Air-incident Ontology

Brian Mc George (MCGBRI004) and Charles Du (DXXCHA001)

1. Introduction

Air-incident is an ontology made to facilitate an ontology driven information system on flight Incidents. Finding the relevant information quickly and querying further about particular data points are aspects that Air-incident aimed to facilitate. Air-incident aims detect inconsistencies in entries by being restrictive yet open enough to allow unusual yet plausibly correct entries to exist.

2. Description of the ontology

Figure 1 shows the classes for the ontology developed. The core classes are Flight, Incident and Investigation. All relationships branch out from those three classes. Ideally, the aircraft class would be imported and make use of an existing aircraft ontology. However, no suitable aircraft ontology was found. As a result, a sub-set of the classes that could exist under the aircraft class was provided. Figure 2 shows the various object properties that were used. The DOLCE object properties were used where applicable. Additional properties were added to ensure the relationship was easy to understand and explicit.

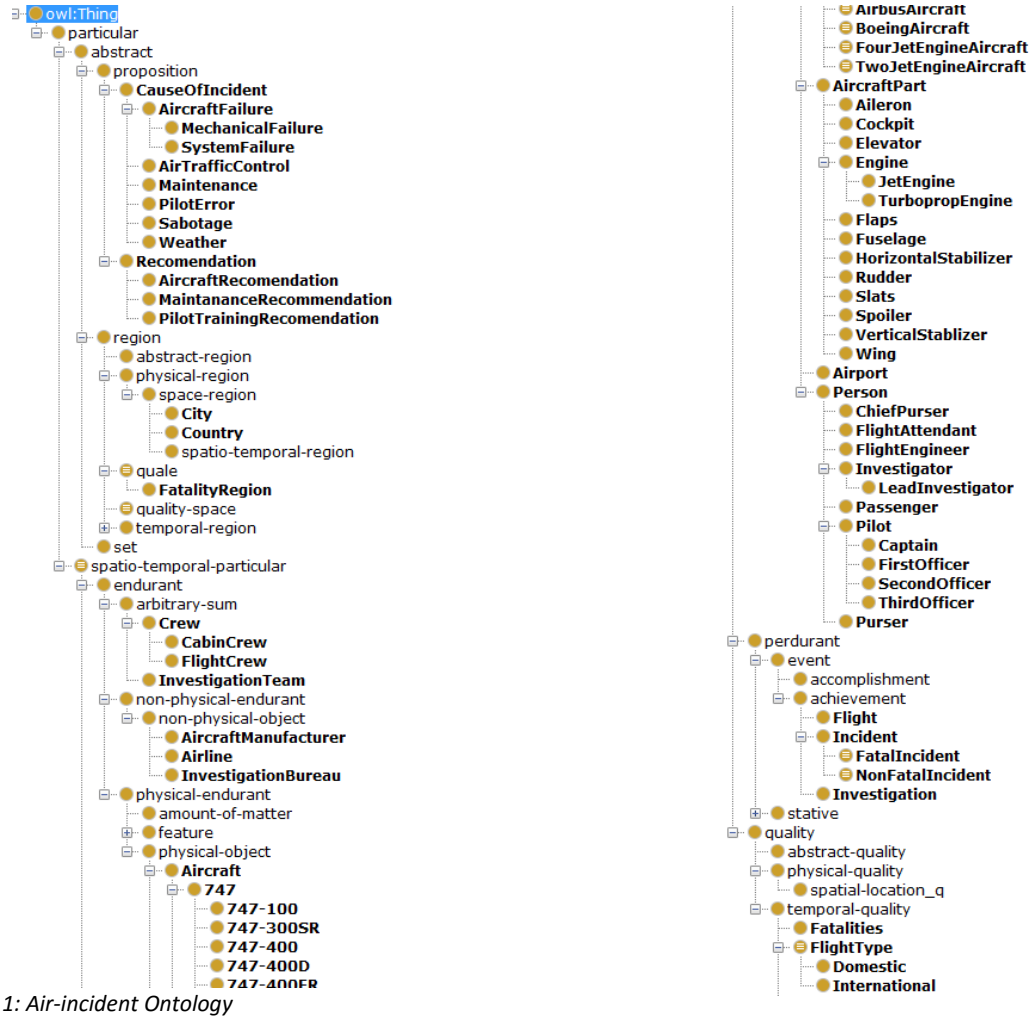

Figure 1: Air-incident Ontology

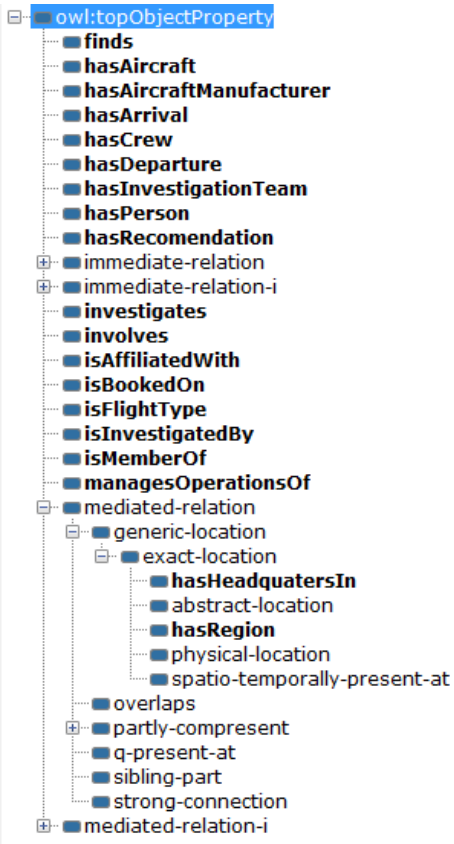

Figure 2: Object properties found in Air-incident

3. Goals of the ontology

Air incidents are complex events that take place across the world. This ontology aims create a knowledge base that facilitates easy access to information on past air incidents, their investigations, the causes of the incidents and the recommendations that arose from the investigation.

3.1. Interoperability

It was determined that an important feature of an information system is interoperability amongst other information systems **(**Sheth, 1999). A foundational ontology was therefore used to increase inoperability. The ONSET (Khan & Keet, n.d) tool was used to determine the foundational ontology that would be used. DOLCE (Masolo et al, 2003) was recommended and upon investigation, it was found that the DOLCE foundational ontology encouraged the design of an ontology that was of higher quality (Keet, 2011). Using DOLCE as a foundational ontology allows for easier integration with other ontologies that also use DOLCE as a foundational ontology (Keet, 2011).

Using the DOLCE foundational ontology ensured the intention of a class was clear since defined and primitive classes needed to be categorized into the classes provided by the DOLCE foundational ontology **(**Schneider, 2003).

The ontology development started before the foundational ontologies had been found by Air-incident Modellers. The Air-incident ontology was then integrated with the DOLCE foundational ontology. This resulted in different naming schemes in the ontology since Air-incident used camel case naming scheme while the DOLCE foundational ontology used lowercase naming scheme with a dash as a separator.

3.2. Data Querying

Relations or otherwise known as object properties in protégé showed and described how certain entries were related and connected to one another. The relations or object properties were used for the traversal of and further querying (otherwise known informally “digging deeper”) of data within Air-incident. An example within Air-incident would be, a user looking at a particular investigation would see the “hasInvestigationTeam” relation showing which investigation team was connected to the investigation. If user wished to inquire further about the investigation team, the user can through the hasInvestigationTeam object property.

Initially there had been difficulty in distinguishing what should be a subclass and what should be an object property. This resulted in a deep class hierarchical structure as object properties were mistakenly sub classed. Clarification of is-a relations for sub classing led to the correct usage of the object properties. This led to flatter class hierarchical which according to Rector et al. (2004) is a better for ontology design.

As an investigator it is important to know where an airplane part fits within the overall structure of the aircraft. A transitive relationship between aircraft parts is used to facilitate such queries. While this relationship is also asymmetric and reflexive, OWL (W3C, 2012) was unable to represent this as the language would become undecidable (W3C, 2009).

3.3. Classification

Classification in the context of an information system groups information together which allows users of the information system to find information as well as compare information about similar objects. Using the automatic reasoner for classification allowed new individuals to be created without explicitly classifying each entry. An example of this in Air-incident is that all incidents which had 0 fatalities were automatically classified under “NonFatalIncident”. The Ethiopian Airlines Flight 702 Hijacking incident in 2014, where no fatalities occurred, would be automatically classified under “nonFatalIncident” as seen in Figure 3. To achieve this, the Incident class has a quality “Fatalities” (as a flower would have a colour quality) which maps to a “FatalitiesRegion” which then has a dataProperty “hasDataValue” which maps to an rdf:PlainLiteral datatype. The literal datatype was chosen to allow interoperability between data sources.

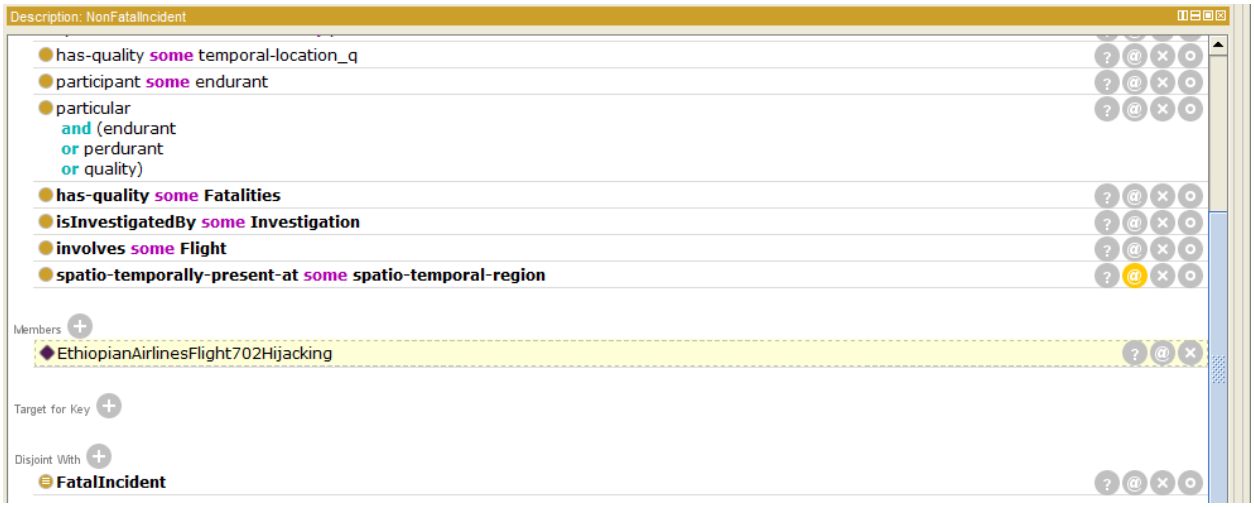

Figure 3: Showing the automatic instance classification of EthiopianAirlinesFlight702Hijacking

Relations were partly used for classification by using range and domain restrictions. For instance, the "isMemberOf" object property had the domain restricted to "Person". By using the automatic reasoner, any entry with the isMemberOf relation would be classified to "Person".

Classification of classes was done by using defined classes. Aircraft classes were automatically classified depending on manufacturer and number of engines. Figure 4 and Figure 5 show how the classes are automatically classified based on their object property relationships.

Figure 4: Class Hierarchy

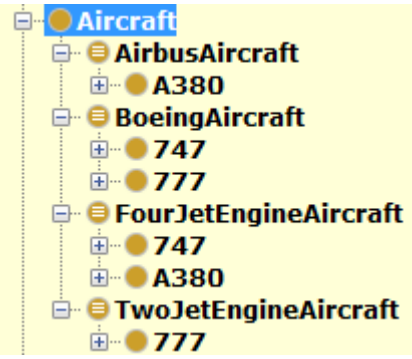

Figure 5: Inferred class hierarchy

3.4. Inconsistency and error detection

Restrictions of domain and ranges in object properties ensures consistency and that object properties are used as intended. For example, the "hasAircraftManufacturer" object property has the range set to "AircraftManufacturer" and domain to "Aircraft". Misuse of the "hasAircraftManufacturer" relation will be detected by the automatic reasoner. It will determine inconsistencies in entries and satisfiability issues which may arise from a poorly defined class.

Cardinality was used to ensure information is complete and that instances are not classified incorrectly. This was used on specific aircraft models which required a fixed number of a specific airplane part. An example of this would be the 747 aircraft as shown in Figure 6.

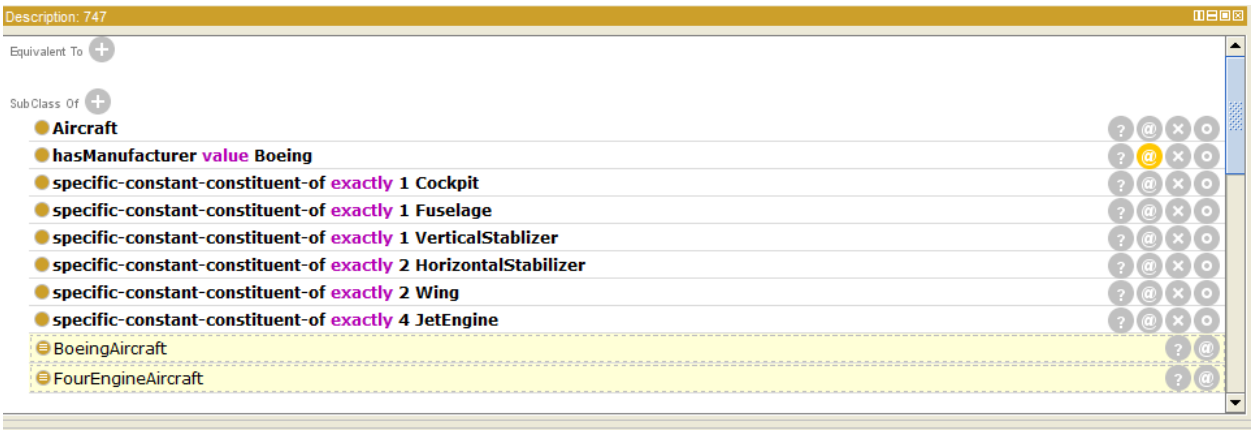

Figure 6: 747 class description

Restrictions to the number of relations between concepts has also been imposed. By classifying the object property "isType" as a functional relation, one ensures that a particular flight cannot be both the type "Domestic" and the type "International" at the same time.

4. Testing

A number of test cases were used to ensure that the reasoner classified classes and instances as expected. Protégé 5.0.0 Beta 23 (Gennari et al., 2003) was used to develop the ontology and the HermiT (Shearer et. al., 2008) reasoner was used throughout the development process. Both success and failure probes were used. The success probes were highlighted in Section 3.3. A class 747-FailureProbe was used to ensure that the reasoner picked up cases where classes were inconsistent. In this case, two wings were defined in the parent class and three wings were defined in the failure probe resulting in an inconsistency.

The OntOlogy Pitfall Scanner (OOPS) (Poveda-Villalón et al., 2012) was used to identify various pitfalls within the ontology. The main pitfall identified was the domain and range of various properties not being set.

5. Future possibilities

The Air-incident ontology lays the ground work for an extensive ontology. The ontology has a large scope and can be extended in a number of different ways. One of them being the Aircraft class discussed in Section 2, the other being the AircraftPart class. It would be better to move the aforementioned classes to their own ontology which is then imported into the Air-incident ontology as this would allow Air-incident to be more focused towards to its primary goals.

The relationships between the various investigation components could also be extended as required.

6. Conclusion

The Air-incident ontology has represented the knowledge of air incidents such that its goals have been satisfied: interoperability, data query, classification and error detection. These goals facilitate the overall aim of creating ontology driven information systems which make use of the data encapsulated in this knowledge base.

7. References

Gennari, J. H. et. al. 2003. The evolution of Protégé: an environment for knowledge-based systems development. Int. J. of Hum.-Comp. Studies 58, 1 , 89-123.

Keet, C.M. 2011. The use of foundational ontologies in ontology development: an empirical assessment. 8th Extended Semantic Web Conference (ESWC'11), G. Antoniou et al (Eds.), Heraklion, Crete, Greece, 29 May-2 June. Springer, Lecture Notes in Computer Science LNCS 6643, 321-335.

Khan, Z., Keet, C.M. n.d. ONSET: Automated Foundational Ontology Selection and Explanation.18th International Conference on Knowledge Engineering and Knowledge Management (EKAW'12), A. ten Teije et al. (Eds.). Oct 8-12, Galway, Ireland. Springer, Lecture Notes in Artificial Intelligence LNAI 7603, 237-251.

Masolo, C., Borgo, S., Gangemi, A., Gaurino, N. and Oltramari, A. 2003. Ontology Library. ICT (33053)13-25

Poveda-Villalón, M., Suárez-Figueroa, M.C. and Gómez-Pérez, A., 2012. Validating ontologies with OOPS!. In *Knowledge Engineering and Knowledge Management* (pp. 267-281). Springer Berlin Heidelberg.

Rector, A., Drummond,N., Horridge,M., Rogers,J., Knublauch, H.,Stevens,R., Wang ,H. and Chris Wroe. 2004. OWL Pizzas: Practical Experience of Teaching OWL-DL: Common Errors & Common Patterns. Engineering Knowledge in the Age of the Semantic Web Lecture Notes in Computer Science: 63-81. Print.

Schneider, L. 2003. How to Build a Foundational Ontology. KI 2003: Advances in Artificial Intelligence Lecture Notes in Computer Science: 120-34. Print.

Shearer, R., Motik, B. and Horrocks, I., 2008, October. HermiT: A Highly-Efficient OWL Reasoner. In *OWLED* (Vol. 432, p. 91).

Sheth, Amit P. 1999. Changing Focus on Interoperability in Information Systems:From System, Syntax, Structure to Semantics.Interoperating Geographic Information Systems : 5-29. Print.

W3C, 2009. *OWL 2 Web Ontology Language Structural Specification and Functional-Style Syntax.* [Online]
Available at: https://www.w3.org/TR/2009/REC-owl2-syntax-20091027/#The_Restrictions_on_the_Axiom_Closure
[Accessed 10 March 2016].

W3C, 2012. *OWL 2 Web Ontology Language Document Overview (Second Edition).* [Online]
Available at: https://www.w3.org/TR/owl2-overview/
[Accessed 10 March 2016].

8. Ontologies Imported

Laboratory for Applied Ontology, n.d. *DOLCE-lite.* [Online] Available at: http://www.loa-cnr.it/ontologies/DOLCE-Lite.owl [Accessed 8 March 2016].