# Main Presentation

# Chris Lum 233 – Sprint 1403 Simulation Project

As you just saw from Bao and Kevin’s user stories, we’ve matured our back end analysis algorithms to the point of being ready for integration into the overall plug-in system. Although we are unit testing these algorithms and components, we wanted to put together an environment and framework which simulates and exercises these components as they are intended to be used within ICOMC2 and during an actual mission. This user story involves creating a simulation environment which mimics the core functionality of ICOMC2 that our systems will rely on in order perform rapid verification and validation of our back end algorithms in an ICOMC2 agnostic fashion.

To show how the simulator functions, I thought I would present a high level overview of some of the modules in our collision awareness system. To start, one of the primary modules in our system is what we are calling the entity manager. This is an entity which is in charge of maintaining a list of entities that our system cares about. This could be Insitu aircraft, general aviation traffic, restricted airspaces, or anything else that the operator wants to keep track of during a mission. Effectively this acts as a database which can be used to hand out information to other modules for analysis.

For example, this information could be used by the forward state estimation module that Kevin and Bao have developed. As they mentioned, this module provides predictive aspects to our system by projecting the vehicles forward in time to give operators an idea of potential future conflicts.

In addition to the forward state estimators, we have instantaneous separation notification which provides operators with warnings of traffic which are currently violating their airspace. This module could also be driven by data from the entity manager.

So as you can see, this modular structure allows us to develop different analysis packages and quickly integrate them into our system since each module acts in a somewhat encapsulated manner.

Now this is the vision for how the plugin will operate but of course, the plugin will be integrated into ICOMC2 and be driven by data that it reads from various object models within ICOMC2. In essence, our plugin will read information such as vehicle positions, velocities, routes, etc. from ICOMC2. Once it has this information, it can run the appropriate analysis algorithms and in effect, “write” or display information to the user in the form of visual cues or audio alarms. As you can imagine, this requires ICOMC2 to drive most of this but in the absence of ICOMC2, we wanted to be able to still test our systems in a programmatic fashion, so we developed a simulation engine to replace some of the core functionality of ICOMC2 that our system depends on. Furthermore, we wanted a way to visualize the data and results from our plug-in so we have our simulator output information compatible with both Google Earth and Unity for visualization purposes.