AFRL-VA-WP-TP-2004-310

IMPLEMENTATION OF HIGH LEVEL ARCHITECTURE INTO THE MULTIUAV RESEARCH SOFTWARE

Brian Stolarik
Bill Niland

FEBRUARY 2004

Approved for public release; distribution is unlimited.

AIR VEHICLES DIRECTORATE
AIR FORCE RESEARCH LABORATORY
AIR FORCE MATERIEL COMMAND
WRIGHT-PATTERSON AIR FORCE BASE, OH 45433-7542

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YY)*
February 2004

2. REPORT TYPE
Conference Paper Preprint

3. DATES COVERED *(From - To)*

4. TITLE AND SUBTITLE
IMPLEMENTATION OF HIGH LEVEL ARCHITECTURE INTO THE MULTIUAV RESEARCH SOFTWARE

5a. CONTRACT NUMBER
F33615-01-D-3105

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER
62201F

6. AUTHOR(S)
Brian Stolarik
Bill Niland

5d. PROJECT NUMBER
2403

5e. TASK NUMBER
01

5f. WORK UNIT NUMBER
62

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Institute for Scientific Research, Inc.
Fairmont, WV 26554

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Air Vehicles Directorate
Air Force Research Laboratory
Air Force Materiel Command
Wright-Patterson Air Force Base, OH 45433-7542

10. SPONSORING/MONITORING AGENCY ACRONYM(S)
AFRL/VACA

11. SPONSORING/MONITORING AGENCY REPORT NUMBER(S)
AFRL-VA-WP-TP-2004-310

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES
Conference paper to be presented at the American Control Conference, Boston, MA, June 30-July 2, 2004. This work, resulting from Department of Air Force contract number F33615-01-D-3105, has been submitted for publication in the Proceedings of the 2004 American Control Conference. If published, the American Automatic Control Council (AACC) may assert copyright. If so, the United States has for itself and others acting on its behalf an unlimited, nonexclusive, irrevocable, paid-up, royalty-free worldwide license to use for its purposes.

14. ABSTRACT
This paper describes the implementation of the Department of Defense's High Level Architecture (HLA) into the U.S. Air Force Research Laboratory's multiple unmanned aerospace vehicles research software (MultiUAV). MultiUAV allows simulations of multiple UAV's cooperating as a team to accomplish strongly coupled tasks. Since it operates in MathWorks' Simulink simulation environment, the HLA was integrated through a series of S-functions written in C++. The addition of the HLA into MultiUAV enables more realistic inter-vehicular communication modeling to include noise, latency, and data dropouts. It also enables the distribution of MultiUAV across a network of computers. Two mission scenarios were simulated both with and without the HLA. Identical behaviors in all four simulations show a successful implementation of the HLA into the MultiUAV research tool.

15. SUBJECT TERMS
High Level Architecture, simulation, cooperative control

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT:	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON (Monitor)
a. REPORT	b. ABSTRACT	c. THIS PAGE			Corey Schumacher
Unclassified	Unclassified	Unclassified	SAR	12	19b. TELEPHONE NUMBER *(Include Area Code)* (937) 255-8494

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. Z39-18

Implementation of High Level Architecture into the MultiUAV Research Software

Brian Stolarik and Bill Niland
Institute for Scientific Research, Inc.
Fairmont, West Virginia 26554
bstolarik@isr.us, wniland@isr.us

Abstract

This paper describes the implementation of the Department of Defense's High Level Architecture (HLA) into the U.S. Air Force Research Laboratory's multiple unmanned aerospace vehicles research software (MultiUAV). MultiUAV allows simulations of multiple UAV's cooperating as a team to accomplish strongly coupled tasks. Since it operates in MathWorks' Simulink simulation environment, the HLA was integrated through a series of S-functions written in C++. The addition of the HLA into MultiUAV enables more realistic inter-vehicular communication modeling to include noise, latency, and data dropouts. It also enables the distribution of MultiUAV across a network of computers. Two mission scenarios were simulated both with and without the HLA. Identical behaviors in all four simulations show a successful implementation of the HLA into the MultiUAV research tool.

Introduction

Future generations of UAV's will be able to autonomously cooperate with either manned or other unmanned aerial vehicles to accomplish strongly coupled tasks. Such cooperative tasks envisioned by military planners include combat intelligence, surveillance, and reconnaissance, aerial-based communication nodes, suppression of enemy air defense, identification and destruction of time critical targets, close air support, cooperative search, and persistent denial[1,2,3].

Researchers at the U.S. Air Force Research Laboratory have written software to enable their investigation into such UAV teaming arrangements. MultiUAV[4,5] is a MathWorks' MATLAB/Simulink based simulation program that allows cooperative algorithms to be easily tested in a simulated mission. Two limitations of early versions of the software were a lack of realistic inter-vehicular communication modeling and the inability to distribute the software across a computer network. To eliminate these issues we integrated the Department of Defense's networking standard High Level Architecture (HLA) into the software.

Brief overviews of the HLA and MultiUAV are given before a discussion of the work completed to merge the two. The simulation results of two missions with increasing complexity are presented in the results section. Finally, we discuss some conclusions and future work.

The High Level Architecture

In 1996 the U.S. Department of Defense released the High Level Architecture as its standard for communication between distributed simulations. In 2000, the Institute of Electronics and Electrical Engineers (IEEE) adopted the HLA as IEEE Standard 1516. A brief overview of the HLA will be given below but interested readers are urged to consult one of the published references[6,7,8] for a more detailed explanation.

The HLA is a software architecture designed to allow simulations to be interoperable, distributed and reusable. Such capabilities provide the simulation designer the ability to interact their HLA compliant simulation with any other HLA compliant simulation. These individual simulations, or federates, can be mixed and matched to create a master simulation, or federation. The HLA is additionally flexible in that it does not require simulations to be written in a specific programming language. In fact, separate components of the simulation can be written in different languages and still be interoperable.

An HLA federation is a set of federates defined by the HLA interface specifications that interact by exchanging data. This data exchange is accomplished through an interface common to all federates on the same network known as the Run Time Infrastructure (RTI). The RTI provides a set of services that a federate can use to send data to and receive data from other federates. The RTI also manages the federation.

Four of the six services provided by the RTI to a federate were used in this research:

1) Federation management services provide a set of functions a federate can call to create, join, resign, and destroy a federation.

ASC - 04 - 0145

2) Object management services allow a federate to register and discover objects within the federate.
3) Declaration management services specify what data will be provided and required by the federate during execution.
4) Time management services enable federates to send and receive time stamped data. It also allows federates to be time synchronized, an important feature for distributed simulations.

MultiUAV

The MultiUAV simulation software allows researchers to examine cooperative control algorithms in multiple UAV mission scenarios. In the latest public release of MultiUAV[9], researchers can adjust the number of vehicles (maximum of 8) and targets (maximum of 10) in the simulated mission. The goal of the mission is to find, classify and attack targets in a search area. Since the UAV's are homogeneous and modeled as a Low Cost Autonomous Attack System (LOCAAS), an attack by a vehicle terminates its existence. Thus, the number of vehicles alive during the simulation decline as targets are attacked. After an attack, another vehicle must fly over the target to do battle damage assessment. Vehicles that remain after all targets are destroyed simply continue their search mission.

The goal of the coordinated control algorithms under investigation within MultiUAV is to optimally[10], if possible and feasible, allocate the mission's tasks among the vehicles. At the beginning of the mission the vehicles start a predetermined search pattern. Once a vehicle detects a target it notifies all other vehicles, triggering a replan. Once triggered, the coordinated control algorithm, duplicated on each vehicle to maximize autonomy, calculates the next set of tasks for all known vehicles. The algorithm duplication allows each vehicle to compute the same set of tasks for all vehicles without substantial inter-vehicular communication.

Implementing the HLA into MultiUAV

Since MultiUAV was written in the MATLAB/Simulink environment, making it easily understandable and modifiable to outside researchers, the options for integrating the HLA into the software is limited to Simulink S-Functions[11]. S-Functions are compiled code, C++ in this research, which users write to extend Simulink's predefined functionality. Simulink simply passes data to and receives data from a user defined S-Function.

Figures 1 and 2 illustrate conceptually how S-Functions were used to remove the communication bus within MultiUAV. Figure 1 shows the connections of n targets and m vehicles. The simulation bus, shown in red, only passes truth information between vehicles and targets. The communication bus, shown in green, models the communication between vehicles. Again, these figures are only meant to convey a concept since the latest MultiUAV (not publicly released at the time of this writing) actually uses a common communications memory structure instead of direct block connections. The elimination of the communication bus, not the simulation bus, was the focus of this research.

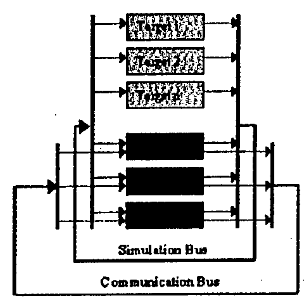

Figure 1: Original communication design

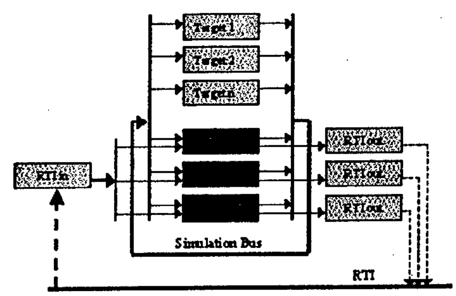

Figure 2: Modified communication design

Figure 2 illustrates the new communication design that incorporates the HLA. Notice that in the new design inter-vehicular communication is achieved through the RTI via S-Functions. Each vehicle has its own S-Function, labeled "RTI out", that sends the data received from Simulink to the RTI. Also, all vehicles share a single S-Function, labeled "RTI in", that receives the data from the RTI and makes it available to Simulink. Each S-Function is seen as a federate to the RTI. The forcing of inter-vehicular communication to pass through the RTI is a first step toward making MultiUAV distributable across a network. Additionally, other federates can connect to the RTI outside of Simulink or MultiUAV, enabling them to receive any data the vehicles send. Such a federate might be a passive data logger, a bandwidth filter, or a communication model to allow researchers to artificially inject noise, dropouts, or latencies.

The implementation of Figure 2 occurred at the top levels of the MultiUAV Simulink model. The original vehicles subsystem can be seen in Figure 3. Only the first two vehicles out of a possible eight are shown for brevity.

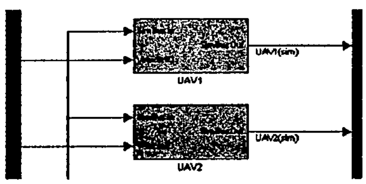

Figure 3: Original Vehicle Subsystems

Since each S-Function is seen by the RTI as a federate, each requires unique parameters to operate, e.g. a federate name. For this reason and ease of use, each vehicle's S-Function was placed outside of the vehicle model. A small modification was made to the vehicle model to allow passing of the communication outside of the vehicle and into the S-Function. Figure 4 illustrates the vehicles tied to the HLA interface S-Functions, which are shown in green. The additional input to the vehicle block is simply a flag that allows the user to toggle between the original communication system and our HLA system for testing purposes.

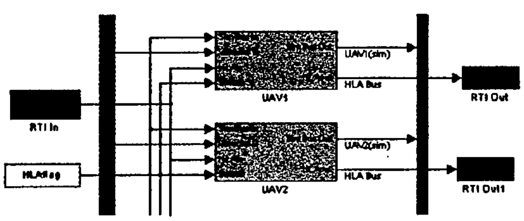

Figure 4: Modified Vehicle Subsystems

Messages in the original MultiUAV were triggered and written to memory only when pertinent for other vehicles. Figure 5 shows the original message passing system found in each vehicle.

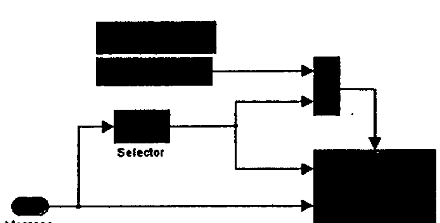

Figure 5: Original Message Passing Subsystem

This message triggering system worked very well for an HLA implementation. Extra output ports were created, allowing for the message trigger and the data to be passed outside the vehicle block. The triggering concept keeps network bandwidth to a minimum. The modified message passing system can be seen in Figure 6.

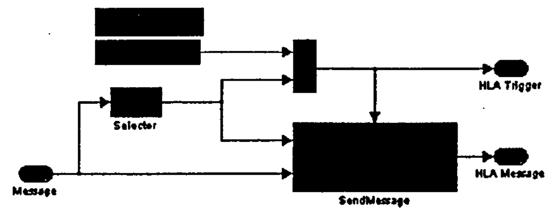

Figure 6: Modified Message Passing Subsystem

Each message passing subsystem's outputs are then multiplexed together to form a single HLA bus. Referring to Figure 4, this bus is subsequently injected into the RTIout S-Function. Once the data is received, the S-Function parses the vector, checking to see if any particular message has a trigger associated with it. Any messages with triggers are wrapped together and sent to the RTI.

The triggered data that is sent through the RTI is reflected at the RTIin S-Function. At each Simulink time-step, the S-Function invokes a *tick()* command to the RTI, yielding time for its callbacks to be executed. These callbacks dynamically create and fill a vehicle memory table, storing the data temporarily until the S-Function can read and send it to the Simulink model. A total of ten messages can be reflected at the RTIout S-Function. When a message is reflected, the data along with its timestamp are placed on individual output ports, thus 20 outputs can be found. After expanding the RTIin sub-system in Figure 4, the S-Function can be found. This is seen in Figure 7.

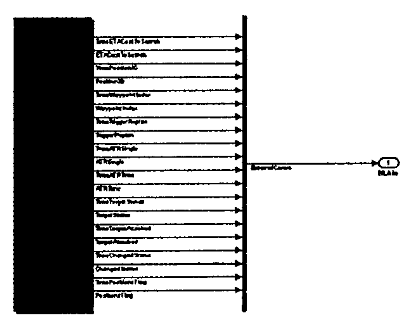

Figure 7: RTIin Subsystem

This configuration of output ports is used since each vector can change in width without effecting the simulation. The bus hierarchy is created to place no emphasis on indexing the vector to extract messages. This concept was directly modeled from the existing MultiUAV communication passing[4].

Simulation Results

Two scenarios using the same control algorithm were tested with increasing complexity to verify the HLA communication's dependability. The individual scenarios were executed twice, the first using the original communications design for data exchange while the second used the HLA. The data passed to and received from the S-Functions was then compared. The results presented are for the HLA design versus the original MultiUAV communications architecture.

Mission 1: The first mission scenario tested included three vehicles and one target. In this mission, vehicle 1 finds the target, vehicle 3 is tasked to confirm and kill the target, and vehicle 1 finishes the mission by confirming the kill. Vehicle 2 is not tasked for target 1 so it continues its original search mission.

Figure 8 shows a snapshot of this simulation using the original communications design at 62.30 seconds. Vehicle 1 has already detected the target while vehicle 3 is approaching for the kill. Figure 9 shows the same simulation using the HLA communications design.

The change in background color is intentional to distinguish between the two comparisons. This snapshot shows an exact match between the original communication and the HLA design.

Table 1 shows the event flow for the first mission scenario. The left two columns of the table document the major simulation events before HLA, while the right two columns document the events after HLA.

The matching of events and event times verifies that the cooperative control algorithm behaves identically with either communication design. Thus, the added capabilities of the HLA have been integrated into MultiUAV's communication scheme without affecting the algorithm in this simple mission scenario

Mission 2: The second mission scenario tested is significantly more complicated in that it included eight vehicles and five targets. Vehicles 1, 2, 4, 5, and 6 locate the five targets to begin the simulation.

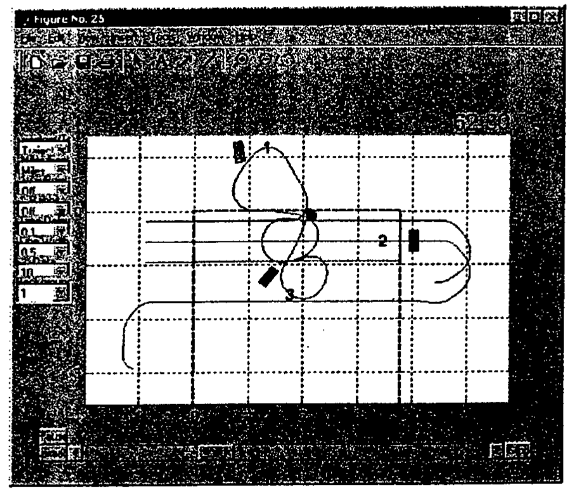

Figure 8: Mission Scenario 1 Without HLA

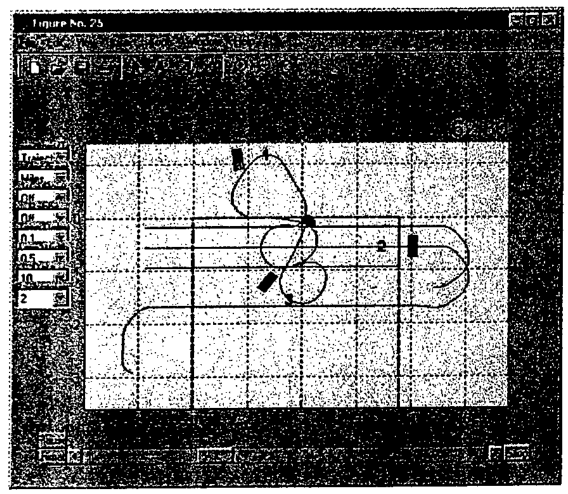

Figure 9: Mission Scenario 1 With HLA

Original Communication Design		HLA Communication Design	
Time	Event	Time	Event
33.52	UAV1 Finds Target1	33.52	UAV1 Finds Target1
35.90	Target1 Detected	35.90	Target1 Detected
75.96	UAV3 Finds Target1	75.96	UAV3 Finds Target1
78.20	Target1 Classified	78.20	Target1 Classified
84.60	Target1 Killed	84.60	Target1 Killed
90.12	UAV1 Finds Target1	90.12	UAV1 Finds Target1
92.30	Target1 Confirmed	92.30	Target1 Confirmed

Table 1: Even Flow Chart for Scenario 1

Vehicles 1, 5, 6, 7, and 8 carry out the target execution. The simulation is finished when vehicles 2, 3, and 4 perform the kill verification. The simulation before HLA introduction is highlighted in Figure 10.

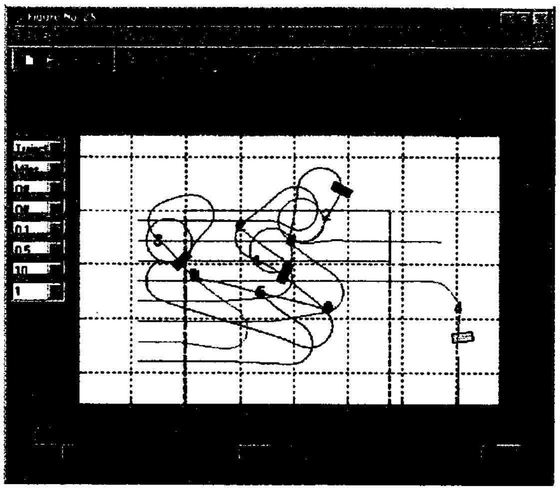

Figure 10: Mission Scenario 3 Before HLA

This snapshot is taken 88.20 seconds into the simulation. All targets have been identified and tasking has been decided upon through the algorithm.

For this last scenario, the HLA simulation in Figure 11 is identical to the original communication design simulation. All vehicles are on the same course of action, showing the algorithm is behaving the same. Table 2 shows a simulation event flow chart of the individual steps for this mission scenario. As in the first scenario, the integration of the HLA has not affected MultiUAV's algorithm performance.

Time Managed HLA

It is important to note that the above HLA results use RTI time-management services. The initial HLA integration used no time-management and yielded similar results. However, a detailed analysis of the data revealed that each message contained a small percentage of dropouts. Depending on the message, the dropout rate varied from 0.1 to 15 percent.

Using the provided routines in the RTI timing queues, the data dropout was corrected and the messages passed using either communication design were identical. This will prove to be useful when the simulation is distributed across multiple Simulink models.

Though MultiUAV runs dependable with time management, it runs six to seven times slower due to the synchronization requests between each vehicle at every time-step. Finding ways to speed up the RTI will be the focus of future work.

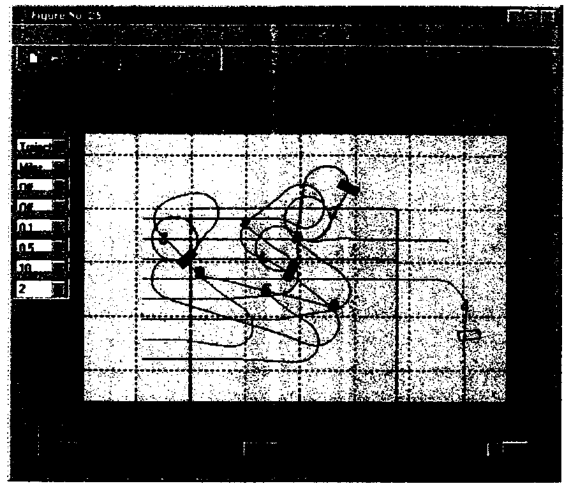

Figure 11: Mission Scenario 3 After HLA

Original Communication Design		HLA Communication Design	
Time	Event	Time	Event
5.56	UAV4 Finds Target4	5.56	UAV4 Finds Target4
7.90	Target4 Detected	7.90	Target4 Detected
17.18	UAV1 Finds Target2	17.18	UAV1 Finds Target2
19.50	Target2 Detected	19.50	Target2 Detected
22.60	UAV5 Finds Target5	22.60	UAV5 Finds Target5
24.90	Target5 Detected	24.90	Target5 Detected
30.70	UAV2 Finds Target1	30.70	UAV2 Finds Target1
33.10	Target1 Detected	33.10	Target1 Detected
39.56	UAV1 Finds Target3	39.56	UAV1 Finds Target3
41.74	UAV7 Finds Target4	41.74	UAV7 Finds Target4
42.50	Target3 Detected	42.50	Target3 Detected
50.40	Target4 Killed	50.40	Target4 Killed
53.20	UAV5 Finds Target2	53.20	UAV5 Finds Target2
58.62	UAV8 Finds Target5	58.62	UAV8 Finds Target5
61.90	Target2 Killed	61.90	Target2 Killed
66.84	UAV6 Finds Target1	66.84	UAV6 Finds Target1
67.30	Target5 Killed	67.30	Target5 Killed
68.32	UAV1 Finds Target2	68.32	UAV1 Finds Target2
70.60	Target2 Confirmed	70.60	Target2 Confirmed
75.50	Target1 Killed	75.50	Target1 Killed
92.06	UAV3 Finds Target4	92.06	UAV3 Finds Target4
94.20	Target4 Confirmed	94.20	Target4 Confirmed
100.20	UAV1 Finds Target3	100.20	UAV1 Finds Target3
108.50	UAV3 Finds Target5	108.50	UAV3 Finds Target5
108.90	Target3 Killed	108.90	Target3 Killed
110.70	Target5 Confirmed	110.70	Target5 Confirmed
117.60	UAV2 Finds Target1	117.60	UAV2 Finds Target1
119.80	Target1 Confirmed	119.80	Target1 Confirmed
125.60	UAV3 Finds Target3	125.60	UAV3 Finds Target3
127.80	Target3 Confirmed	127.80	Target3 Confirmed

Table 2: Event Flow Chart for Scenario 2

Conclusions and Future Work

The paper described the successful replacement of the original communication design within MultiUAV with an HLA communication design. The software was tested in two mission scenarios designed to challenge the new design at extreme scenarios and stress the RTI. In both scenarios the modified MultiUAV performed identically to the original MultiUAV. Therefore, the added capabilities of the HLA have been integrated into MultiUAV's communication scheme without affecting the performance of the coordinated control algorithm.

Future work on MultiUAV will include the elimination of the simulation bus, the addition of a central control federate that will act as a data logger, bandwidth filter, and a communication model, and the formatting of messages into Link 16 to allow MultiUAV to communicate with the Joint Integrated Mission Model.

Acknowledgments

The U.S. Air Force Research Laboratory Air Vehicles Directorate and General Dynamics Advanced Information Systems supported this research. Additionally, the authors thank Steve Rasmussen of General Dynamics for his assistance.

References

1. S. Banda, "Future Directions in Control for Unmanned Air Vehicles", *AFOSR Workshop on Future Directions in Control*, Arlington, VA, April 2002.

2. McLain, T., "Coordinated Control of Unmanned Air Vehicles", Technical report to the Air Force Research Laboratory, Air Vehicles Directorate, September 1999.

3. C. Schulz, D. Jacques and M. Pachter, "Cooperative Control Simulation Validation Using Applied Probability Theory", *Proceedings of the 2003 AIAA Guidance, Navigation and Control Conference*, Austin, TX, AIAA 2003-5587, August 2003.

4. S. Rasmussen, J. Mitchell, C. Schulz, C, Schumacher, and P. Chandler, "A Multiple UAV Simulation For Researchers", *Proceedings of the 2003 AIAA Guidance, Navigation and Control Conference*, Austin, TX, AIAA 2003-5684, August 2003.

5. S. Rasmussen and P. Chandler, "MultiUAV: A Multiple UAV Simulation for Investigation of Cooperative Control", *Proceedings of the 2002 Winter Simulation Conference*, San Diego, CA, December 2002.

6. "RTI 1.3-Next Generation Programmer's Guide Version 5", DoD Defense Modeling and Simulation Office, May 2002.

7. J. Dahmann, R. Fujimoto and R. Weatherly, "The Department of Defense High Level Architecture", *Winter Simulation Conference*, December 1997.

8. R. Dumond and R. Little, "A Federation Object Model Flexible Federate Framework", Technical Note CMU/SEI-2003-TN-007, March 2003.

9. http://www.isr.us/Projects.asp?s=4

10. P. Chandler, M. Pachter, S. Rasmussen and C. Schumacher, "Multiple Task Assignment for a UAV Team", Proceedings of the 2002 AIAA Guidance, Navigation and Control Conference, AIAA 2002-4587, August 2002.

11. "Writing S-Functions", The MathWorks, Inc., 2003.