Abstract*—Recent research has shown that wilderness search and rescue (WiSAR) can be aided through the use of unmanned arial systems (UASs). A single UAS, however, requires several human operators to manage the interface between the UAS vehicle and the larger search and rescue efforts on the ground and in the air that are coordinated by the central command center. For UASs to scale to real-world wilderness search and rescue scenarios, it is important to reduce operator workload and mitigate the effects of stress and fatigue through effective distributed control and augmented autonomy. A primary challenge in any effort to understand distributed control is in effectively modeling the various roles in the system from the humans, to the GUI, to the actual UAS in the physical environment. This paper discusses a Java model that explicitly formalizes the individual roles of the WiSAR UAS that can be model checked by Java Pathfinder to establish its intended behavior. The model is the basis for research on human machine interfaces to support combined human roles that reduce operator workload. In essence, by modeling each individual role in WiSAR, it is possible to then perform role fusion and show that the new UAS with combined roles, increased autonomy, and new interfaces is a correct implementation of the original system. The experience of this modeling activity suggests that modeling WiSAR or any system will be at least as hard as any solution to distributed control or role fusion.

***Index Terms*—Keywords goes here.**

I. Introduction

Problem Statement: UASs require several human operators to monitor and administer.

II. Related Work

Creating digital models of the real world is important to many disciplines. From video games creators to NASA researchers, more people are obsessed with finding, producing, and examining these digital models in order to better understand and verify real world events. Brahms is a robust modeling language that involves agents, geographies, and objects. These can also be thought of as the people involved, their environment, and the objects or tools they have to work with.

NASA Ames Research Center is using Brahms to model interactions between operators and their aerial equipment. These complex models have given new understanding to both the instances studied and the language itself. In their study of the Uberlingen collision the model, produced using the Brahms language, Neha Rungta and her colleagues were able to correctly predict the collision. Such a model could have forewarned the air traffic controller of the collision.

WiSAR, Wilderness Search and Rescue, is primarily concerned with the finding people who have become lost in rugged terrain. Research has shown that UAVs, unmanned aerial vehicles, facilitate the work this group is doing. In principle the vehicle is used to search for victims in areas that would be difficult for ground teams to reach. Michael Goodrich and his colleagues tested the effectiveness of these types of operations. They found that altitude and video clarity determined the success of the mission. Furthermore they supposed that these factors could be enhanced if the roles of the UAV operator and video operator were combined.

III. Results

We chose to use JPF, also known as Java Pathfinder, because it can explore all possible paths our simulation could have taken. In other words JPF would check all possible combinations of agent inputs to help us find errors. This tool was an essential part of our verification process. Using unhandled assertion statements in the code enhanced JPFs ability to find instances when our code wasnt acting in the intended way. JPF did its job. Right away it found multiple circumstances that would lead to an infinite loop. One of these loops involved an extremely improbable sequence of successive method calls that would have been nearly impossible to otherwise test for.

Our model consisted of multiple state machines that took a non-deterministic amount of time to form outputs. We choses this implementation because no human would take exactly the same amount of time to do the same thing every time they did it. The events also non-deterministically chose whether or not to occur at any time during a given time frame. This allowed for a more realistic test of the system since events could happen at any time. These two factors, while necessary for the realism of the simulator, would have been incredibly difficult to test without a tool like JPF. Only JPF could have found all possible paths our simulator could have taken. Because of the incredible amount of non-determinism it took a long time for JPF to traverse every path. The time waited was only a small impediment in light of the amount of work it saved us.

Creation of the model was not easy. The intricacies and nuances of the simulator made each class difficult to write. It was, however, easier than learning a new language, where many more intricacies and nuances would have to be overcome before modeling could begin. Furthermore, we can reuse much of what we created to perform verification of future models, and since we used Java more people can understand our work by simple examination. We aimed for a high level of abstraction. We didnt want to simulate keyboard inputs or specific mouse clicks, but instead chose roles that significantly impacted the likelihood of positive sightings. This level of abstraction was achieved by definition of specific inputs and outputs for each role. The basic initialization of the search was the easiest part to model. It didnt take very long to get everything sorted and the UAV into simulated air. This involved an event that began the search as well as giving each role the ability to process the commands it received to begin the search. Modeling the sub roles of the UAV was difficult. It took a while before we could simulate flights that included crashes, video anomalies, and new search areas. Most of these difficulties came from the complexity each new event gave to the whole system. Simply tracing the expected flow of data between the agents was a huge task.

IV. Conclusion

The conclusion goes here.

Acknowledgment

The authors would like to thank Neha Rungta of NASA Ames Intelligent Systems Division for her help with JPF and

Brahms.

References

[1] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.
[2] http://www.dagstuhl.de/Materials/Files/07/07122/07122.SierhuisMaarten.Other.pdf
[3] Neha's Article
[4] http://onlinelibrary.wiley.com/doi/10.1002/rob.20226/pdf