UCUENCA

UNIVERSIDAD DE CUENCA

FACULTAD DE INGENIERÍA

COMPUTACIÓN

Trabajo Final - Ontología para un sistema de vuelos

Autores:

Bryan Steven Mendoza Barahona

David Patricio Romero Alemán

Asignatura: Representación del Conocimiento

Docente: Ing. Mauricio Espinoza

Fecha de entrega: 27/07/2025

Breve descripción del propósito del modelo creado

El modelo ontológico “vuelo” fue diseñado para representar los elementos clave de un sistema de vuelos comerciales, incluyendo clases para aviones, vuelos, aerolíneas, ciudades, boletos y pasajeros, así como también a pilotos y copilotos con sus respectivas habilitaciones. El modelo cuenta con 39 clases, 17 propiedades de objeto y 9 propiedades de datos, además de 35 individuos que ejemplifican instancias reales de vuelos, equipos y usuarios. El objetivo principal era automatizar inferencias sobre asignaciones de recursos y facilitar consultas SPARQL para obtener información sobre horarios, disponibilidad y roles.

Reglas que ayudaron

Las reglas SWRL definidas redujeron la carga de anotación manual al automatizar reclasificaciones recurrentes. Por ejemplo la regla PilotoAutorizado reconoce directamente a cualquier tripulante con la habilitación adecuada para un tipo de avión como miembro de la clase correspondiente, ayudando a evitar tener que marcar uno a uno quién está capacitado para volar cada modelo. De manera similar la regla VueloInternacional agrupa automáticamente como “Internacional” aquellos vuelos cuyo origen y destino pertenezcan a ciudades de países distintos, eliminando la necesidad de asignar manualmente el atributo de tipo de vuelo.

Con todas las reglas propuestas se lograría acelerar la carga y el mantenimiento de datos, además de que estas reglas refuerzan la coherencia del modelo, reducen la posibilidad de errores manuales y permiten que el razonador aporte conclusiones fiables de manera automática y escalable.

Descripción sobre los problemas del modelado

Al ejecutar el razonador se vio que la ontología no generó información sobre el número de asientos libres en cada avión, de modo que no se mostraron conceptos que identificaran las aeronaves con capacidad completa ni las que todavía contaban con plazas disponibles. De igual forma no se produjo ninguna clasificación que enlazara cada vuelo de manera exclusiva con una sola nave ni que asignara o cada boleto a un pasajero único, por lo que no aparecieron agrupaciones de boletos ya ocupados ni de asientos pendientes de asignar. Tampoco surgieron clases diferenciadas para distinguir aviones en estado ocupado de aquellos declarados libres, ni se distinguió entre pilotos y copilotos con servicio activo frente a los que se encontraban desocupados, con lo que no se obtuvo información acerca de la disponibilidad de tripulación. También al no contar con un esquema temporal que definiera ventanas de tiempo para los vuelos y los turnos de los tripulantes, el razonador no detectó cruces de horario ni conflictos en la programación, dejando sin clasificar aquéllos vuelos o personas que debían aparecer como solapados o en conflicto.

También se trató de agregar una regla llamada MejoraAsientoAncianos. Esta regla intentaba identificar a cada pasajero clasificado como Anciano que tuviera un boleto en clase Turista y, a partir de esa condición, inferir dos nuevos hechos: declarar su boleto como BoletoUpgrade y asignarle la clase Ejecutiva. Al cargarla en Protégé apareció el mensaje Invalid OWL individual name ‘Ejecutiva’ porque el editor de SWRL esperaba que el término usado en la posición de objeto de la propiedad claseAsiento fuera un individuo previamente

definido y no una clase de la ontología. Debido a que ya se penso toda la ontología con ‘Ejecutiva’ como clase no se agregó la regla (Figura 1).

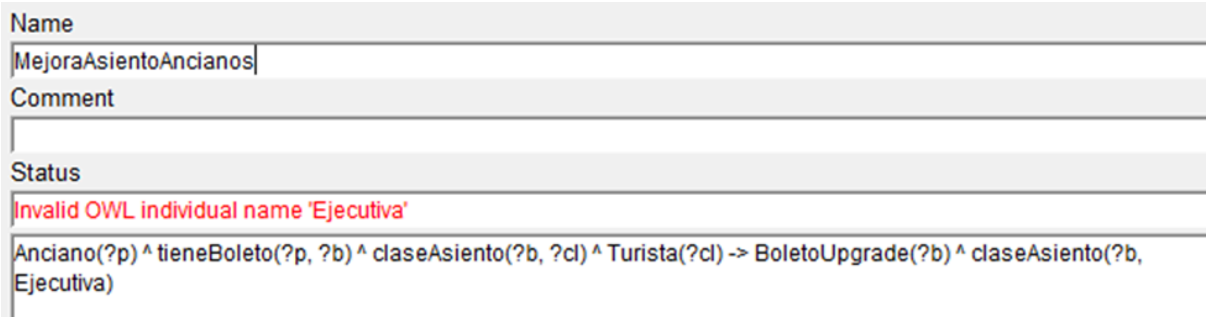

Figura 1. Regla que no se agregó

Dominio sin inferir vs Dominio inferido

La versión original del modelo definía clases y propiedades de forma básica sin generar nuevas categorías. Con el modelo inferido el sistema añadió PilotoAutorizado y PilotoNoAutorizado para separar tripulantes según su habilitación. También creó PilotoSenior para identificar a los pilotos con más experiencia. Agrupó vuelos bajo las clases VueloInternacional cuando cruzan fronteras y VueloConRiesgo en rutas largas o complejas. Etiquetó boletos con BoletoConMascota o BoletoCostoExtra y señaló casos de EquipajeSobrepeso y RequiereCheckInEspecial. De esta forma las reglas que antes estaban escritas ahora aparecen como clases reales. Esto elimina la necesidad de etiquetar cada caso a mano y nos ayuda a realizar consultas más potentes que apoyan la toma de decisiones.

Validaciones

Se añadieron validaciones que ayudaron en reforzar la integridad del modelo y mejorar la detección automática de conflictos. Por un lado se estableció que cada vuelo debe contar con un piloto y un copiloto distintos, lo que permitió al razonador identificar de inmediato aquellos itinerarios en los que faltaba un miembro de la tripulación o se había asignado la misma persona a ambos roles. Otro ejemplo es que se impuso la regla de que ningún avión puede figurar en dos vuelos cuya franja horaria coincida, de modo que se pudiera detectar solapamientos y advertir sobre posibles conflictos de programación en la flota. Gracias a estas validaciones, el modelo ahora no solo valida la coherencia de las asignaciones, sino que también apoya consultas más fiables sobre disponibilidad de aviones y recursos humanos.

Conclusión

Este trabajo ha consistido en diseñar y refinar una ontología para un sistema de vuelos que estructura las clases de aeronaves, vuelos, pasajeros y tripulación y define propiedades y reglas para automatizar tareas como distinguir pilotos autorizados y clasificar vuelos internacionales. La comparación entre el modelo original y su versión inferida ha demostrado cómo las reglas SWRL generan nuevas categorías que enriquecen la información y reducen la carga de anotación manual. Al incorporar inferencias para permisos de vuelo, análisis de riesgo en rutas y gestión de equipajes especiales se facilita la

ejecución de consultas complejas y se mejora la toma de decisiones. A pesar de las limitaciones iniciales en la gestión de asientos y en la definición temporal de asignaciones, la recomendación de añadir restricciones de cardinalidad, axiomas de funcionalidad y un esquema temporal permitirá convertir la ontología en un sistema completamente automatizado que garantice coherencia y disponibilidad de recursos en tiempo real.