A graphical user interface application for querying the unmanned aerial system integration safety and security technology ontology

By

Matthew Richard Beach

Approved by:

Samee Khan (Major Professor)
John Ball
Chaomin Luo
Jenny Q. Du (Graduate Coordinator)
Jason M. Keith (Dean, Bagley College of Engineering)

A Thesis
Submitted to the Faculty of
Mississippi State University
in Partial Fulfillment of the Requirements
for the Degree of Master of Science
in Electrical and Computer Engineering
in the Department of Electrical and Computer Engineering

Mississippi State, Mississippi

August 2024

Name: Matthew Richard Beach

Date of Degree: August 08, 2024

Institution: Mississippi State University

Major Field: Electrical and Computer Engineering

Major Professor: Samee Khan

Title of Study: A graphical user interface application for querying the unmanned aerial system integration safety and security technology ontology

Pages of Study: 27

Candidate for Degree of Master of Science

Unmanned aerial systems (UAS) have become wildly popular over the past decade. With the increased demand of these systems, it is imperative for agencies such as the Federal Aviation Administration (FAA) to integrate rules and regulations of UAS into the National Airspace System (NAS). In 2023, a UAS Integration Safety and Security Technology Ontology (ISSTO) was developed in the Web Ontology Language (OWL) to aid in this integration. To further aid UAS integration into the NAS, it becomes necessary to develop methods for training new pilots, air traffic control operators, drone operators, etc. This paper proposes a query application that allows new operators to query through ISSTO and efficiently access UAS and counter-UAS information from the knowledge domains contained within the ontology. These knowledge domains include information on various FAA regulations and authorizations, National Airspace classifications, counter-UAS procedures and more. The application developed in this thesis serves as a proof of concept for a commercialized training product.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

In 2022, the Federal Aviation Administration (FAA) contracted the Alliance for System Safety of UAS through Research Excellence (ASSURE) for the research project titled "Evaluation of Unmanned Aircraft Systems (UAS) Integration Safety and Security Technologies in the National Airspace System (NAS) Program [10]." One of the purposes of this project was to develop an ontology that covered the domain of knowledge concerning UAS and related concepts in order to facilitate the integration of UAS safety and security technologies into the NAS. This came to be known as ISSTO (Integration Safety and Security Technologies Ontology). Throughout 2022 and 2023, ISSTO was developed. It contains smaller domains such as Aircraft Types and Manufacturers (AC), National Airspace System (NAS), Operations, Actions, and Outcomes (OPS), Sensor Types and Components (SEN), UAS Components (UAS), UAS Air Traffic Management (UTM), Waivers, Authorizations, ATC Advisories (WAA), and Weather (WTHR). In order to make this ontology useful, an application was developed that would query information from the ontology, and display that information to a user. The information would display to the user the concepts as they are related to each other in the ontology, as well as definitions and a means of gathering further information of the given topic. The purpose of building this application is to help someone new to UAS and counter-UAS systems to easily gain knowledge from the ontology as well as give them a

starting point to better their understanding of various UAS concepts. This thesis outlines how the application was developed and the tools used in its development.

The contributions added to integrating UAS safety and security technologies into the National Airspace include

- Developing a proof of concept application that allows a user to query for information contained within ISSTO
- Testing the application using multiple user case scenarios
- Integrated existing python package and libraries, and other technologies into the application

CHAPTER II

BACKGROUND

2.1 ISSTO

The Integration Safety and Security Technologies Ontology (ISSTO) was developed throughout 2022 and 2023 by Rebeca Garcia et al [10]. It contains many smaller domains of knowledge related to larger UAS and counter-UAS domains. The ontology was built using Protégé, a free, open-source ontology editor and framework. The knowledge domains are stored using class entities organized in a hierarchy. Figure 2.1 below shows the ontology as seen in Protégé.

ISSTO is made up of eight, smaller domains of knowledge. Each of these domains focuses on different aspects of a National Airspace System with UAS integration. These domains are shown in table 2.1.

2.2 The Semantic Web

The semantic web, also known as web 3.0, is an extension of the world wide web. While most of the web is designed for humans to read, computers can only parse web pages for routine processing and layouts. The semantic web brings structure to meaningful content found within web pages by giving machines a way to process the semantics, and understand the present data. According to its founder Tim Berners-Lee as defined in the May 2001 Scientific American article, "The Semantic Web," [1].

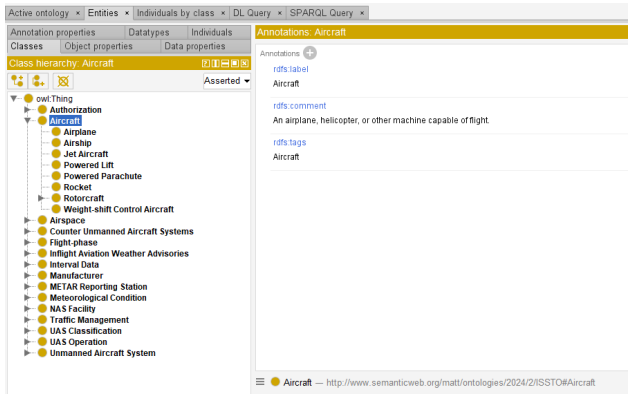

Figure 2.1

ISSTO as seen in Protégé

The Semantic Web is an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in cooperation.

The semantic web is built off Linked Open Data (LOD) and Semantic Metadata that allows for meaning to be inferred through a web of explicitly defined data contained within a vocabulary of knowledge. Linked Open Data contains the factual data and ontologies. Ontologies contain classes of objects, relationship types and their attributes. Semantic Metadata are tags that allow for meaning to be assigned to a piece of data. The semantic web was adopted as a set of standards by the World Wide Web Consortium (W3C) through 2001 to 2005 in what is known as the first phase of the semantic web [27]. The semantic web is organized as seen in figure 2.2 [6] in a stack of layers known as the semantic web cake.

Unicode and URIs make up the base with XML, Namespaces (NS) and xmlschemas built on top. The next layers are comprised of RDF and rdfschemas, ontology vocabulary, logic and proof with digital signatures for each layer. The topmost layer is trust because a user trusts that each

Table 2.1

Domains within ISSTO

Namespace	Content
AC	Aircraft Types and Manufacturers
NAS	National Airspace System
OPS	UAS Operations, Actions, and Outcomes
SEN	Sensor Types and Components
UAS	Components of Unmanned Aerial Systems
UTM	UAS Air Traffic Management
WAA	Waivers, Authorization, ATC Advisories
WTHR	Weather

of these technologies are implemented correctly and the information contained in each layer is accurate.

For the semantic web to work, automated reasoning is conducted by applying a series of inference rules to structured collections of information. Two technologies that were crucial to the development of the semantic web are eXtensible Markup Language (XML) and the Resource Description Framework (RDF). XML allows for users to tag their web pages with hidden tags that give structure to their documents. RDF allows for the meaning of the data to be expressed by encoding the data in a set of triples in the form of subject, predicate, object. By combining XML with RDF, data can be expressed in XML tags in a set of RDF triples. The result is a natural way to describe how machines process data. The subject and object are identified using a Universal Resource Identifier (URI) that links to a piece of data and the predicate is what relates these URIs to each other, similar to how Universal Resource Locators (URLs) provide a link to a web page. URLs are the most common type of URI. Machines require explicit definitions and the URI is how

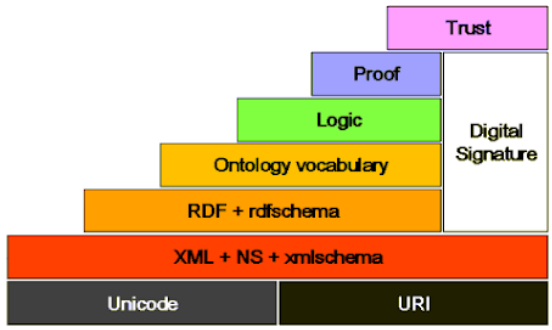

Figure 2.2

Semantic Web Cake

information is explicitly defined in RDF. A query language known as SPARQL is needed to query for the data found within an RDF/XML document.

2.3 SPARQL

SPARQL (SPARQL Protocol and RDF Query Language) is the query language used to query data conforming to the RDF data model. Data in an RDF data model is stored in triples of subject, object and predicate, which allows the data to be easily queried for. This is shown in table 2.2.

To query for the phone number, firstName and phoneNum would need to be declared first, pointing to the associated property value. A simple SPARQL query is made up of two clauses, SELECT and WHERE. The SELECT clause selects the variables being queried for, the WHERE clause tells SPARQL where to find them by matching the SPARQL triple pattern to an RDF triple pattern. An example SPARQL query taken from Learning SPARQL [7] is shown in figure 2.3.

Craig's email can be found because that property value exists in the triple pattern containing `ab:craig` and `ab:email`. The property value is stored in the variable `?craigEmail`, selected in the SELECT clause. The SPARQL keyword PREFIX is used to make the query statements more

Table 2.2

RDF Triple Patterns

subject (resource identifier)	predicate (property name)	object (property value)
firstName	phoneNum	(111) 111-1111
firstName	email	firstName@email.com

```
# filename: ex010.rq

PREFIX ab: <http://learningsparql.com/ns/addressbook#>

SELECT ?propertyName ?propertyValue
WHERE
{ ab:cindy ?propertyName ?propertyValue . }
```

Figure 2.3

Performing a SPARQL query

concise by binding a namespace to a variable. Another common keyword is DISTINCT which eliminates possible duplicates in the results.

2.4 Web Ontology Language

According to the W3C website, the Web Ontology Language (OWL) is a semantic markup language for publishing and sharing ontologies on the World Wide Web [5]. It is a vocabulary extension of RDF. There exists two specific subsets of the OWL language. OWL DL (Description Logic) and OWL Lite. OWL Lite is designed for easy implementation to help users get started with OWL, OWL DL is used for maximum possible expressiveness of the ontology to be used with powerful reasoning systems. If an ontology is not specified to be DL or Lite, then the ontology is OWL Full. OWL Full is not a sublanguage like with OWL DL or OWL Lite and instead contains all

of the OWL language constructs while providing free, unconstrained use of RDF constructs. This means that in OWL Full, the resource owl:Class is equivalent to rdfs:Class whereas in OWL DL or OWL Lite, owl:Class would be a subclass of rdfs:Class. In OWL Full, all data values are in the individual domain, so object properties and data properties are not disjoint. The reason for using OWL Full is to have the flexibility and metamodelling features of RDF with the expressiveness of OWL [5].

A common method for developing an ontology is with Protégé, a free, open-source ontology editor and knowledge-base framework [24]. Protégé is a Java based application and supports RDF, RDFS, OWL and XML Schema formats. Protégé is also plug-in based. Some useful plug-ins include OWL Viz which can be used to visualize the ontology through GUI of nodes and edges, and Snap SPARQL which allows the user to query the ontology with SPARQL commands.

2.5 UAS Systems

UAS stands for Unmanned Aerial Systems. According to Title 14, Chapter 1, Subchapter A Part 1 of the Code of Federal Regulations [25], which includes the general definitions used throughout the chapter, an unmanned aircraft means an unmanned aircraft and its associated elements (including communication links and the components that control the unmanned aircraft) that are required for the safe and efficient operation of the unmanned aircraft in the airspace of the United States. An unmanned aircraft system means an unmanned aircraft and its associated elements (including communication links and the components that control the unmanned aircraft) that are required for the safe and efficient operation of the unmanned aircraft in the airspace of the United States [25].

Unmanned aircraft are used in a wide variety of applications from military and police applications of surveillance and national security measures, to civilian use in hobbyist circles and scientific research. Drones can be outfitted with a wide variety of sensors and payloads to accomplish this wide variety of applications. They can also vary in size and weight, and are classified by the DoD (Department of Defense) and the FAA (Federal Aviation Administration) as such. Unmanned aircraft systems are classified into two main groups according to Title 14, Chapter 11 of the Code of Federal Regulations [25]. A drone is classified as a small UAS or a large UAS according to its MGOW (mass gross operating weight). This specific regulation is known as Part 107. It states that a UAS must weigh less than 55 pounds to be considered a small UAS. Large UAS are further grouped according to MGOW, normal operating altitude and speed. A large UAS can be classified under any group 1 - 5 based on its specifications. These groupings are useful for controlling which airspace a UAS can fly since they also share the national airspace system (NAS) with civilian aircraft [25]. Regulating airspace is essential for national security, privacy and safety.

According to Drone Industry Insights [17], a leading source for data on commercial drones, the global drone market size is expected to reach USD 54.6b by 2030 with a compound annual growth rate of 7.7 percent. The top application method is said to be mapping and surveying followed by photography and filming [16]. A global forecast to 2028 by marketsandmarkets.com [19] shows that the military drone market is projected to reach USD 18.2b with a compound annual growth rate of 7.0 percent. This is attributed to a growing defense budget and enhanced military capabilities. This rising demand has fueled investment in research and development. As the technology is further integrated into daily life, more regulations will be needed to ensure the national airspace is safe and secure.

2.6 Related Works

In building an application that can parse through an ontology based on UAS and counter-UAS technologies and procedures, it is important to establish a literature review on existing UAS and counter-UAS ontologies. Ontologies are useful for building a knowledge framework that can guide researchers and other individuals seeking to know more about a given topic by having various concepts and their relationships within a domain of knowledge visualized and neatly organized. In a paper by Juliette Chevallier [4], a UAS Mission ontology is developed in order to guide individuals engaged in research and product development of autonomous vehicles. The ontology focuses on a systems-level description of relevant concepts for UAS missions. The UAS Mission Ontology uses industry-wide accepted resources to ensure individuals across the industry have common understanding of the related concepts. An ontology by Susan Ferreira and Jenny Tejeda [8] is developed to help support individuals seeking to know more about the testing and evaluation systems of unmanned autonomous systems. A paper by David Miller [22] discusses the development of the Airspace System Ontology, an ontology that conceptualizes airspace system architectures capable of supporting future airspace operations of 2035 and beyond.

Ontologies are capable of housing data that can be used for automated purposes. A paper by David Martín-Lammerding et al. [20] develops a novel ontology called Dronetology-cas for managing a collision avoidance system (CAS). The ontology allows the CAS to make autonomous decisions based on UAS gathered data. Ontologies have also been developed for path planning adaptation systems [21], detecting active and passive drone threats [18], and automated decision making processes to aid air traffic controllers with maintaining airspace safety [13] [15] [14]. Another paper [28], uses an ontology-based knowledge built through web crawling data and natural

language processing to control drones with voice commands. One unique ontology application proposed by D. Cavaliere et al. adds semantic information to the video tracking process to create a reliable object detection solution for aerial surveillance [3].

CHAPTER III

METHODOLOGY

This chapter covers the method for developing the proposed application for querying ISSTO. Concepts in ISSTO are stored in an RDF/XML document, built using Protégé. The RDF/XML document is parsed using the RDFLib library and displayed in a GUI built using the TKInter library.

3.1 Ontology

The information retrieved by the application is stored in an RDF/XML document developed using Protégé. A reasoner is used to check that the ontology is logically consistent and no errors exist that could cause the final application to crash or fail to run. The plug-in Snap SPARQL was used to test query commands that would be implemented in the SPARQL class, shown in Figure 3.1.

`?x` is used a generic dummy variable for the resource identifier. The rest of the statement follows the RDF triple pattern. The resource identifiers in the ontology are limited to those with the property value of owl:Class, selecting every entity that is a class entity. The label annotation property value is reached by further limiting the query to pull entities that have the property name of rdfs:label, then finally storing those property values in the variable `?label`. The FILTER

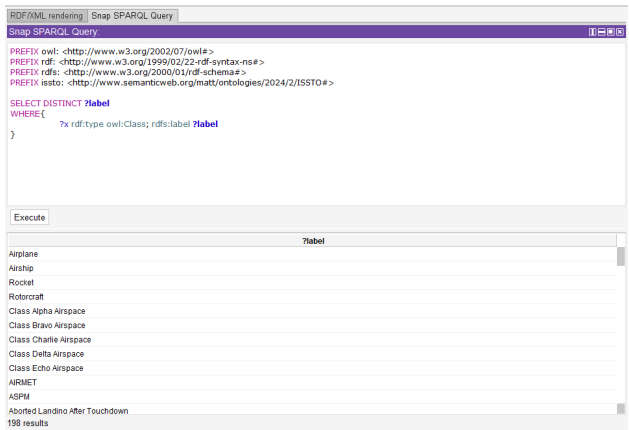

Figure 3.1

SPARQL query for retrieving every class entity label

command with a regex function can be used to further limit the results based on a string, shown in figure 3.2.

The label variable is selected for ease of searching. Every entity in the ontology is given a label annotation for this reason. SPARQL checks every label for an exact match to whatever word is input into the regex function. This ability to search the ontology is useful for the building of the application to allow for user input.

3.2 Parser

The RDF/XML document is parsed using RDFLib, a pure Python package for working with RDF. It contains Parsers and serializers for RDF/XML, N3, NTriples and more. RDFLib is chosen for its extensive documentation, ease of use within Python, and SPARQL language support [29]. The maximum amount of results are achieved by extracting each word in the user input, finding synonyms to each word, and appending them all to a keywords list. SpaCy, a free open-source

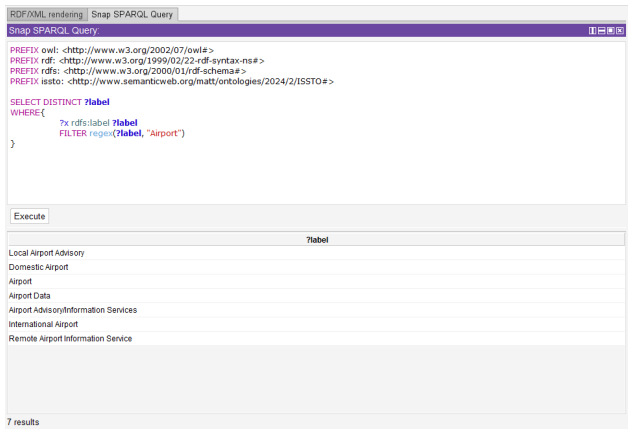

Figure 3.2

SPARQL query for a string

library for Natural Language Processing, is used to lemmatize the user input [12]. Synonyms for each word are found by searching through the WordNet corpus [2] provided through the Natural Language Toolkit (NLTK) [23]. A SPARQL query queries each word in the keywords list. Each word is passed as a string in a `regex` function and the query results are filtered using the SPARQL `FILTER` command.

The `search` method of the `MySPARQL` class returns a list of lists, where each nested list is a result retrieved from the SPARQL query search. The SPARQL query returns a result if any annotation in a triple pattern contains an instance of any word in the keywords list. This includes labels, comments, and meta tags defined within the RDF file. Each result is evaluated on relevancy, given a score, and sorted such that the items most closely associated with the user input appear towards the top of the list of results. The score is calculated by counting how often a word in the keywords list appears in an annotation, with weights attached to the label and tags annotations.

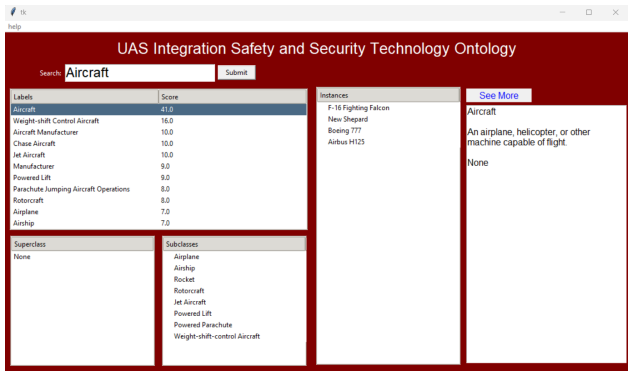

Figure 3.3

Simple query

3.3 Proposed Application

The user searches by typing into the input search box and pressing submit. The results and their respective subclasses, superclasses, and instances can be seen in the figure below. The list of results is sorted from highest score to lowest such that every class entity that is closer related to what the user searches is pushed towards the top of the list. In figure 3.3, the user queries "Aircraft". The result is that the subclasses are correctly filled as well as the instances of aircraft within the ontology.

If the class entity contains subclasses, or is the subclass of another entity, these related entities will appear in their respective boxes. A double click event is enabled in the first table which allows the user to double click an item in the list, populating the lower two tables with that label's respective superclass or subclasses as well as the instances table. In the case of Figure 13, no direct superclass exists, but eight subclasses and four instances exist.. Therefore, "None" is filled in for the superclass table while the subclasses and instances tables are populated. A single click event is enabled for every table. This populates the output window with a definition and hyperlink that

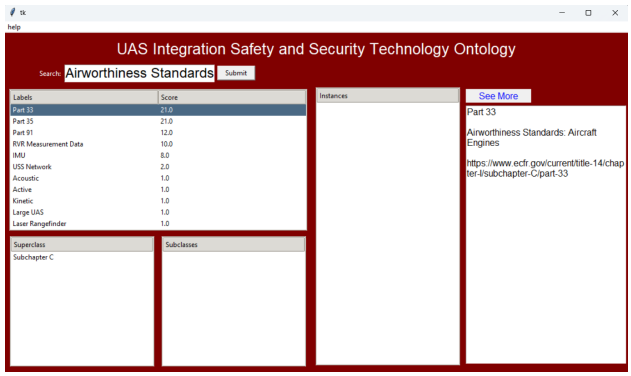

Figure 3.4

Complex query

corresponds to the selected label. If a hyperlink exists, it can be accessed by clicking on the “See More” button, opening the user’s default browser.

3.4 Complex Queries

A complex query shall be defined as a query that contains two or more words. Figure 3.4 shows the user performing the complex query ”Airworthiness Standards.”

More results are returned with a complex query with the caveat that computation time increases with each added word in the user input. Part 33 regulates the standards for the airworthiness of aircraft engines, defined in its `rdfs:comment` annotation. As a result, this class entity receives a higher relevancy score and appears in the first position of the Labels table.

CHAPTER IV

EVALUATION

This chapter covers three case studies used to evaluate the performance of the proposed application for parsing the Integration Safety and Security Technology Ontology. These case studies set requirements to be met by the parsing application. According to these performance requirements, a user is able to easily discover instructions on how to use the application, a user is able to perform complex queries and receive the relevant results, a user is able to cycle through entities contained within the ontology and discover their relationships to other entities in the ontology.

4.1 Case Study I

The first case study involves teaching the user how to use the application. A menu bar is added to the application that contains a button labeled ”help,” as seen in Figure 4.1 below.

By selecting ”help,” a popup window containing a short list of instructions is generated. The user may close out of this window by either pressing the button labeled ”okay,” or clicking the ”X” button in the top right of the popup window.

4.2 Case Study II

The second case study is to query the ontology to find applicable laws and or regulations regarding the use of UAS. The user searches “UAS regulations” and the labels table is populated

Figure 4.1

User is instructed on how to use the application

with relevant results. Included in these results are classes such as "Authorizations," "Part 91," etc. These results appear because the terms "UAS" and "regulations" appear somewhere in the annotations attached to that class label. The user clicks on "Authorization," subclass "Chapter 1," then which contains FAA regulations for aircraft, pilots and UAS operators as seen in Figure 4.2.

The user may click through each direct subclass showing their subsequent subclasses. Figure 4.3 shows the user selecting the class "Chapter 1."

"Chapter 1" is selected and the Superclass and Subclasses tables are populated. The user may select any of the subchapters to see the parts contained within the ontology. Subchapter C contains federal regulations and authorizations for aircraft, Subchapter D contains regulations and authorizations for airmen, and Subchapter F contains regulations and authorizations for air traffic and general operating rules.

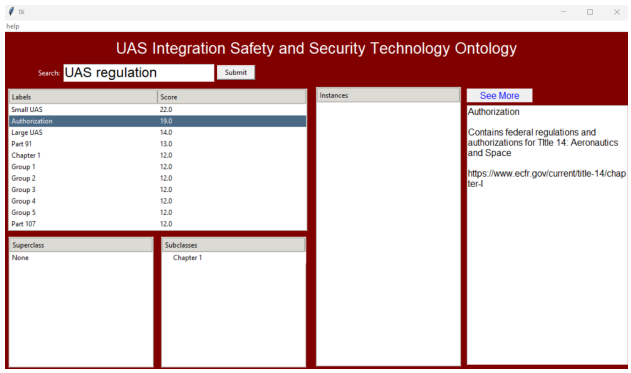

Figure 4.2

User searches for UAS regulations

4.3 Case Study III

The next use case involves the instance table. The instance table contains all of the instances for a class that exist in the ontology. The user queries the ontology for different types of devices used to mitigate small UAS to inform potential procurement of a counter-UAS device. In Figure 4.4, the user queries "counter UAS device." The user then selects "Counter Unmanned Aircraft Systems." This class contains the classifications for different responses to an unauthorized UAS.

"Active" and "Passive" classes appear in the subclasses table. The user selects "Active" as shown in Figure 4.5.

Four options for an active response to an unauthorized UAS appear in the subclasses table, as well as the KAPAN Counter-UAS System in the instances table. This is a system that can detect, track and neutralize a target. If the user wishes to know more information, they can select the item in the instance table to see the product description and website link. The user may also select "Passive," as seen in Figure 4.6.

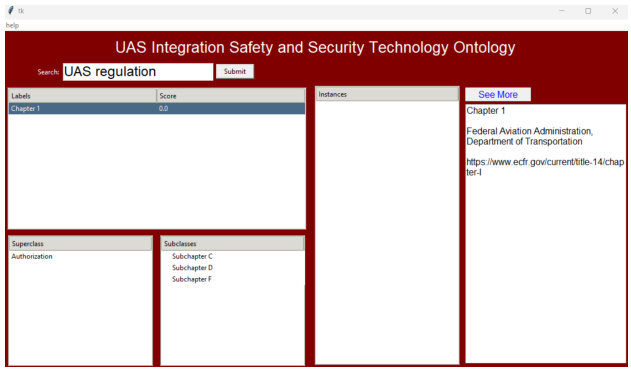

Figure 4.3

User discovers UAS regulations

Two systems are shown for a passive response in the instances table. By implementing these case studies, it is shown that a user can learn how to use the application as well as retrieve information relevant to their search.

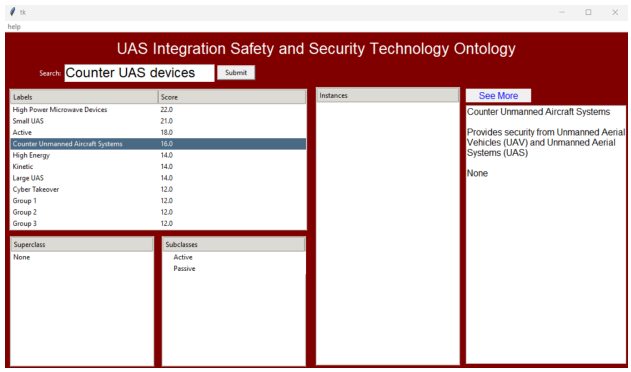

Figure 4.4

User searches for counter-UAS devices

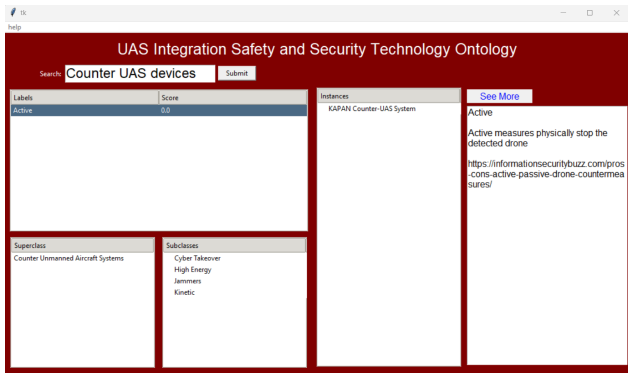

Figure 4.5

User selects ”Active”

Figure 4.6

User selects ”Passive”

CHAPTER V

CONCLUSION AND FUTURE WORK

The application developed in this paper serves the purpose of providing an uninformed user with an efficient manner of retrieving knowledge contained within ISSTO. Three case studies were developed to test various aspects of the application and the requirements of each case study were met. This application serves as a proof of concept towards developing a commercial ready product that could be installed either on a mobile device or desktop computer or developed as a web application. Further work on the application is needed for it to become a final product such as offering the user more ways of visualizing the relationships between entities and expanding the ISSTO ontology to include more UAS and counter-UAS information.

In order to offer more visualization, an interactive diagram could be developed that would allow the user to click through each class entity and discover the entity relationships as they exist in the ontology. OWLViz, a Protégé plug-in [24], offers a way for users to visualize entity relationships. An example of this plug-in can be seen in Figure 5.1.

To improve upon this concept for the application proposed in this paper, the user might have the option to see all of the annotations and instances associated with the selected class entity.

The application could also be further improved by developing it through the web. This way, the application would more accessible, as it would not need to be installed locally. The ontology

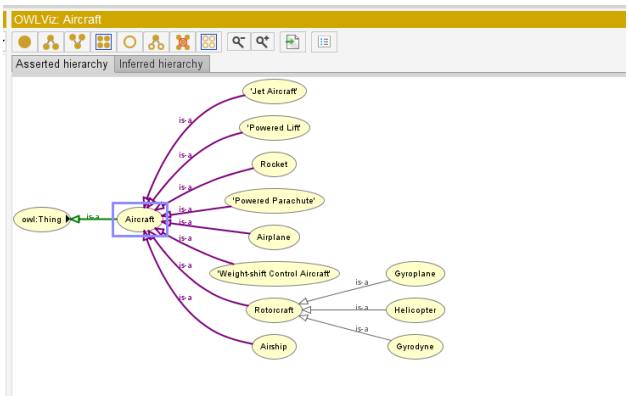

Figure 5.1

OWLViz view of ISSTO

requires expansion in all of its current knowledge domains as these domains have been kept small for the sake of brevity. If this application were developed for the web, the ontology could be updated efficiently by storing the information on a cloud server. Updates could be rolled out as new information is added to the ontology.

As more information is added to the application, the likelihood of the user coming into contact with unfamiliar terms increases. A potential solution for this would be for an in-house dictionary to be added to the application. This dictionary would supplement the currently existing dictionary (the comments annotations for each class and instance entities) by providing further context to the user as they click through the ontology. This dictionary would be a separate file or database that can be updated as the ontology is expanded.

This application integrates existing technologies such as SpaCy [12], the NLTK [2], and Protégé [24]. For this application to become commercial ready, licensing of these technologies is to be considered. All of these technologies are open-source and their use in the application product

is subject to the terms and conditions laid out in their respective license agreements [11], [9], [26].

REFERENCES

[1] T. Berners-Lee and O. Lassila, "The Semantic Web," *Scientific American*, vol. 284, no. 5, May 2002.

[2] E. L. Bird, Steven and E. Klein, *Natural Language Processing with Python*, O'Reilly Media Inc, 2009.

[3] D. Cavaliere, S. Senatore, M. Vento, and V. Loia, "Towards semantic context-aware drones for aerial scenes understanding," *2016 13th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 2016, pp. 115–121.

[4] J. Chevallier, *Enabling Autonomy in Commercial Aviation: An Ontology and Framework for Automating Unmanned Aircraft Systems (UAS)*, master's thesis, MIT, June 2021.

[5] W. W. W. Consortium, "OWL Web Ontology Language Reference,", https://www.w3.org/TR/owl-ref/.

[6] W. W. W. Consortium, "The Semantic Web Made Easy,", https://www.w3.org/RDF/Metalog/docs/sw-easy.

[7] B. DuCharme, *Learning SPARQL*, 2nd edition, chapter 1, O'Reilly Media, 2013, p. 9.

[8] S. Ferreira and J. Tejeda, "An Ontology for Unmanned and Autonomous Systems of Systems Test and Evaluation," *INCOSE International Synmposium*, vol. 9, no. 1, June 2011, pp. 1082–1091.

[9] A. S. Foundation, "Apache License,", https://www.apache.org/licenses/LICENSE-2.0, January 2004.

[10] R. A. Garcia, *Unmanned aerial system integration safety and security technology ontology*, master's thesis, Department of Electrical and Computer Engineering, Mississippi State University, May 2023.

[11] M. Honnibal, "The MIT License,", https://github.com/explosion/spaCy/blob/master/LICENSE, 2015.

[12] M. Honnibal, I. Montani, S. Van Landeghem, and A. Boyd, "spaCy: Industrial-strength Natural Language Processing in Python," 2020.

[13] C. C. Insaurralde and E. Blasch, "Ontological knowledge representation for avionics decision-making support," *2016 IEEE/AIAA 35th Digital Avionics Systems Conference (DASC)*, 2016, pp. 1–8.

[14] C. C. Insaurralde and E. Blasch, "Framework Prototype to Test Decision Support System for Avionics Analytics," *2021 IEEE/AIAA 40th Digital Avionics Systems Conference (DASC)*, 2021, pp. 1–11.

[15] C. C. Insaurralde and E. Blasch, "Situation Awareness Decision Support System for Air Traffic Management Using Ontological Reasoning," *Aerospace Research Center*, vol. 19, no. 3, Feb 2022.

[16] D. I. Insights, "237 Ways Drone Applications Revolutionize Business,", https://droneii.com/237-ways-drone-applications-revolutionize-business, May 2021.

[17] D. I. Insights, "Drone Market Report 2023 - 2030,", https://droneii.com/product/drone-market-report, 2023.

[18] R. P. Loui and J. Smith, "An ontology for active and passive aerial drone threat automatic plan recognition," *2016 IEEE National Aerospace and Electronics Conference (NAECON) and Ohio Innovation Summit (OIS)*, 2016, pp. 62–67.

[19] Markets and Markets, "Military Drone Market by Platform,", https://www.marketsandmarkets.com/Market-Reports/military-drone-market-221577711.html, Feb 2024.

[20] D. Martín-Lammerding, J. J. Astrain, A. Córdoba, and J. Villadangos, "An ontology-based system to avoid UAS flight conflicts and collisions in dense traffic scenarios," *Expert Systems with Applications*, vol. 215, 2023, p. 119027.

[21] D. Martín-Lammerding, A. Córdoba, J. J. Astrain, P. Medrano, and J. Villadangos, "An Ontology-Based System to Collect WSN-UAS Data Effectively," *IEEE Internet of Things Journal*, vol. 8, no. 5, 2021, pp. 3636–3652.

[22] D. Miller, "An ontology for future airspace system architectures," *2017 IEEE/AIAA 36th Digital Avionics Systems Conference (DASC)*, 2017, pp. 1–8.

[23] G. A. Miller, "WordNet: A Lexical Database for English," *Communications of the ACM*, vol. 38, no. 11, 1995, pp. 39–41.

[24] M. Musen, "The Protégé project: A look back and a look forward," *AI Matters*, vol. 1, no. 4, June 2015, pp. 4–12.

[25] O. of Federal Register, "Code of Federal Regulations,", https://www.ecfr.gov, 2024.

[26] T. B. of Trustees of Leland Stanford Junior University, "Copyright,", https://github.com/protegeproject/protege?tab=License-1-ov-filereadme, 2014.

[27] S. B. Palmer, "What Happened to the Semantic Web?,", https://lists.w3.org/Archives/Public/public-lod/2017Oct/0003.html, Oct 2017.

[28] S. Rajapaksha, V. Illankoon, N. D. Halloluwa, M. Satharana, and D. Umayanganie, "Responsive Drone Autopilot System for Uncertain Natural Language Commands," *2019 International Conference on Advancements in Computing (ICAC)*, 2019, pp. 232–237.

[29] R. Team, "rdflib 7.0.0,", https://rdflib.readthedocs.io/en/stable/index.html.