A Simulation Environment for Testing and Evaluating Multiple Cooperating Solar-powered AUVs

Rick J. Komerska and Steven G. Chappell
Autonomous Undersea Systems Institute
Lee, NH 03824 USA
{komerska, chappell}@ausi.org

***Abstract* - This paper describes the joining of an Autonomous Undersea Vehicle (AUV) with a pre-existing distributed simulation environment. Together, they provide the capability to test and evaluate Solar-powered AUV (SAUV) system components and multiple cooperative vehicle mission profiles before going in the water. This simulation facility allows for complete testing of SAUV onboard high-level software, including underwater networking protocol logic. The facility also has a training functionality in that top level mission planning and vehicle monitoring applications used by SAUV operators can also be tested as if they were in a field setting. Hardware components, such as radio frequency (RF) modems (typically connected to the vehicle monitoring application) as well as acoustic modems can also be tested within a systems context. The SAUV PC-104 system, running a Linux OS and the high-level software, can be tested as a bench-level component. In addition, significant portions of the standalone SAUV vehicle can be put into simulation mode, thereby allowing test of other on-board vehicle electronics and subsystems.**

Figure 1. Solar-powered AUVs.

I. INTRODUCTION

As difficult as it is designing, testing, fielding and evaluating the mission level logic for a single AUV, the problems are greatly compounded when two or more cooperating AUVs are considered. This is due to the plethora of high level interactions possible amongst the participants of any cooperative mission. The logic controlling the mission in each participant must be made robust enough to stand up in the face of missed or garbled communications, limited spatial position information, relative levels of participants' energy reserves, vehicle subsystem failures and the like. Testing multiple vehicle mission logic is particularly difficult because all vehicles should be present in the test in order to adequately evaluate each vehicle's response to its partners' actions and communications. Obviously, some manner of laboratory bound testing of multiple vehicle mission control logic prior to that first test in the water is desirable. Efforts by others [1,2,3,4] demonstrate various approaches to addressing this issue.

Particularly important is the ability to visualize the trajectories of the participants' motion in a cooperative exercise. Much of typical AUV mission control logic is concerned with getting a vehicle to a particular place at a particular time. When multiple vehicles are involved, it is important that the mission logic correctly implement the overall fleet "choreography" as desired by the operator. Real time 3D visualization of such activity is crucial for testing the mission logic prior to going into the water.

Testing mission logic is best done within the context of an operating vehicle. Pulling that logic out of the vehicle environment and attempting to install it in some manner of test harness risks removing it too far from it's originally intended runtime environment. That, in turn, increases the chances for logic failure once returned to the vehicle. Multiple vehicle mission testing, therefore, should be performed with as much of the vehicle system operating as possible.

This paper details recent work done to connect together the high level software architecture of the Solar-powered AUV (SAUV) platform [5,6] to an AUV simulation environment. Section II introduces the SAUV architecture and the simulation environment to which it was interfaced. Section III delves into the pertinent details of both systems and how they were mated. Section IV describes how the resulting simulation harness is being put into use to facilitate ongoing research and operations. Section V presents conclusions.

This research was supported in part by grant N00014-05-1-0666 from the Office of Naval Research.

II. Background

A. Solar-powered AUV (SAUV)

The Autonomous Undersea Systems Institute (AUSI), in cooperation with Falmouth Scientific, Inc. (FSI), Technology Systems, Inc. (TSI) and the Naval Undersea Warfare Center - Newport (NUWCDIVNPT) is conducting basic research and development in the deployment of multiple AUV for defense, security, and distributed underwater sensing applications. The primary AUV platform for these programs is the SAUV-II, a solar-powered autonomous underwater vehicle shown in Fig. 1. The goal of the SAUV program has been to develop an AUV system capable of autonomous underwater sampling with long endurance. Five SAUV vehicles have been built to date; one is owned and operated by NUWCDIVNPT and four by AUSI. The vehicles have been used in technology demonstrations including AUVFest 2005 (Keyport, WA) and science data collection missions in support of the Environmental Protection Agency (Greenwich Bay, RI – 2004) [7] and the Layered Organization in the Coastal Ocean (LOCO) team (Monterey Bay, CA – 2006).

The main SAUV computer is a PC-104 stack providing a 100 MHz ZFx86 CPU, 32 MB of RAM, a 192 MB disk on chip, 10 serial ports, and two Ethernet ports. The system's OS is a stripped down Slackware (a mix of 7.0 and 8.0) Linux installation. Important vehicle subsystems are: the Navigator for positioning, the Propulsion and Motion Controller (PMC) housing the motion controller, and the Energy Manager (EM) which controls the batteries and solar panels. Communication for the vehicle is via a FreeWave RF modem, a Benthos ATM-855 acoustic modem and an Iridium satellite modem. The Navigator, PMC, and EM subsystems form the real-time section of the SAUV, which is monitored and controlled by the PC-104 system.

B. Cooperative AUV Development Concept (CADCON)

The Cooperative AUV Development Concept (CADCON) effort was initiated by AUSI in 1998 and was designed to be a distributed simulation environment to support research in cooperative behaviors among a team of AUVs [8]. It employs a client-server architecture, where a Linux-based server provides the virtual underwater environment composed of simple models, including water current, temperature, salinity and solar insolation. Along with models of the physical environment, it also provides simple models for acoustic, RF and satellite communications. CADCON clients include *AUVSim*, a program which can represent one of several AUV body/control types in the virtual environment, and the *Visualizer*, a program which provides a god's eye visualization of the environment and vehicles in it. Both of these clients run on the Windows OS.

The emphasis in CADCON is to support high-level interaction among multiple heterogeneous AUVs; only first order physics is used in modeling vehicle motion and environment interaction. CADCON continues to provide a testbed environment for development of an AUV common control language [9] and underwater networking protocols [10].

III. Simulation Architecture

The simulation architecture, shown schematically in Fig. 2, is essentially based upon the CADCON client-server architecture, with modifications to the SAUV software and *AUVSim* client software to permit them to be conceptually represented as a single AUV entity in the virtual environment. In this role, *AUVSim* takes the place of the low-level SAUV Navigator, Energy Manger, and communication and subsystems (modems) while also maintaining the connection to the simulation server. *AUVSim* was also modified to permit a serial connection to an external program, such as a GUI-

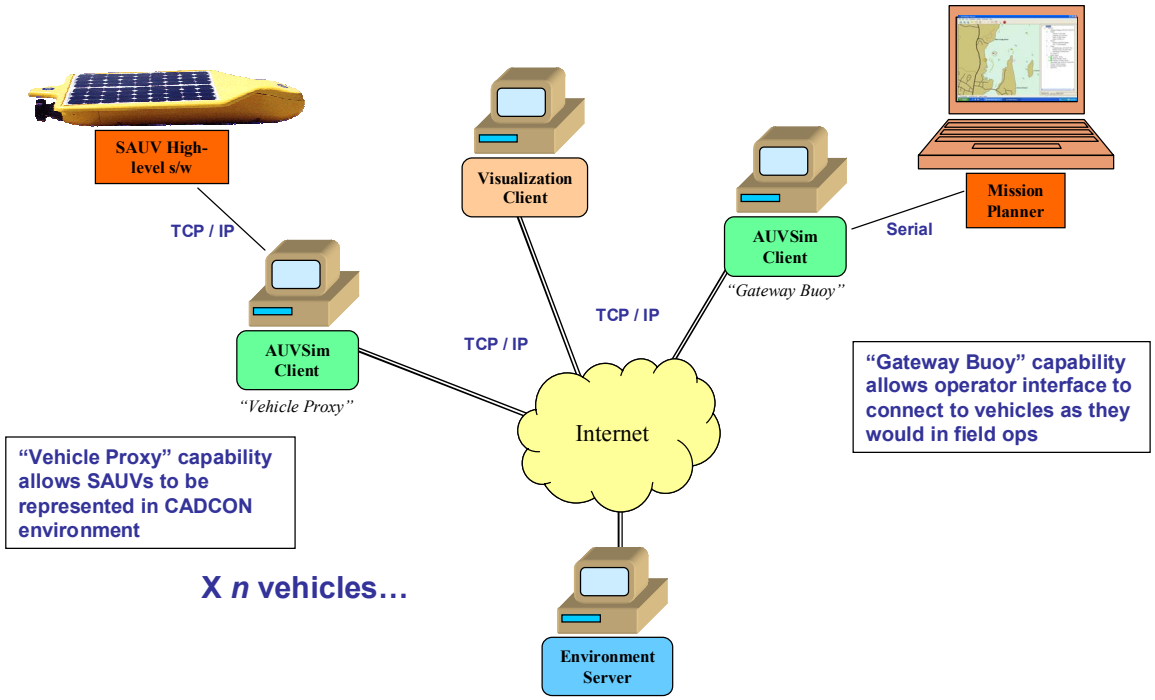

Figure 2. Components in the SAUV-CADCON simulation environment.

oriented mission planning and monitoring program. In this mode, *AUVSim* can represent a gateway buoy in the virtual environment to allow actual vehicle commands and data to be transmitted and received by an AUV operator program. On the SAUV platform, the layered control architecture was exploited in the development of a process which can divert commands and data from the actual subsystems to the *AUVSim* client, as described below.

A. SAUV Modifications

In order to connect a SAUV to the pre-existing CADCON infrastructure, some redesign and implementation work had to be performed on the SAUV software architecture. The first step was to find where in the architecture a two way connection for communication with CADCON could be made. The vehicle's hardware arrangement provided just such a location. SAUV system components are arranged as a set of real-time subsystems (and modems) all connected to a PC-104 system via serial ports. The PC-104 applications software interfaces with those serial ports through a set of special purpose "handler" processes, whose duties are to handle or service the subsystem on the other side of its serial port. Each handler process contains the expertise needed to manipulate its particular subsystem. This is shown in Fig 3.

These handlers form a kind of interfacing layer between the SAUV high level applications (*Mission Manager, Communications Manager* and *Data Manager*) and the outside world. This arrangement frees the managers' code from becoming too complex in dealing with subsystem manipulation concerns. Whereas all SAUV input and output passes through some handler process, it was decided that those handlers would be an ideal site to effect a switch between dealing with a real subsystem or a simulated one (CADCON or otherwise). Note that all communication between managers and handlers uses UDP/IP messaging.

The second step in this conversion was to modify the handlers such that, at runtime, they could be directed to either connect to their normal hardware system's serial port or to the CADCON infrastructure. This decision would be made at the time of SAUV applications level boot up. The actual control is governed through a specific configuration file (see Fig. 3) that is given to the SAUV application boot up script.

The third step in the conversion was to implement the actual connection of the various SAUV handler processes to CADCON. Since CADCON communications occur as a unitized packet (all data values ride in a single packet as opposed to individual packets for each and every simulated device), one process was implemented onboard the SAUV to make that connection. This process, called *sauvSim*, serves as the collection point on the SAUV for the handlers as well as the connection point with CADCON.

The result is shown in Fig. 4, where *sauvSim* has been inserted between the SAUV handler processes and the CADCON infrastructure. In this arrangement, the various handlers receive their input from *sauvSim* and they send their output to *sauvSim*. The *sauvSim* process handles the IO

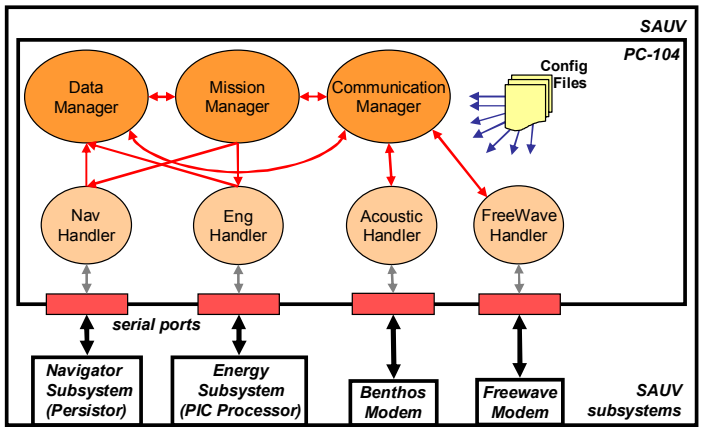

Figure 3. SAUV high-level processes (field configuration).

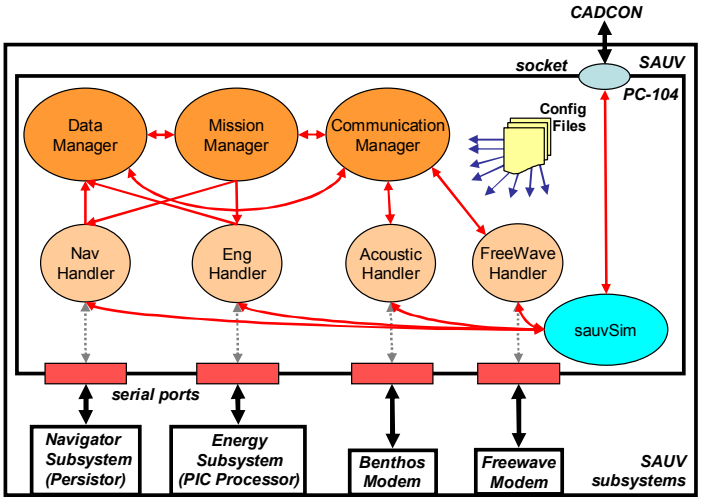

Figure 4. SAUV high-level processes (simulation configuration).

mapping between the SAUV subsystem and the CADCON *AUVSim* client using TCP/IP and honors each side's timing requirements. The end result is the SAUV participating in a CADCON mediated scenario, complete with communications.

The diagram shows dotted lines between the handlers and their subsystems and solid lines connecting them with *sauvSim*. This indicates a situation of full connection to CADCON; all handlers are dealing with simulated subsystems. It is possible to tune the configuration file so as to individually select handler to CADCON connectivity. For example, in order to test the acoustic handler's ability to correctly manipulate a real Benthos modem, the configuration file can be set such that only the acoustic handler connects with its hardware, while all the others connect through *sauvSim* to CADCON.

The *sauvSim* process was also developed to provide a "self-simulation" capability, whereby *sauvSim* can completely simulate the SAUV subsystems without the assistance of the CADCON infrastructure. This enables single SAUV testing, albeit with less realistic external data feeds.

B. CADCON Modifications

The *AUVSim* client was the only program in the CADCON suite of applications which required modification. To support

this effort, two important modifications were made. With these changes in place, *AUVSim* could be connected to the simulation environment either alone (as previously done) or in concert with a running SAUV high-level process suite.

The first key addition was to include the ability to connect *AUVSim* via a TCP/IP network link to a running *sauvSim* process. This process could be running on an actual SAUV, or on a dedicated desktop or virtual machine running the SAUV-flavored Linux OS. Standard CADCON packet structures are used to send and receive initialization, command and vehicle state information between *AUVSim* and *sauvSim*. Inter-vehicle and operator-vehicle commands and data (such as those which would occur in actual field operations) are also routed through this link.

The second modification was to provide an ability to emulate an RF/acomms gateway buoy. Gateway buoys, which form a transparent bridge between RF and acoustic modems, are used by the SAUV team and others as a convenient way to communicate with undersea AUVs from a shore station. For this effort, an interface was implemented in *AUVSim* to allow the user to specify a serial port in which to connect a 3rd party application, typically a mission command and monitoring tool. When logged into the virtual environment in this gateway mode, *AUVSim* transparently routes all communications to and from the (simulated) acoustic modem to the 3rd party application.

C. *Virtualization*

An important capability of our simulation environment is to be able to run experiments which employ multiple SAUV systems executing group behaviors which rely upon inter-vehicle communication. The logic for these behaviors resides in the high-level processes located on the PC-104. In the past when we conducted single vehicle testing, we would use an actual PC-104 stack on the bench and connect directly to it through its acoustic serial port in order to issue vehicle commands and monitor outgoing acomms messages. Putting s*auvSim* into service improved the situation by allowing us to route vehicle acomms traffic through the CADCON simulator, thus opening the door for other vehicles to hear and respond. We were still restrained, however, to the use of one PC-104 stack per simulated SAUV. This did not appear to be a cost effective way for setting up a fleet of simulated SAUVs.

Our initial solution to this problem was going to be the installation of the SAUV Linux OS on older PCs in house, and then installing copies of the SAUV high-level processes on them. While this would obviate buying a PC-104 stack for each SAUV we wanted to simulate, the prospect of dealing with a motley collection of various aging desktop machines seemed less than ideal.

Instead, we turned to the use of virtualization software [11], which allows us to install multiple SAUV-flavored Linux "guest" systems on a given "host" computer. We use VMware Workstation, by VMware Corp., to support those guest systems. These "virtual SAUVs" are configured to have the same runtime resources and run the same Linux OS and applications

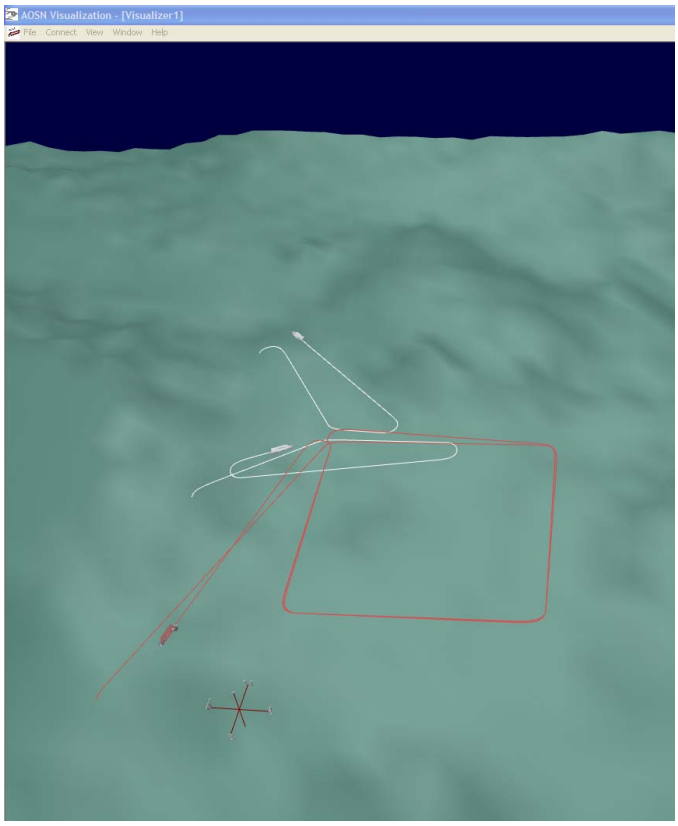

Figure 5. Testing a three vehicle cooperative mission using the CADCON *Visualizer* application.

suite as real SAUVs. Since VMware runs on both Linux and Windows platforms, both can host our virtual SAUV guests. In addition to making it easier to create (about 15 minutes) and manage virtual SAUVs in house, we can now also easily distribute new virtual SAUVs to our development partners, who can use the free VMware Player application to run their virtual SAUVs.

IV. Putting It To Work

Since the modifications described in this paper were completed in spring 2006, we have been actively using this new capability. Below we describe areas where we have been using this facility, or expect to in the near future.

A. *SAUV High-level Process Debugging*

This new capability allows us to now easily, routinely, and extensively exercise and evaluate various parts of the SAUV high level software. Developers at AUSI utilize the facility to exercise the various manager and handler processes of the SAUV. Researchers at NUWCDIVNPT are developing the Distributed Control Environment (DICE) to enable multiple vehicle cooperation, and a version of this has been implemented as higher level control logic on the SAUV [9]. The distributed nature of our simulation environment allows these Navy researchers in Rhode Island to test, evaluate and debug this code logic in real-time with AUSI researchers located in New Hampshire.

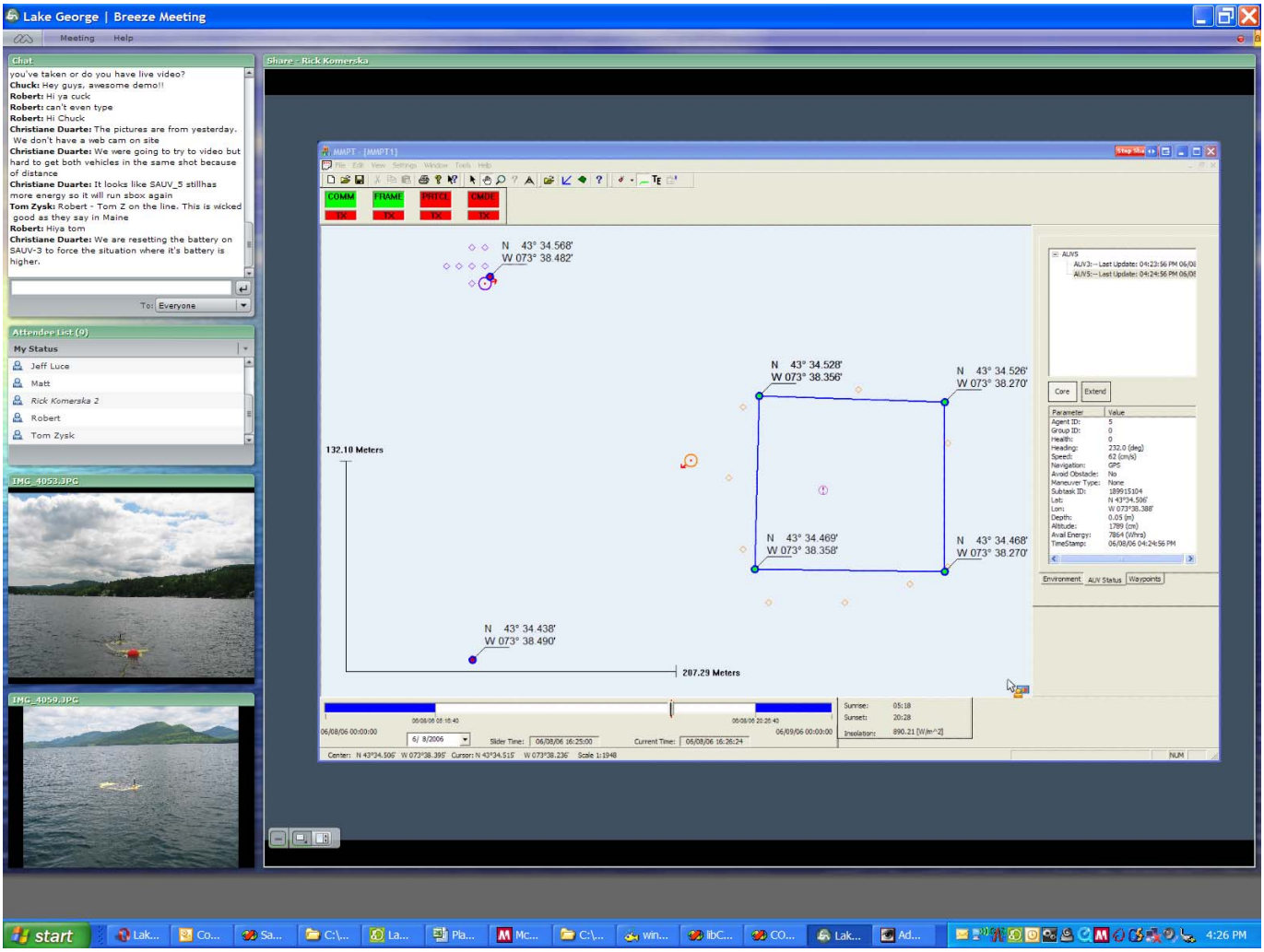

Figure 6. Collaborating in mission planning, execution and monitoring using the web-based Breeze room environment.

B. Cooperative Vehicle Mission Development and Rehearsal

To date, we have used our simulation environment to develop and test cooperative level behavior logic to support three SAUV field test events: Stillwater Basin, RI (May 2006), Lake George, NY (June), and Monterey Bay, CA (July). Each of these testing events focused on variations of a role swapping mission involving two or three vehicles. In this mission, one vehicle would perform a survey pattern (typically a box or lawnmower pattern) while the other vehicles would maintain station on the surface recharging their batteries. When the survey vehicle completed its pattern, the group would then decide who should next take the survey role. The decision process relies on the vehicles listening to each other's periodic status updates and using information therein (such as available energy and current task state) to decide who does what.

Developers distributed across different locations were able to use the CADCON *Visualizer* tool to view aspects of the mission simulation. Fig. 5 shows a visualization of the trajectory for such a mission. We have also been working with Navy researchers experimenting with other collaboration tools; Fig. 6 shows a prototype of the SAUV mission planning tool being used to monitor an actual field experiment at Lake George, NY within the Breeze room environment developed by Macromedia Corp. We continue to use the Breeze room for both display of actual field data as well as simulation testing and debugging in a distributed collaborative environment.

C. Mission Planning Tools Development

AUSI researchers are currently supporting TSI in the development of a Modular Mission Planning Toolkit (MMPT) application to be used for planning, monitoring and executing multiple cooperative AUV missions. TSI developers in Brunswick, ME will typically connect an MMPT application to an *AUVSim* in gateway mode, which is then connected to the CADCON virtual environment server at AUSI. In addition, they may run one or two virtual SAUVs (using VMware Player) to provide data input into the MMPT application, or passively participate in an ongoing simulation, eavesdropping on simulated acomms status message traffic to test monitoring aspects of MMPT.

D. Network Protocols Development

As part of our current research, we are actively working with faculty from the University of New Hampshire (UNH) to further develop protocols to support undersea networking and media access control. Our goal is to be able to have UNH researchers working on this project be able to implement new communications protocol algorithms within their virtual SAUVs (using VMware Player) while working in their own lab and test them in the CADCON environment. AUSI researchers will be able to monitor and support the testing as appropriate.

E. SAUV Operator Training

While nothing compares to actual in water training, basic familiarity with the SAUV Mission Planner operator software can now be gained before committing to the expense of actually fielding the vehicles. This allows for beginning operators to work with the Mission Planner as well as the RF and acoustic modem hookups before going into the water for the first time. The setup allows the operator to focus on learning the SAUV applications, without the real world distractions of being in the field.

V. Conclusions

The simulation capability we have created has allowed us to better develop and test the various pieces of the SAUV system. It has enabled the ability to evaluate system components, such as vehicle planning tools, within the context of a fully simulated mission, with developers located in disparate locations.

We hope to use this capability to reach out to other investigators and developers interested in multiple AUV cooperation. We believe our environment is flexible and rich enough to accommodate other AUVs in a manner similar to what was accomplished with the SAUV integration effort. We believe it would be fruitful in the research into heterogeneous vehicle cooperation to have other AUV systems represented in CADCON. In addition, researchers would have the ability, through the gateway buoy aspect of *AUVSim,* to exercise their own AUV fleet management applications.

References

[1] J. Sousa and A. Gollu, "A Simulation Environment of the Coordinated Operation of Multiple Autonomous Underwater Vehicles," in *Proceedings of the Winter Simulation Conference'97*, 1997.

[2] J. Weekley, D. Brutzman, A.J. Healey, D. Davis and D. Lee, "AUV Workbench: Integrated 3D for Interoperable Mission Rehearsal, Reality, and Replay," *2004 Mine Countermeasures & Demining Conference*, Australian Defence Force Academy, Canberra, Australia, February 2004.

[3] P. Ridao, E. Batlle, D. Ribas and M. Carreras, "NEPTUNE: A HIL Simulator for Multiple UUVs," in *Proceedings of MTS/IEEE Oceans'04 Conference*, November 2004.

[4] F. Mahieu, "Software development of a hardware-in-the-loop simulation and a three-dimensional viewer for autonomous underwater vehicles," M.S. Thesis, *Florida Atlantic University Libraries*, August 2000.

[5] J. Jalbert, et al., "Solar-Powered Autonomous Underwater Vehicle Development," in *Proceedings of the Thirteenth International Symposium on Unmanned Untethered Submersible Technology*, August 2003.

[6] S.G. Chappell, S.S. Mupparapu, R.J. Komerska, and D.R. Blidberg, "SAUV II High Level Software Architecture," in *Proceedings of the Fourteenth International Symposium on Unmanned Untethered Submersible Technology*, August 2005.

[7] D.M. Crimmins, et al., "Use of a long endurance solar powered autonomous underwater vehicle (SAUV II) to measure dissolved oxygen concentrations in Greenwich Bay, Rhode Island, U.S.A," in *IEEE OCEANS'05 EUROPE Conference Proceedings*, 2005.

[8] S.G. Chappell and R.J. Komerska, "An Environment for High-Level Multiple AUV Simulation and Communication," in *Proceedings of the Collaborating and Leveraging Outer Space and Undersea Technologies (CLOUT) NOAA/NASA Workshop*, August 2000.

[9] C.N. Duarte, et al., "A Common Control Language to Support Multiple Cooperating AUVs," in *Proceedings of the Fourteenth International Symposium on Unmanned Untethered Submersible Technology*, August 2005.

[10] S.S. Mupparapu, R. Bartos and M. Haag. "Performance Evaluation of Ad Hoc Protocols for Underwater Networks," in *Proceedings of the 14th International Symposium on Unmanned Untethered Submersible Technology*, August 2005.

[11] M. Rosenblum and T. Garfinkel, "Virtual Machine Monitors: Current Technology and Future Trends," *Computer*, vol. 38, no. 5, pp. 39-47, May 2005.