10TH INTERNATIONAL COMMAND AND CONTROL RESEARCH AND TECHNOLOGY SYMPOSIUM
EXPERIMENTATION TRACK

Title: A Semantic Web Application for the Air Tasking Order

Authors: Albert Frantz (POC), Milvio Franco

Organization: United States Air Force Research Laboratory/Command and Control Engineering
Address: AFRL/IFSA
525 Brooks Road
Rome NY 13441-4505, USA
Phone: (315) 330-1456, Fax: (315) 330-2807
Albert.Frantz@RL.AF.Mil

10TH INTERNATIONAL COMMAND AND CONTROL RESEARCH AND TECHNOLOGY SYMPOSIUM
EXPERIMENTATION TRACK

Title: A Semantic Web Application for the Air Tasking Order

Authors: Albert Frantz (POC), Milvio Franco
Organization: United States Air Force Research Laboratory/Command and Control Engineering
Address: AFRL/IFSA
525 Brooks Road
Rome NY 13441-4505, USA
Phone: (315) 330-1456, Fax: (315) 330-2807
Albert.Frantz@RL.AF.Mil

Abstract

The purpose of this in-house exploratory development was to investigate using Semantic Web technologies for Command and Control (C2) applications. This paper describes a Semantic Web application we developed for the *Air Tasking Order* (*ATO*), the document used to assign aircraft to perform specific missions. We used existing Semantic Web tools to construct an ATO knowledge base. The knowledge base is used to select potential air missions to reassign to strike time sensitive targets by the computer. This paper introduces Semantic Web technologies, followed by a discussion of the design and implementation of our ATO knowledge base. We conclude that the current Semantic Web tools are mature enough for computers to assist in fairly sophisticated C2 domain modeling and reasoning.

Introduction to the Semantic Web

The Semantic Web extends the World Wide Web by adding computer understandable semantics (meaning). This creates a *computer-processable*, *collaborative* communication medium. The vision is to allow computers to examine multiple Semantic Web pages and then *reason*, to create new facts from the existing facts. This enables a true query capability. It promises better knowledge management, electronic commerce and personal agent applications.

Key Semantic Technologies

The Semantic Web will consist of knowledge bases, reasoning tools and *Semantic Web Services*. The knowledge bases are created with the *Web Ontology Language (OWL*[1]*)*, *Resource Description Framework (RDF)* and *RDF Schema (RDFS)*. Ontology building tools such as *Protégé* are used to simplify constructing the knowledge bases. *Semantic Web Services* are software applications on the Web that can be discovered, described, accessed and understood by computers allowing them to process the data, relationships and meaning in knowledge bases.

Taxonomies, Ontologies and Knowledge Bases

"The first step toward the Semantic Web and using Semantic Web services is expressing the taxonomy in a machine-readable form."[DACONTA]. The *taxonomy* is a classification based on a tree structure that categorizes some specific domain. The best known taxonomies are the plant and animal kingdoms in biology. Taxonomies are based on classes and subclasses that form a

[1] The W3C OWL acronym mimics Winnie the Pooh character Owl misspelling his name as Wol.

tree structure. Taxonomies are pervasive in our lives because we construct classifications of things to better understand them.

Ontologies extend taxonomies and are semantically richer than taxonomies. *Ontologies* provide a shared and common understanding of a domain to be communicated among people and computers to facilitate knowledge sharing and reuse. Ontologies provide a formal explicit conceptualization (i.e. meta-information) that describes the semantics of information of the static domain knowledge of knowledge based systems. When the ontology is populated with specific dynamic *instances* (facts) of information, it becomes a *knowledge base*. The knowledge base can be reasoned over to create new facts from the knowledge base. Informally, the knowledge base is a set of sentences written in a knowledge representation language that represents some assertions about the domain. Similar to a database, the knowledge base must have the ability to add new information through updates and reasoning, as well as the ability to query the knowledge base. A knowledge base shares many of the concepts of a database, for instance, complex relationships, but also adds machine readable semantics and reasoning.

The key concepts to remember about the Semantic Web are that it is distributed, collaborative and computer readable.

Knowledge Engineering

In [RUSSELL] they discuss the **knowledge engineering process** for a knowledge base in detail. Summarizing the engineering process results in the following steps:

1. Identify the task
2. Assemble the relevant knowledge (*knowledge acquisition*)
3. Decide on a vocabulary of predicates, functions and constants
4. Encode general knowledge about the domain
5. Encode a description of the specific problem instance
6. Pose queries to the inference procedures and get answers
7. Debug the knowledge base

We used Protégé to construct the ATO knowledge base. Protégé is an ontology and knowledge base editor produced by Stanford University. Protégé is a tool that enables the construction of domain ontologies, customized data entry forms to enter data. Protégé allows the definition of classes, class hierarchies, variables, variable-value restrictions, and the relationships between classes and the properties of these relationships. Protégé is free and can be downloaded from http://protégé.stanford.edu . Protégé comes with visualization packages such as OntoViz, EZPal, etc.; all of these help the user visualize ontologies with the help of diagrams. Stanford University is doing a magnificent job of continually improving Protégé. As part of its last update, Protégé now includes an interface for SWRL (Semantic Web Rule Language), which sits on top of OWL to do math, temporal reasoning, and adds Prolog-type reasoning rules. Stanford has a tutorial that covers the basics of using Protégé with the OWL plug-in. Additional support can be obtained by consulting others on the Protégé/OWL news group.

We have primarily used the RACER[2] reasoner because it is connected to the Protégé tool. RACER can be found at http://www.sts.tu-harburg.de/~r.f.moeller/racer . RACER is a Semantic

[2] Renamed ABox and Concept Expression Reasoner

Web inference engine that is used for reasoning, queries and it supports publish and subscribe capabilities for the knowledge bases.

The Air Tasking Order Ontology

For the most part, as we implemented our ATO ontology, we tried to follow the knowledge engineering process steps defined above.

1. Identify the task

We identified the task of building ontologies for the ATO. The ATO is the formal document that the military uses to assign aircraft to missions. The ATO is very rich in classes and relationships and serves as a good demonstration for potential C2 applications. We decided that the first reasoning the ATO ontology would perform was to check the consistency of instances against the ontology constraints and to find potential missions to reassign to engage time-sensitive targets. An example of the consistency checking is the flagging of an aircraft carrying an inappropriate configuration (weapon). For our ontology experiment, we included constraint rules that state that bombers can only carry air-to-ground weapons, fighters can only carry air-to-air weapons and fighter-bombers can carry both. We also included only ground time sensitive targets.

2. Assemble the relevant knowledge (knowledge acquisition)

We served as our own domain experts for most of our knowledge acquisition. We have over twenty years of combined experience with air campaign planning and are intimately familiar with most of the ATO concepts. We did use a consultant, C3I Associates Inc., for their expertise in air campaign planning to confirm our design was correct.

We used ArgoUML, a very good, free Unified Modeling Language (UML) tool from http://argouml.tigris.org to extract the domain knowledge and model the ontologies. The ontologies are based on the standard NATO/US Message Text Format (USMTF), XML ATO message and the XML Schema document that defines the correct syntax and allowable fields for the ATO. We extracted most of our ontologies out of an actual ATO text document and the XML Schema to construct the taxonomy of classes and data relationships. The ATO message and schema do not contain object property relationships; they are implied in the document tree structure and we derived them from our own domain experience with the meaning of the tags in the document. We started with the basic ATO message and then added additional classes and properties to it to support the knowledge base. A discussion of several of the UML designs is covered in section 4.

3. Decide on a vocabulary of predicates, functions and constants

We decided to use the Web Ontology Language as the vocabulary markup language to develop our ontology and knowledge base. We made this decision primarily because OWL has become an official World Wide Web Consortium (W3C) recommendation. OWL is rapidly becoming the accepted standard language for the Semantic Web. Protégé automatically generates OWL code from the encoded graphical knowledge base.

4. Encode general knowledge about the domain

We used Stanford's Protégé ontology building tool with the OWL plug-in to code our ontologies and knowledge base. Protégé has an intuitive hierarchy-based drag and drop expandable interface to build the classes. Protégé also permits defining data properties and object properties commonly referred to as *relationships*.

Figure 1 shows the geographic location in latitude and longitude class in the Unified Modeling Language class diagrams. In the top line is the class name "Geographic LocationLatLong" and in the middle section are the *attributes* (variables) of the class. The data variables of the Latitudinal Hemisphere are constrained to be a character either N (North) or S (South). The bottom rectangle of a class would be methods or functions for a class. Ontologies focus on relationships rather than the object decomposition into classes and methods that operate on the classes in object-oriented languages like Java, C# and C++.

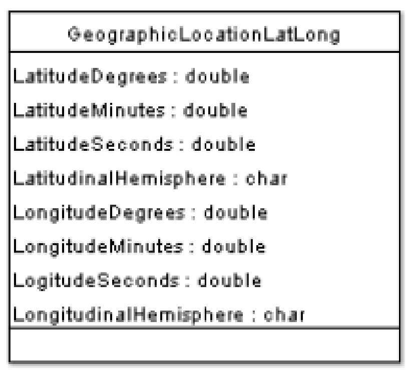

Figure 1 GeographicLocationLatLong UML Class

The most obvious taxonomy in our ATO ontology is shown in Figure 2; the decomposition of aircraft. The decomposition starts by classifying the aircraft into combat, refueling and cargo aircraft classes. The combat aircraft are further broken down into fighters, bombers and fighter-bombers. The taxonomy becomes an ontology when the KC135/10 aircraft are subclasses of both the cargo and refueling aircraft classes. This is referred to as *multiple inheritance*, which deviates from the tree structure of a taxonomy, making it an ontology. The triangle arrows are "*is-a*" relationships. A bomber is a subclass of combat aircraft.

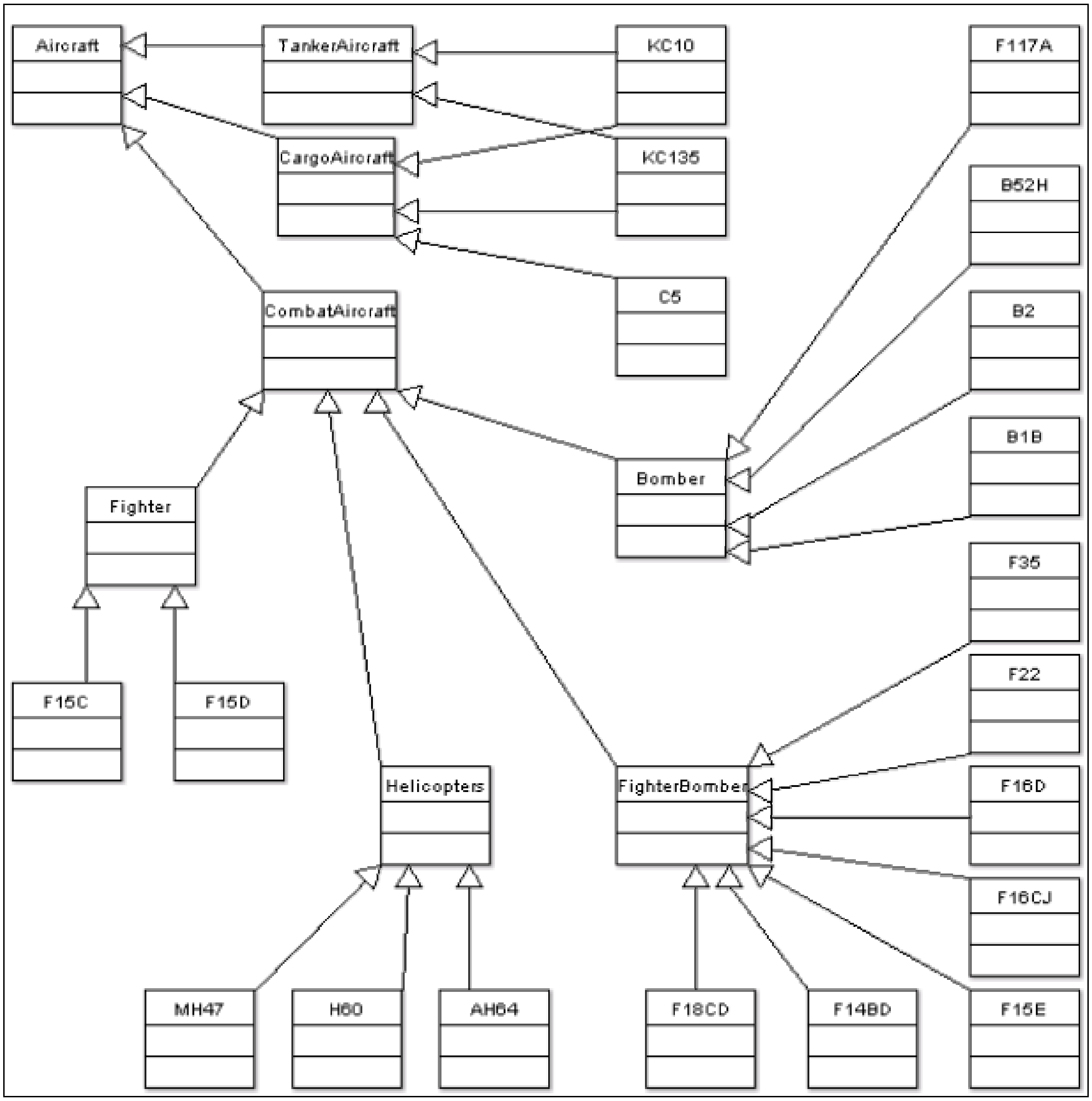

Figure 2 Aircraft Ontology

Figure 3 shows the aircraft mission ontology. All of the attributes of the classes are not shown in order for the diagram to fit on the page. The Aircraft Mission class has five relationships. Two new types of relationships are added in this diagram. The plain arrow is the relationship "*has-a*" indicating the aircraft mission has a location and also has a mission type. The parallelogram is the *composition* relationship in which missions are composed of aircraft and air objectives, and packages are composed of missions. Note that the mission ontology uses the aircraft and aircraft configuration load (weapons) ontologies. The numbers and the *n* on the relationships represent how many of each classes can be related. A mission can have 1 to *n* aircraft.

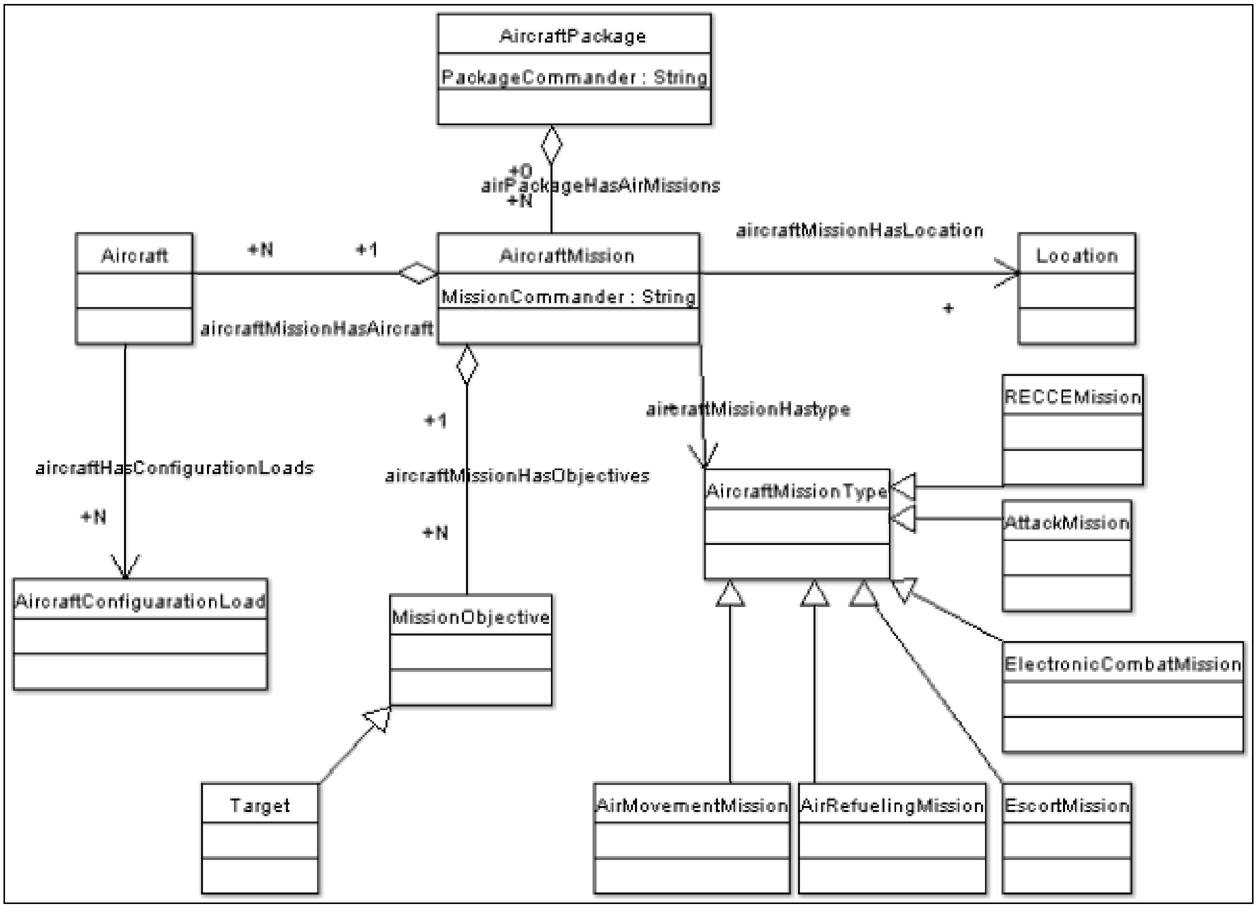

Figure 3 Aircraft Mission Ontology

Figure 4 shows the aircraft ontology in Protégé. The indentations of the classes on the left indicate subclasses. Arrows facing right indicate there are more subclasses under a class. Note that KC10 and KC135 are under both the cargo and refueling aircraft classes indicating inheritance of both.

In Figure 4 we should point out that it would seem reasonable that fighter-bombers should be subclasses of both fighter and bomber. The reason we did not do this is that fighters have a constraint that they can't carry air-to-ground weapons and the fighter-bomber can, and bombers can only carry air-ground weapons. Protégé does not allow overriding these inherited constraints in the fighter-bomber class the way Java allows overriding of methods.

5. Encode a description of the specific problem instance

Figure 5 shows an example of a specific instance class F15C_1Class in the knowledge base. It also shows a constraint violation picked up by the reasoner that the F15C_1 fighter individual, circled in red at the bottom, showing it is improperly carrying an air-to-ground guided bomb unit GBU12. In Figure 5, also note the p2: in front of the classes is the ontology abbreviation (XML namespace) in OWL for the imported aircraft ontology.

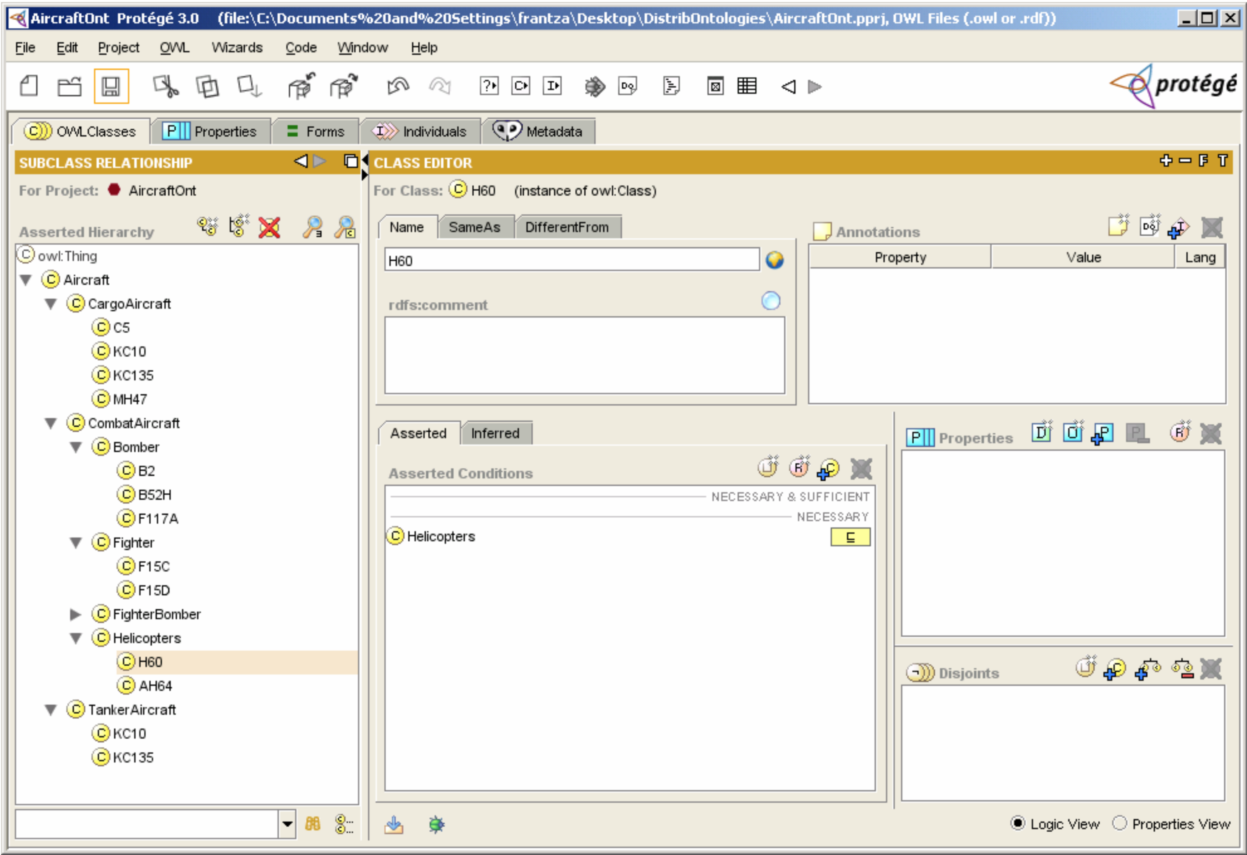

Figure 4 Aircraft Ontology in Protégé

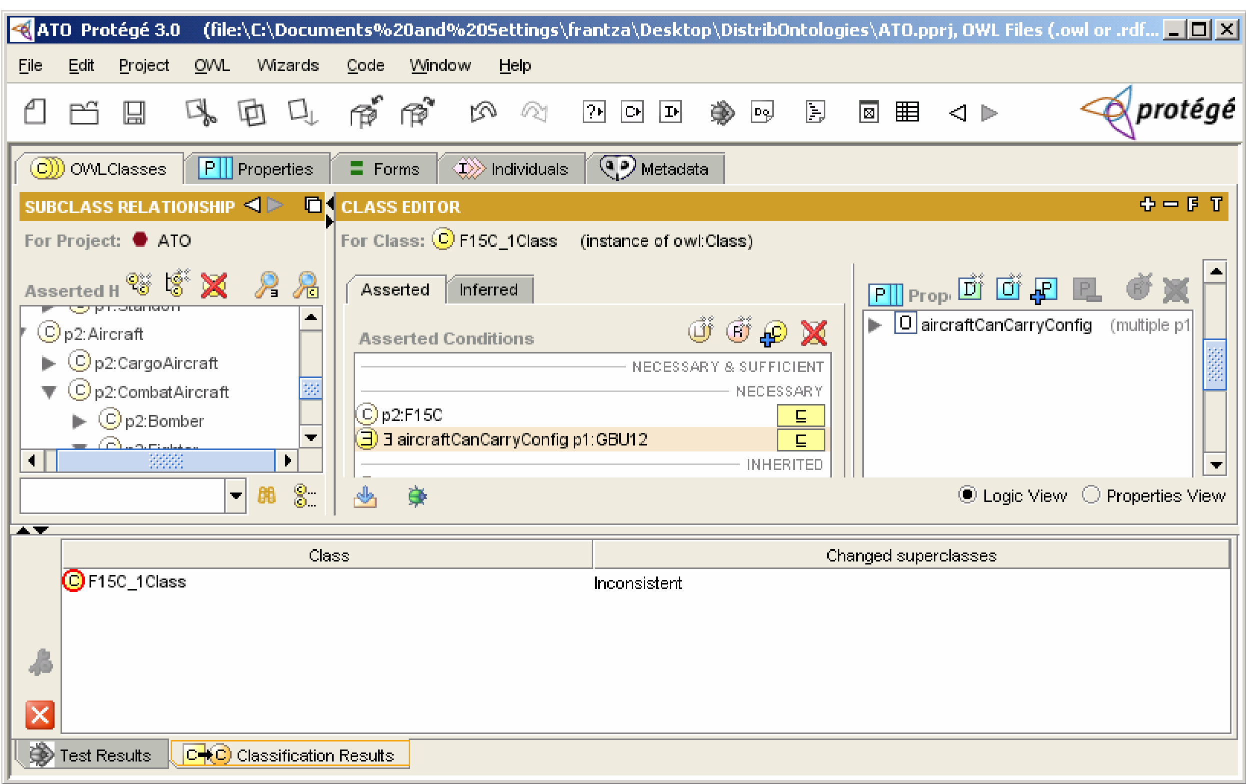

Figure 5 Constraint Violation

6. Pose queries to the inference procedures and get answers

Figure 6 shows the results of running the RACER reasoning engine on the ATO knowledge base in the bottom pane where new facts discovered from reasoning are shown. There are three time sensitive targets in our knowledge base; an SA20 (very long range surface to air missile), an SA20 near a Mosque and a command post. There are also three air missions named AL, Beyerle[3] and Milvio. In this case, all three missions can engage the first SA20 and the command post.

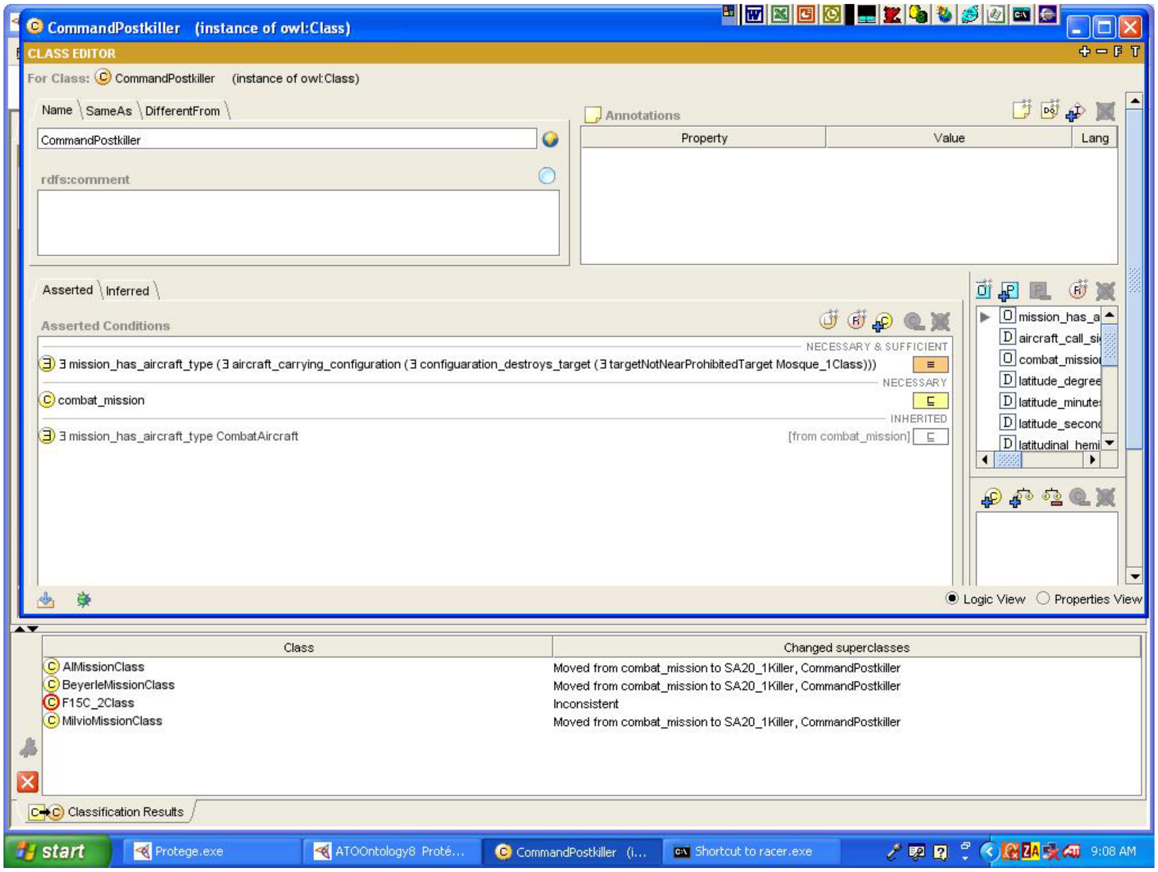

Figure 6 Reasoning on the Knowledge Base

When we add a popup target such as an SA20 to the knowledge base, we use a Java application to determine if there are any air missions that can reach the target within the time constraints. The Java application uses the latitude and longitude locations of the target and the combat missions and their speed capability to determine which missions can reach a target in the required time window. It also calculates the distance of the targets to protected assets such as hospitals. The Java application then updates the knowledge base to reflect those potential missions and targets near prohibited assets. The RACER reasoner then uses the knowledge base to determine which of the potential divertible missions has the aircraft with the correct weapons to destroy the target. The second SA20 is near a protected asset, so is not matched with any missions.

[3] John Beyerle is our subject matter expert from C3I Associates, Inc.

In the center of Figure 6 is the rule that determines if missions can engage a command post. The rule in English says that a mission can engage the command post if the mission has aircraft with the correct weapons to destroy the command post, and the command post is not near the protected Mosque. The missions' knowledge base has individual missions. The missions have individual aircraft from the aircraft knowledge base. The aircraft carry weapons from the configuration knowledge base. The target knowledge base contains the individual targets with rules of what weapon can destroy them. The reasoning takes roughly three seconds for our small problem.

7. Debug the knowledge base

Originally, when we started the ATO ontology it was fairly small and manageable. As we added to and fixed the ontology, it became more complex and less manageable. When it became too unwieldy, we decomposed the ontology into four separate smaller ontologies; the aircraft, target, mission and configurations (air weapons) ontologies. This had several advantages; it made the ontologies more specific for understanding the smaller knowledge domains and it also made them more manageable and reusable for other applications. Consistency rules are then added to the ontology relating the four ontologies. Protégé can import distributed ontologies from Web pages, giving a good demonstration of the collaborative capability of the Semantic Web.

Potential Future Work

We are currently in the process of improving our ATO knowledge base by importing an independently developed Effects-Based Operations (EBO) ontology. This will add the ability for the computer to help the decision maker to determine which of the potential missions to be reassigned would have the least impact, based on the original desired effects of the mission in the overall air campaign. We are also looking at the potential of developing an Operational Net Analysis (ONA) ontology to use the computer to help find centers of gravity. We would like to scale to much larger military sized applications to stress Protégé and RACER.

Conclusions

We have developed an ontology and knowledge base for the ATO that does some fairly sophisticated reasoning.

The Key findings of this experiment are as follows:

a. The hardest part of the knowledge engineering process is designing a good ontology before implementing it with Protégé.

b. Working with Protégé to generate OWL is much easier and faster than conventional programming. Once you have a design, implementing it in Protégé only takes hours.

c. The knowledge bases are much easier to maintain and modify than conventional programs.

d. Writing reasoning rules and getting them right is fairly challenging.

e. Existing Semantic Web technologies appear ready for small to medium sized applications.

f. Overall, the ATO ontology has been an interesting and successful demonstration of applicability of Semantic Web technologies in a military domain, our original goal.

g. We really need a semantic capable Web browser with built-in reasoning to promote the rapid acceptance of the Semantic Web for both military and commercial applications.

h. As the tools continue to evolve, the Semantic Web will become a reality with great potential to realize the collaborative, computer readable Semantic Web vision and where computers will have even more promise to improve and advance C2 and commercial applications.

Acknowledgements

We thank our branch technical advisor, Daniel Fayette, who started this project, and branch supervisor, Alex Sisti, who allocated the time for us to work on it in-house. We also thank Professor Bart Selman, our mentor at Cornell University, who challenged us to advance the project to the next level. Thanks to John Beyerle and Rob Ihle, our real domain experts for verifying our design. Thanks also for the individuals who reviewed and edited the paper, Bonnie Manning, Carl DeFranco, John McNamara, Rob Ihle, Henry Simmons, Sqn Ldr Alan Holmes, Cathy and Rebecca Frantz.

References

[DACONTA] "The Semantic Web", Michael C. Daconta, Leo J. Orbst and Kevin T. Smith, Wiley Publishing, Inc., 2003, ISBN 0471432571

[LACY] "OWL: Representing Information Using the Web Ontology Language", Lee W. Lacy, Trafford Publishing, 2005, ISBN 1412034485

[RUSSELL] Artificial Intelligence a Modern Approach, Second Edition, Stuart Russell and Peter Norvig, Pages 261-266, Prentice Hall Publishing 2003 ISBN 0137903952