [nas:AviationIndustryManufacturer]: Links to the manufacturer of the engineered system.
- **Datatype properties**:
  - eqp:manufactureYear [integer]: Year that the system was manufactured.

- eqp:modelID [string]: The system model identifier.

5.4.1 eqp:DecomposableSystem

- **Description:** Represents a complex engineering system that can be further decomposed into a set of decomposable subsystems and primitive components.
- **Superclasses**:
  - eqp:EngineeredSystem
- **Subclasses:**
  - eqp:AircraftSubsystem
  - eqp:NavigationAid
  - eqp:Aircraft
- **Object properties:**
  - eqp:hasComponent [eqp:UnitAssembly]:Links to any primitive portion of the decomposable system. This corresponds to a unit assembly that is not modeled at a finer level of granularity.
  - eqp:hasSubsystem [eqp:DecomposableSystem]: Links to a further decomposable portion of the engineered system.

5.4.1.1 eqp:AircraftSubsystem

- **Description**: An engineering subsystem of an aircraft.
- **Superclasses**:
  - eqp:DecomposableSystem
- **Subclasses**:
  - eqp:AircraftCommunicationSystem
  - eqp:AircraftEngine
  - eqp:AircraftNavigationSystem
  - eqp:ElectricalPowerSystem
- **Note**: The listed subclasses are for illustration purposes only and have not been further modeled in the ontology

5.4.1.1.1 eqp:AircraftCommunicationsSystem

- **Description**: An aircraft subsystem involving all voice and data communications functions.
- **Superclasses**:
  - eqp:AircraftSubsystem

5.4.1.1.2 eqp:AircraftEngine

- **Description**: An aircraft engine is the component of the propulsion system for an aircraft that generates mechanical power.
- **Superclasses**:
  - eqp:AircraftSubsystem

5.4.1.1.2.1 eqp:EngineType

- **Description:** Represents generic types of aircraft engines. Specific types are represented as instances of this class, including types for electric engine, jet engine, propeller engine, and turboprop engine. This class is used to map aircraft types into one of these engine types.
- **Datatype properties:**
  - eqp:engineTypeDescription [string]: A text description of the engine type characteristics.

5.4.1.1.3 eqp:AircraftNavigationSystem

- **Description**: An aircraft subsystem responsible for navigation functions.
- **Superclasses**:
  - eqp:AircraftSubsystem

5.4.1.1.4 eqp:ElectricalPowerSystem

- **Description**: An aircraft subsystem responsible for generating and distributing electrical power.
- **Superclasses**:
  - eqp:AircraftSubsystem

5.4.1.2 eqp:NavigationAid

- **Description**: A navaid (navigational aid) is an engineered system on the ground that airplanes can detect based on their emission of radio signals that enable the aircraft to navigate. Modern examples include NDBs and VORs, which both transmit radio signals that aircraft can follow and home in on.
- **Superclasses**:
  - eqp:DecomposableSystem
- **Note**: This class is for illustration purposes only and has not been modeled in detail in the ontology

5.4.2 eqp:UnitAssembly

- **Description**: A primitive, non-decomposible component in an engineered system.
- **Superclasses**:
  - eqp:EngineeredSystem
- **Subclasses**:
  - eqp:BallBearing
- **Note**: The listed subclasses are for illustration purposes only and have not been further modeled in the ontology

5.4.2.1 eqp:BallBearing

- **Description**: A unit assembly in an aircraft mechanical subsystem, a ball bearing is positioned between a wheel and a fixed axle, in which the rotating part and the stationary part are separated by a ring of small solid metal balls that reduce friction.
- **Superclasses**:
  - eqp:UnitAssembly

5.4.3 eqp:Aircraft

- **Description:** This class represents the physical realization of an aircraft, produced by a manufacturer according to the specifications defined for a specific model. Properties of aircraft instances, such as serial number, registration (tail) number, aircraft registrant, and others are derived from the FAA's Aircraft Registry (https://www.faa.gov/licenses_certificates/aircraft_certification/aircraft_registry/releasable_aircraft_download/).
- **Superclasses:**
  - eqp:DecomposableSystem
  - gen:SequencedItem
- **Object properties:**
  - eqp:hasAircraftModel [eqp:AircraftModel]: Links to the model for this aircraft. The model is the abstract specification used in the manufacture of the physical aircraft.
- **Datatype properties**:
  - eqp:aircraftModelCertainty [float]: A numeric measure between 0 and 1 reflecting how certain the eqp:hasAircraftModel link is to be correct between the aircraft and the model. A value of one indicates 100% certainty. A lesser value is not to be taken as a percentage, but is a heuristic similarity value taken from the scoring of the TF/IDF similarity assessment performed between the FAA aircraft make/model in the FAA registration database and ICAO make/model names used in the ontology (see eqp:AircraftModel for details).
  - eqp:aircraftRegistrant [string]: The entity that registered the aircraft with the FAA. (Note: The registrant is represented as a string, but at some point should instead be linked to a named business entities. Often the entities are carrier, which are already represented in the ontology.)
  - eqp:aircraftSerialNumber [string]: The manufacturer serial number of the aircraft.
  - eqp:certificateIssueDate [date]: The date that the airworthiness certificate was issued for the aircraft.
  - eqp:modeSCode[string]: The unique Mode S address of the transponder aboard the aircraft. Mode S equipment on aircraft are assigned a unique ICAO 24-bit address or (informally) Mode S hex code upon national registration and this address becomes a part of the aircraft's Certificate of Registration.
  - eqp:numberOfEngines [integer]: The number of engines specified for this aircraft model or found on this aircraft.
  - eqp:numberOfSeats [integer]: The number of seats on this aircraft.
  - eqp:registrationNumber [string]: The aircraft FAA registration number (the tail number, or the N-Number in the US).

5.4.3.1 eqp:AircraftModel

- **Description:** An aircraft model represents a generic specification describing the characteristics of a specific type of aircraft to be manufactured. The subclasses and instances beneath eqp:AircraftModel are derived from the IACIS (International Aircraft Categorization And Identification Standard) Aircraft Taxonomy produced by the CAST/ICAO Common Taxonomy Team (http://www.intlaviationstandards.org). The subclasses of eqp:aircraftModel correspond to sets of related aircraft models, as defined by the

taxonomy notion of a 'master model'; instances correspond to an individual 'make/model/series' in the taxonomy.

- **Object properties:**
  - eqp:isAircraftType [eqp:AircraftType]: Links an aircraft model to its corresponding aircraft type. (Each aircraft type encompasses a set of related models.)
  - eqp:designedBy [nas:AirframeManufacturer]: Links an aircraft model to the airframe manufacturer that designed the model.
- **Datatype properties**:
  - eqp:cicttNumber [integer]: CAST/ICAO Common Taxonomy Team (CICTT) identifier for this model as specified by the International Aircraft Categorization And Identification Standard (IACIS).
  - eqp:numberOfEngines [integer]: The number of engines specified for this aircraft model or found on this aircraft.

5.4.3.2 eqp:AircraftType

- **Description:** The aircraft type designator is a coding scheme that specifies aircraft models with similar operational flight characteristics. ICAO maintains the list of aircraft type designator codes. Each code links to one or more aircraft models using the model's property eqp:isAircraftType. In addition, the type designator is linked to other properties common to the set of aircraft models represented by the type designator. The aircraft type designator is similar to the CAST/ICAO Common Taxonomy (http://www.intlaviationstandards.org) notion of a 'master model', except an aircraft type may cover a broader set of aircraft models than a 'master model'.
- **Object properties:**
  - eqp:hasAircraftEngineType [eqp:EngineType]: links to the type of engine used by this aircraft type
  - eqp:hasAircraftWakeCategory [eqp:AircraftWakeCategory]: links to the aircraft wake category assigned by FAA to this type of aircraft
  - eqp:hasAircraftWeightClass [eqp:AircraftWeightClass]: links to the weight class assigned by FAA to this type of aircraft
- **Datatype properties**:
  - eqp:aircraftTypeDesignator [string]: The type designator code for this type of aircraft (e.g., B777).

5.4.3.3 eqp:AircraftWakeCategory

- **Description:** A category defined in terms of aircraft wake turbulance characteristics (including takeoff weight and wingspan). The wake turbulence category is useful for the purpose of FAA separation assurance. See FAA Order JO 7110.659B, Wake Turbulence Recategorization, effective March 01, 2015. There are 8 defined categories, which are represented as instances of this class. This class is used to map the set of aircraft types into one of these categories.
- **Datatype properties:**
  - eqp:maxTakeoffWeightHighBound [integer]: The upper bound of the maximum takeoff weight for this category.
  - eqp:maxTakeoffWeightLowBound [integer]: The lower bound of the maximum takeoff weight for this category.

- eqp:wakeCategoryID [string]: The wake category identification character (A through F) as defined in the FAA Order.
- eqp:wingSpanHighBound [integer]: The upper bound of the wingspan for this category. If not specified, the upper range of the wingspan interval is unbounded.
- eqp:wingSpanLowBound [integer]: The lower bound of the wingspan for this category.

5.4.3.4 *eqp:AircraftWeightClass*

- **Description:** A category defined in terms of a specified aircraft weight interval. There are three defined categories of aircraft weight (high, medium, low), which are represented as instances of this class. This class is used to map the set of aircraft types into one of these categories.
- **Datatype properties:**
  - eqp:aircraftWeightHighBound [integer]: The upper bound of the aircraft weight interval. If not provided, the upper range of the weight interval is unbounded.
  - eqp:aircraftWeightLowBound [integer]: The lower bound of the aircraft weight interval.

5.5 *Illustrative Figures*

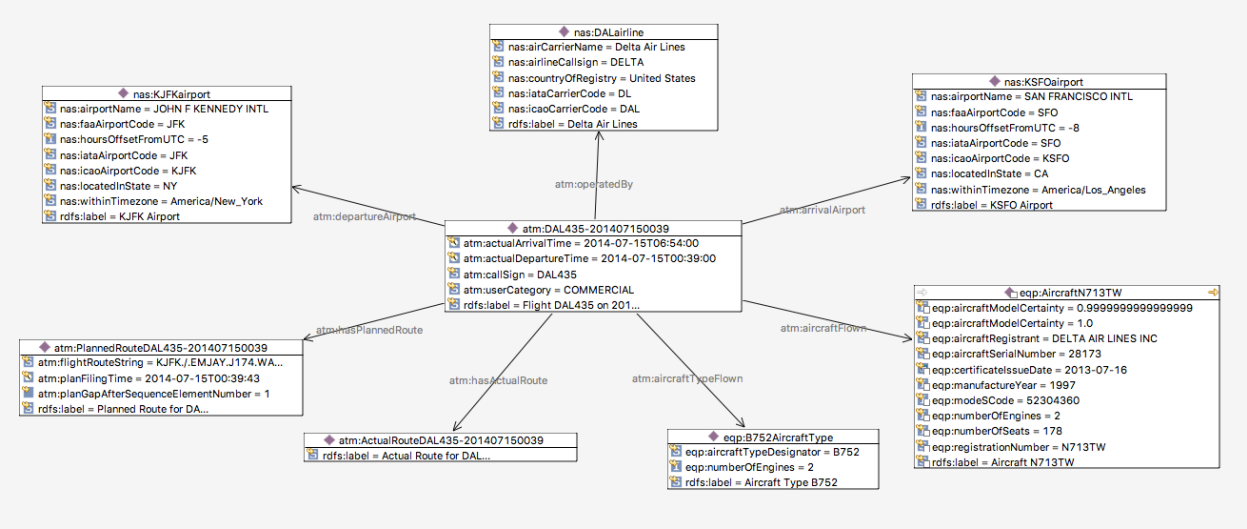


Figure 10: Structure of a Flight. This figure illustrates the basic components of the ontology representation of a flight. Each flight is associated with its departure and arrival airports; the aircraft, aircraft type, and operating carrier; and the actual and planned flight route. The representation for the actual and planned flight is described in Figure 2 and Figure 3, respectively.

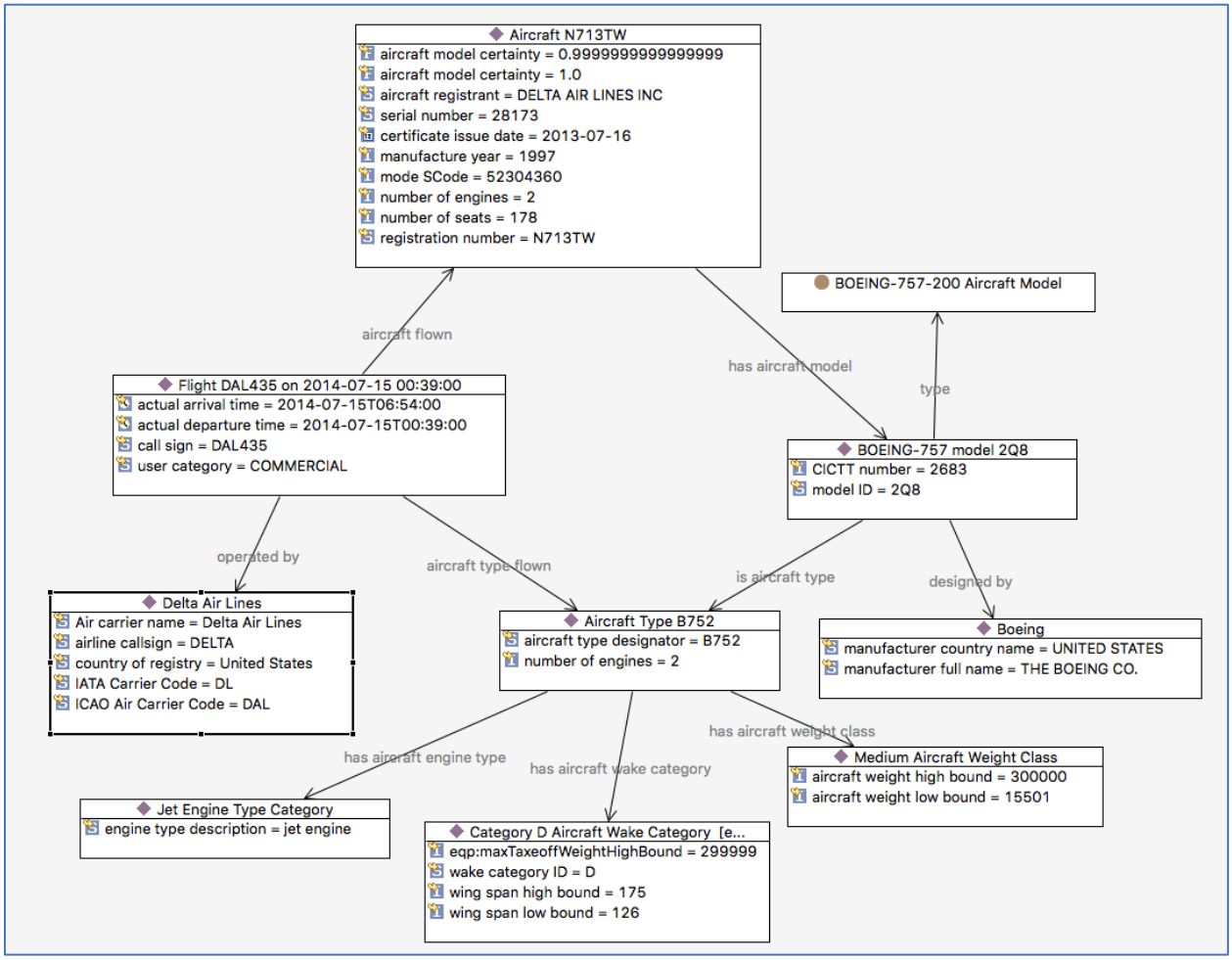


Figure 11: Relationships among instances of aircraft, carrier, flight, model, manufacturer, and other classes associated with Delta Airlines flight DAL435 on 2014-07-15. The aircraft flown for this flight is N713TW, a Boeing model 757-2Q8, one of the B757-200 family of aircraft. The aircraft family is represented as a model class and the the specific model is represented as an instance of that class. The FAA also designates an aircraft type, which may cover models in multiple aircraft families. The aircraft type for model 757-2Q8 is B752. Associated with type B752 aircraft are a set of instances that describe the engine type, wake turbulence category, and weight class of all B752 type aircraft.

6 Airport and Surface Operations

The classes in this section center on defining an airport and its physical infrastructure, including the structures involved in surface movement and operations.

6.1 nas:Airport

- **Description:** A facility where aircraft arrive and depart.
- **Superclasses**:
  - atm:NavigationElement
  - data:METARreportingStation
  - nas:NASfacility
- **Subclasses:**
  - nas:InternationalAirport
    - nas:CanadianAirport
  - nas:USairport
    - nas:CONUSairport: Airports in the continental US
    - nas:NonCONUSairport: Airports outside the continental US
- **Note**: The instances of this class were derived from two sources:
  - For domestic airports: FAA published data, including the ATC Tower and Satellite Airport Communications data file from the 56-Day NASR subscription available at the NFDC portal (https://nfdc.faa.gov)
  - For international airports: A list published by openflights.org at http://openflights.org/data.html#airport.
- **Object properties:**
  - data:hasAirportData [data:AirportData]: Links an airport to a collection of temporally-dependent data pertinent to the airport conditions, including a variety of data collected in the ASPM system.
  - data:hasTAFreport [data:TAFreport]: Associates an airport with TAF weather reports generated for that airport.
  - nas:airportBoundary [gen:Polygonal2DRegion]: Links an airport to a representation of its surface area, including its perimeter boundary.
  - nas:airportLocation [gen:PointLocation]: Links to a geographic point designated as the airport's location.
  - nas:hasRunway [nas:PhysicalRunway]: Links an airport to the physical runways for that airport.
  - nas:hasSID [nas:SID]: Associates an airport with the SIDs designated to route aircraft departing from that airport.
  - nas:hasSTAR [nas:STAR]: Associates an airport with the STARs designated to route aircraft arriving at that airport.
  - nas:hasTaxiway [nas:Taxiway]: Links an airport to the taxiways comprising the airport surface network
  - nas:hasTerminal [nas:Terminal]: Links an airport to the terminal(s) for that airport.
  - nas:hasTower [nas:ATCT]: Links an airport to an air traffic control tower(s) for that airport.
  - nas:withinARTCC [nas:ARTCC]: Links the airport to the ARTCC within which it is located.
- **Datatype properties**:
  - nas:airportName [string]: The official name of the airport.

- nas:faaAirportCode [string]: The alphanumeric FAA code of length 3 or 4 designated for the airport. As contrasted with the ICAO and IATA codes, which in general differ from the FAA code. Often, however, the IATA and FAA codes are identical.
- nas:hoursOffsetFromUTC [integer]: The number of hours that the airport is offset from the UTC timezone.
- nas:iataAirportCode [string]: The three-letter IATA code assigned to the airport. As contrasted with the ICAO and FAA airport codes.
- nas:icaoAirportCode [string]: The three-letter ICAO code assigned to the airport. As contrasted with the IATA and FAA airport codes.
- nas:isInstrumentControlEligible [boolean]: Indicates whether this airport is capable of an instrument approach.
- nas:isWeatherReportingStation [boolean]: Indicates whether this airport is a METAR weather reporting station.
- nas:locatedInState [string]: The 2-letter US postal code for the state in which the airport is located.
- nas:withinTimezone [string]: The time zone identifier for the airport, where the time zone is specified as Area/Location, e.g. 'America/New_York'.

6.1.1 nas:InternationalAirport

- **Superclasses:**
  - nas:Airport
- **Subclasses**:
  - nas:CanadianAirport

6.1.1.1 nas:CanadianAirport

- **Superclasses:**
  - nas:InternationalAirport

6.1.2 nas:USairport

- **Superclasses:**
  - nas:Airport
- **Subclasses:**
  - nas:CONUSairport
  - nas:NonCONUSairport

6.1.2.1 nas:CONUSairport

- **Superclasses:**
  - nas:USairport

6.1.2.2 nas:NonCONUSairport

- **Superclasses:**
  - nas:USairport

6.2 data:AirportData

- **Description**: Represents a collection of airport data for a given time period, as reported by the FAA's ASPM system (http://aspm.faa.gov).
- **Superclasses**:
  - data:IntervalData
- **Object properties:**
  - data:hasASPMmetCondition [data:ASPMmeteorologicalCondition]: Links meteorological information to the collection of temporally-dependent data associated with an airport.
- **Datatype properties**:
  - data:airportArrivalRate [integer]: The arrival rate per hour set by the airport during the specified interval. This is the total number of aircraft that can arrive on all runways combined during an hour.
  - data:airportDepartureRate [integer]: The departure rate per hour set by the airport during the specified interval. This is the total number of aircraft that can depart from all runways combined during an hour.
  - data:arrivalDemand [integer]: The number of aircraft intending to arrive at an airport during the specified time period.
  - data:aspmFlightRules [string: "I" , "V"]: Indicates what flight rule conditions the airport is operating under during the specified time period (I-instrument, V-Visual).
  - data:departureDemand [integer]: The number of aircraft intending to depart during the specified time period.
  - data:edctArrivalHold [float]: EDCT (Estimated Departure Clearance Time) hold minutes at other airports arriving this airport.
  - data:edctDepartureHold [float]: EDCT (Estimated Departure Clearance Time) hold minutes at other airports departing this airport.
  - data:etmsArrivals [integer]: Count of arrivals at airport based on ETMS (Enhanced Traffic Management System) data.
  - data:etmsDepartures [integer]: Count of departures at airport based on ETMS (Enhanced Traffic Management System) data.
  - data:highWindWITIdaily [float]: The high wind weather impacted traffic index (WITI) computed for the airport during the specified timeframe (daily). WITI is an estimate of the number of flights potentially impacted due to inclement weather based on the scheduled traffic demand for a defined geographic region of the air traffic system (in this case, an airport). If the winds at the airport are above a set threshold, then the high wind WITI is set to the number of scheduled arrivals for that timeframe (daily).
  - data:highWindWITIhourly [float]: The high wind weather impacted traffic index (WITI) computed for the airport during the specified timeframe (hourly). WITI is an estimate of the number of flights potentially impacted due to inclement weather based on the scheduled traffic demand for a defined geographic region of the air traffic system (in this case, an airport). If the winds at the airport are above a set threshold, then the high wind WITI is set to the number of scheduled arrivals for that timeframe (hourly).
  - data:lowCeilingWITIdaily [float]: The low ceiling weather impacted traffic index (WITI) computed for the airport during the specified timeframe (daily). WITI is an estimate of the number of flights potentially impacted due to inclement weather based on the scheduled traffic demand for a

defined geographic region of the air traffic system (in this case, an airport). If the ceiling at the airport is below a set threshold, then the low ceiling WITI is set to the number of scheduled arrivals for that timeframe (daily).

- data:lowCeilingWITIhourly [float]: The high wind weather impacted traffic index (WITI) computed for the airport during the specified timeframe (hourly). WITI is an estimate of the number of flights potentially impacted due to inclement weather based on the scheduled traffic demand for a defined geographic region of the air traffic system (in this case, an airport). If the ceiling at the airport is below a set threshold, then the low ceiling WITI is set to the number of scheduled arrivals for that timeframe (hourly).
- data:lowVisibilityWITIdaily [float]: The low visibility weather impacted traffic index (WITI) computed for the airport during the specified timeframe (daily). WITI is an estimate of the number of flights potentially impacted due to inclement weather based on the scheduled traffic demand for a defined geographic region of the air traffic system (in this case, an airport). If the visibility at the airport is below a set threshold, then the low visibility WITI is set to the number of scheduled arrivals for that timeframe (daily).
- data:lowVisibilityWITIhourly [float]: The low visibility weather impacted traffic index (WITI) computed for the airport during the specified timeframe (hourly). WITI is an estimate of the number of flights potentially impacted due to inclement weather based on the scheduled traffic demand for a defined geographic region of the air traffic system (in this case, an airport). If the visibility at the airport is below a set threshold, then the low visibility WITI is set to the number of scheduled arrivals for that timeframe (hourly).
- data:oagArrivalDelay [integer]: Minutes of OAG (Official Airline Guide) based arrival delay in excess of 15 minutes.
- data:oagGateDepartureDelay [integer]: Minutes of OAG (Official Airline Guide) based gate departure delay in excess of 15 minutes.
- data:scheduledArrivals [integer]: Count of scheduled arrivals. (Most probably from OAG (Official Airline Guide))
- data:scheduledDepartures [integer]: Count of scheduled departures. (Most probably from OAG (Official Airline Guide))
- data:totalAirborneDelay [float]: Total airborne flight delay in minutes for this airport during the specified period.

6.2.1 *data:WITIproperty*

- **Description**: A property class containing various instance properties relating to WITI (Weather-Impacted Traffic Index), including data:highWindWITIdaily, data:highWindWITIhourly, data:lowCeilingWITIdaily, data:lowCeilingWITIhourly, data:lowVisibilityWITIdaily, and data:lowVisibilityWITIhourly.
- **Superclasses**:
  - owl:DatatypeProperty

6.3 *nas:AirportInfrastructureComponent*

- **Description:** Part of an airport's physical infrastructure, including gates, terminals, runways, taxiways, etc.
- **Subclasses:**
  - nas:AirportServiceVehicle
  - nas:ATCT

- nas:DeicingPad
- nas:Gate
- nas:OperationalRunway
- nas:PhysicalRunway
- nas:RampTower
- nas:Taxiway
- nas:Terminal

- **Object properties**:
  - nas:associatedAirport [nas:Airport]: Links an airport infrastructure component to its associated airport.

6.4 nas:AirportServiceVehicle

- **Description:** A class of vehicles that service the flights and maintain the airport physical infrastructure.
- **Subclasses:** (Note: the following subclasses are only placeholders. No details have been modeled.)
  - nas:DeicingTruck
  - nas:RefuelingTruck

6.4.1 nas:DeicingTruck

- **Description**: A vehicle that holds deicing fluid and pumping equipment to support deicing.
- **Superclasses**:
  - nas:AirportServiceVehicle

6.4.2 nas:RefuelingTruck

- **Description**: Vechicle that transports and pumps jet fuel to refuel aircraft.
- **Superclasses**:
  - nas:AirportServiceVehicle

6.5 nas:ATCT

- **Description:** The Air Traffic Control Tower facility is responsible for managing arrivals, departures, and surface movement of aircraft on runways and taxiways.
- **Superclasses**:
  - nas:AirportInfrastructureComponent
  - nas:NASfacility
- **Datatype properties:**
  - nas:towerID [string]: The FAA identifier for air traffic control tower.

6.6 nas:RampTower

- **Description:** A control tower from which ramp controllers guide aircraft movements as they enter and exit the taxiways, and arrive and leave the gates. The ramp is the area of the airport surface where planes, service vehicles, and people meet. Note: This class is only a placeholder; no detail has been modelled.
- **Superclasses:**
  - nas:AirportInfrastructureComponent

6.7 nas:DeicingPad

- **Description:** A physical location in the airport where deicing is performed.
- **Superclasses:**
  - nas:AirportInfrastructureComponent
- **Object properties**:
  - nas:deicingPadLocation [gen:Polygonal2DRegion]: Links a deicing pad with its location as a two-dimensional region with a bounding polygon.
  - nas:hasQueue [nas:DeicingQueue]: Links a deicing pad to its queue of aircraft waiting to be deiced.

6.7.1 nas:DeicingQueue

- **Description**: An ordered sequence of aircraft awaiting deicing at a deicing pad.
- **Superclasses**:
  - gen:Sequence

6.8 nas:Gate

- **Description:** A physical interface between the aircraft and the terminal, used to board passengers and crew.
- **Superclasses:**
  - nas:AirportInfrastructureComponent
- **Subclasses:** The full set of gates for each airport is represented as a subclass, and this subclass, in turn, has several subclasses breaking the full set into subsets corresponding to the gates for each terminal. The actual instances of gates are associated with these terminal-related gate subclasses.
  For example:
  - nas:KDFWGate: The class of all gates at KDFW
    - nas:KDFWterminalAgate: The subclass of gates at Terminal A.
      - nas:KDFWtermAgateA10 (instance of gate at Terminal A)
      - nas:KDFWtermAgateA11 (instance of gate at Terminal A)
      - etc.
    - nasKDFWterminalBgate: The subclass of gates at Terminal B.
    - etc.

  Note: Subclasses and instances were created only for four airports: KDFW, KEWR, KJFK, and KLGA.

- **Datatype properties:**
  - nas:gateID [string]: The gate identifier assigned by the airport authority.

6.9 nas:PhysicalRunway

- **Description:** A physical runway, defined as a delimited rectangular surface region of the airport. Each physical runway is associated conceptually with two operational runways, 180 degrees apart, to represent the use of the runway taking off or landing in either direction.

- **Superclasses:**
  - nas:AirportInfrastructureComponent
- **Subclasses:** There is a subclass of nas:PhysicalRunway corresponding to each airport. Each physical runway at the designated airport is an instance of this class. Note: Subclasses and instances were created only for US and select Canadian airports.
- **Object properties**:
  - nas:associatedOpRunway [nas:OperationalRunway]: Associates a physical runway with its two operational runways (180 degrees apart, representing approaches from opposite ends of the physical runway).
  - nas:runwayFootprint [gen:Polygonal2DRegion]: Associates a physical runway with its bounding two-dimensional rectangle.
- **Datatype properties:**
  - nas:runwayID [string]: An identifier for the runway. The convention is to label runways by their heading, dropping the last digit. For operational runways, the runway identifier specifies only one heading corresponding to the direction of travel on the runway; for physical runways, the runway identifier specifies both headings (180 degrees apart). For example, at Liverpool airport, the operational runway labeled '09' (approximately magnetic heading 90 degrees) is distinct from the operational runway in the opposite direction of travel along the same physical runway, which is labeled '27' (approximately magnetic heading 270 degrees). The identifier for the physical runway is '09/27'. If an airport has parallel runways, these would then be marked Left, Center and Right, e.g. 09L, 09C, 09R.
  - nas:runwayLengthInFeet [float]: The length in feet of the physical runway.
  - nas:runwayWidthInFeet [float]: The width in feet of the physical runway.

6.9.1 data:RunwayStatusData

- **Description**: This class describes a temporally-dependent set of data about the operating status of a physical runway.
- **Superclasses**:
  - data:IntervalData
- **Object properties**:
  - data:runwayReportedOn [nas:PhysicalRunway]: Associates a runway status report with the physical runway being measured.
- **Datatype properties**:
  - data:runwayStatus [string: "open", "closed"]: An indicator of whether the runway is open or closed during the specified period.
  - data:runwaySurfaceFriction [float]: The surface friction (mu) value of the runway for the specified period. Runway surface friction is directly relevant to the braking action which will be available to an aircraft decelerating after touch down, or after a decision to reject a takeoff.

6.10 nas:OperationalRunway

- **Description:** An operational runway is a named runway used in airport operations. There are two operational runways corresponding to a single underlying physical runway; these operational runways are

180 degrees apart and are named based on the runway heading as determined by the direction of aircraft travel on the runway.

- **Superclasses:**
  - nas:AirportInfrastructureComponent
- **Subclasses:** There is a subclass of nas:OperationalRunway for each airport. Each operational runway at the designated airport is an instance of this class. Note: Subclasses and instances were created only for US and select Canadian airports.
- **Object properties:**
  - nas:touchdownPoint [gen:PointLocation]: Links to the geographical point location that is the designated touchdown spot on the operational runway.
- **Datatype properties:**
  - nas:runwayID [string]: An identifier for the runway. The convention is to label runways by their heading, dropping the last digit. For operational runways, the runway identifier specifies only one heading corresponding to the direction of travel on the runway; for physical runways, the runway identifier specifies both headings (180 degrees apart). For example, at Liverpool airport, the operational runway labeled '09' (approximately magnetic heading 90 degrees) is distinct from the operational runway in the opposite direction of travel along the same physical runway, which is labeled '27' (approximately magnetic heading 270 degrees). The identifier for the physical runway is '09/27'. If an airport has parallel runways, these would then be marked Left, Center and Right, e.g. 09L, 09C, 09R.

6.11 nas:Taxiway

- **Description:** An airport surface pathway that aircraft traverse in traveling between the ramp area (i.e., the area in the vicinity of the gates) and the runways.
- **Superclasses:**
  - nas:AirportInfrastructureComponent
  - gen:SequencedItem
- **Object properties:**
  - nas:taxiwayFootprint [gen:Polygonal2DRegion]: Associates a taxiway with its bounding two-dimensional rectangle.
- **Datatype properties:**
  - nas:taxiwayID [string]: The airport authority's official identifier for the taxiway.

6.11.1 atm:Taxipath

- **Description**: A sequence of taxiways followed by an aircraft taxiing en route to/from a runway.
- **Superclasses**:
  - gen:Sequence

6.12 nas:Terminal

- **Description:** A structure on the airport surface that serves as an interface between people, baggage, and aircraft.
- **Superclasses:**
  - nas:AirportInfrastructureComponent

- **Subclasses:** There is a subclass of nas:Terminal for each airport, and the terminal instances are distributed under those subclasses . Note: Subclasses and instances were created only for four airports: KDFW, KEWR, KJFK, and KLGA.
- **Object properties**:
    - nas:hasGate [nas:Gate]: Associates an airport terminal with a aircraft gate located at that terminal.
    - nas:hasRampTower [nas:RampTower]: Associates an airport terminal with a ramp tower that controlls the aircraft and vehicle traffic in the vicinity of that terminal.
- **Datatype properties:**
    - nas:terminalID [string]: The alphanumeric identifier of the airport terminal.

6.13 *Illustrative Figures*

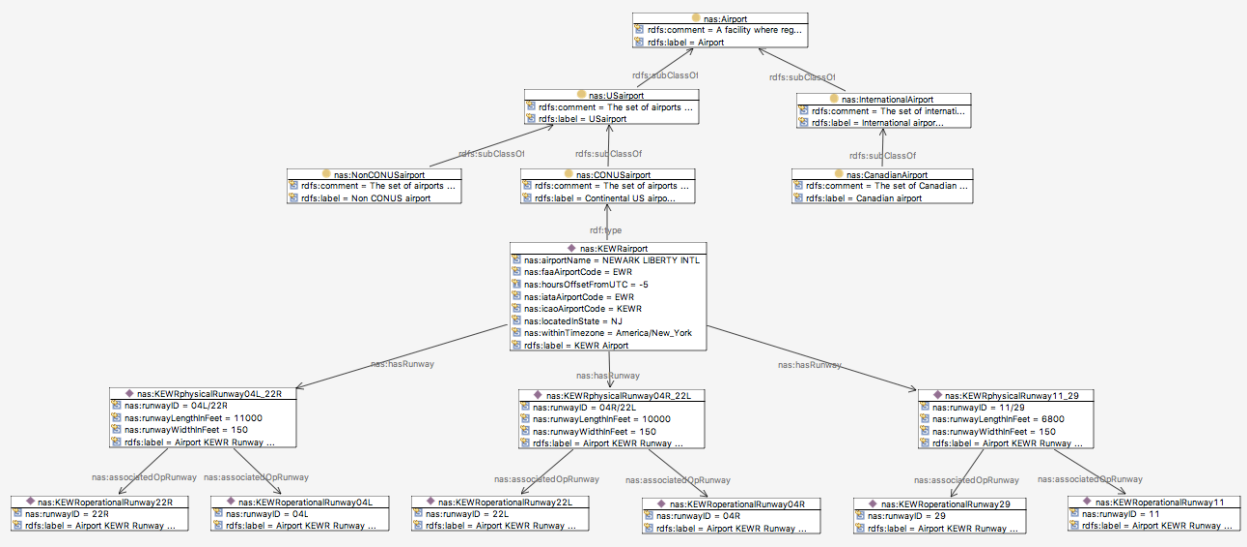


Figure 12: Airports and Runways. This figure illustrates a portion of the representation for runways at Newark airport (KEWR). In general, airports are divided into US and international airports. Due to the close proximity of Canadian airports to the US airspace in the Northeast, there is a special subclass of nas:InternationalAirport for Canadian airports. US airports are split into subclasses for those in the continental US and those outside. These distinctions are relevant due to different air traffic procedures in place when flying outside the continental US and between Canadian and US airports. Newark airport has three physical runways, each of which can be operated with planes landing or taking off in one direction or the other. Thus conceptually, each physical runway has two associated operational runways in the ontology. For example, the physical runway with runway identifier '04L/22R' is associated with two operational runways oriented 180° apart: 04L (40° heading) and 22R (220° heading). The operational directionality of the runway is important because there are different properties associated with either direction (the runway touchdown point and the runway visible range, for example).

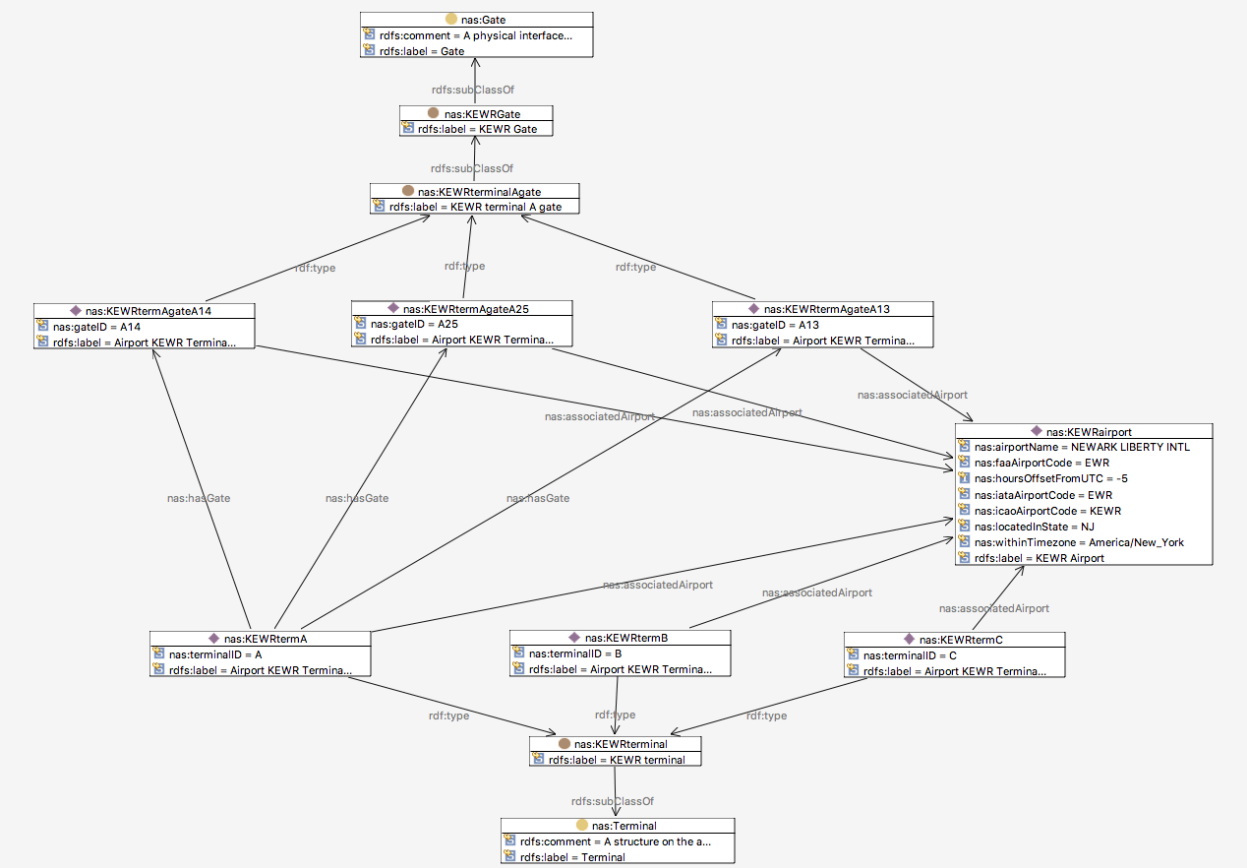


Figure 13: Terminals and Gates. This figure illustrates a portion of the ontology representation for terminals and gates at Newark airport (KEWR). (Note that to reduce clutter, only a small number of gates are depicted.) Terminals and gates are structured similarly. The class of terminals is partitioned into subclasses corresponding to airports. The instances of terminals at KEWR (Terminals A, B, and C) all belong to the subclass nas:KEWRterminal. The class of gates is partitioned into subclasses corresponding to terminal. The instances of gates at KEWR Terminal A (including A14, A25, and A13) all belong to the subclass nas:KEWRterminalAgate. In addition, the instances of terminals and their corresponding gates are linked by the property nas:hasGate.

7 Weather

The classes in this section pertain to the reporting of meteorological data. An attempt has been made to generalize over several different sources of surface weather data reported for airports (specifically, METAR, TAF, ASPM); no attempt has been made as yet to capture enroute, spatially-distributed meteorological conditions at high altitude (e.g., as reported in CWAM, CCFP data sources).

7.1 data:MeteorologicalCondition

- **Description:** A representation of the meteorological status for a specified time period, including sky, wind, visibility, and weather subcomponents. The class data:MeteorogicalCondition is the central organizing class for describing weather conditions, including present and projected/forecast conditions. The class data:MetCondition links to this class and its subclasses provides details of the sky, surface, weather phenomena, and visibility conditions. This basic class is used to uniformly describe meteorological conditions as reported by ASPM, METAR, and TAF. In TAF reports (data:TAFreport), forecasts are represented as sequences (gen:Sequence) of meteorological conditions, each with its own validity timeframe.
- **Superclasses:**
  - data:IntervalData
- **Subclasses:**
  - data:ASPMmeteorologicalCondition
  - data:METARreport
  - data:TAFmeteorologicalCondition
- **Object properties**:
  - hasSkyCondition [data:SkyCondition]: Links to sky conditions
  - hasSurfaceWindCondition [data:SurfaceWIndCondition]: Links to surface wind conditions
  - hasVisibilityCondition [data:VisibilityCondition]: Links to visibility conditions
  - hasWeatherCondition [data:WeatherCondition]: Links to weather conditions
- **Datatype properties:**
  - data:meteorologicalConditionStatus [string: “observed”, “forecast”]: indicates whether this meteorological condition represents an actual observation or a forecast
  - data:metConditionProbability [float]: a number between 0 and 1 representing the probability associated with a forecast meteorological condition
  - data:dewpoint [float]: The temperature in degrees Celsius to which a given parcel of air must be cooled at constant pressure and constant water-vapor content in order for saturation to occur.
  - data:seaLevelPressure [float]: Sea-Level pressure in hectopascals. Sea-level pressure is computed by adjusting the measuring station pressure to compensate for the difference between the station elevation and sea-level.
  - data:surfaceTemperature [float]: Surface temperature in degrees Celsius.

7.1.1 data:ASPMmeteorologicalCondition

- **Description**: Meteorological conditions reported as part of the ASPM hourly airport data. Includes sky, wind, visibility and weather components.
- **Superclasses**:
  - data:MeteorologicalCondition

7.1.2 data:METARreport

- **Description**: Curent meteorological conditions are reported in a periodic (nominally, hourly) METAR report. Includes sky, wind, visibility and weather components.
- **Superclasses**:
  - data:MeteorologicalCondition
- **Object properties**:
  - data:associatedMETARreportingStation [data:METARreportingStation]: Links a METAR report to the reporting station where the data were collected.
- **Datatype properties**:
  - data:metarReportModifier [string: "AUTO" , "COR"]: Automation status information pertaining to a METAR report: AUTO indicates a fully automated report with no human intervention; COR indicates a corrected observation; no modifier indicates either human observer/reporter or automated report with human oversight.
  - data:metarReportString [string]: Entire text of METAR report.
  - data:metarReportType [string: "METAR" , "SPECI"]: METAR report type indicator specifies whether this report is a regularly scheduled hourly report (METAR) or a special unscheduled report (SPECI). SPECIs are issued more frequently than hourly when adverse weather conditions prevail.
  - data:metarStationiHasPrecipitationSensor [boolean]: A boolean value indicating whether the reporting METAR station has a precipitation sensor.

7.1.2.1 data:METARreportingStation

- **Description:** A reporting station that provides sensor data for a METAR report. Many airports are also METAR reporting stations, but there are also non-airport reporting stations where operational weather monitoring hardware is installed.
- **Subclasses:**
  - nas:Airport
  - nas:StandAloneWeatherStation
- **Object properties**:
  - data:hasMETARreport [data:METARreport]: A link to the METAR reports for this reporting station

7.1.2.1.1 nas:StandAloneWeatherStation

- **Description**: A non-airport location where operational weather monitoring and reporting hardware and software is installed.
- **Superclasses**:
  - data:METARreportingStation
- **Note**: This class is a placeholder and is not modeled in any detail.

7.1.3 data:TAFmeteorologicalCondition

- **Description**: A forecast meteorological condition, reported as part of a TAF forecast. Includes sky, wind, visibility and weather components for current and future time periods.
- **Superclasses**:
  - data:MeteorologicalCondition

- **Datatype properties**:
  - data:rapidityOfWeatherChange [string: "rapid" , "gradual" , "transient"]: Indicates how fast change is expected from the prior meteorological conditions to this forecast condition.

7.1.3.1 data:TAFreport

- **Description:** A TAF (Terminal Aerodrome Forecast) report predicts meteorological conditions at successively later timepoints, starting at with the current timepoint (and the current actual conditions). TAF reports are modeled as a sequence (gen:Sequence) of forecast meteorological conditions over a specified time period.
- **Superclasses:**
  - gen:Sequence
  - data:IntervalData
- **Object properties:**
  - data:forecastingAirport [nas:Airport]: Links to the airport for which the TAF (Terminal Area Forecast) report was compiled.
- **Datatype properties**:
  - data:forecastIssueTime [datetime]: The time that the TAF (Terminal Area Forecast) report was issued.
  - data:tafReportString [string]: The full text of the entire TAF report.
  - data:tafReportType [string: "routine" , "amended" , "corrected" , "delayed"]: Type of Terminal Area Forecast report. An amended TAF is issued when the current TAF no longer adequately describes the ongoing weather or the forecaster feels the TAF is not representative of the current or expected weather.

7.2 data:MetCondition

- **Description:** A superclass over the a set of classes that define different aspects of a current or forecast meteorological condition (data:MeteorologicalCondition). In general, the properties and values modeled in the subclasses of data:MetCondition are derived from the international standard METAR definitions.
- **Subclasses:**
  - data:SkyCondition
  - data:SurfaceWindCondition
  - data:WeatherCondition
  - data:VisibilityCondition

7.2.1 data:SkyCondition

- **Description:** The sky condition is represented as a sequence of stacked, homogenous cloud layers, starting with the surface and moving up to the ceiling altitude, above which the sky is clear. This is done using a structure called a cloud layer profile, which is linked to data:SkyCondition.
- **Superclasses:**
  - data:MetCondition
- **Object properties:**
  - data:hasCloudLayerProfile [data:CloudLayerProfile]: The class data:CloudLayerProfile is a sequence composed of cloud layers (data:CloudLayer). Each cloud layer has distinct properties:

its base level and topmost altitudes; the type of cloud present ("towering cumulus", "cumulonimbus", "altocumulus castellanus"); and the type of cloud cover in the layer ("clear", "few clouds", "scattered clouds", "broken clouds", "overcast", "vertical visibility layer")

- **Datatype properties**:
  - data:ceiling [integer]: The vertical visibility (in feet) measured from the ground to the lowest cloud layer reported as broken or overcast.

7.2.1.1 data:CloudLayer

- **Description**: A cloud layer is a horizontal slice of the sky that exhibits uniform characteristics, including the amount of coverage and the type of cloud present.
- **Superclasses**:
  - gen:SequencedItem
- **Datatype properties**:
  - data:baseAltitude [integer]: the altitude in feet of the bottom edge of the layer
  - data:topAltitude [integer]: the altitude in feet of the top edge of the layer
  - data:cloudCover [string: "clear", "few clouds", "scattered clouds", "broken clouds", "overcast", "vertical visibility layer"]: the type of cloud cover present in the layer
  - data:cloudType [string: "towering cumulus", "cumulonimbus", "altocumulus castellanus"]: the type of cloud present in the layer

7.2.1.1.1 data:CloudLayerProfile

- **Description**: A sequence of cloud layers, ordered from the ground upward. Represents a vertical slice through the distinct layers of clouds starting at the surface.
- **Superclasses**:
  - gen:Sequence

7.2.2 data:SurfaceWindCondition

- **Description:** The surface winds are specified in terms of their speed – both steady (data:surfaceWindSpeed) and gusting (data:surfaceGustSpeed) – and direction.
- **Superclasses:**
  - data:MetCondition
- **Datatype properties:**
  - data:surfaceGustSpeed [float]: Reported maximum speed of wind when gusting, in knots
  - data:surfaceWindDirectionStatus [string: "fixed", "varyingWithinRange","variable"]: When data:surfaceWindDirectionStatus is “fixed”, the property data:windDirectionFixed stores the wind direction in degrees; when data:surfaceWindDirectionStatus is “varyingWithinRange”, the properties data:windDirectionLower and data:windDirectionUpper hold the wind direction interval; and when data:surfaceWindDirectionStatus is “variable”, there is no specification of the wind direction.
  - data:windDirectionFixed [float]:Stores wind direction in degrees when data:surfaceWindDirctionStatus is “fixed”
  - data:surfaceWindSpeed [float]: Reported constant speed of wind, in knots

- data:windDirectionLower [float]: Stores lower bound of wind variability interval in degrees, where the interval is defined clockwise between the lower and upper bound. Note that the 'lower' bound may be numerically greater than the 'upper' bound if the interval includes 0 degrees (North).
- data:windDirectionUpper [float]: Stores upper bound of wind variability interval in degrees, where the interval is defined clockwise between the lower and upper bound. Note that the 'lower' bound may be numerically less than the 'upper' bound if the interval includes 0 degrees (North).
- data:windShearHeight [float]: Wind shear height in feet above ground level

7.2.3 *data:WeatherCondition*

- **Description:** This class holds properties describing observed or forecast weather phenomena, along with their intensity, proximity, and other related characteristics. The properties and their values are derived from the international standard WMO METAR definitions.
- **Superclasses:**
  - data:MetCondition
- **Datatype properties:**
  - data:hourlyPrecipitation [float]: precipitation in inches, to the hundredth of an inch
  - data:weatherIntensity [string: "light" , "moderate" , "heavy"]
  - data:weatherPhenomenon [string: "drizzle" , "rain" , "snow" , "snow grains" , "ice crystals" , "ice pellets" , "hail" , "small hail and/or snow pellets" , "unknown precipitation" , "mist" , "fog" , "smoke" , "volcanic ash" , "widespread dust" , "sand" , "haze" , "spray" , "well-developed dust/sand whirls" , "squalls" , "funnel cloud" , "tornado/waterspout" , "sandstorm" , "duststorm"]
  - data:weatherProximity [string: "immediate proximity" , "in vicinity"]
  - data:weatherQualifier [string: "shallow" , "partial" , "patchy" , "low drifting" , "blowing" , "shower" , "thunderstorm" , "freezing"]

7.2.4 *data:VisibilityCondition*

- **Description:** This class describes visibility conditions at an airport in terms of two types of measurements: the prevailing visibility measured at an airport and the runway visible range measured along one or more airport runways. The prevailing visibility is a measurement of the greatest distance visible throughout at least half of the horizon, not necessarily continuously. That distance is either limited or unlimited. If the property data:unlimitedVisibility is either false or not specified, then the limit of visibility is to be specified in data:limitedVisibilityDistance; if data:unlimitedVisibility is true, then the distance limit is left unspecified. Any available measurements of runway visible range are linked using the property data:runwayVisibleRange.
- **Superclasses:**
  - data:MetCondition
- **Object properties:**
  - nas:runwayVisibleRange [nas:RunwayVisibleRangeMeasurement]: An indication of the range of distance past which the runway surface markings become unreadable for the pilot
- **Datatype properties**:

- data:limitedVisibilityDistance [float]: If prevailing visibility is limited, this property holds the surface visibility distance in statute miles
- data:unlimitedVisibility [boolean]: True if prevailing visibility is unlimited

7.2.4.1 nas: RunwayVisibleRangeMeasurement

- **Description:** A measurement of the distance over which a pilot of an aircraft on the centerline of a runway (linked via nas:runwayMeasured) can see the runway surface markings delineating the runway and its centerline. RVR (Runway Visible Range) is reported as either constant or variable. Variable prevailing visibility is reported if the prevailing visibility is less than 3 miles and rapidly increases or decreases by 1/2 statute mile or more during the time of observation. If the visibility is constant, then the nas:minVisibility and nas:maxVisibility properties will be equal; if the visibility is variable, then these properties specify the minimum and maximum number of feet visible.
- **Superclasses:**
  - data:IntervalData
- **Object properties:**
  - nas:runwayMeasured [nas:OperationalRunway]: The runway for which visibility is being measured.
- **Datatype properties**:
  - nas:maxVisibility [integer]: Maximum distance (in feet) at which surface markings become invisible
  - nas:minVisibility [integer]: Minimum distance (in feet) at which surface markings become invisible

7.3 Illustrative Figures

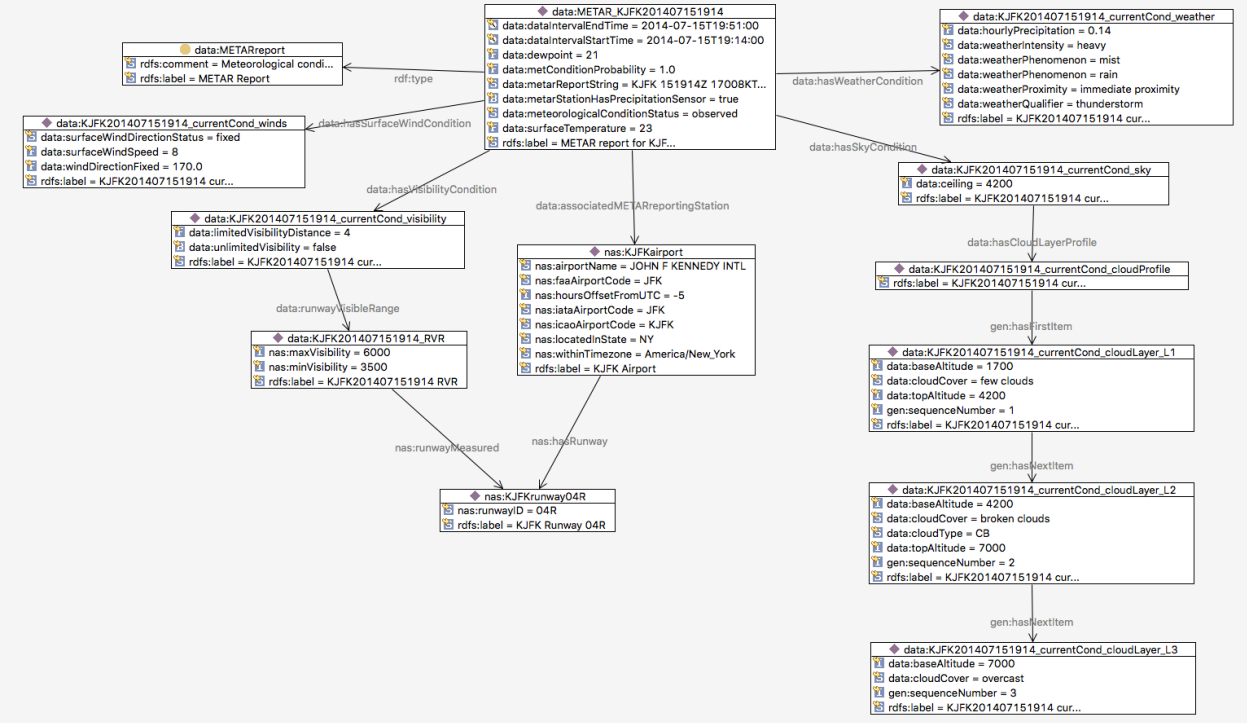


Figure 14: METAR Report. This figure illustrates the structure of a METAR report (data:METARreport) and related subclasses of data:MeteorologicalCondition, including TAF reports and ASPM weather indicators. The illustrated METAR report was issued on 7/15/2014 at 19:14 UTC based on data from sensors at KJFK. The raw METAR report consists of the following WMO standards-compliant METAR text string: 'KJFK 151914Z 17008KT 4SM R04R/3500VP6000FT +TSRA BR FEW017 BKN042CB OVC070 23/21 A2980 RMK AO2 FRQ LTGICCG NE-S TS NE-S MOV NE P0014'. Meteorological conditions were extracted from this string and re-encoded within the ontology structure shown in this figure. At the root of this structure is a node representing the overall METAR report instance (data:METAR_KJFK201407151914). There is a link from this node to the reporting station that generated the data for the report (in this case KJFK airport). The report includes data about four different meteorological components: weather, visibility, sky cloud cover, and winds. Each of these components is represented in a separate instance linked to the METAR report instance. The weather and wind components include only datatype properties, and have no linking substructure. The visibility condition includes a link to a runway visible range (RVR) measurement instance. That instance includes values for the maximum and minimum runway visibility and a link to the actual runway at KJFK being measured (Runway 04R). The sky condition component is linked to a structure called a cloud layer profile, which is a sequence of cloud layers (data:CloudLayer) ordered from the ground upward. The cloud layer profile represents a slice through the skies above the reporting station. Each sequenced cloud layer has a lower and upper altitude and represents a homogeneous cloud layer with a specified cloud cover type and density. See 7.2.1.1.

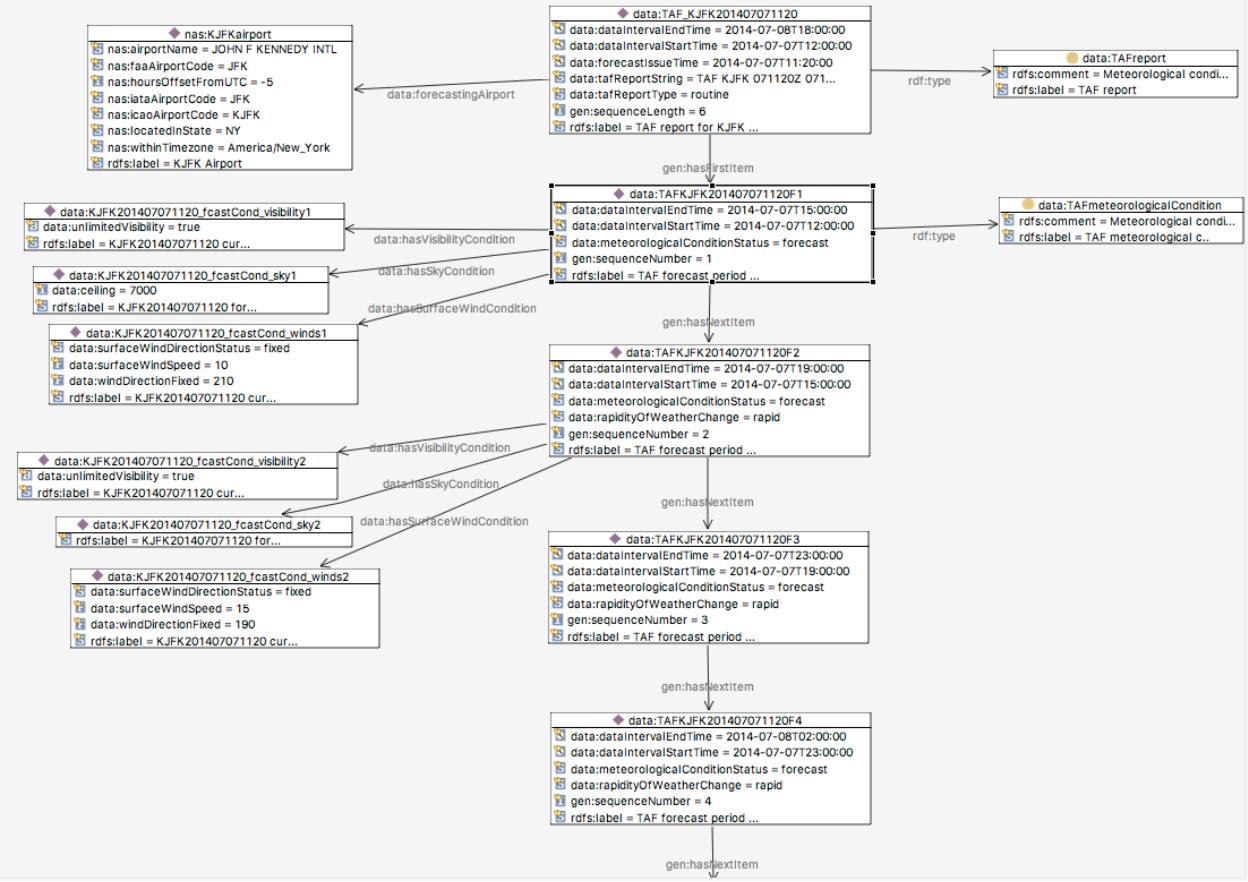


Figure 15: TAF Report. This figure illustrates the structure of a TAF report (data:TAFreport), which provides meteorological forecasts over a specified period of time – typically 24 hours. The graph structure in this figure illustrates a TAF report as specified in the following WMO standard TAF text string: 'TAF KJFK 071120Z 0712/0818 21010KT P6SM BKN070 FM071500 19015KT P6SM SCT060 FM071900 19018G25KT P6SM SCT060 BKN250 FM072300 19016KT P6SM SCT060 BKN250 FM080200 20010KT P6SM SCT060 BKN250 FM081300 22012KT P6SM SCT060 SCT250'. Meteorological conditions were extracted from this string and re-encoded within the ontology structure shown in this figure. The forecast period for this TAF is from 7/7/2014 at 12:00 UTC through 7/8/2014 at 18:00 UTC, and the report is a forecast for KJFK airport. The root node of the TAF report includes some properties related to the report's time of issue and includes the TAF text string, but the main function of this node is to point to the sequence of nodes that comprise the TAF forecast. The items in the TAF sequence each correspond to a different predicted TAF meteorological condition (data:TAFmeteorologicalCondition). Each condition covers a specified time range and provides the forecast sky, visibility, wind, and weather conditions. For example, the first forecast period (data:TAFKJFK201407071120F1) is linked to the root node via the property gen:hasFirstItem, and exhibits the same basic structure as the METAR report in Figure 14. But in addition, the first forecast period is linked to the next period, which has the same structure as the first period. (The sky, visibility, wind, and weather conditions are not displayed for each forecast period to reduce clutter.) In essence, a TAF report can be viewed structurally as a sequence of METAR reports. A key difference is that METAR reports actual observations at a specific time, whereas TAF reports a sequence of successive forecast conditions, each valid for the duration of a specified time period.

8 Sequences, Subsequenes, Sequenced Items

The classes in this section define the notion of a sequence, which is an important structuring concept used throughout the ontology. A sequence (gen:Sequence) represents an ordered, linked list of typed instances. For example, an aircraft trajectory is represented as a sequence of aircraft trackpoints, where each trackpoint records the GPS coordinates and altitude of the aircraft at successive timepoints throughout the flight. Another example of a sequence is a weather forcast, which is represented as a sequence of meteorological conditions, each projected at a specified timepoint in the future. Any class can be sequenced as long as it inherits the necessary properties from gen:SequencedItem. Note that the sequence is distinct from the items being sequenced. The sequence points to the first, last, and intervening items in the sequence; each of those items points to the next and previous item in the sequence.

8.1 gen:Sequence

- **Description:** An ordered sequence of instances. A sequence is modeled as a linked list of a certain length, containing instances of the class gen:SequencedItem. A sequence is linked to all items in its ordered list, with special links to the first item and the last item. Each item in the sequence is numbered sequentially and points to the next in the list. For a given subclass of gen:Sequence, the contents of the list are generally restricted to only one subclass of gen:SequencedItem. (This restriction is implemented as an OWL restriction of type owl:allValuesFrom.) In other words, the list may be constrained to contain only homogeneous types of instances. For example, a polygon boundary (gen:PolygonBoundary) is a sequence containing only point locations (gen:PointLocation).
- **Subclasses:**
    - atm:FlightSequence [sequence of atm:AircraftTrackPoint or atm:NavElementContainer]
        - **Description**: An end-to-end flight plan or actual flight path represented as a sequence of navigational elements followed from origin to destination airport.
        - **Subclasses**:
            - atm:ActualFlightRoute [sequence of atm:AircraftTrackPoint]: See details in Section 3.8.
            - atm:PlannedFlightRoute [sequence of atm:NavElementContainer]: See details in Section 3.4.
    - atm:NavigationPath [sequence of atm:NavElementContainer]
        - **Description**: An ordered sequence of navigation elements (fixes, routes, airports) representing a path through the airspace.
        - **Subclasses**:
            - atm:PlannedFlightRoute [sequence of atm:NavElementContainer]
            - nas:AirspaceRoute [sequence of atm:NavElementContainer]
    - atm:PopupFactorSequence [sequence of atm:PopupFactorContainer]
        - **Description:** An ordered sequence of hourly popup factors used in specifying a Ground Delay Program (GDP).
    - atm:ProgramArrivalRateSequence [sequence of atm:ProgramArrivalRateContainer]
        - **Description:** An ordered sequence of hourly program arrival rates used in specifying a Ground Dela Program (GDP).

    - atm:Taxipath [sequence of nas:Taxiway]
        - **Description:** A sequence of taxiways followed by an aircraft taxiing en route to/from a runway.
    - data:CloudLayerProfile [sequence of data:CloudLayer]
        - **Description:** A sequence of cloud layers, ordered from the ground upward. Represents cloud conditions.
    - data:TAFreport [sequence of data:TAFmeteorologicalCondition]
        - **Description**: Meteorological conditions provided in a periodic TAF forecast. Includes sky, wind, visibility and weather components. (See further details on data:TAFreport in Section 7.1.3.1.)
    - gen:PolygonBoundary [sequence of gen:PointLocation]
        - **Description:** An ordered sequence of point locations where the last point connects back to the first point, closing the polygon.
    - nas:DeicingQueue [sequence of eqp:Aircraft]
        - **Description:** An ordered sequence of aircraft awaiting deicing at a deicing pad.
- **Object properties**:
    - gen:hasFirstItem [gen:SequencedItem]: Links a sequence to the first item being sequenced.
    - gen:hasLastItem [gen:SequencedItem]: Links a sequence to the last item being sequenced.
    - gen:hasSequencedItem [gen:SequencedItem]: Links a sequence to an item being sequenced. The first and last items in the sequence should be linked using both this property and the distinguished properties above.
- **Datatype properties**:
    - gen:sequenceLength [integer]: The number of items in the ordered sequence.

8.1.1 gen:SubSequence

- **Description:** Represents a subsequence of an existing base sequence. The subsequence does not replicate the items of the sequence, but merely points to the start and end positions within the base sequence. This is used, for example, to specify that a flight is to follow a portion of an established airspace route as part of their flight plan.
- **Subclasses:**
    - *atm:NavigationSubPath*: A contiguous portion of an existing navigation path.
        - *atm:AirspaceRouteSegment:* A contiguous portion of an existing airspace route.
        - *atm:FlightPlanSegment*: A contiguous portion of a complete flight plan.
- **Object properties**:
    - gen:subsequenceOf [gen:Sequence]: Links a subsequence to its base sequence.
- **Datatype properties**:
    - gen:subsequenceStartIndex [integer]: A numeric position within a base sequence that represents the start of the subsequence.
    - gen:subsequenceEndIndex [integer]: A numeric position within a base sequence that represents the end of the subsequence.

8.2 gen:SequencedItem

- **Description:** The set of instances that can be sequenced using gen:Sequence. This class defines two properties that facilitate sequencing: a pointer to the next instance in the sequence, and a sequence number indicating the position within the sequence.
- **Subclasses:** Sequenced instances must be members of one of the following subclasses. (Note that these subclasses share nothing in common conceptually, except that they can be sequenced for some purpose.)
    - atm:AircraftTrackPoint
    - atm:NavElementContainer
    - atm:NumericParameterContainer
        - atm:PopupFactorContainer
        - atm:ProgramArrivalRateContainer
    - data:CloudLayer
    - data:TAFmeteorologicalCondition
    - eqp:Aircraft
    - gen:PointLocation
- **Object properties**:
    - gen:hasNextItem [gen:SequencedItem]: Links an instance in a sequence to the subsequent instance in the sequence.
- **Datatype properties**:
    - gen:sequenceNumber [integer]: Indicates the ordered position of this instance within the sequence, where 1 signifies the first position in the sequence.

8.3 Illustrative Figures

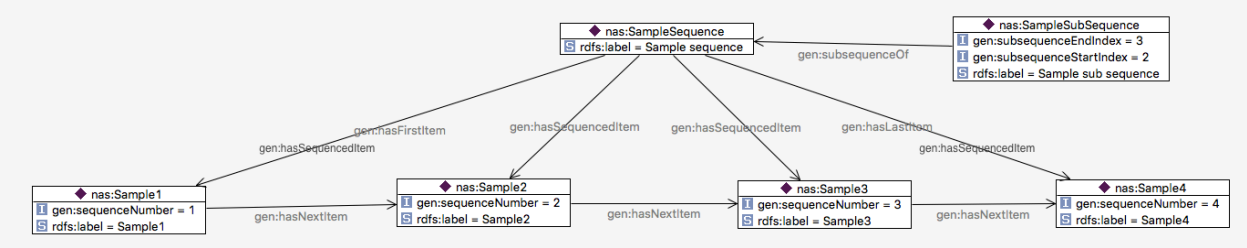


Figure 16: Sequences and Subsequences. In previous figures, there are many concrete examples of sequences. This figure illustrates an abstract instance of a sequence named nas:SampleSequence, which contains an ordered sequence of samples (nas:Sample1 through nas:Sample4). Each sample is linked to the sequence instance using the property gen:hasSequencedItem. In addition, there are special properties to identify the first and last items in the sequence (gen:hasFirstItem and gen:hasLastItem). Each item in the in the sequence is linked to its successor through the gen:hasNextItem property, and the position within the sequence is recorded in the datatype property gen:sequenceNumber. Finally, to represent the subsequence of elements in nas:SampleSequence that starts in position 2 and ends in position 3, we use a subsequence instance (nas:SampleSubSequence), which avoids duplicating the items in the sequence. Datatype properties in nas:SampleSubSequence hold the sequence number of the starting and ending subsequence positions and the property gen:subsequenceOf links to the main sequence.

9 Temporal/Spatial

The classes in this section describe general concepts pertaining to time and space. Very little here is specific to the ATM domain. (In fact, most of these concepts are so basic that they could have – and probably should have – been imported from other well-established external ontologies. This would be an area for future improvement.)

9.1 data:IntervalData

- **Description:** This class includes data that has a temporal lifetime. All subclasses of data:IntervalData inherit the basic properties that enable temporally-dependent data – the data's period of validity, as defined by a starting and an ending time.
- **Subclasses:**
  - atm:AircraftFlowCapacity
  - data:AirportData
  - data:MeteorologicalCondition
  - data:RunwayStatusData
  - data:TAFreport
  - nas:RunwayVisibleRangeMeasurement
- **Object properties**:
  - data:dataIntervalEndDay [nas:NASday]: Links temporally-dependent data to the ending day of its period of validity.
  - data:dataIntervalStartDay [nas:NASday]: Links temporally-dependent data to the starting day of its period of validity.
- **Datatype properties**:
  - data:dataIntervalEndTime [dateTime]: For temporally-dependent data, the ending time of the data's period of validity.
  - data:dataIntervalStartTime [dateTime]: For temporally-dependent data, the starting time of the data's period of validity.

9.2 gen:TimeInterval

- **Description:** A time interval with starting and ending time points. Intervals can be open (excluding the end points) or closed (including the end points) or mixed (including one but not the other end point).
- **Notes:** This is the proper way to do a time interval and this representation eventually should replace the less general representations currently used in various classes
- **Datatype properties:**
  - gen:closedEndTimeInterval [boolean]: A property that specifies whether or not the endpoint of the time interval is closed (inclusive of the point).
  - gen:closedStartTimeInterval [boolean]: A property that specifies whether or not the starting point of the time interval is closed (inclusive of the point).
  - gen:endTime [dateTime]: The ending time of a defined time interval.
  - gen:startTime [dateTime]: The starting time of a defined time interval.

9.3 nas:NASday

- **Description:** A class that explicitly represents a day of the year. Events, such as flights, weather forecasts, and other events are linked to the instance of the day they occurred.

- **Datatype properties:**
  - nas:calendarMonth [integer]: The month number associated with a nas:NASday.
  - nas:calendarYear [integer]: The year number associated with a nas:NASday.
  - nas:date [date]: The date represented by the nas:NASday, stored in date format.
  - nas:dayOfMonth [integer]: The day number associated with a nas:NASday.

9.4 nas:NAShour

- **Description:** A class that explicitly represents an hour within a day. Events, such as flights, weather forecasts, and other events are linked to the instance of the hour they were initiated.
- **Datatype properties:**
  - nas:startingTime [time]: The beginning of the hour represented by the nas:NAShour, stored as a datetime value.

9.5 gen:Location

- **Description:** A place defined by a point or a contiguous geographic region.
- **Subclasses:**
  - gen:GeographicRegion
    - gen:Region2D
      - gen:CircularRegion
      - gen:Polygonal2DRegion
    - gen:Region3D
      - gen:ShearSidedPolygonalVolume
  - gen:PointLocation
    - atm:AbsoluteFix
- **Datatype properties**:
  - gen:WKTgeoRepresentation [string]: The WKT string representation of a location. Well-Known Text (WKT) is an ISO/IEC standards-based a text markup language for representing vector geometry objects on a map, spatial reference systems of spatial objects, and transformations between spatial reference systems. WKT is stored as an alternative to the explicit polygon boundary representation employed in this Ontology (see gen:PolygonBoundary).

9.5.1 gen:GeographicRegion

- **Description:** A class representing a demarcated region on or above the surface of the Earth.
- **Superclasses:**
  - gen:Location

9.5.1.1 gen:Region2D

- **Description**: A two-dimensional geographic region.
- **Superclasses***:*
  - gen:GeographicRegion
- **Subclasses:**
  - gen:CircularRegion
  - gen:Polygonal2DRegion

9.5.1.1.1 gen:CircularRegion

- **Description:** A two-dimensional region defined by a geographic centerpoint and radius.
- **Object properties:**
  - gen: centerpoint [gen:PointLocation]: Links to the center point of the 2-D circular region.
- **Datatype properties:**
  - gen:radius [float]: The radius of a circular region. Note: Unfortunately, the units are currently dependent the type of circular region defined. This needs to be improved.

9.5.1.1.2 gen:Polygonal2DRegion

- **Description:** A two-dimensional region defined by a polygonal boundary.
- **Object properties:**
  - gen:hasPolygonBoundary [gen:PolygonBoundary]: Links a polygonal region to its boundary representation.

9.5.1.2 gen:Region3D

- **Description**: A three-dimensional geographic region.
- **Superclasses***:*
  - gen:GeographicRegion
- **Subclasses:**
  - gen:ShearSidedPolygonalVolume

9.5.1.2.1 gen:ShearSidedPolygonalVolume

- **Description:** A three-dimensional volume defined by a two-dimensional polygon plus a length component that stretches the polygon along an axis perpendicular to the polygon surface.
- **Object properties:**
  - gen:hasPolygonBoundary [gen:PolygonBoundary]: Links a polygonal region to its boundary representation.
- **Datatype properties**:
  - gen:polygonHeight [float]: Specifies the length dimension of a polygonal volume. Note: The length units are use-dependent.

9.5.2 gen:PointLocation

- **Description:** A location in three-space defined by a latitude, longitude, and altitude.
- **Superclasses:**
  - gen:Location
  - gen:SequencedItem
- **Subclasses**:
  - atm:AbsoluteFix
- **Datatype properties:**
  - gen:altitude [float]: The altitude of a point location in feet.
  - gen:latitude [float]: The latitude of a point location in decimal degrees.
  - gen:longitude [float]: The latitude of a point location in decimal degrees.

9.5.2.1 gen:PolygonBoundary

- **Description:** An ordered sequence (gen:Sequence) of individual point location instances (gen:PointLocation), where the last point connects back to the first point, closing the polygon.
- **Note**: This is an explicit, alternative representation to the WKT string representation stored in the gen:WKTgeoRepresentation property of the gen:Location class.
- **Superclasses:**
    - gen:Sequence

9.6 Illustrative Figures

Figure 17: Sector layer representation. This figure extends Figure 1 and illustrates the spatial representation for one of the sector layers composing Sector 075 in New York Center (ZNY ARTCC). Each sector consists of a set of horizontally-stacked polygonal layers with shear vertical sides, something like a layer cake. Each sector is represented with three components, a lower altitude, an upper altitude, and a layer geometry. The geometry provides the height of the layer (gen:polygonHeight) and a link (gen:hasPolygonBoundary) to a representation of the polygon boundary. The boundary is represented as an ordered sequence of point locations (gen:PointLocation), each of which specifies a latitude and longitude. Note that the property gen:WKTgeoRepresentation in gen:ZNYsector075Layer1Polygon contains an alternative representation of the boundary in the Well Known Text (WKT) format.

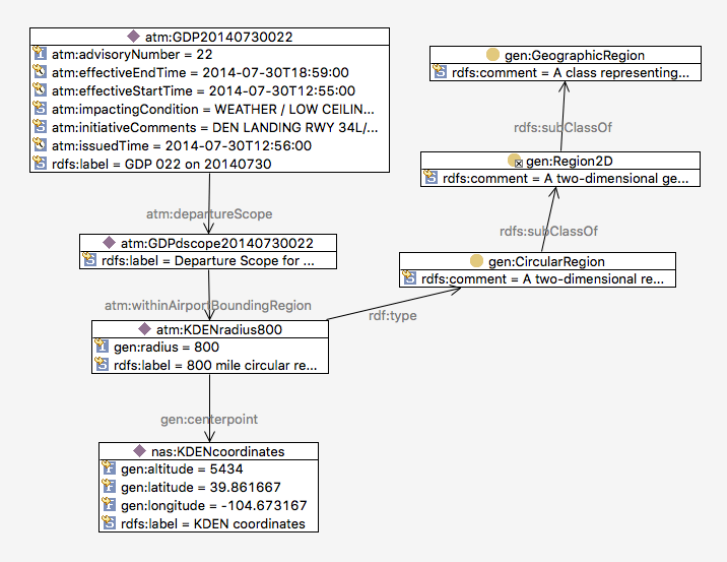


Figure 18: Circular airport radius. This figure illustrates the use of a circular geographic region to define the scope of a Ground Delay Program (GDP) TMI. The GDP's departure scope specifies that the TMI is applicable to any aircraft departing within an 800 mile radius around KDEN. The instance atm:KDENradius800 is a circular region with a centerpoint specified by nas:KDENcoordinates, which specifies the airport's geographic center.

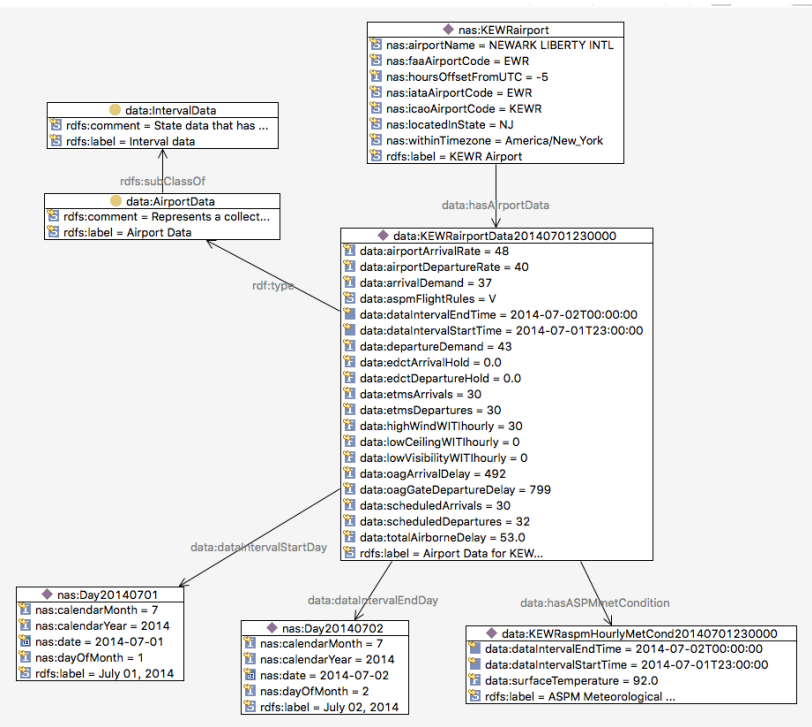


Figure 19: Temporal representation for airport data. In this figure, data:KEWRairportData20140701230000 represents an instance of data:AirportData. This instance stores data that are measured hourly at major airports in the US, such as KEWR. The instance inherits the properties of data:IntervalData, including the data's period of validity, as specified by a starting and ending validity time and links to the starting and ending days (nas:Day20140701 and nas:Day20140702).

10 Appendices

Appendix A References

[1] A. Doan and A. Y. Halevy, "Semantic integration research in the database community: A brief survey," AI magazine, vol. 26, p. 83, 2005.

[2] N. F. Noy, "Semantic integration: a survey of ontology-based approaches," ACM Sigmod Record, vol. 33, pp. 65-70, 2004.

[3] R. M. Keller, S. Ranjan, M. Y. Wei, and M. M. Eshow, "Semantic Representation and Scale-up of Integrated Air Traffic Management Data," in International Workshop on Semantic Big Data, San Francisco, California, 2016.

[4] R. M. Keller, "Ontologies for Aviation Data Management," in 2016 IEEE/AIAA 35th Digital Avionics Systems Conference (DASC), Sacramento, 2016.

[5] R. M. Keller, "Data Integration Using the NASA Air Traffic Management Ontology," presented at the Semantic Web for Air Transportation (SWAT-2015), Washington, DC, 2015. Available: https://www.faa.gov/nextgen/programs/swim/governance/servicesemantics/media/NASA ATM Ontology for SWAT revised.pdf

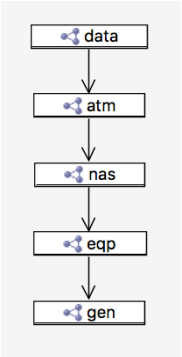
[6] R. M. Keller, "NASA's ATM Ontology," presented at the Air Transportation Information Exchange Conference, Silver Spring, MD, 2015. Available: http://www.aixm.aero/sites/aixm.aero/files/imce/library/ATIEC_2015/38_day3_nasas_atm_ontology_semantic_integration_and_querying_across_nas_data_sources.pdf

[7] W3C. (2014). OWL 2 Web Ontology Language Document Overview (Second Edition). Available: https://www.w3.org/TR/owl2-overview/

[8] W3C. (2014). RDF 1.1 Concepts and Abstract Syntax. Available: http://www.w3.org/TR/rdf11-concepts/

[9] W3C. (2014). RDF Schema 1.1. Available: http://www.w3.org/TR/rdf-schema/

Appendix B Ontology Namespaces



Classes defined in the ontology are separated into different XML namespaces to facilitate modularity and reuse, and to reduce complexity. The primary purpose for using namespaces in this ontology is to separate the large number of classes into a smaller clusters of thematically-interrelated classes. Although there is a high degree of interconnectedness and cross-linking among classes within the same namespace, classes also can make reference to classes outside their own namespace. These references are resolved by the use of *import statements*, the mechanism by which one namespace imports another.

In the ATM Ontology, the namespace import relationships are structured linearly, as depicted on the left. The *data* namespace imports all of the classes in the ontology because it directly imports classes in the *atm* namespace, which indirectly imports concepts from *nas*, *eqp*, and *gen*. These five namespaces are described in the following subsections.

Note that the partitioning of classes into namespaces is not always straightforward and unambiguous; rational arguments are possible for including a given class in a different namespace than the one herein assigned.

B.1 Namespace gen: Generic, domain-independent classes

- **Prefix**: gen
- **Namespace URI**: http://atmweb.arc.nasa.gov/ontology/general#
- **Description**: This namespace contains classes that are general-purpose, and not specific to the air traffic or aviation domain. These include temporal, spatial, and sequencing classes. This is the base namespace that is inherited (directly or indirectly) by all other namespaces.
- **Classes**:
  - gen:CircularRegion
  - gen:FloatParameter
  - gen:GeographicRegion
  - gen:IntegerParameter
  - gen:Location
  - gen:NumericParameter
  - gen:PointLocation
  - gen:Polygonal2DRegion
  - gen:PolygonBoundary
  - gen:Region2D
  - gen:Region3D
  - gen:Sequence
  - gen:SequencedItem
  - gen:ShearSidedPolygonalVolume
  - gen:SubSequence
  - gen:TimeInterval

B.2 Namespace eqp: Equipment-related classes

- **Prefix**: eqp
- **Namespace URI**: http://atmweb.arc.nasa.gov/ontology/eqp#

- **Imports**: gen
- **Description**: This namespace contains classes related to aircraft, aircraft models, aircraft type designators, aircraft weight and wake characteristics, and decomposable engineering systems, in general.
- **Classes**:
    - eqp:Aircraft
    - eqp:AircraftCommunicationSystem
    - eqp:AircraftEngine
    - eqp:AircraftModel
    - eqp:AircraftNavigationSystem
    - eqp:AircraftSubsystem
    - eqp:AircraftType
    - eqp:AircraftWakeCategory
    - eqp:AircraftWeightClass
    - eqp:BallBearing
    - eqp:DecomposableSystem
    - eqp:ElectricalPowerSystem
    - eqp:EngineeredSystem
    - eqp:EngineType
    - eqp:NavigationAid
    - eqp:UnitAssembly

B.3 Namespace nas: National Airspace System-related classes

- **Prefix**: nas
- **Namespace URI**: http://atmweb.arc.nasa.gov/ontology/nas#
- **Imports**: eqp
- **Description**: This namespace defines classes that define the overall National Airspace System (NAS) infrastructure, including NAS facilities (the FAA command center, Airports, ARTCCs/Centers, TRACONs), airport surface structures (control towers, runways/taxiways, terminals, gates), airspace structures (routes, sectors, approach and departure procedures), and aviation service providers (air carriers, manufacturers, air navigation service providers),
- **Classes**:
    - nas:AirCarrier
    - nas:AircraftEngineManufacturer
    - nas:AirframeManufacturer
    - nas:Airport
    - nas:AirportInfrastructureComponent
    - nas:AirportRoute
    - nas:AirportServiceVehicle
    - nas:AirspaceInfrastructureComponent
    - nas:AirspaceLayer
    - nas:AirspaceRoute
    - nas:ARTCC
    - nas:ARTCCtier
    - nas:ATCSCC

- nas:ATCT
- nas:AviationIndustryManufacturer
- nas:AviationServiceProvider
- nas:CanadianAirport
- nas:CommonRoute
- nas:CONUSairport
- nas:DeicingPad
- nas:DeicingQueue
- nas:DeicingTruck
- nas:FederalAirway
- nas:Gate
- nas:GovernmentAviationServiceProvider
- nas:InternationalAirport
- nas:JetRoute
- nas:NASday
- nas:NASfacility
- nas:NAShour
- nas:NonCONUSairport
- nas:OperationalRunway
- nas:PhysicalRunway
- nas:QRoute
- nas:RadialRoute
- nas:RampTower
- nas:RefuelingTruck
- nas:RNAVroute
- nas:RunwayVisibleRangeMeasurement
- nas:Sector
- nas:SID
- nas:SIDSTAR
- nas:SIDSTARroute
- nas:StandAloneWeatherStation
- nas:STAR
- nas:Taxiway
- nas:Terminal
- nas:TRACON
- nas:TransitionRoute
- nas:TRoute
- nas:USairport
- nas:VictorRoute
- nas:VORroute

B.4 Namespace atm: Air Traffic Management-related classes

- **Prefix**: atm
- **Namespace URI**: http://atmweb.arc.nasa.gov/ontology/atm#

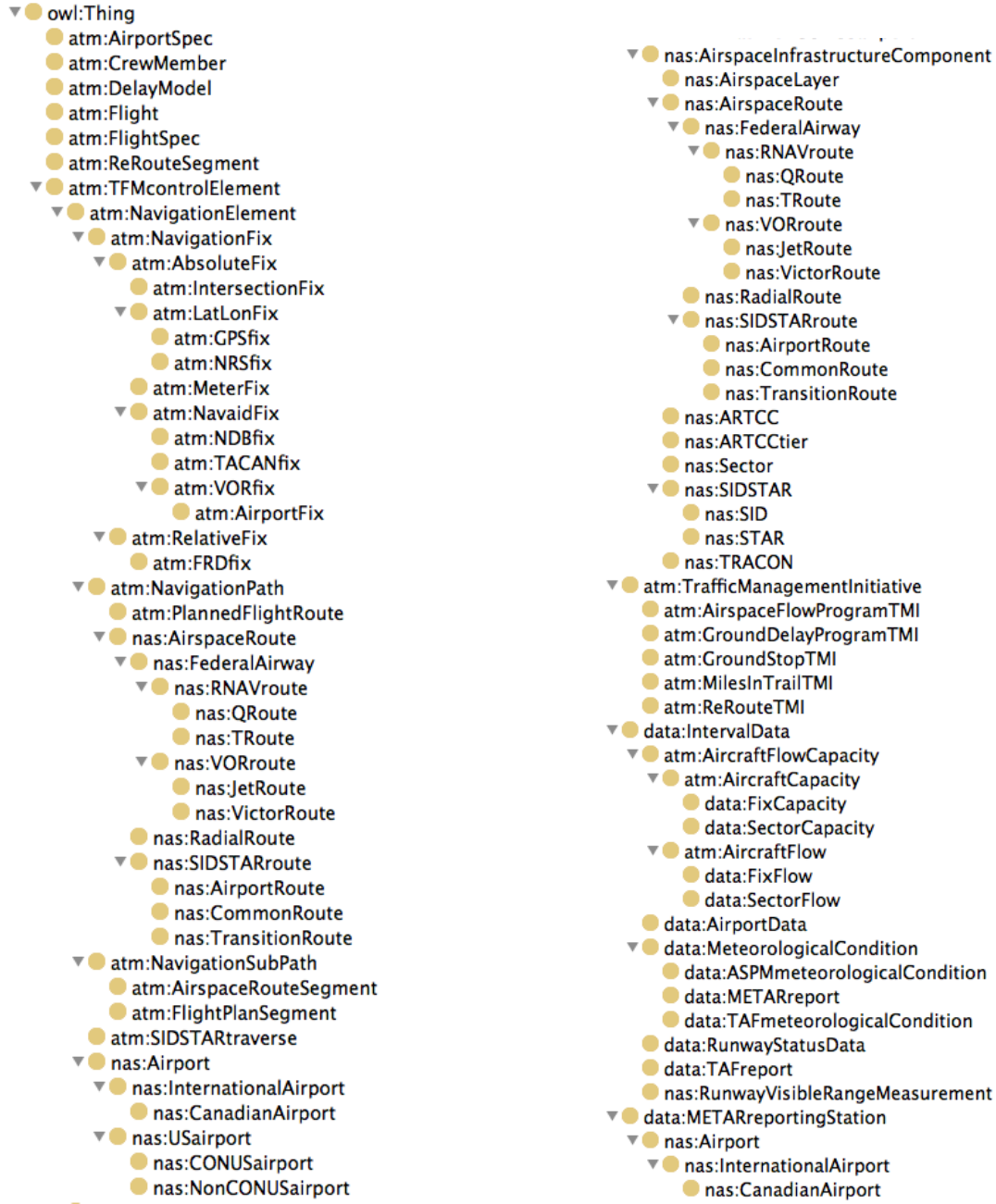
- **Imports**: nas
- **Description**: This namespace defines classes that specifically relate to the control of air traffic through the NAS, including Traffic Management Initiatives (TMIs) and air navigation structures (fixes, paths/routes, flight plans).
- **Note**: The borderline between the classes in this namespace and the *nas* namespace (which is imported by *atm*) is not clearcut; reasonable arguments can be made about which namespace is more appropriate for a given class.
- **Classes**:
    - atm:AbsoluteFix
    - atm:ActualFlightRoute
    - atm:AircraftCapacity
    - atm:AircraftFlow
    - atm:AircraftFlowCapacity
    - atm:AircraftTrackPoint
    - atm:AirportFix
    - atm:AirportSpec
    - atm:AirspaceFlowProgramTMI
    - atm:AirspaceRouteSegment
    - atm:CrewMember
    - atm:DelayModel
    - atm:Flight
    - atm:FlightPlanSegment
    - atm:FlightSpec
    - atm:FRDfix
    - atm:GPSfix
    - atm:GroundDelayProgramTMI
    - atm:GroundStopTMI
    - atm:IntersectionFix
    - atm:LatLonFix
    - atm:MeterFix
    - atm:MilesInTrailTMI
    - atm:NavaidFix
    - atm:NavElementContainer
    - atm:NavigationElement
    - atm:NavigationFix
    - atm:NavigationPath
    - atm:NavigationSubPath
    - atm:NDBfix
    - atm:NRSfix
    - atm:NumericParameterContainer
    - atm:PlannedFlightRoute
    - atm:PopupFactor
    - atm:PopupFactorContainer
    - atm:PopupFactorSequence

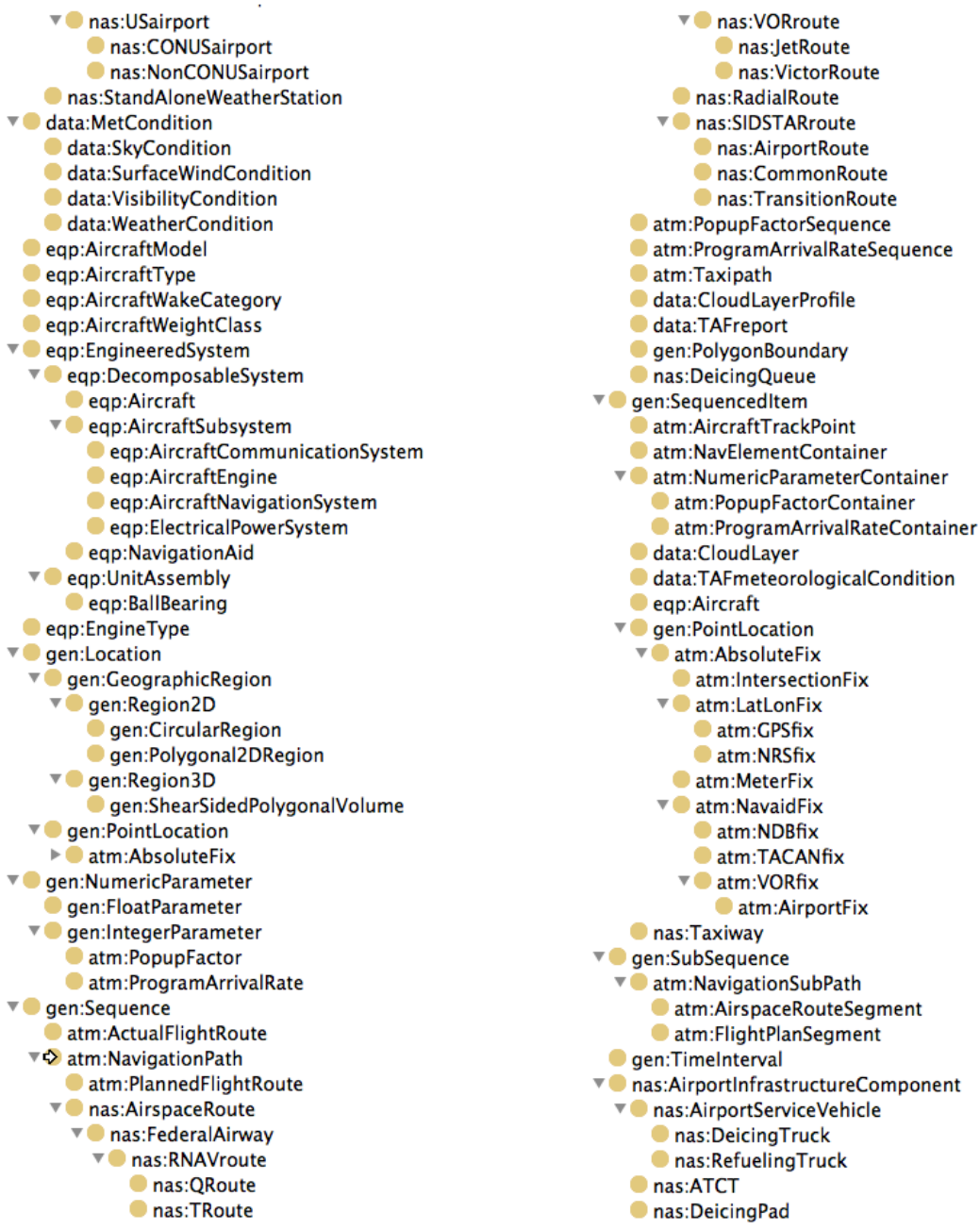
- atm:ProgramArrivalRate
- atm:ProgramArrivalRateContainer
- atm:ProgramArrivalRateSequence
- atm:RelativeFix
- atm:ReRouteSegment
- atm:ReRouteTMI
- atm:SIDSTARtraverse
- atm:TACANfix
- atm:Taxipath
- atm:TFMcontrolElement
- atm:TrafficManagementInitiative
- atm:VORfix

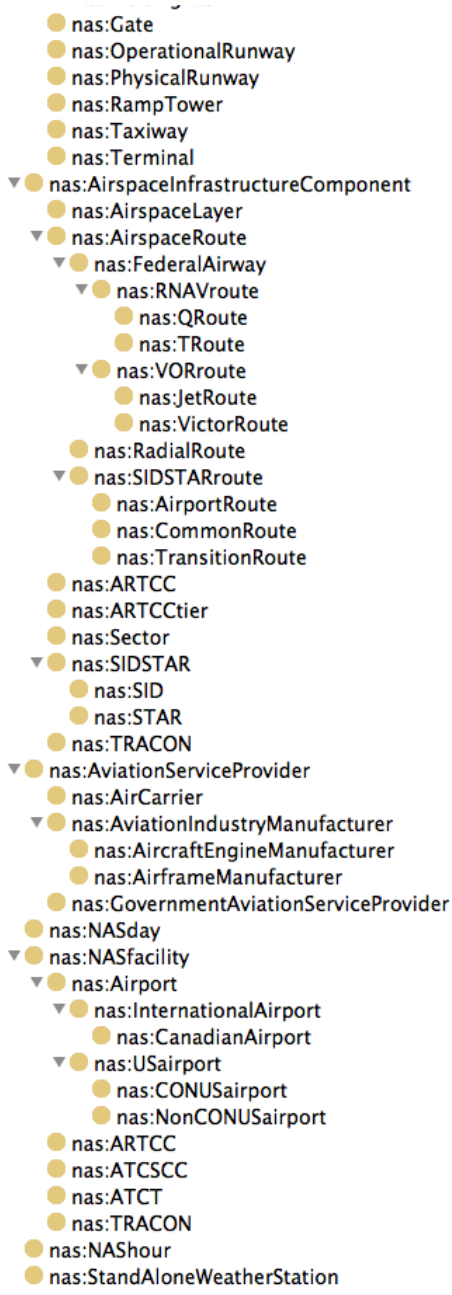
B.5 Namespace data: Data-specific classes

- **Prefix**: data
- **Namespace URI**: http://atmweb.arc.nasa.gov/ontology/data#
- **Imports**: atm
- **Description**: This namespace includes classes required to define data-specific concepts related to meteorological data (including sky, wind, visibility, and weather conditions), as well as runway status and airport arrival, departure, flow, and capacity statistics. These classes cover data associated reported from METAR, TAF, and ASPM sources.
- **Classes**:
  - data:AirportData
  - data:ASPMmeteorologicalCondition
  - data:CloudLayer
  - data:CloudLayerProfile
  - data:FixCapacity
  - data:FixFlow
  - data:IntervalData
  - data:METARreport
  - data:METARreportingStation
  - data:MetCondition
  - data:MeteorologicalCondition
  - data:RunwayStatusData
  - data:SectorCapacity
  - data:SectorFlow
  - data:SkyCondition
  - data:SurfaceWindCondition
  - data:TAFmeteorologicalCondition
  - data:TAFreport
  - data:VisibilityCondition
  - data:WeatherCondition
  - data:WITIproperty

Appendix C Subsumption Hierarchy




nas:USairport
nas:CONUSairport
nas:NonCONUSairport
nas:StandAloneWeatherStation
data:MetCondition
data:SkyCondition
data:SurfaceWindCondition
data:VisibilityCondition
data:WeatherCondition
eqp:AircraftModel
eqp:AircraftType
eqp:AircraftWakeCategory
eqp:AircraftWeightClass
eqp:EngineeredSystem
eqp:DecomposableSystem
eqp:Aircraft
eqp:AircraftSubsystem
eqp:AircraftCommunicationSystem
eqp:AircraftEngine
eqp:AircraftNavigationSystem
eqp:ElectricalPowerSystem
eqp:NavigationAid
eqp:UnitAssembly
eqp:BallBearing
eqp:EngineType
gen:Location
gen:GeographicRegion
gen:Region2D
gen:CircularRegion
gen:Polygonal2DRegion
gen:Region3D
gen:ShearSidedPolygonalVolume
gen:PointLocation
atm:AbsoluteFix
gen:NumericParameter
gen:FloatParameter
gen:IntegerParameter
atm:PopupFactor
atm:ProgramArrivalRate
gen:Sequence
atm:ActualFlightRoute
atm:NavigationPath
atm:PlannedFlightRoute
nas:AirspaceRoute
nas:FederalAirway
nas:RNAVroute
nas:QRoute
nas:TRoute
nas:VORroute
nas:JetRoute
nas:VictorRoute
nas:RadialRoute
nas:SIDSTARroute
nas:AirportRoute
nas:CommonRoute
nas:TransitionRoute
atm:PopupFactorSequence
atm:ProgramArrivalRateSequence
atm:Taxipath
data:CloudLayerProfile
data:TAFreport
gen:PolygonBoundary
nas:DeicingQueue
gen:SequencedItem
atm:AircraftTrackPoint
atm:NavElementContainer
atm:NumericParameterContainer
atm:PopupFactorContainer
atm:ProgramArrivalRateContainer
data:CloudLayer
data:TAFmeteorologicalCondition
eqp:Aircraft
gen:PointLocation
atm:AbsoluteFix
atm:IntersectionFix
atm:LatLonFix
atm:GPSfix
atm:NRSfix
atm:MeterFix
atm:NavaidFix
atm:NDBfix
atm:TACANfix
atm:VORfix
atm:AirportFix
nas:Taxiway
gen:SubSequence
atm:NavigationSubPath
atm:AirspaceRouteSegment
atm:FlightPlanSegment
gen:TimeInterval
nas:AirportInfrastructureComponent
nas:AirportServiceVehicle
nas:DeicingTruck
nas:RefuelingTruck
nas:ATCT
nas:DeicingPad


nas:Gate
nas:OperationalRunway
nas:PhysicalRunway
nas:RampTower
nas:Taxiway
nas:Terminal
nas:AirspaceInfrastructureComponent
nas:AirspaceLayer
nas:AirspaceRoute
nas:FederalAirway
nas:RNAVroute
nas:QRoute
nas:TRoute
nas:VORroute
nas:JetRoute
nas:VictorRoute
nas:RadialRoute
nas:SIDSTARroute
nas:AirportRoute
nas:CommonRoute
nas:TransitionRoute
nas:ARTCC
nas:ARTCCtier
nas:Sector
nas:SIDSTAR
nas:SID
nas:STAR
nas:TRACON
nas:AviationServiceProvider
nas:AirCarrier
nas:AviationIndustryManufacturer
nas:AircraftEngineManufacturer
nas:AirframeManufacturer
nas:GovernmentAviationServiceProvider
nas:NASday
nas:NASfacility
nas:Airport
nas:InternationalAirport
nas:CanadianAirport
nas:USairport
nas:CONUSairport
nas:NonCONUSairport
nas:ARTCC
nas:ATCSCC
nas:ATCT
nas:TRACON
nas:NAShour
nas:StandAloneWeatherStation

Appendix D Notes on the Ontology

- **Ontology scoping**: The scoping and focus of this ontology was primarily application driven. We designed datatype properties (and corresponding classes) that would enable us to store all the types of data values that were captured by the different aviation data sources we sought to integrate. While we attempted to make the classes and properties as general and reusable as possible, achieving generality while satisfying specific needs and application requirements can be a difficult proposition. In cases where no immediate requirements were present to define a certain class, but the class is obviously warranted, the modeling is thin, and serves only as a placeholder.
- **Improvement areas:**
    - Representation of time intervals: Currently, there is an mixture of representations used within the Ontology for time intervals. For example, the actual operating time for a flight is currently represented by two separate properties atm:actualDepartureTime and atm:actualArrivalTime. This might be represented alternatively with one property that links to an instance of gen:TimeInterval. The situation is similar for atm:TrafficManagementInitiative, which uses atm:effectiveStartTime and atm:effectiveEndTime to mark the interval that a TMI is to be enforced. The class gen:TimeInterval allows for more flexibility concerning whether the interval endpoints are open or closed, and is a more appropriate, more fine-grained representation to use in some situations. For instance, the class nas:NAShour is defined to have a start time only. Implicitly, the end time is the end of the hour. A more correct representation would use a time interval with a closed starting timepoint (at the beginning of the hour) and an open ending timepoint (at the beginning of the next hour).
    - Treatment of units: In general, measurement units are implicit in this Ontology. Sometimes the measurement units are embedded in the property names (e.g., nas:runwayLengthInFeet); other times, the units are defined in this document (e.g., gen:altitude [float]: The altitude of a point location in feet); other times the units are unstated or defined by convention (e.g., data:hourlyPrecipitation [float], which is in millimeters by WMO standards). This treatment is unsatisfactory and should be remedied. Note, however, that the processing and storage burden imposed by explicitly associating unit information with each measurement would likely be unacceptable for very large datasets.
    - Lack of OWL constructs: The ATM Ontology uses few features from OWL and the classes can mostly be expressed using RDFS. However, the ontology does include the use of some OWL restrictions. See, for example the description of sequences (gen:Sequence). In general, additional OWL constructs could be incorporated into the Ontology to further constrain semantics and to enable inferences to be drawn. However, poor inference engine performance on large datasets limit the effectiveness of inference in the context of NASA data. This is an area for further cost/benefit analysis and more testing and evaluation.
    - Lack of concept reuse from established external ontologies: The ATM Ontology was developed without referencing classes imported from externally-developed ontologies. Although this runs counter to the spirit of ontology reuse, externally-defined classes were not typically available for the aviation-specific concepts required in this Ontology or were defined in a manner that did not meet NASA's application requirements. No upper ontology was used as a scaffolding for building this ontology; our assumption – correct or misinformed – was that an upper ontology would add complexity but provide little value-added content or structure to the ontology. This assumption must be critically reexamined.

Appendix E Organization of Ontology Files

To facilitate reuse, ontology files are divided into base model files and instance files. Typically, the base model files contain only classes and subclasses, with no instances; instance files contain instances only. Occasionally, however, instance files will contain classes and subclasses, especially when those classes have been programmatically generated and may not be of general interest. These cases are noted below.

Base model files are organized by namespace (see Appendix B). The classes associated with a given namespace are stored in a single OWL file in Turtle format (.ttl), with the following filenames:

- ATM.ttl: All classes defined for namespace prefix *atm.*
- data.ttl: All classes defined for namespace prefix *data.*
- equipment.ttl: All classes defined for namespace prefix *eqp.*
- general.ttl: All classes defined for namespace prefix *gen.*
- NAS.ttl: All classes defined for namespace prefix *nas.*

Class instances are stored in a separate set of files. Each instance file imports the base model files in which the corresponding classes are defined. The instance files are:

- acInst.ttl: Contains aircraft instances associated with processed flights. (eqp:Aircraft)
- acManufInst.ttl: Contains aircraft manufacturer instances. (nas:AirframeManufacturer)
- acModelInst.ttl: Contains instances of aircraft type designators (eqp:AircraftType), as well as subclasses of eqp:AircraftModel *and* instances of those subclasses. Both aircraft model and type are derived from the CAST/ICAO Aircraft Taxonomy. See Sections 5.4.3.1 and 5.4.3.2. Note: This is one of only a few 'instance files' that contains subclasses as well as instances. Typically all classes and subclasses are defined in a base model file (see above).
- airlineInst.ttl: Contains instances of airline carriers (nas:AirCarrier).
- airportInst.ttl: Contains instances of nas:Airport, nas:PhysicalRunway, nas:OperationalRunway, nas:Gate, and gen:PointLocation, which provides the airport coordinates. In addition, this file contains subclasses of nas:Gate, nas:PhysicalRunway, and nas:OperationalRunway that are airport-specific. See subclasses discussion in Section 6.8 6.9 6.10. Note: This is one of only a few 'instance files' that contains subclasses as well as instances. Typically all classes and subclasses are defined in a base model file (see above).
- allInst.ttl: This file can be used to load the entire ontology, because it imports all instance files.
- ARTCCLocationInst.ttl: Contains instances of nas:ARTCC, plus instances of gen:Polygonal2DRegion, representing the boundaries of each ARTCC (including gen:PointLocation instances specifying the points that define the polygon boundary).
- ASPMinst.ttl: Contains airport weather instances generated from the ASPM database, including instances of data:ASPMmeteorologicalCondition and associated instances of various sky, wind, visibility, and weather conditions.
- dayInst.ttl: Contains instances of all days from 2012-2017.
- equipmentInst.ttl: Contains instances of eqp:EngineType, eqp:AircraftWakeCategory, and eqp:AircraftWeightClass.
- fixInst.ttl: Contains instances corresponding to all named fixes (all subclasses of atm:NavigationFix). These fixes were generated from FAA's ERAM data, which includes a comprehensive listing of navigation fixes.
- flightInst.ttl: Contains instances of 100 flights during July 2014 either arriving or departing New York area airports JFK, EWR, and LGA. Includes instances of actual flights and planned flight paths, plus associated

flight plans (including the planned routes, fixes, and SID/STAR traverses) and actual track points (including the associated fixes and sectors through which the flights passed).

- METARinst.ttl: Contains instances of all METAR reports (data:METARreport) issued at JFK, EWR, and LGA during July 2014, plus all of the associated instances (data:SkyCondition, data:SurfaceWindCondition, data:VisibilityCondition, data:WeatherCondition, data:CloudLayerProfile, data:CloudLayer, nas:RunwayVisibleRangeMeasurement).
- NASinst.ttl: Contains an instance representing the FAA command center (ATCSCC) plus ARTCC instances covering the entire NAS and the set of NAShour instances representing each hour in a day.
- routeInst.ttl: Contains instances of defined FAA airspace routes (nas:AirspaceRoute and its various subclasses) plus the corresponding instances of atm:NavElementContainer that sequence the navigational elements comprising each route.
- SectorLocationInst.ttl: Contains instances of nas:Sector, plus instances of gen:ShearSidedPolygonalVolume, representing the volume of each airspace layer that comprises the sectors (including gen:PointLocation instances specifying the points that define the polygon boundary for each layer). See 2.3.
- sidStarInst.ttl: Contains instances of SIDs and STARs, along with instances of the various routes through them (atm:SIDSTARroute) plus the corresponding instances of atm:NavElementContainer that sequence the navigational elements comprising each route.
- TAFinst.ttl: Contains instances of all TAF weather forcasts (data:TAFreport) issued at JFK, EWR, and LGA during July 2014, plus all of their associated instances (data:TAFmeteorologicalCondition, data:SkyCondition, data:SurfaceWindCondition, data:VisibilityCondition, data:WeatherCondition, data:CloudLayerProfile, data:CloudLayer).
- TMIinst.ttl: Contains instances of all Ground Delay, Ground Stop, and ReRoute traffic management initiatives issued during July 2014. Included are auxiliary instances describing flight and airport constraints (atm:FlightSpec, atm:AirportSpec). Also included are the route segments permitted for each ReRoute TMI and all the associated instances necessary for defining those route segments. See Section 4.
- TRACONinst.ttl: Contains instances of all TRACONs (nas:TRACON).

Appendix F Acronyms

Acronym	Acronym Expansion
ADL	Aggregate Demand List
ARTCC	Air Route Traffic Control Center (or "Center")
ASPM	Aviation System Performance Metrics
ATCSCC	Air Traffic Control System Command Center
ATCT	Air Traffic Control Tower
ATM	Air Traffic Management
CDR	Coded Departure Route
CONUS	Continental US
DAS	Delay Assignment
EDCT	Estimated Departure Clearance Time
ERAM	En-Route Automation Modernization
ETA	Estimated Time of Arrival
ETD	Estimated Time of Departure
ETMS	Enhanced Traffic Management System
FAA	Federal Aviation Administration
FCA	Flow-Controlled Area
GAAP	General Aviation Airport Program
GDP	Ground Delay Program
GMT	Greenwich Mean Time
GS	Ground Stop
IEC	International Electrotechnical Commission
ISO	International Standards Organization
LOA	Letter of Agreement
METAR	Meteorological Terminal Aviation Routine Weather Report
MIT	Miles-in-Trail
NAS	National Airspace System
NASA	National Aeronautics and Space Administration
OAG	Official Airline Guide
PAR	Program Arrival Rate
RVR	Runway Visible Range
SID	Standard Instrument Departure
STAR	Standard Terminal Arrival Route
TAF	Terminal Aerodrome Forecast
TFMS	Traffic Flow Management System
TMI	Traffic Management Initiative

TRACON	Terminal Radar Approach Control
UDP	Unified Delay Program
UTC	Coordinated Universal Time or "ZULU" Time
VOR	Very-high Frequency Omni Directional Radio Range
WITI	Weather Impacted Traffic Index
WMO	World Meteorological Organization