

**An Experimental Platform
for Investigating
Decision and Collaboration Technologies
in Time-Sensitive Mission Control Operations**

**S. D. Scott
M. L. Cummings**

**Massachusetts Institute of Technology*
Prepared For Boeing, Phantom Works**

**HAL2007-04
August, 2007**



<http://halab.mit.edu>

e-mail: halab@mit.edu

*MIT Department of Aeronautics and Astronautics, Cambridge, MA 02139

An Experimental Platform for Investigating Decision and Collaboration Technologies in Time-Sensitive Mission Control Operations

Stacey D. Scott & Mary L. Cummings

Executive Summary

This report describes the conceptual design and detailed architecture of an experimental platform developed to support investigations of novel decision and collaboration technologies for complex, time-critical mission control operations, such as military command and control and emergency response. In particular, the experimental platform is designed to enable exploration of novel interface and interaction mechanisms to support both human-human collaboration and human-machine collaboration for mission control operations involving teams of human operators engaged in supervisory control of intelligent systems, such as unmanned aerial vehicles (UAVs). Further, the experimental platform is designed to enable both co-located and distributed collaboration among operations team members, as well as between team members and relevant mission stakeholders.

To enable initial investigations of new information visualization, data fusion, and data sharing methods, the experimental platform provides a synthetic task environment for a representative collaborative time-critical mission control task scenario. This task scenario involves a UAV operations team engaged in intelligence, surveillance, and reconnaissance (ISR) activities. In the experimental task scenario, the UAV team consists of one mission commander and three operators controlling multiple, homogeneous, semi-autonomous UAVs. In order to complete its assigned missions, the UAV team must coordinate with a ground convoy, an external strike team, and a local joint surveillance and target attack radar system (JSTARS). This report details this task scenario, including the possible simulation events that can occur and the logic governing the simulation dynamics.

In order to perform human-in-the-loop experimentation within the synthetic task environment, the experimental platform also consists of a physical laboratory designed to emulate a miniature command center. The Command Center Laboratory comprises a number of large-screen displays, multi-screen operator stations, and mobile, tablet-style devices. This report details the physical configuration and hardware components of this Command Center Laboratory. Details are also provided of the software architecture used to implement the synthetic task environment and experimental interface technologies to facilitate user experiments in this laboratory.

The report also summarizes the process of conducting an experiment in the experimental platform, including details of scenario design, hardware and software instrumentation, and participant training. Finally, the report suggests several improvements that could be made to the experimental platform based on insights gained from initial user experiments that have been conducted in this environment.

Table of Contents

1	Introduction	4
2	Representative Task Scenario: UAV Team Operations	4
3	Command Center Laboratory: Hardware Setup	5
3.1	Networked, Large-screen Wall Displays	7
3.2	Reconfigurable Operator Consoles	7
3.3	Mobile Team Workstations.....	8
3.4	Collaboration Server / Data Analysis Workstation	8
3.5	Data Capture Equipment	8
3.6	Command Center Laboratory Configuration	8
4	Software Architecture and User Interfaces.....	10
4.1	Conceptual Software Design	10
4.2	Implementation Details	13
4.2.1	Simulation Server	13
4.2.2	Functionality Common to All Team Displays	15
4.2.3	Map Display.....	16
4.2.4	Mission Status Display	18
4.2.5	Remote Assistance Display	20
4.2.6	Mission Commander Interface	22
4.2.7	Operator Display	24
5	Conducting a User Experiment.....	25
5.1	Scenario Development.....	25
5.2	Instrumentation	25
5.3	Participant Training.....	26
5.4	Running the Experimental Software	27
6	Lessons Learned from Initial User Experiments	28
7	Suggested Improvements.....	29
7.1	Software Improvements	29
7.2	Participant Training Improvements	30
8	Conclusions	30
9	References	31

1 Introduction

This report details an experimental platform developed to support a research project aimed at investigating decision and collaboration technologies for complex, time-critical mission control operations, such as military command and control and emergency response. This experimental platform is designed to enable exploration of novel interface and interaction mechanisms that support both human-human collaboration and human-machine collaboration for mission control operations involving teams of human operators engaged in supervisory control of intelligent systems, such as unmanned aerial vehicles (UAVs). Further, the experimental platform is designed to enable both co-located and distributed collaboration among operations team members, as well as between team members and relevant mission stakeholders.

To enable initial investigations of new information visualization, data fusion, and data sharing methods, the experimental platform provides a synthetic task environment for a representative collaborative time-critical mission control task scenario. This task scenario involves a UAV operations team engaged in intelligence, surveillance, and reconnaissance (ISR) activities. To provide further background and context for the details of the experimental platform, the report first describes this task scenario in detail, including a discussion of the team composition and their task roles within the experimental environment.

The report then details the hardware setup of a Command Center Laboratory that was built to house the collaborative technology research experiments at the MIT Humans and Automation Laboratory. Next, the architecture of the experimental software is then presented, along with the design and functionality of the application interfaces currently available in the experimental platform. The process of conducting a user study in the experimental platform is then described. In particular, details are provided on designing an experimental scenario, instrumenting the experimental setup, training participants, and running the experimental software. Finally, some lessons learned and recommended improvements are discussed based on our experiences with initial studies conducted in the environmental platform.

2 Representative Task Scenario: UAV Team Operations

To enable initial investigations of decision and collaboration support technologies in an experimental setting, a representative collaborative time-critical mission control task scenario was developed as the foundation for a synthetic task environment. The task scenario involves a military unmanned aerial vehicle (UAV) operations team engaged in intelligence, surveillance, and reconnaissance (ISR) activities to ensure the safe passage of a political ground convoy that is traveling through a potentially hostile geographic area. During the task, the UAV team surveils the area for potential threats. If any emergent, time-sensitive targets are discovered in the UAV team's area of interest (AOI), the team coordinates with a local strike team to prosecute these targets before they are within weapons range of the convoy. The team also monitors incoming intelligence reports in order to extract information relating to their AOI and potentially communicate with other teams as necessary to clarify intelligence reports.

In order to secure the AOI, the team utilizes a number of semi-autonomous unmanned aerial vehicles (UAVs). Various team members monitor the progress of these UAVs as they provide surveillance of the large AOI and to reroute the UAVs from their original surveillance course, as necessary to secure the area. The team may also be required to coordinate with other teams to utilize assets outside of their immediate control to help secure the AOI.

As shown in Figure 1, the UAV operations team consists of three UAV operators, each responsible for controlling multiple UAVs, and one mission commander overseeing the team's mission progress. The UAV operators are responsible for supervising the progress of several UAVs surveilling the AOI, confirming and classifying potential targets identified by the UAVs' onboard automatic target recognition (ATR) systems, and coordinating with a strike team to destroy confirmed targets. This task scenario assumes advanced onboard ATR capability, as well as the use of a distributed ISR Cell that would liaise with this UAV team for any necessary detailed image analysis.

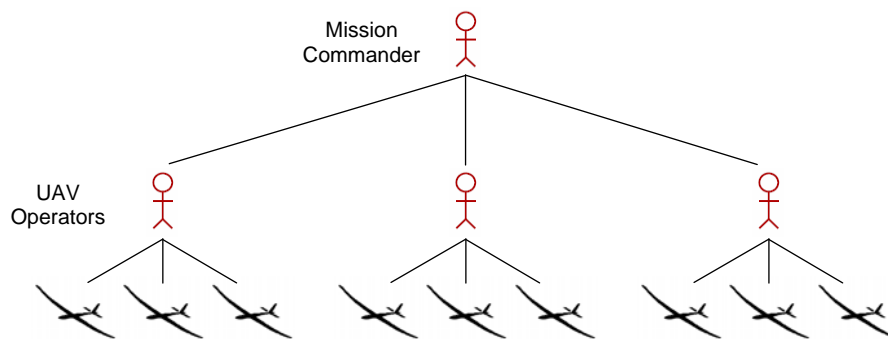


Figure 1. UAV team structure.

The team's mission commander is responsible for ensuring the safety of the convoy and for managing the workload of the UAV operators on their team throughout the mission. To achieve these mission objectives, the mission commander can make several strategic decisions, which include requesting the convoy hold its current position if its intended path is not deemed safe for passage, requesting supplementary surveillance data from a nearby joint surveillance and target attack radar system (JSTARS), and re-tasking of one of the team's UAV assets to a different sub-AOI (requiring the handoff of the UAV asset between operators).

3 Command Center Laboratory: Hardware Setup

In order to facilitate the evaluation of new interface technologies to support decision-making and collaboration in the UAV operations team described above, and for time-critical mission operations in general, a physical laboratory-based testing environment was developed to emulate a miniature command center. This section describes this team testing laboratory and the rationale for its overall design and constituent components.

In complex, collaborative mission command environments, such as those in which military personnel are engaged in ISR activities, there are typically many people performing various related, yet distinct task activities. Each team member contributes their own skills and duties to the overall performance of the team activities. Often different team members need access to different types of information in order to perform their particular duties. For instance, an operator supervising several UAVs during a surveillance task may need to know the status of each UAV, and may also need to access video feed from each UAV to monitor his area of responsibility. On the other hand, that operator's mission commander may be more interested in knowing which areas of a larger geographical area their team has currently surveilled and which areas still need some attention. Knowing the overall status of the team's mission will also help the commander report back to their superiors, who are busy managing many teams, performing many parts of the overall mission.

The physical characteristics of different display technologies (e.g., size, proximity, number of displays, and mobility) that may be present in these complex environments make each display technology more or less suitable for particular individual and collaborative uses and interactions. This issue has begun to be explored in corporate and design environments (Mandryk et al., 2002; Russell et al., 2002; Ryall et al., 2004; Inkpen et al., 2005), but has received little systematic investigation in complex, time-critical environments. However, since the consequences of communication and coordination breakdowns in these environments can be more severe than in typical corporate or design settings, understanding how to best utilize, and design appropriate decision and collaboration interfaces for, different display technologies in these environments may help reduce such collaboration breakdowns and enhance the overall effectiveness of these technologies for both individual and collaborative use.

To facilitate this research, a laboratory with the following components was developed (see Figure 2):

1. 3 x Networked, large-screen wall displays
2. 4 x 3-Screen, reconfigurable operator consoles
3. 4 x Mobile team workstations (2 tablet-style and 2 handheld-style displays)
4. 1 x Collaboration server / data analysis workstation
5. Data capture equipment

The following section provides more details for each of these components, and the rationale for including them in the Command Center Laboratory.

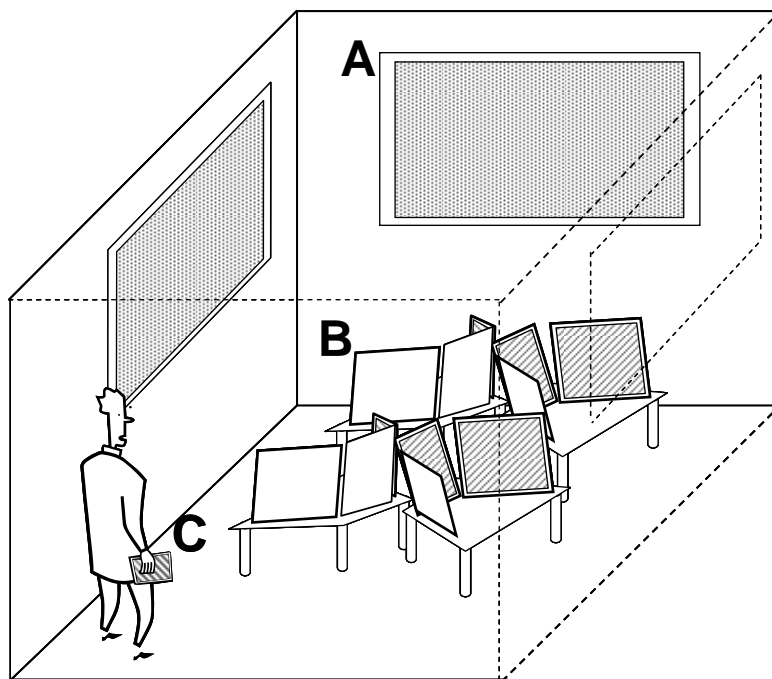


Figure 2. Concept diagram of the Command Center Laboratory: (A) large-screen wall displays, (B) reconfigurable, multi-display operator consoles, and (C) small, mobile displays (e.g., a TabletPC) (collaboration server and data capture equipment not shown).

3.1 Networked, Large-screen Wall Displays

Large-screen wall displays are becoming a ubiquitous part of current mission command environments (Jedrysik et al., 2000; Dudfield et al., 2001; Smallman et al., 2001; Jenkin, 2004). These displays are often used to provide team members with the “big picture” and are often referred to as “situation displays” because they are typically used to display maps and status information related to the overall battlefield situation (Jenkin, 2004; Peel and Williams, 2004). Yet, the best use of these large displays in command center operations remains a challenging issue because various team members, each performing different aspects of the overall task, are all potential consumers of the information on these displays.

The experimental platform is designed to enable research on teamwork that occurs in relatively confined spaces, similar to teamwork that might occur in the tight confines of a naval vessel or in a mobile surveillance truck. Even in such tight quarters, however, there is still a need for a mission commander or the team members in general to maintain an overall awareness of the “big picture” and an understanding of how their activities fit into this picture. In order to explore various software visualizations to provide relevant situation awareness information, the experimental platform contains several large-screen displays, mounted along the walls of the team testing laboratory (Figure 2 (A)).

The large-screen wall displays are each driven by a separate display server, and are connected via the Internet and underlying collaboration software to the other computers in the laboratory. Finally, the wall displays each include touch-screen overlay technology to enable team members to interact directly with the digital data being displayed on these large-screen displays in order to facilitate collaborative planning and decision-making among team members.

3.2 Reconfigurable Operator Consoles

In order to investigate the impact of the design of individual operator consoles on team task performance and the impact of the relative arrangement of multiple individual operator consoles on individual and team task performance, the Command Center Laboratory contains several reconfigurable individual operator consoles. Since most futuristic operator consoles consist of multiple displays (Osga, 2000; Peel and Williams, 2004), the operator consoles in the laboratory comprise three flat-screen LCD displays (Figure 2 (B)). To provide flexibility in the team-related research questions that can be explored in the experimental platform the operator consoles are designed to be reconfigurable in the following ways:

- The relative positions of consoles within the Command Center Laboratory are reconfigurable to enable the investigation of the impact of seating arrangement and visibility of other team members’ task activities on individual and team task performance. To provide this flexibility, the operator consoles are equipped with sturdy, yet small and mobile computer desks and mobile computer chairs.
- The multi-display setups at each console are adjustable to enable investigation of the impact of individual console design on individual and team task performance. To enable this display reconfiguration, the operator consoles are equipped with an adjustable monitor stand capable of arranging three monitors in a variety of positions.

The information displayed at each operator consoles is provided by separate computer workstations that are connected to the other computers in the Command Center Laboratory via the Internet and underlying collaboration software. Each console also includes wireless headsets

to enable communication among all team members in the room and to other potential collaborators in other external team nodes, via an Internet connection.

3.3 Mobile Team Workstations

Smaller, more mobile displays, such as TabletPCs and handheld computers are also becoming common in complex, mission control environments. These displays provide the advantage of ultimate portability, especially as wireless network connectivity improves. These displays can also offer the advantage of being semi-private (Mandryk et al., 2002; Inkpen et al., 2005). Thus, they may provide a suitable medium for team members, such as mission commanders, who need to access classified information or wish to review pending tasks that the team will soon have to handle.

In order to explore appropriate uses of such mobile technology in the context of mission command operations, the Command Center Laboratory includes several mobile displays (Figure 2 (C)), including two TabletPCs and two handheld computers. While both of these technologies are very portable, the relative difference in their display sizes will likely impact the most suitable usage of each device.

3.4 Collaboration Server / Data Analysis Workstation

In order to enable user input to the simulated task environment from any team display and to support data sharing across these displays, the experimental platform includes a main collaboration server. This server also doubles as a data analysis workstation between experimental testing phases. To facilitate quantitative data analysis, this workstation includes the SPSS statistical data analysis software package. To facilitate qualitative data analysis from video data captured during team experiments, this workstation includes the Altas.ti qualitative data analysis software package. Finally, to facilitate creation of demonstration videos of new technology developed during this project, this workstation includes the Adobe Premiere video editing software package.

3.5 Data Capture Equipment

To facilitate careful analysis of the individual and team behavior that occurs during experimental testing, the Command Center Laboratory contains several types of data capture equipment. First, a digital mini DVD camcorder, with a wide angle lens, is available to capture video and audio data of team members' interactions. These data help provide context to the quantitative measures that are collected through the software datalogs during experimental trials. This context enables richer interpretation of the data and, ultimately, a better understanding of how well the technological solutions being tested are able to support individual and team task performance. Several accessories for the digital camcorder are also available, including a tripod, and a supply of mini DVDs for capturing and archiving the experimental data. In order to capture an accurate account of the dynamic changes to the software interfaces during the experiments, the computers in the Command Center Laboratory are also equipped with real-time screen-capture software that can be replayed during the data analysis phase. This software is also useful in replaying all or part of a participant's session during retrospective interviews.

3.6 Command Center Laboratory Configuration

This section describes the Command Center Laboratory equipment's current physical configuration. Figure 3 illustrates the current setup of the main equipment components in the laboratory. Figure 4 shows a schematic diagram of the current setup, illustrating the relative

placement of these components. Appendix A provides further details on these components, including their hardware specifications.

The collaboration server is currently located just outside of the laboratory, beside the observation window (Figure 3a). This location enables an experimenter to control the simulation software during experimental trials without disturbing the experiment participants and to monitor the participants' behavior.

Two of the three large-screen wall displays are mounted by stationary, but tiltable wall mounts (the left and center displays in Figure 3b), while the third display is mounted by a wall mount with an articulated arm that enables a variety of positions, including one that covers a bookshelf mounted in the back, right corner of the laboratory (the right display in Figure 3b).

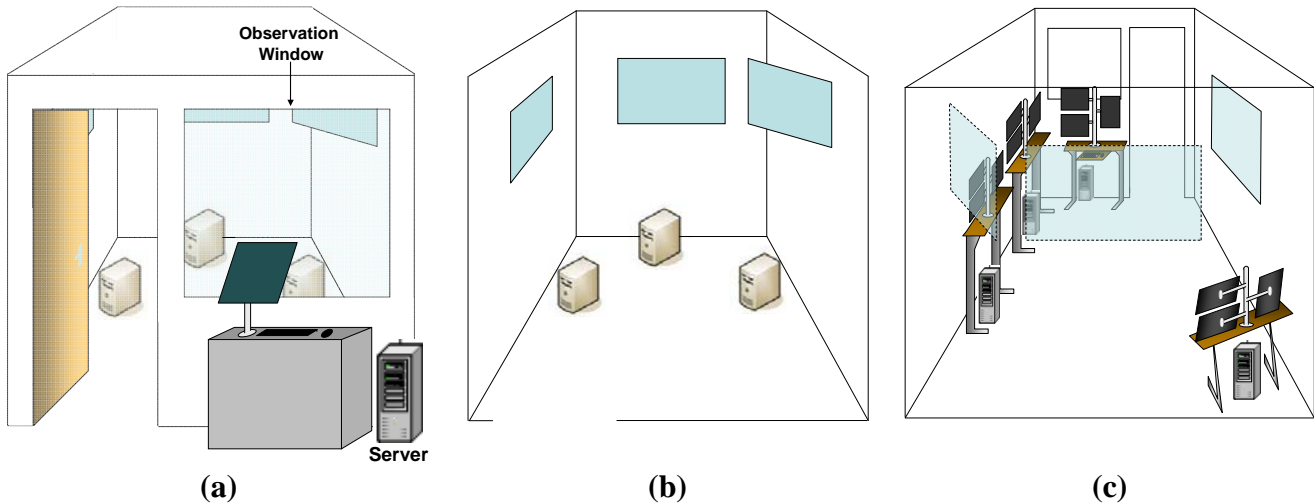


Figure 3. Command center laboratory equipment in its current configuration: (a) collaboration server, (b) three large-screen wall displays, and (c) four reconfigurable operator consoles (note, this diagram is show from the back of the lab looking towards the observer window).

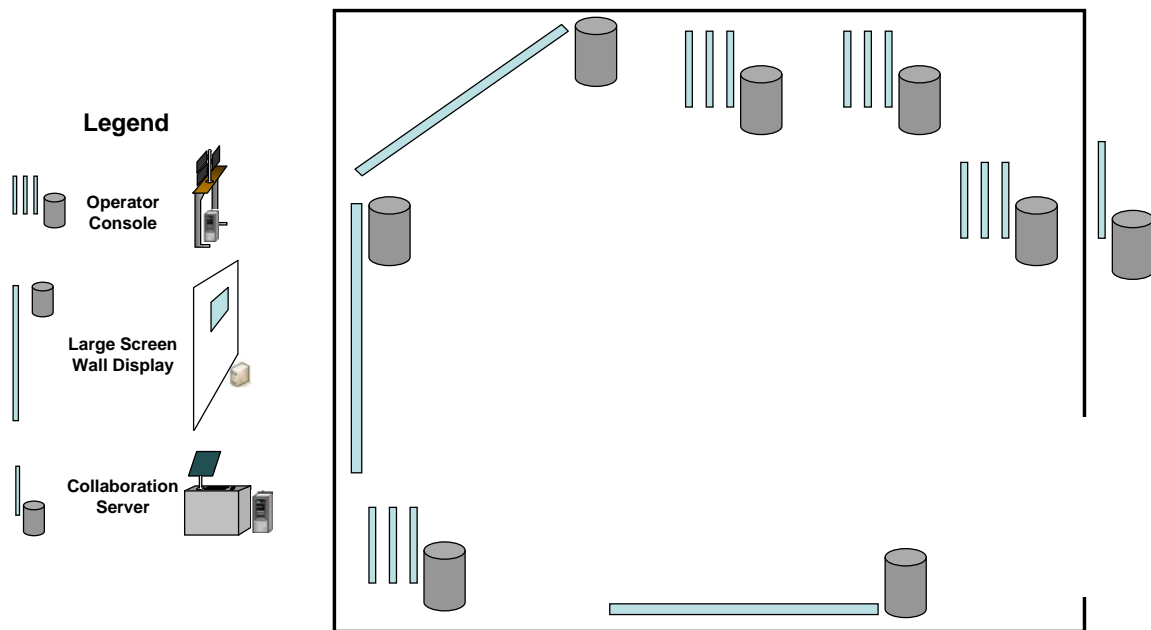


Figure 4. Current equipment configuration of the Command Center Laboratory.

The four reconfigurable operator consoles shown in Figure 3c are easily re-positioned as they are self contained on a wheeled computer station. During experiments including local operators, these consoles support up to four local operators (i.e. UAV or communications operators), and can also be used to facilitate supplementary data capture during experiments.

Finally, the Command Center Laboratory also includes several mobile displays (tablets and handhelds (not shown in Figure 3). These displays can be positioned anywhere in the laboratory. For extended experimental sessions, a portable wooden shelf is provided for participants to rest the mobile displays, since some participants find these displays (particularly the tablet displays) quite heavy to hold for long periods of time.

4 Software Architecture and User Interfaces

This section summarizes the underlying software architecture of the current synthetic task environment in the experimental platform. It also provides details of the available user interfaces, including both the proposed UAV team user interfaces and the experimenter displays. The team displays described in this section were developed based on information and functional requirements generated from a cognitive task analysis (CTA) performed on the task scenario described above (Scott et al., 2006; Scott et al., 2007).

As an initial starting point to enable preliminary research studies in the experimental platform, the CTA, and the subsequent displays focus on supporting the mission commander's role in the UAV team task scenario. Thus, the behaviors of the UAVs and UAV operators are currently simulated in the software testbed, and experiment participants assume the role of the UAV mission commander. The type of user experiments currently supported in the experimental platform will be described in more detail below. Ongoing efforts are focused on extending the capabilities of this experimental platform to include active UAV operator displays in order to provide complete support for experiment participants to assume all roles of the UAV team.

4.1 Conceptual Software Design

The conceptual design of the currently available software in the experimental platform is shown in Figure 5.

The complete testbed consists of five separate executable applications, each representing a separate UAV team or experimenter interface. Represented as rounded boxes in Figure 5, these applications include:

- **Simulation Server.** This application provides the main simulation engine for the synthetic task environment. It governs the entity and events behaviors during the experimental scenarios and provides all the main calculations for any data displayed on the team displays, based on the current and expected state of the simulated world and its components. This application also provides the functionality for creating test scenarios to use in the synthetic task environment, including functionality to create world entities (e.g., UAVs), to set their attributes (e.g., UAV planned routes), and to create planned scenario events (e.g., communication link outages). These test scenarios can be saved to XML files and reloaded later into the Simulation Server.

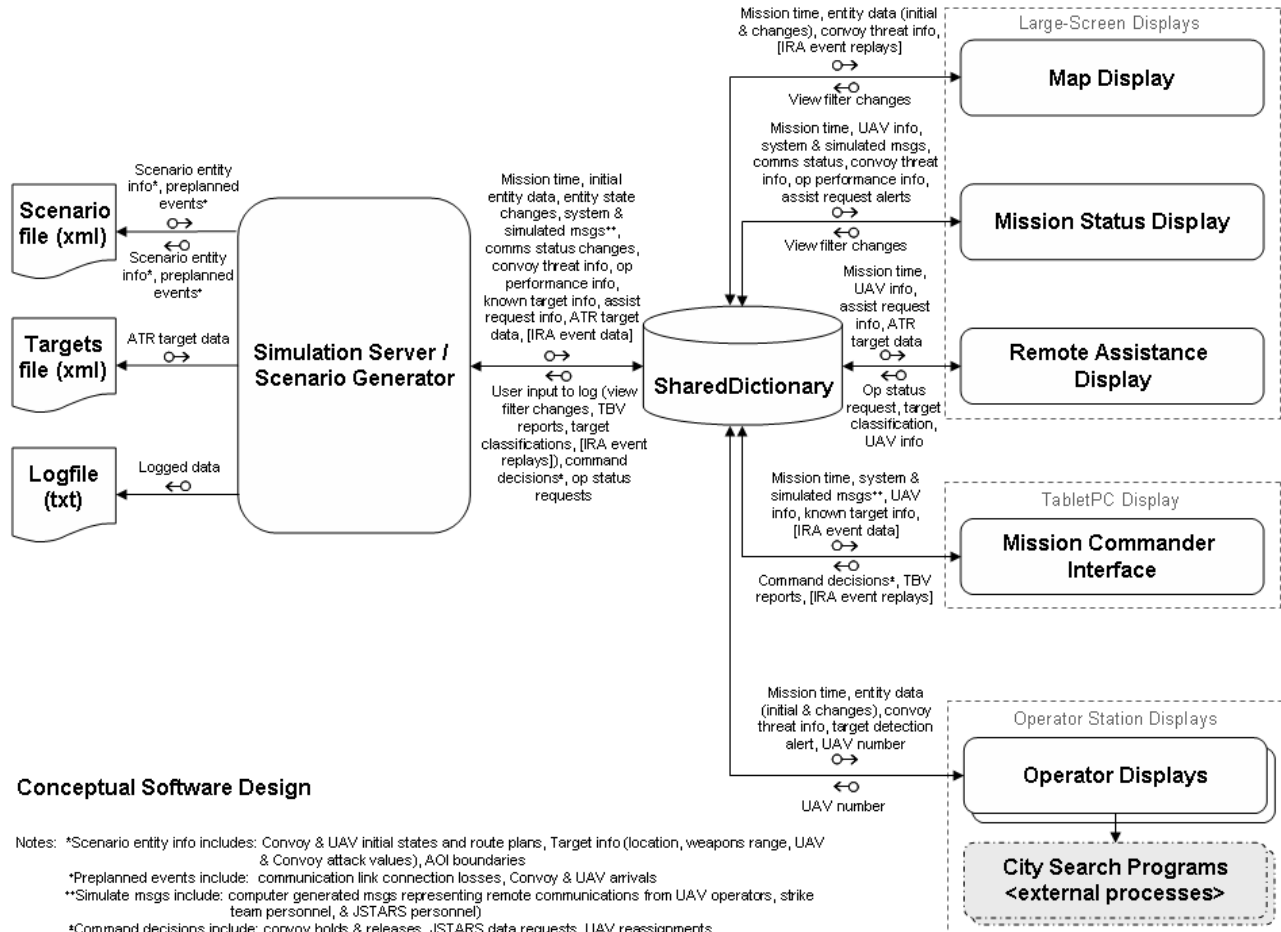


Figure 5. Conceptual design of the experimental platform software (see Appendix B for expanded view).

- **Map Display.** This application provides the functionality for one of the supervisory-level interfaces, designed to be displayed on a large-screen wall display. It provides an updated view of the main mission assets (e.g., convoy, UAVs, targets) in the context of the team’s area of interest (AOI). This application also provides a visual timeline display that provides an updated representation of the current and expected threat level of the convoy.
- **Mission Status Display.** This application provides the functionality for a second supervisory-level large-screen wall display. It displays a variety of mission status information, including communication link connectivity status, operator performance data, UAV tasking information, and team communication and system status messages.
- **Remote Assistance Display.** This application provides the functionality for a third (optional) supervisory-level, large-screen wall display. This display enables the UAV mission commander to assist a (simulated) remote UAV operator with a target classification operation (explained in detail below).
- **Mission Commander Interface.** This application provides the functionality for executing command decisions during experimental scenarios. These command decisions are the main mechanisms by which the mission commander can effect changes in the synthetic task environment during experimental trials. This interface, designed to be run on a tablet

display, enables software actions such as stopping or starting the convoy's progress, reassigning a UAV from one area in the team's AOI to another, and requesting additional surveillance data from a nearby intelligence source.

- **Operator Display.** This application provides limited, optional functionality for supporting local UAV operators during experimental scenarios. This display provides a very basic UAV operator display, that essentially mimics that map-related functionality of the Map Display, though filtered to show only the AOI of one of the three possible operators (the desired operator view can be selected in the provided interface). While the majority of the UAV operator behavior is still simulated by the Simulation Server (e.g. control of the UAVs), when this display is used instead of the Remote Assistance Display, a separate, external process map-based search program (call the City Search program) is launched to emulate a target identification activity whenever one of the operator's UAVs' detects a potential target in their AOI.

To facilitate data storage and sharing in this software environment, the Grouplab.Networking collaboration software toolkit (a .NET derivative of Bolye's (2005) Collabrary toolkit) is used. This toolkit handles all of the low-level networking coordination between the distributed software applications and provides a data storage and data sharing mechanism called the SharedDictionary. The SharedDictionary serves as a central data store for all connected applications (Figure 5). It uses a subscription-based approach to data sharing. In particular, the Simulation Server creates and initializes an instance of the SharedDictionary class upon start-up. Each application, including the Simulation Server, then opens a TCP connection to the SharedDictionary process and identifies the particular data items¹ about which they wish to receive notifications, that is, they "subscribe" to those data items. Any application connected to the SharedDictionary can add, modify, or delete a data item from the SharedDictionary. Through the built-in functionality of the Grouplab.Networking toolkit, these changes are automatically propagated to any application that is subscribed to that particular data item. Thus, connected applications do not need to continually poll the SharedDictionary for updates, nor do they need to know which other applications need to be informed of particular data updates.

In order to enable the experimental platform to emulate a number of different task situations, the Simulation Server utilizes a number of data files (Figure 5). As previously mentioned, the Simulation Server can store and load scenario XML data files. The scenario generation process and the contents of the scenario data files are detailed below. The Simulation Server can also load target-related information from an XML data file (generated externally). This information is used in conjunction with the Remote Assistance Display to simulate data that would be received from a UAV's onboard automatic target recognition (ATR) system (e.g. target imagery, initial target classification, and certainty of classification). Finally, the Simulation Server can also create text-based data log files to facilitate analysis of experiment participants' system interaction and overall performance. These files are formatted to be easily imported into Microsoft Excel for processing and analysis.

¹ The SharedDictionary supports storing and sharing of many types of data items, including complete classes. Any class object that can be serialized can be stored in and retrieved from the SharedDictionary.

4.2 Implementation Details

This section provides further details of the software design, including the graphical user interface designs and the particular functionality of each application display. The experimental platform software applications were developed using the C# programming language and the Microsoft .NET framework in the Microsoft Visual Studio (Visual C# 2005 Express Edition) integrated development environment. The Grouplab.Networking toolkit that provides the SharedDictionary functionality is a publicly available² .NET library (Boyle and Greenberg, 2006). To facilitate modularity and code reuse, two helper libraries were created and are used by the experimental platform software applications: an entity library and a draw library (Appendix B). The entity library, called TST Entity Library, provides the class descriptions for all world entities used in the synthetic task environment, such as the UAV, Convoy, Target, and Strike Schedule classes. The draw library, called the TST Draw Library, provides standard methods for drawing the visual representations of world entities and common interface components.

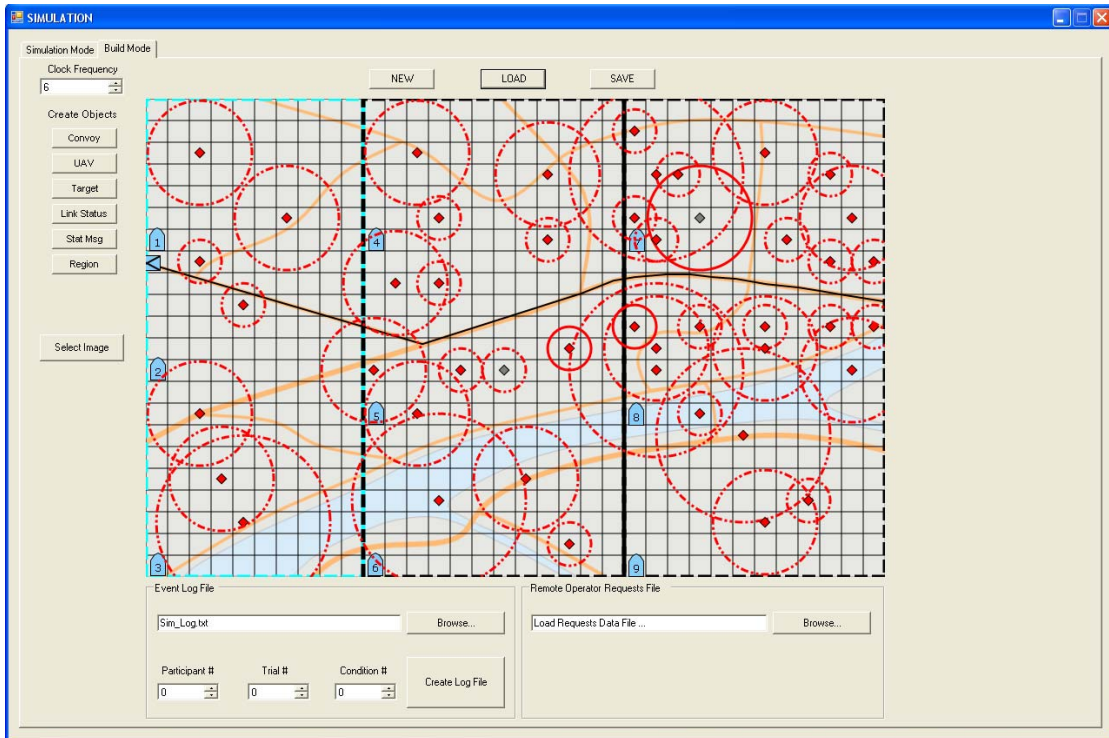
4.2.1 Simulation Server

Figure 6 shows the graphical user interface (GUI) of the Simulation Server. This interface consists of two panels, a Build Mode panel (Figure 6a) and a Simulation Mode panel (Figure 6b). The Build Mode panel provides the functionality to create, save, and load scenario files, to load target-related data files, and to create data log files. The Simulation Model panel provides the functionality to connect to the SharedDictionary and to run, pause, and restart experimental scenarios. An experimental scenario can be run as a standalone application, that is, offline from the SharedDictionary, with no other testbed applications running. The ability to run the Simulation Server offline enables experimenters to test potential scenario files without the hassle of setting up each team display. Likewise, each team display can be executed offline, however no entity data will be shown until they are connected to the SharedDictionary and the Simulation Server begins to run a scenario.

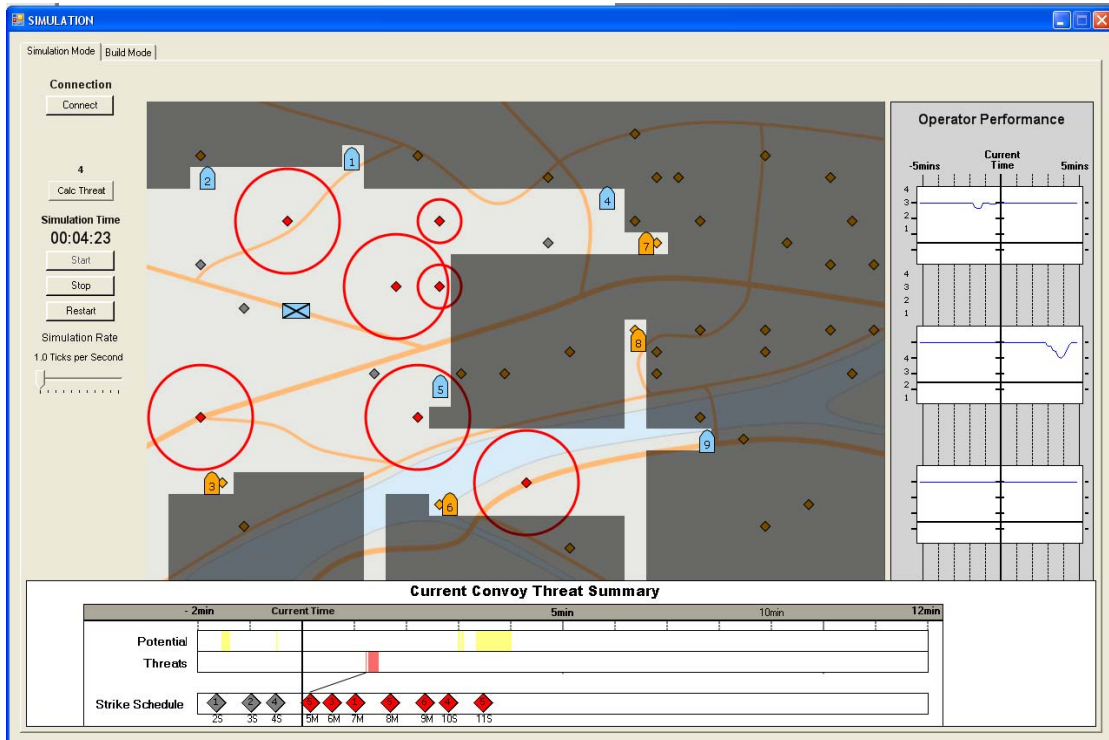
In order to run an experimental scenario in the experimental platform, a scenario file must first be created (Appendix D details the process of creating or modifying an experimental scenario) or loaded in the Build Mode panel. To load an existing scenario file, select the “Load” button at the top of the Build Mode panel, and then select the desired XML scenario file from the file browser dialog that appears. If the Remote Assistance Display is being used during an experimental scenario, a Targets data file must also be loaded in the Build Mode panel. To accomplish this, select the “Browse” button from the “Remote Operator Request File” area at the bottom of the panel, and then select the desired Targets XML file from the file browser dialog that appears. Finally, before an experimental scenario is started, a log file should be created in the Build Mode panel. All scenario events and user interaction will be written into this file. To create a log file, enter the appropriate participant, trial, and condition number data in the “Event Log File” area and select the “Create Log File” button. The program will automatically create a log file in the same directory as the Simulation Server executable file, labeled with these data in the following format.

```
<year>_<month>_<day>_<participant_no>_<trial_no>_<condition_no>.txt
```

² The Grouplab.Networking software toolkit is licensed under the Academic Free License Version 1.2 (<http://www.opensource.org/licenses/academic.php>).



(a)



(b)

Figure 6. Simulation Server Display: (a) the Build Mode panel, and (b) the Simulation Mode panel.

Alternatively, a filename can be manually entered into the filename text box, which by default contains the text “Sim_Log.txt”. If no customized log file is specified, all data will be logged to a file with this default name. This file will be overwritten for each new simulation run, if an alternative file is not created.

Once all of the data files have been loaded or created, an experimental scenario can be run from the Simulation Mode panel. Unless the scenario is being run as an offline process, the Simulation Server must first be connected to the SharedDictionary by selecting the “Connect” button in the upper left corner of the panel. The URL and port “tcp://localhost:hello” (the default URL) should be entered into the connection dialog that appears. Next, each team display that is being used in the experimental scenario should be opened and also connected to the SharedDictionary via the “Connect” button on each respective display (the URL and port of the server computer should be used, e.g. “tcp://boeingserver:hello”). Finally, the “Start” button in the Simulation Mode panel should be selected to initiate the scenario simulation. Once a scenario has been started, it can be paused at any time by selecting the “Stop” button in this panel. A scenario can be restarted from the beginning by selecting the “Restart” and then “Start” buttons in sequence³.

When an experimental scenario is running, the Simulation Mode panel provides a view of the team’s AOI map, similar to the view provided in the Map Display (described below). However, on this display, all world entities in the AOI are shown, even targets that the UAV team has yet to discover (Figure 6b). This enables the experimenter to remain updated on the expected scenario events and participants’ mission progress. This panel also provides a view of the Convoy Threat Summary timeline that is provided on the Map and Mission Status Displays, as well as the operator performance panel that is provided on Mission Status Display. These visualizations are provided on this panel to facilitate pre-experiment scenario testing when the Simulation Server is being run as a standalone application. During this activity, the experimenter will test the scenarios to ensure that they provide the desired situation complexity. These visualizations also enable the experimenter to easily monitor the status of the scenario during experimental trials.

Finally, the Simulation Model panel enables scenarios to be run faster than real-time during pre-experiment scenario testing by providing a Simulation Rate slider⁴.

4.2.2 Functionality Common to All Team Displays

For experimental purposes, all of the team displays in the experimental platform (Map Display, Mission Status Display, Remote Assistance Display, Mission Commander Interface, and Operator Display) provide several features designed to assist the experimenters and participants in understanding their temporal progress within the current mission and the state of the connected displays. In particular, all team displays provide a Mission Clock that depicts an updated view of the elapsed and current mission times located at the top of each display. Additionally, all team displays provide a “Connect” button that enables the application to be

³ The Restart feature currently does not correctly reset all data items in the SharedDictionary. It is highly recommended that all applications, including the Simulation Server, be reopened between experimental trials.

⁴ Due to computational and redrawing inefficiencies in the current software, the Simulation Rate feature currently does not provide significant simulation speed-ups. Switching to the Build Mode panel while a scenario is running tends to provide a slight increase in simulation rate.












connected to the SharedDictionary during the setup phase of any experimental scenario. The current connection status is displayed in a small adjacent textbox. When connecting to the SharedDictionary, the team displays should use the URL and port that corresponds to the computer currently running the Simulation Server (“tcp://boeingserver:hello” is currently the default, which corresponds to the current Simulation Server machine).

4.2.3 Map Display

The main purpose of the Map Display, shown in Figure 7, is to provide an up-to-date view of the main mission assets (e.g., convoy, UAVs, targets) in the context of the UAV team’s AOI. The symbology used on this display is primarily based on standard military display symbology from MIL-STD-2525B (DOD, 1999), modified to satisfy the information requirements generated by the cognitive task analysis (CTA) conducted on the UAV task scenario (Scott et al., 2006; Scott et al., 2007).

In particular, the map symbology is designed to dynamically change through the mission to enhance the mission commander’s awareness of possible threat and operator performance issues. For example, areas of the map which have not yet been surveilled are indicated by a semi-transparent black overlay. When a UAV surveils an area, its overlay is cleared. Thus, the current surveillance progress across the UAV team is indicated by the relative amount of clear and black areas in each operator’s AOI. Though not currently supported, ideally these areas would fade back to black as time passes and the surveillance data ages (Bisantz et al., 2006). Table 1 describes the current map symbology.

Table 1. Map Symbology used throughout the team displays.

Map Entity	Symbology		
Convoy			
Convoy route			
UAV	 (nominal)	 (reviewing ATR imagery)	 (down)
Targets	 (identified)	 (potential target detected)	 (destroyed)
AOI boundary	 (nominal)	 (request for assistance received from operator)	 (OPL critically low)

ATR = automatic target recognition
 WR = weapons range
 OPL = operator performance level

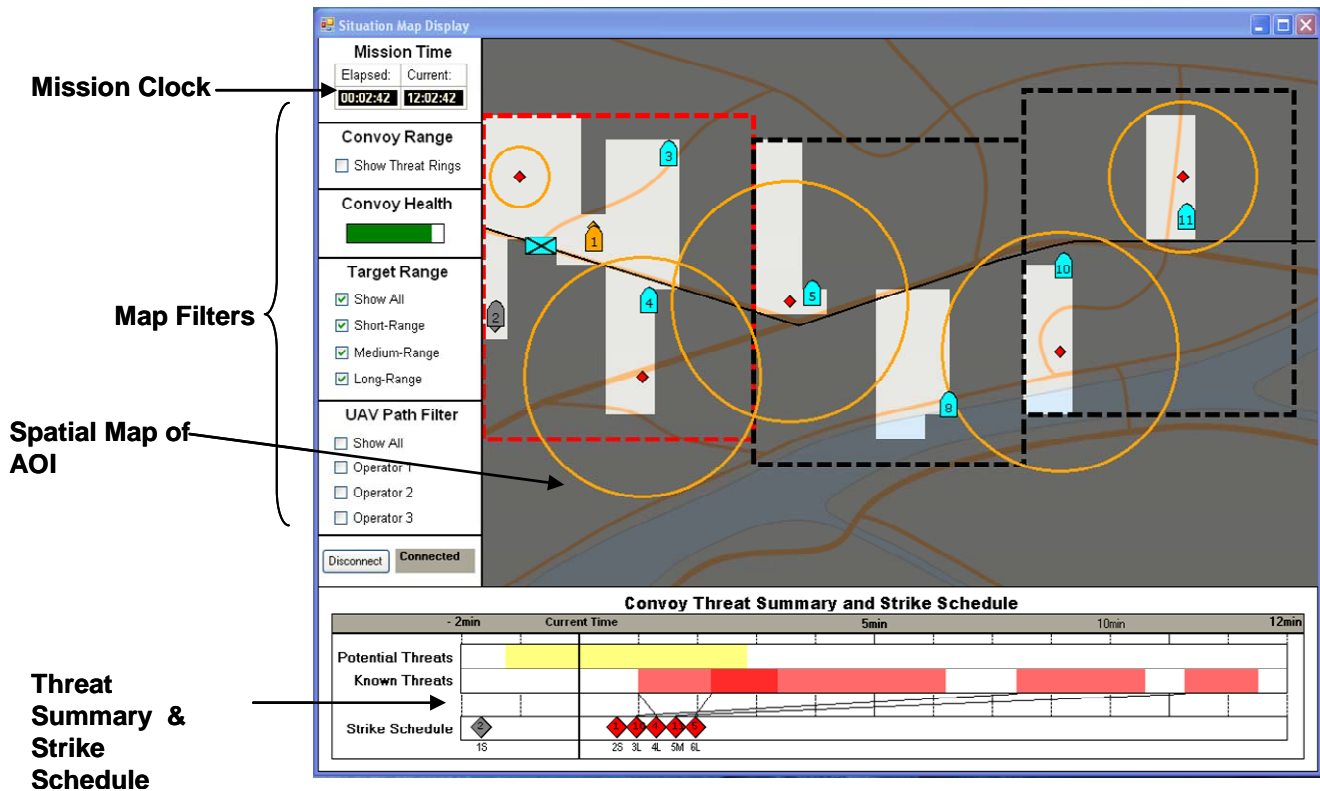


Figure 7. Map Display.

When an operator is in the process of confirming a possible target detected by a UAV’s onboard automatic target recognition (ATR) system, an orange target symbol is displayed on the map in the location of the detected target and the UAV that detected the target is displayed as orange. When the operator has finished confirming the target, the UAV returns to its nominal blue color and the target is displayed as red, indicating a known threat.

The color of the AOI boundaries also changes depending on operator performance, which is tracked and displayed in more detail on the Mission Status Display. A black boundary indicates the corresponding operator is expected to meet their ISR responsibilities. That is, the operators are predicted to surveil areas in their AOI that are within at least long range weapons range of the convoy in the near future. If the system detects a critical drop in operator performance, their AOI boundary will change to red. Currently, critically low operator performance indicates that an operator has significantly fallen behind schedule in checking areas directly along the route of the convoy, perhaps due to UAV losses. Also, if the operator requests command assistance with their current tasking (currently implemented for assistance with target identification tasking only), their AOI boundary will change to orange.

The Map Display also provides various view filters to enable the mission commander to show or hide extra display information, as needed during the mission. These filters include “Target Range,” which displays the weapons range rings around identified targets, “UAV Path,” which displays the planned route for the team’s UAVs, and “Convoy Range,” which displays a set of rings centered on the convoy’s position the correspond to short, medium, and long range weapons distance ranges expected in the area.

The Map Display also provides an up-to-date Threat Summary timeline below the AOI map (Figure 8). This timeline indicates when the convoy is or is expected to be in range of any un surveilled areas (i.e., a potential threat, shown as a yellow time window) or in range of a known threat (shown as a red time window). These time windows will be referred to as threat envelopes, that is, durations of time in which the convoy will be in potential or known threat situations.

The timeline also shows the up-to-date target strike schedule in the context of the current and expected convoy threats. Known threats are shown as red diamonds in the last row of the timeline. The position of a known threat on the timeline indicates the scheduled time when it will be destroyed by the external strike team. If the convoy is or is expected to be within weapons range of a known threat, a black line is displayed between the target’s symbol in the strike schedule and the beginning of its corresponding threat envelope in the preceding row.

Since humans are adept at perceiving differences in line angles (Ware, 2000), this connector line creates an emergent feature to help the mission commander identify off-nominal situations when a threat strike will not happen before the convoy will be within its weapons range. For example, when the mission commander sees a threat connector line at a vertical angle or sloping downwards to the right (e.g., the strike will be later than the convoy’s arrival within the threat’s weapons range), they should take action to delay the convoy and let the strike team destroy the threat before the convoy is allowed to continue.

4.2.4 Mission Status Display

The Mission Status Display shows various types of information designed to provide the mission commander current and expected status of the UAV operators’ task performance, the convoy’s safety level, and the UAV team’s communication connections to remote contacts (Figure 9). The CTA highlighted the importance of supporting the mission commander’s analysis of the ongoing temporal relationships between the UAV team’s activities and the convoy’s safety; thus, the critical status information presented on this display is provided in the form of timelines and time graphs that show the current situation, along with the recent history and the expected future status of mission related data.

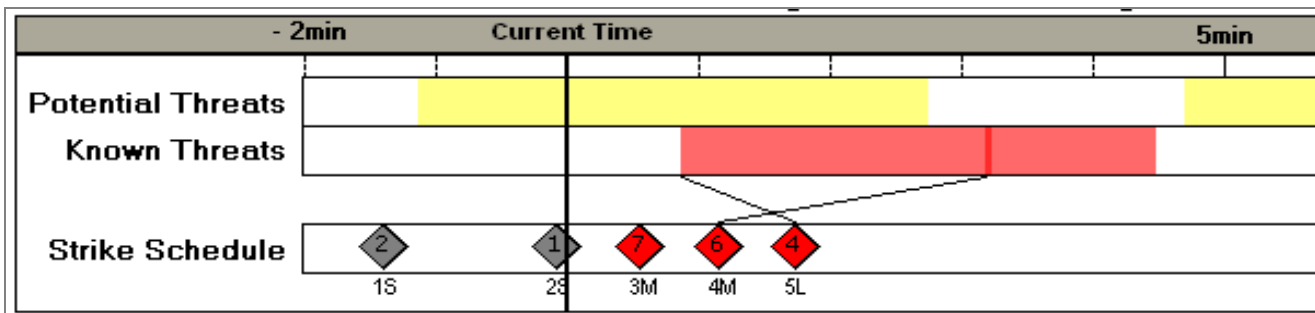


Figure 8. Strike schedule example: Threat 4M is scheduled to be destroyed 2 minutes *before* the convoy will be within its weapons range. Threat 5L is scheduled to be destroyed 1 minute *after* the convoy will be within its weapons range. Threat 3M is far enough away from the convoy’s route that the convoy is not expected to pass within its weapons range, thus no corresponding “threat window” is shown.

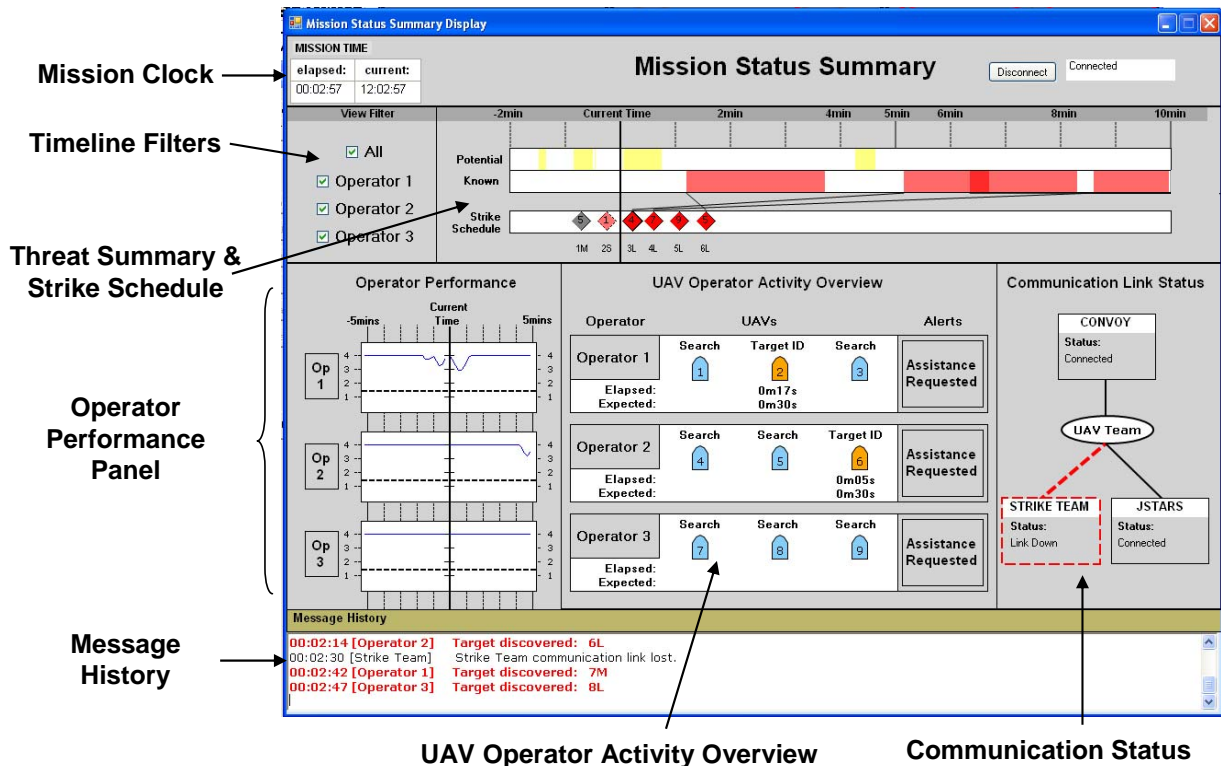


Figure 9. Mission Status Display.

In particular, the Mission Status Display contains a Convoy Threat Summary timeline (mirrored on the Map Display as described above) and Operator Performance time graphs. The Operator Performance time graphs show the current and expected operator performance, currently based on each operator’s ISR performance and its current and expected impact on convoy safety. In particular, each point on the graph indicates the corresponding operator’s ISR performance for the previous 30 seconds (for points in the future, this calculation is based on expected performance, based on current surveillance patterns, and may change if the operator subsequently detects a target). If an operator’s ISR performance begins to degrade, putting the convoy’s safety in jeopardy, the operator’s performance score decreases. When an operator’s performance is or expected to become critically low (i.e., their surveillance performance is putting the convoy in eminent danger of being attacked), the alert to the left of the corresponding time graph will turn red. Also, the corresponding operator AOI boundary will turn red in the Map Display. The Operator Performance score is calculated using a four-point scale (4 = high performance, 1 = low performance), which is based on the following heuristic:

- 4: Probable convoy safety level is high - all areas within expected weapons range have been or are expected to be surveilled
- 3: Probable convoy safety level is somewhat uncertain - convoy is within or expected to be within long-range weapons distance to unsurveilled areas
- 2: Probable convoy safety level is very uncertain - convoy is within or expected to be within medium-range weapons distance to unsurveilled areas
- 1: Probable convoy safety level is low - convoy is within or expected to be within short-range weapons distance to unsurveilled areas

The Mission Status Display also provides a visual summary of the current tasking of each of the team’s UAVs, arranged by UAV operator. For example, when a UAV and its operator are engaged in target identification activities, the symbol representing that UAV will turn orange, corresponding to the UAV symbol color change on the Map Display. This view also provides timing data to assist with supervisory-level decision making. For example, when a UAV and its operator are engaged in target identification, the time-on-task information is given, as well as the estimated time this task should take.

This UAV tasking panel also provides an “Assistance Request” alert that corresponds to the Remote Assistance Display (see Section 4.2.5). When that display is in use, the synthetic task environment can simulate a request from a remotely located (currently simulated) operator for help with the target identification activity that is performed whenever a UAV detects a potential target. When such a request is received, the “Assistance Request” alert corresponding to the sending operator will turn orange. Also, the corresponding operator AOI boundary will turn orange in the Map Display.

The Mission Status Display also provides an up-to-date view of the UAV team’s current connection status to the external contacts. When the UAV team is connected to all external contacts, the connecting lines between the UAV team icon and the contacts are shown as solid black lines. When a communication link is lost, the corresponding connecting line is shown as a dashed red line and the corresponding contact icon is also outlined in a dashed red line.

Finally, the Mission Status Display contains a message history box, which displays communication messages sent to the mission commander from team members and external contacts, as well as status messages from the system. Messages are displayed in the following format:

```
<time> [sender] <message text>
```

4.2.5 Remote Assistance Display

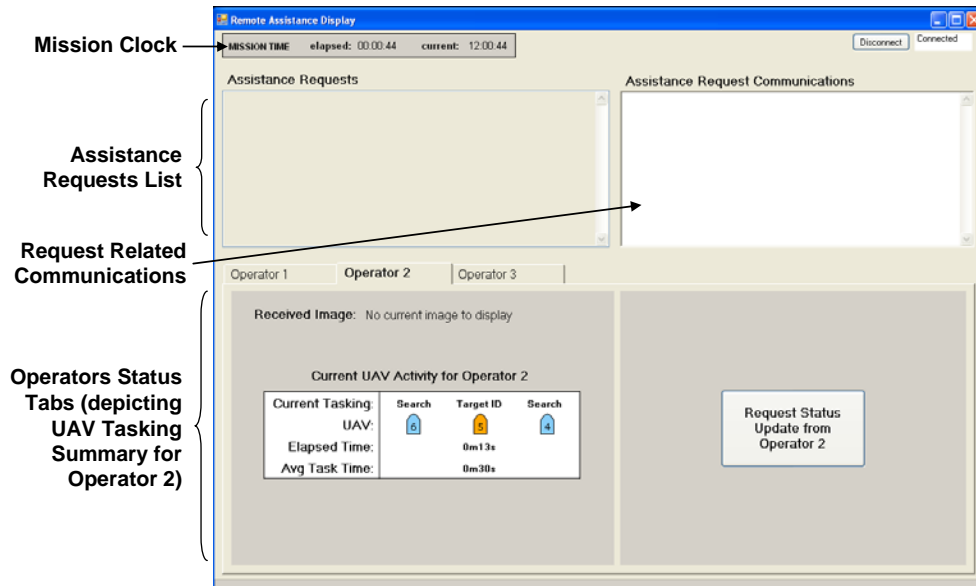
The Remote Assistance Display enables the mission commander to monitor the current UAV related activities of remotely located (current simulated) operators, as well as assist an operator with a target classification if they are having difficulties (Figure 10). This display consists of three main components, a list of assistance requests, a request-related communications message box, and the operator tab panels. Completed requests remain in the requests list, in the order of arrival, and are grayed out and labeled as completed.

When there are no outstanding requests from an operator, their corresponding tab panel displays a visual summary of that operator’s UAV activity (similar to the information provided on the Mission Summary Display). This view also provides a button, “Request Status Update from Operator X,” which enables the mission commander to request a status update from that operator (Figure 10a). For experimental purposes, selecting this button (when at least one of the operator’s UAVs engaged in target identification) triggers the Simulation Server to send an assistance request (via the SharedDictionary)⁵. When the assistance request arrives, it is added to the requests list and the operator panel is populated with details of the assistance request (Figure 10b). These details include the imagery containing the potential target detected by the UAV’s onboard ATR system, contextual information about this image (the UAV that captured the

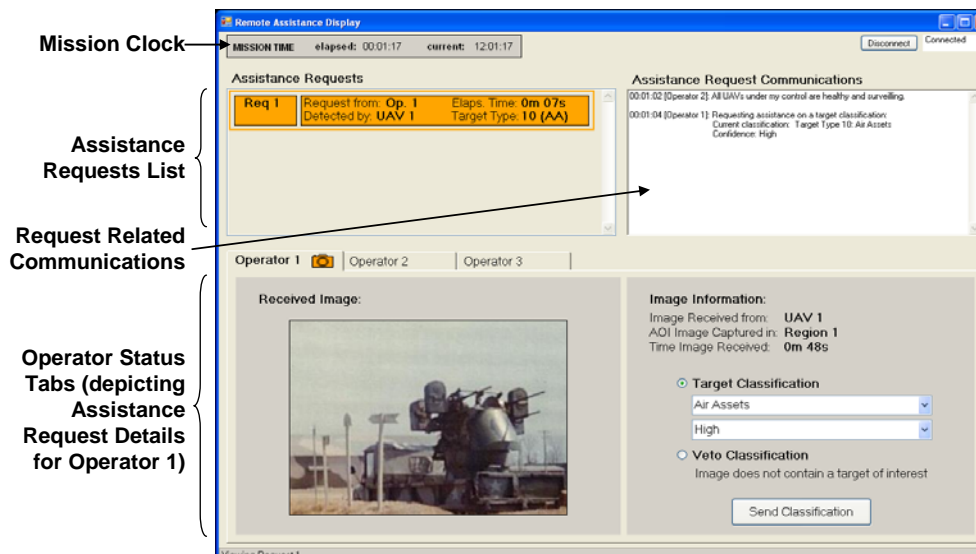
⁵ Assistance requests are created using target details from the Targets XML data file discussed in Section 2.6.1.

image, the location of the UAV, and the time the image was captured), and the operator's proposed target classification and the confidence level of that classification.

The mission commander can confirm and return the classification and confidence level to the operator by selecting the "Send Classification" button. Alternatively, they can change these values from the available drop down lists before returning the classification to the operator. Finally, they can veto the classification, indicating that they do not believe the image contains an immediate threat to the convoy (e.g. it may contain a farmer's tractor). When a remote operator has an outstanding assistance request, a camera icon appears in their corresponding tab label (Figure 10b). The currently displayed request is highlighted in orange in the requests lists, and the camera icon is colored orange in the tab label. Otherwise, the request list entries and camera icons appear as dark gray.



(a)



(b)

Figure 10. Remote Assistance Display.

4.2.6 Mission Commander Interface

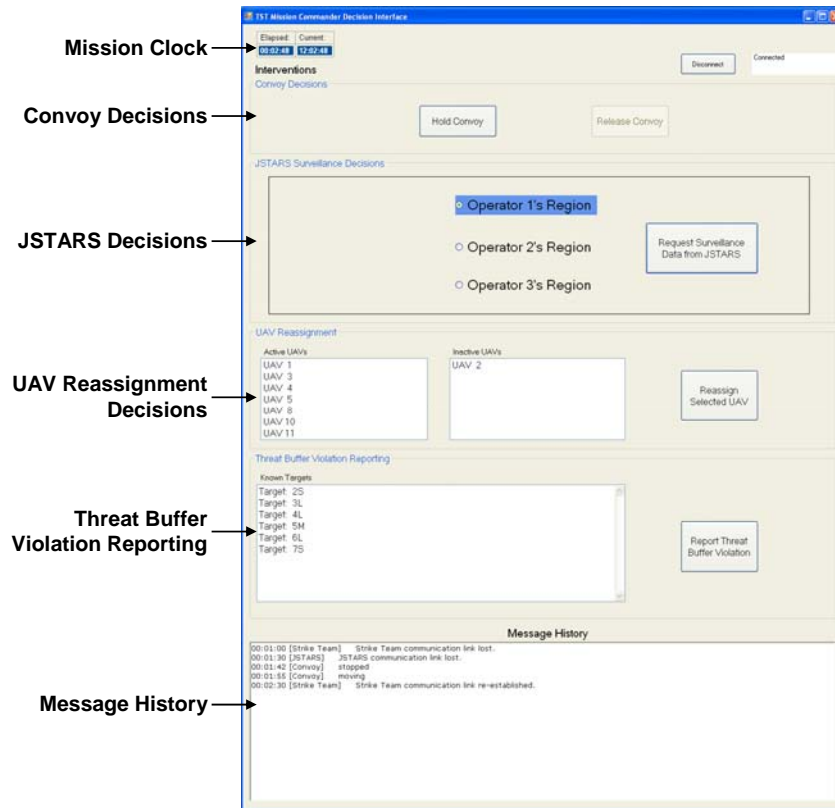
The Mission Commander Interface, shown in Figure 11, is used by the mission commander to execute mission decisions in the synthetic task environment. These decisions include:

- holding or releasing the convoy’s current position,
- requesting additional surveillance data from a nearby JSTARS (Joint Surveillance and Target Attack Radar System), a multi-sensor aircraft, within range of the team’s AOI, and
- reassigning a UAV from one part of the AOI to another to replace a downed UAV.

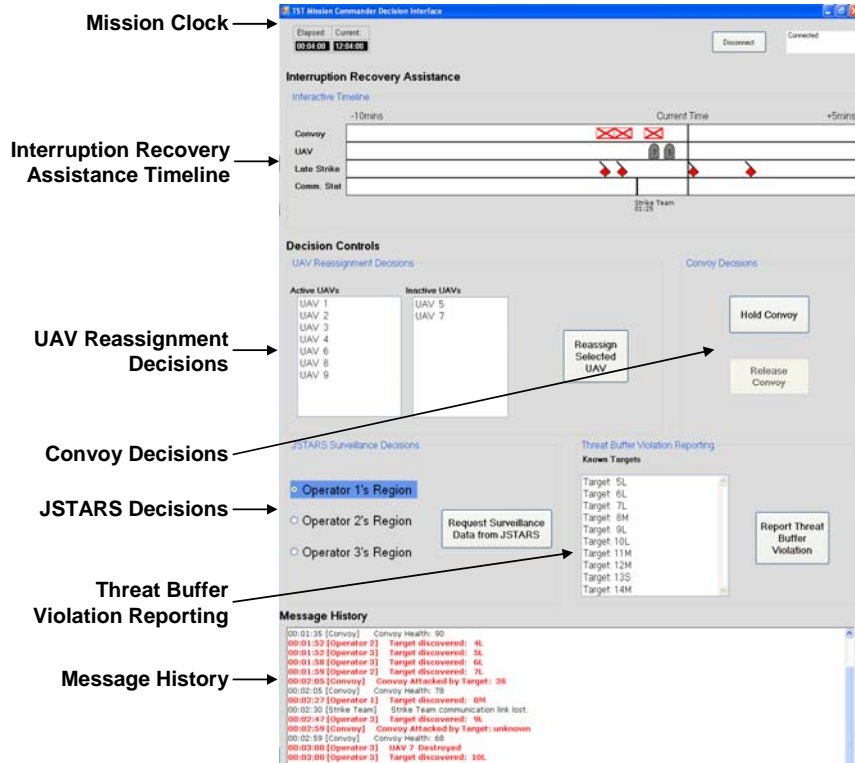
This display also enables the mission commander to report any “threat buffer violations” that occur during the mission. A threat buffer violation is an instance of scheduled target strikes such that the strike is expected to occur after the convoy has passed within weapon’s range of that target (also referred to as a “late strike”).

Similar to the Mission Status Display, a message history is provided at the bottom of this display. This message history is a simple text log of all events that occur during the mission (e.g., convoy attacks, UAV tasking, and communication link failures). The above functionality is provided on the basic version of the Mission Commander Interface (Figure 11a).

The experimental testbed also includes an alternate version of this display that includes an additional timeline designed to support interruption recovery support to the mission commander in the UAV team task (Figure 11b). This timeline provides a visual summary, using iconic bookmarks, of the critical mission events that can occur in the synthetic task environment, including convoy attacks (first row), UAV attacks (second row), late strikes (third row), and communication link outages (last row). With the exception of communication link icons, selecting a bookmark on this timeline causes additional information related to the event to be displayed on the Map Display (e.g., selecting a Convoy attack bookmark results in a red, semi-transparent X to be displayed on the Map Display). Appendix C provides more details on this interruption recovery support functionality.



(a)



(b)

Figure 11. Mission Commander Interface: (a) basic version, and (b) with interruption recovery support.

4.2.7 Operator Display

The main purpose of the Operator Display is to provide limited functionality for supporting local UAV operators during experimental scenarios. This display essentially mimics the map-related functionality provided by the Map Display, filtered to show only the map of the currently selected operator's AOI (Figure 12). The current operator can be chosen from a drop-down list located on the control panel to the left of the screen. Similar to the Map Display, the operator can hide or show additional information on the Map by toggling the view filters (Convoy Threat Rings, UAV paths, Targets, and Target Ranges). While the majority of the UAV operator behavior is simulated by the Simulation Server (e.g. navigational control of the UAVs), this display enables human operators, played by experimental "confederates," to engage in a computer-based activity designed to emulate the image-based, target identification activity an actual operator would perform whenever one of their UAVs detected a potential threat. In particular, when one of the operator's UAVs' onboard ATR systems detects a potential threat, this display launches a map-based search program (City Search program, described in Crandall and Cummings, 2007).

Once the operator has completed the City Search task, the program closes, and the operator selects the "Done Hovering" button to release the UAV from its current hover position over the detected target. To facilitate this functionality, each UAV's "Auto Hover" attribute must be set to false when the Scenario files are being created (see Appendix D scenario creation details). When no UAVs are hovering, this button is grayed out and displays the text "No Hovering UAVs." In order to vary the search task over multiple experimental trials, different trial names (e.g., Practice, Trial 1, etc) should be selected from the Scenario Selector drop-down list. This supplies a unique list or city order to the City Search program for each new instance of target detection (i.e. each UAV hover event).

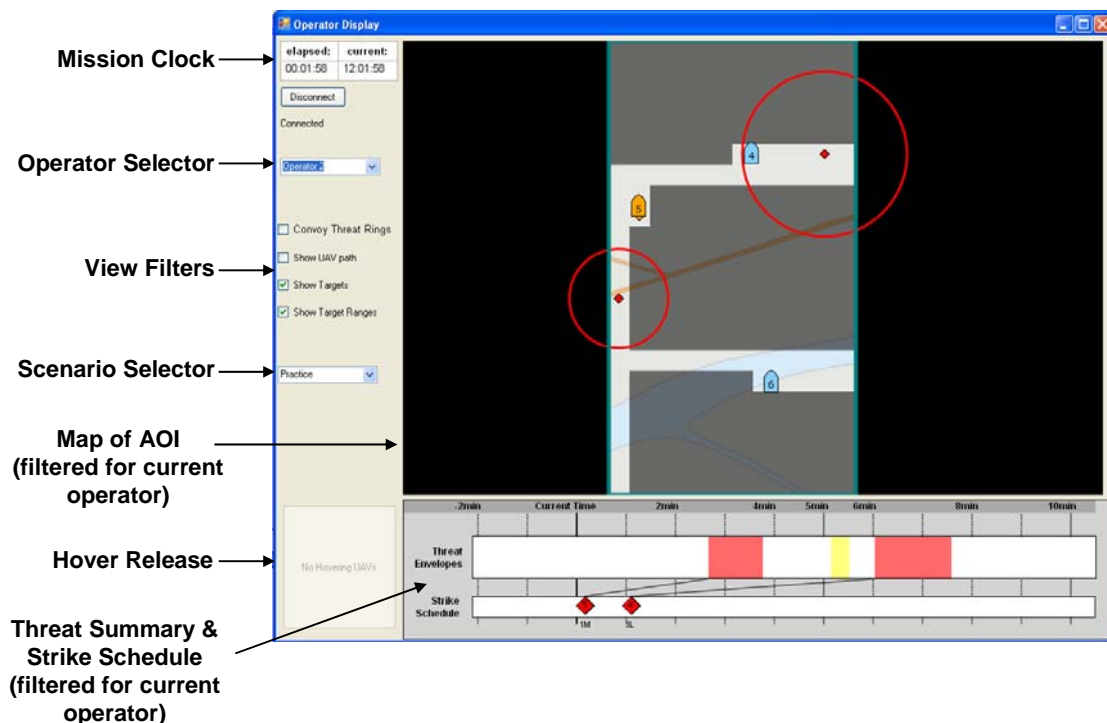


Figure 12. Operator Display.

5 Conducting a User Experiment

This section describes the process of conducting a user study in the experimental platform with the currently available software, as well as future software variants. Specifically, this section summarizes the process of developing a scenario, the current experimental platform instrumentation, participant training methods, and running the experimental software during experimental trials.

5.1 Scenario Development

In order to conduct a user study in the experimental platform, one or more carefully crafted software simulation scenarios are required. These scenarios define the simulated entities (e.g., UAV, targets) that will exist in the simulation environment, as well as the attributes of these entities (e.g. UAV speed, target weapons range).

Since the studies that are conducted in the experimental platform generally concern the investigation of interface technologies designed to facilitate individual and collaborative decision making and task performance in response to complex, time-critical task situations, it is essential that the study scenarios produce appropriately complex, time-critical situations to which participants much respond during the study trials. At the same time, it is important that the scenarios be design to limit the number of uncontrolled situations so the experimenters can be certain (as much as possible) of the cause of observed user behaviors during study trials.

As described in Section 4.2.1, the experimental platform includes functionality to facilitate scenario development as part of the Simulation Server application (Figure 6a). Appendix D details the process of creating a basic scenario using this application. Scenario files can also be created from scratch or by way of another custom made application, as scenarios can be loaded into the Simulation Server application from any XML file containing the correct entity format. Appendix E provides an example of a scenario file in XML format. Scenarios generated in the Simulation Server can be (and are recommended to be) saved to an automatically generated XML file. Therefore these scenarios can be reloaded in the future.

Due to the criticality of using well designed scenarios during user experiments, all scenarios should be developed iteratively. Researchers should schedule time for “debugging” experimental scenarios, not only for completeness and correctness, but in order to determine whether the scenario will produce simulated situations appropriate for the goals of the study. In particular, once the basic scenario components are developed, it is important to test the scenarios with all of the interfaces that will be used in the study, as opposed to running the Simulation Server in the “offline” mode that is convenient to use during basic entity and attribute creation. This will help the experimenter assess and become familiar with what the user will, or might, see during the study trials. Further, it is critical to pilot test the full scenarios with people outside of the research group who may be more likely to exhibit unexpected behaviors.

5.2 Instrumentation

In order to understand the impact of the proposed interface technologies in the experimental platform, data must be collected during study trials and analyzed for (statistical or logical) significance. The experimental platform is currently instrumented in a variety of ways to facilitate collection of both quantitative and qualitative data. Given that the goals of the anticipated studies in the experimental platform will likely focus on understanding the impact of technologies on individual team member system performance and on the overall teamwork

(communication, collaboration, coordination), the experimental platform provides instrumentation to capture user interaction directly *with the technology* and user behavior and interaction *around the technology*, including between human participants. More specifically, the experimental platform software automatically collects and stores any user interactions with the team displays, as well as critical system events to provide context for these actions. These data are stored as textual logfiles by the Simulation Server (see Appendix G for a sample data logfile). These logfiles can be analyzed for performance impacts of the technologies or experimental interventions under study. The experimental platform also enables real-time screen capture using external software, such as TechSmith's Camtasia, which can continuously capture the changing state of the team displays during study trials. This screen capture can be replayed later to facilitate data analysis or retrospective interviews (discussed below). One study performed in the experimental platform used a spare UAV operator console as a data capture computer, on which the large display interfaces were emulated and captured together in a single screen capture video file using Camtasia.

Video data can also be collected during study sessions using an available digital camcorder. A camcorder setup in one corner of the Command Center Laboratory enables video and audio capture of all participant interactions in the environment during study trials. These data help analysts determine the impact of the technology or experimental interventions on interactions between team members and interaction at the large displays. Using a wide-angle lens on the camcorder enables video capture of the entire laboratory space.

Finally, it is also useful to collect direct user feedback through questionnaires and post-experiment interviews. One method of collecting user opinions and clarifying observed user behaviors that has been useful in initial studies in the Command Center Laboratory is to conduct a retrospective interview with experiment participants after the study trials. This method involves conducting a semi-structured interview during which the experimenter shows the participant(s) a replay of the screen capture video that was captured during one or more of their experimental trials. This interviewing method is helpful for prompting participants' memories of their thought processes at different points during the study. It tends to evoke more specific user feedback about the task and interactions with the technology than standard interviewing techniques, which tend to evoke more generalized answers of past events.

5.3 Participant Training

Due to the relatively novel and complex tasks participants are asked to perform in the experimental platform, providing sufficient training is critical to the success of any experiment conducted in this environment. Effective training on the software, hardware, and expected task and collaborative activities is particularly critical in studies involving complex, time-critical tasks so that the researchers can be reasonable certain that the behaviors exhibited during study are a result of the technology or experimental interventions being evaluated. Insufficient training could result in participants reacting to some misunderstood aspect of the experimental environment, rather than reacting to the actual elements of the task to which the researchers assume they are attending.

An important aspect of providing "sufficient" training is to develop a training protocol that attempts to ensure that all participants in similar task roles achieve a similar level of task competency. This approach helps to ensure that any observed differences in individual or team

level task performance can be reasonably attributed to differences in individual or group working styles or experimental or technological interventions, rather than differences in task skill levels.

To date, participant training in the experimental platform has involved two stages. First, participants are asked to read, at their own pace, a computer-based tutorial that explains the experimental task, or “the mission,” and the mission goals that they are expected to meet during the study trials. This tutorial also details the task environment, technology, and the interface displays and their functionality (see Appendix H for an example Participant Training Tutorial for the experimental platform). Upon completion of the computer-based tutorial, participants then complete one or more hands-on practice sessions in the Command Center Laboratory. These sessions begin with a simplified task scenario, during which the experimenter calls attention to critical system features and task events and answers any participant questions. One or more additional practice sessions without the experimenter’s assistance should also be used to provide participants with further task experience. A useful experimental objective for these additional scenarios is to ensure that participants encounter all possible critical system events, possibly multiple times, so that they can gain practice reacting to those events prior to their actual test trials.

Further work is needed to develop a baseline test for core task competencies in the experimental platform that would help determine whether participants are ready to advance to the test trials. Currently, the experimenter’s best judgment is used; however, observations of several ineffective task performers in initial studies indicate that a base competency test would be quite beneficial.

5.4 Running the Experimental Software

Once one or more scenario files (and Targets data files if appropriate) have been created, the scenario can be run with the Simulation Server and applicable team displays⁶. Due to the distributed system architecture of the experimental platform, running the software requires the following multi-step process:

1. First, load a Scenario XML file (and Targets data XML file, if applicable) in the Simulation Server in the Build Mode tab, and then create a data log file if data capture is desired (Section 4.2.1).
2. Next, connect the Simulation Server to the SharedDictionary (SD) (Section 4.2.1) using the local URL and port (e.g. “tcp://localhost:hello”) in the Simulation Mode tab.
3. Next, open all team displays that will be used in the scenario (Sections 4.2.2-4.2.7) on their respective computer displays, and connect each display to the SD, using the URL and port of the Simulation Server computer (e.g. “tcp://boeingserver:hello”).
4. Finally, start the Scenario on the Simulation Server in the Simulation Mode tab (Section 4.2.1).

It is strongly recommended that the experimenter restart all software applications (Simulation Server and all team displays) between each experimental trial to ensure the proper data initialization, as the “Restart” button the Simulation Server does not reliably reinitialize the SD.

⁶ At the time of writing, all necessary executable files corresponding to the Simulation Server and team display applications were located in a “Boeing” folder on the desktop of all computers in the Command Center Laboratory.

6 Lessons Learned from Initial User Experiments

As mentioned above, several user studies have been conducted in the experimental platform. Our experiences with these studies provide several important lessons that may help improve future experimental efforts conducted in this environment.

First, the design of the scenario behavior (i.e., the world rules or logic) can significantly impact participant behavior during experimental trials. As most simulation-type game designers are well aware, “balance” in a simulated game environment between the user goals and incentives and the possible user activities in the world is critical to yield proper and effective user behavior. Consider, for example, designing the scenario behavior governing the loss of convoy health, which is the main visible “performance score” for the UAV team in the experimental platform. Typically the scenarios begin with the convoy at 100% health. There are currently two ways to lose health points: the mission commander holding the convoy in one position and a target attacking the convoy. If the amount of health lost in each situation is not carefully weighted, participants may actively choose to let the convoy be attacked by a target rather than holding it in place for a few minutes. Users quickly learn the incentive structure (i.e. point system) of the environment and how to “game the system” to maximize their scores. This tendency can lead to mismatches between what would realistically be considered good decision making (i.e., protecting the lives of those in the convoy) and what would lead to good game playing behavior (i.e., maximize convoy health). Pilot testing scenarios with a variety of possible scenario rules and considering such behaviors during scenario development can help experimenters set these values to produce “appropriate” behaviors during experimental trials.

A related issue concerns the creation of scenario behavior that is sufficiently complex and realistic in order to produce representative actions and decision making behavior for the type of tasks under investigation. The current scenario behavior is quite predictable, with limited behavioral uncertainty for scenario entities. Thus, participants rarely had to engage in the type of risk-taking decisions during experimental trials that would be quite common for real-world mission control activities. In the current simulation environment, for example, the convoy is always attacked when it passes within weapons range of a target, losing a similar amount of health (between 10 and 15 health points), regardless of how long it remains within range of that target. In reality, it is likely that the convoy would not always be attacked by each known threat, especially if the threat was located at some distance. Also, the longer the convoy was in range of a threat, that more damage it would likely sustain (i.e. convoy health it may lose). Predictable convoy attack behavior can again lead to participants gaming the system, rather than making decisions and taking actions appropriate to the task activities the experimental platform is attempting to emulate. This issue is discussed further in the following section.

As discussed in Section 5.3, another important lesson learned from initial studies in the experimental platform is the importance of providing adequate participant training. The environment and task is novel and relatively complex for most participants (despite the predictable entity behavior). Thus, it is essential to provide them with sufficient background information and hands-on practice so that they can perform the task at a reasonable competency level, to the extent that their behaviors during experimental trials can be confidently considered realistic and representative of expert behavior for this environment.

Finally, our experiences with the initial user studies revealed the importance of making participants feel engaged and part of a team during the experimental trials. This is particularly

important because the technology is designed to support and enhance teamwork. However, if participants feel they are performing the task alone, their behavior may not be representative of someone acting as part of a team in the environment. Said another way, we would essentially be testing the ability of the technology to support an individual, rather than our intention of testing its ability to support team members trying to accomplish a shared goal.

To this end, we have tried a variety of things to emulate a “team” environment, even though to date we have only tested individual participants. We have used confederates (i.e. members of the experimental team who have “played” team members using scripted behaviors). This method requires deception (participants are led to believe that the confederates are also experiment participants), which increases the complexity of the study protocol and complicates participant scheduling. Another approach we have used involved emulating “remote” team members, with whom a participant could indirectly “interact with” through the system by assisting with the task activities of those team members (see Section 4.2.5). Observations made of participants while employing these methods highlighted the importance of designing dependencies between the task activities and goals of the experiment participants and the activities and goals of the confederates or the simulated team members. If participants do not believe their actions impact, or are impacted by, their team members (real, confederate, or simulated) they tend to ignore their team members’ activities and begin acting independently.

7 Suggested Improvements

This section describes several recommendations aimed at addressing some of the issues discussed in the previous two sections. These recommendations relate to modifications to the experimental software and to the available training material and procedures.

7.1 Software Improvements

In order to invoke more realistic collaboration and decision-making behavior during experimental trials, it is recommended that changes be made to the current behavior governing the entities in the simulation environment. The following changes are recommended:

- *Introduction of uncertainty related to convoy attacks.* The level of certainty associated with whether or not a target attacks a convoy that passes within its weapons range may be based on the type of target (e.g., different target classifications could have different probabilities of attack) or on the weapons range of the target (e.g., the probability of attack may be higher for a short-range target versus a long-range target). For experimental control purposes, whether a target will attack the convoy should be deterministic (i.e., the same target will have the same behavior (attack or not attack) for all experimental subjects). However, having some targets attack and others not, governed some underlying logic that subjects can learn, can help convince them that the behavior is probabilistic. This type of uncertainty can currently be implemented by defining the appropriate convoy attack behavior for each defined target during scenario creation, based on manual calculations of probabilities for the targets in the scenario. Yet, more sophisticated methods could also be used by creating rules within the simulation code for governing attack behavior or for producing balanced scenarios during scenario creation.
- *Realistic penalties for poor task performance.* Currently, a convoy loses a predetermined amount of health points when it passes within range of a target, regardless of how long it remains within range of that target. Realistically, a convoy might continue to sustain

damage the longer it was within range of an enemy threat. The simulation code should be modified to reflect this by continuing to deplete the convoy health (or have some increased probability of getting attacked) the longer the convoy is within range of a target.

- *Modification to the UAV attack behavior.* Currently, the simulation environment provides uncertainty related to whether a target will attack a UAV by allowing the experimenter to define whether each target can attack UAVs. However, the simulation currently only attacks the first UAV it encounters. If a subsequent UAV passes overhead and the target is still active, it will not attack the UAV. To more accurately represent the real-world entities the simulation is emulating, targets able to attack a UAV should continue to attack UAVs that pass overhead as long as they are active (possibly with some uncertainty). This behavior will encourage participants to respect this known hazard area as a new spatial constraint for future planning until the target is destroyed.
- *Improved usability of the scenario creation functionality.* As described in Appendix D, creating or modifying a scenario for the experimental platform can be an arduous process, especially with respect to fine-tuning a scenario during pilot testing to ensure that appropriate situations are established during scenario runs to facilitate the study objectives. Significant improvements are recommended, especially related to defining and testing the temporal aspects of a scenario, in order to minimize the time needed for scenario development. In particular, a feature that enables an experimenter to anticipate, rather than through trial and error, the temporal impacts of specific changes to entities and their attributes would be particularly helpful for defining the tempo of a scenario.

7.2 Participant Training Improvements

Finally, as discussed in Section 5.3, improvements are warranted for the participant training procedures. Training in the experimental platform currently consists of tutorial-based training followed by several hands-on practice scenarios. However, observations from our initial studies indicate the need for one or more standardized benchmark tests to determine a sufficient level of task competency. Such adjustments to the training procedure would help ensure that participants are suitably ready to advance to the experimental trials as “task experts” and allow experimenters to be more confident in that participants’ task and system performances are suitably comparable for data analysis.

8 Conclusions

The experimental platform described in this report provides a software and hardware testbed to explore novel interface technologies aimed at supporting critical decision making and effective communication and coordination during complex, time-critical collaborative mission control operations. This platform is currently focused on investigating supervisory-level decision making during UAV Team Operations, however the underlying software and hardware architecture of the testbed could be extended to support investigations of other UAV team member roles, as well as investigations of other mission control operations. Possible expansions to this platform include redesigning the UAV Operator Displays to provide real (i.e. non-simulated) operator tasking behavior, development of “remote” operator stations as part of the physical laboratory configuration to enable investigation of distributed UAV team interactions between real experiment participants, and development of other military and non-military task scenarios to enable investigation of generalized mission control interface technologies.

9 References

- Bisantz, A. M., J. Pfautz, R. Stone, E. M. Roth, G. Thomas-Meyers and A. Fouse (2006). Assessment of Display Attributes for Displaying Meta-information on Maps. *Proceedings of HFES 2006: Human Factors and Ergonomics Society 50th Annual Meeting*, San Francisco, CA, USA, HFES.
- Boyle, M. (2005). Privacy in Media Spaces. *Department of Computer Science*. Calgary, AB, Canada, University of Calgary. Ph.D. Thesis.
- Boyle, M. and S. Greenberg (2006). GroupLab.Networking Library, University of Calgary.
- Crandall, J. W. and M. L. Cummings (2007). Developing Performance Metrics for the Supervisory Control of Multiple Robots. *Proceedings of the 2nd Annual Conference on Human-Robot Interaction*, Arlington, VA, USA, ACM Press.
- Dudfield, H. J., C. Macklin, R. Fearnley, A. Simpson and P. Hall (2001). Big is better? Human factors issues of large screen displays with military command teams. *Proceedings of People in Control (PIC'01): International Conference on Human Interfaces in Control Rooms, Cockpits and Command Centres* Manchester, UK, IEE.
- Inkpen, K., K. Hawkey, M. Kellar, R. Mandryk, K. Parker, D. Reilly, S. Scott and T. Whalen (2005). Exploring Display Factors that Influence Co-located Collaboration: Angle, Size, Number, and User Arrangement. *Proceedings of HCI International 2005*, Las Vegas, NV.
- Jedrysik, P. A., J. A. Moore, T. A. Stedman and R. H. Sweed (2000). Interactive displays for command and control. *Proceedings of IEEE Aerospace Conference*, Big Sky, MT, USA.
- Jenkin, C. (2004). Team Situation Awareness: Display Technologies in Support of Maritime Domain Awareness. *Proceedings of 7th Marine Transportation System Research & Technology Coordination Conference*, Washington, DC.
- Mandryk, R. L., S. D. Scott and K. L. Inkpen (2002). Display Factors Influencing Co-located Collaboration. *Extended Abstracts of CSCW'04: ACM Conference on Computer-Supported Cooperative Work*, New Orleans, LA, ACM Press.
- Osga, G. A. (2000). Task Managed Watchstanding - Concepts for 21st Century Naval Operations. *Proceedings of HFES'00: Human Factors and Ergonomics Society 44th Annual Meeting*, Santa Monica, CA, Human Factors and Ergonomics Society.
- Peel, R. R. and B. J. Williams (2004). Bringing A Common Operating Picture To Fleet UUV Exercises. *Proceedings of AUUSI Unmanned Systems North America 2004 Symposium and Exhibition*, Anaheim, CA.
- Russell, D. M., C. Drews and A. Sue (2002). Social Aspects of Using Large Public Interactive Displays for Collaboration. *Proceedings of UbiComp'02: Conference on Ubiquitous Computing*, Göteborg, Sweden, Springer-Verlag.
- Ryall, K., C. Forlines and C. R.-M. Shen, M. (2004). Exploring the Effects of Group Size and Table Size on Interactions with Tabletop Shared-Display Groupware. *Proceedings of CSCW'04: ACM Conference on Computer-Supported Cooperative Work*, Chicago, IL, ACM Press.
- Scott, S. D., A. E. Rico, C. Y. Furusho and M. L. Cummings (2006). Designing Decision and Collaboration Support Technology for Team Supervision in Multi-UAV, Time-Sensitive Targeting Operations. Cambridge, MA, USA, MIT Humans & Automation Lab.
- Scott, S. D., J. Wan, A. Rico, C. Furusho and M. L. Cummings (2007). Aiding Team Supervision in Command and Control Operations with Large-Screen Displays. *HSIS 2007: ASNE Human Systems Integration Symposium*, Annapolis, MD.

- Smallman, H. S., H. M. Oonk, R. A. Moore and J. G. Morrison (2001). The Knowledge Wall for the Global 2000 War Game: Design Solutions to Match JOC User Requirements. San Diego, CA, Pacific Science and Engineering Group.
- Wan, J. (2007). Assisting Interruption Recovery in Mission Control Operations. *Electrical Engineering and Computer Science*. Cambridge, MA, USA, Massachusetts Institute of Technology. B.S. Thesis.
- Ware, C. (2000). *Information Visualization: Perception for Design*, Morgan Kaufmann Publishers Inc.

Appendix A: Command Center Laboratory Configuration & Equipment Details

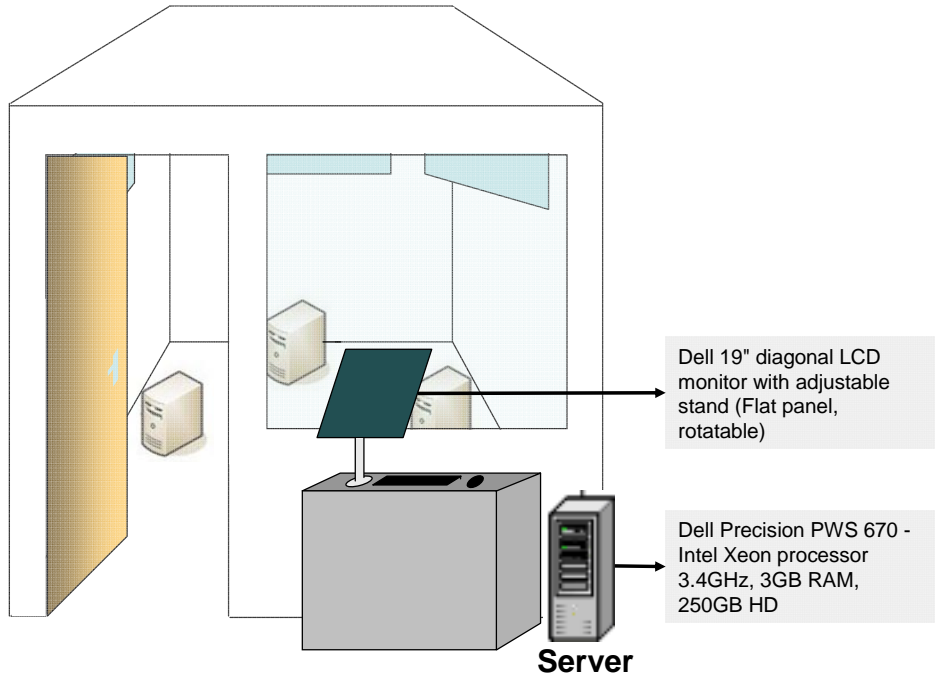


Figure 13. Command center laboratory: observation window and collaboration server / data analysis station.

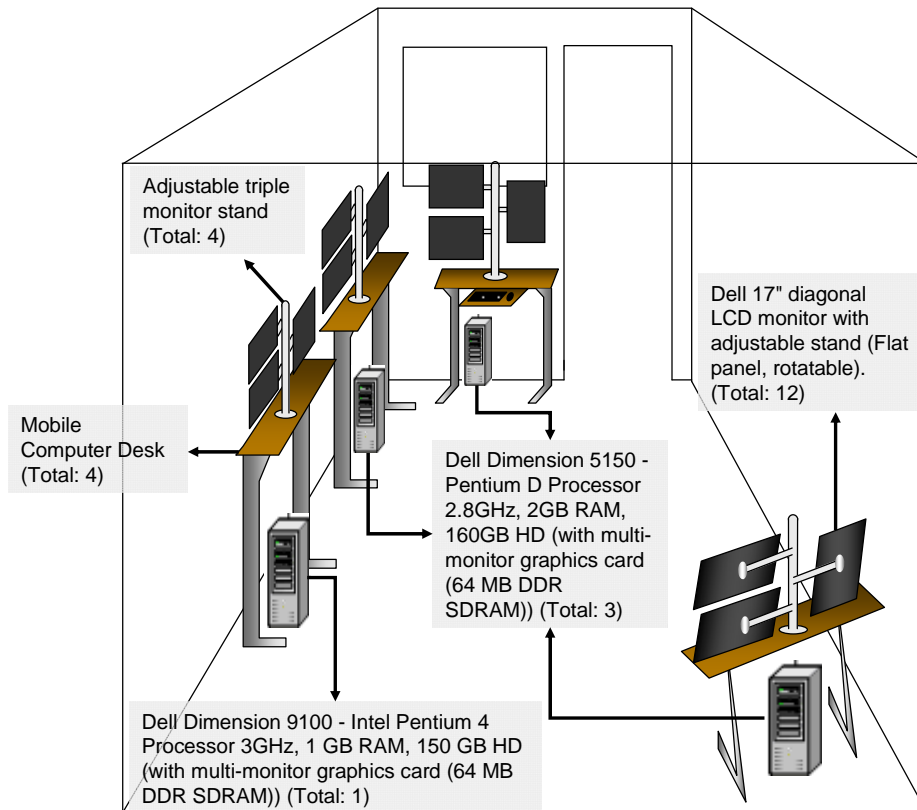


Figure 14. Command center laboratory: four reconfigurable operator consoles.

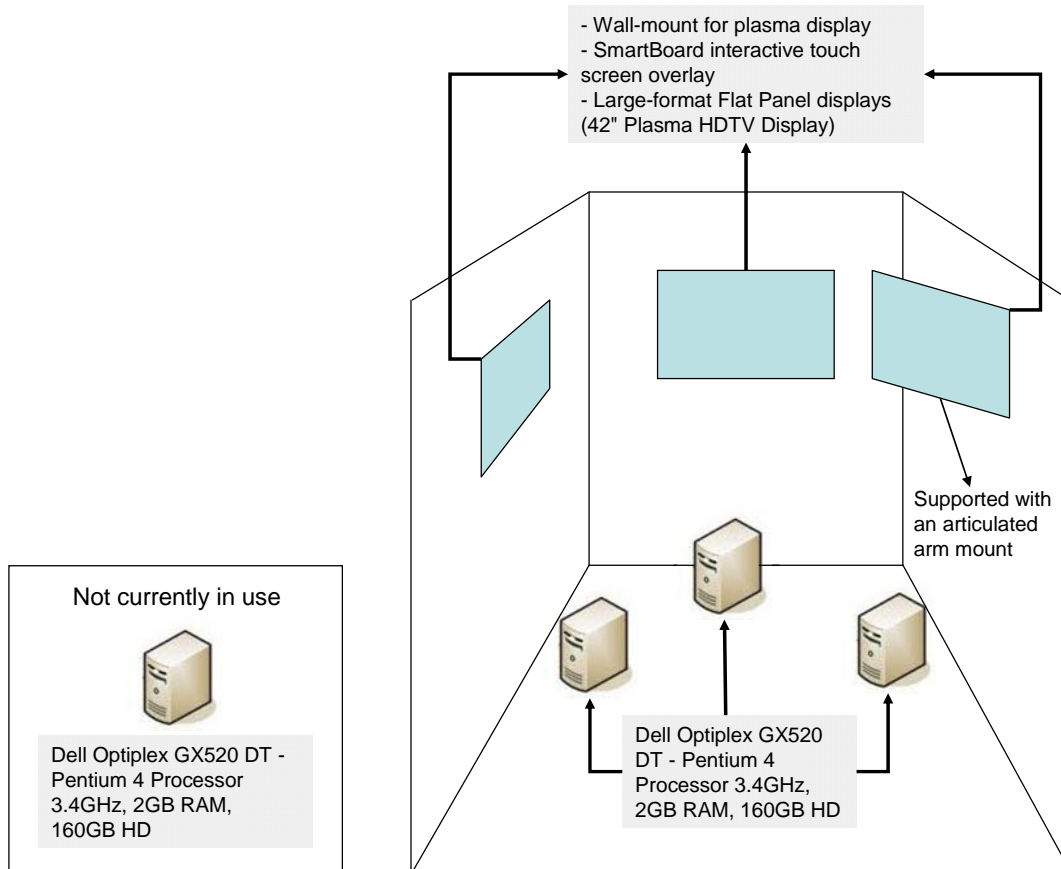


Figure 15. Command center laboratory: three large-screen wall displays, with display servers. There is one spare display server, not currently in use.



Figure 16. Command center laboratory: four mobile team workstations (2 tablets, 2 handhelds).

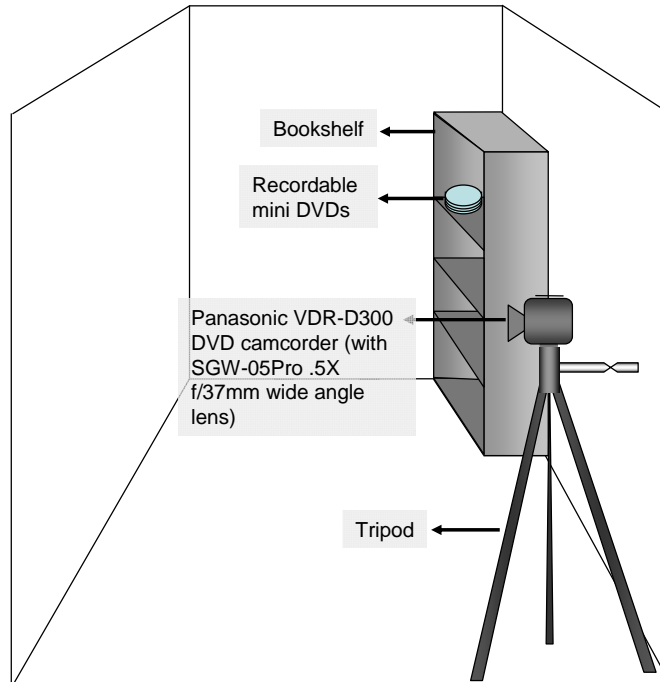


Figure 17. Command center laboratory: data capture equipment (digital video camera (Panasonic VDR-D300 mini DVD digital camcorder) and recordable mini DVDs).

Appendix B: Conceptual Design Diagram

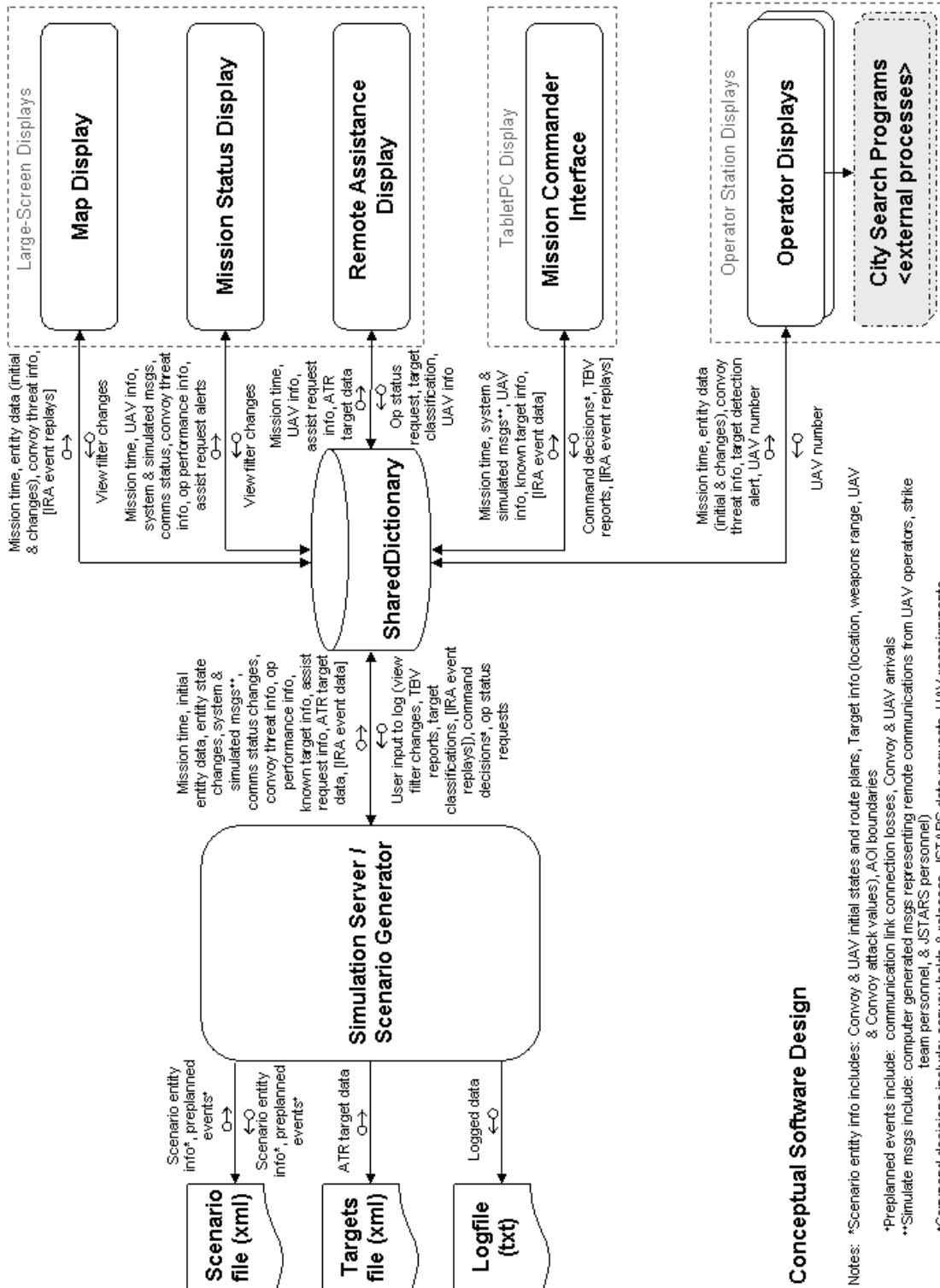
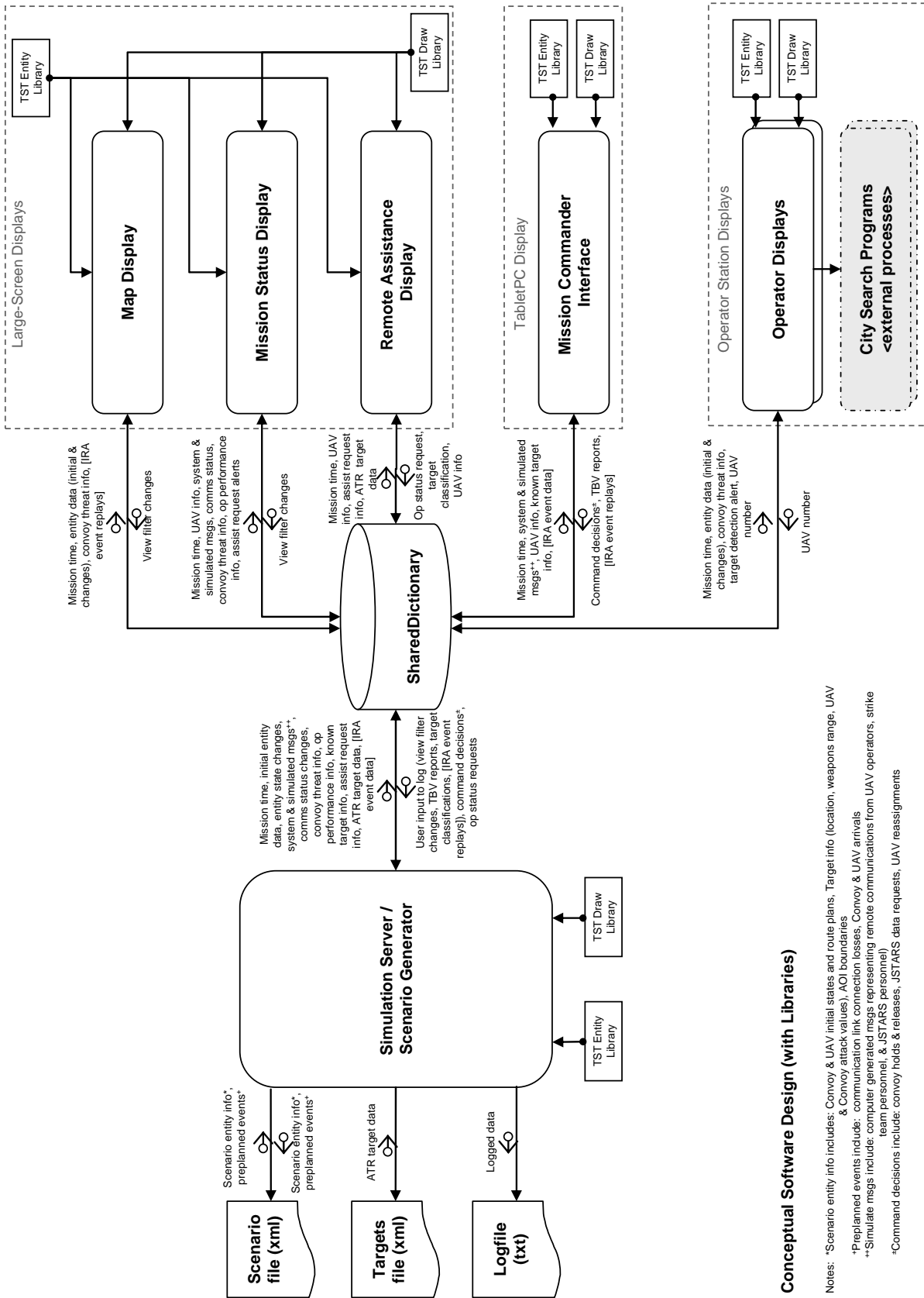


Figure 18. Expanded view of the conceptual design of the experimental platform software (Figure X in the main text).



Conceptual Software Design (with Libraries)

Notes: *Scenario entity info includes: Convoy & UAV initial states and route plans, Target info (location, weapons range, UAV & Convoy attack values), AOI boundaries
 *Preplanned events include: communication link connection losses, Convoy & UAV arrivals
 **Simulate msgst include: computer generated msgst representing remote communications from UAV operators, strike team personnel, & JSTARS personnel
 *Command decisions include: convoy holds & releases, JSTARS data requests, UAV reassignments

Figure 19. Conceptual design of the experimental platform software, including software libraries.

Appendix C: Interruption Recovery Support Functionality

As discussed in Section 4.2.6, when event bookmark icons are selected on the Interruption Recovery Assistance timeline corresponding to the Convoy Attacked, UAV Destroyed, and Late Strike events, additional information is displayed on the Map Display. This information is displayed for five seconds, and then fades to reveal the current state of the map. This time frame was selected based on pilot tests, which indicated five seconds was long enough for someone to select the bookmark from the tablet PC and look to the Map Display to see the change on the map. The respective change to the Map Display was available an additional few seconds as the information was fading back to its normal state. Figures 17-19 show the details of these interactions.

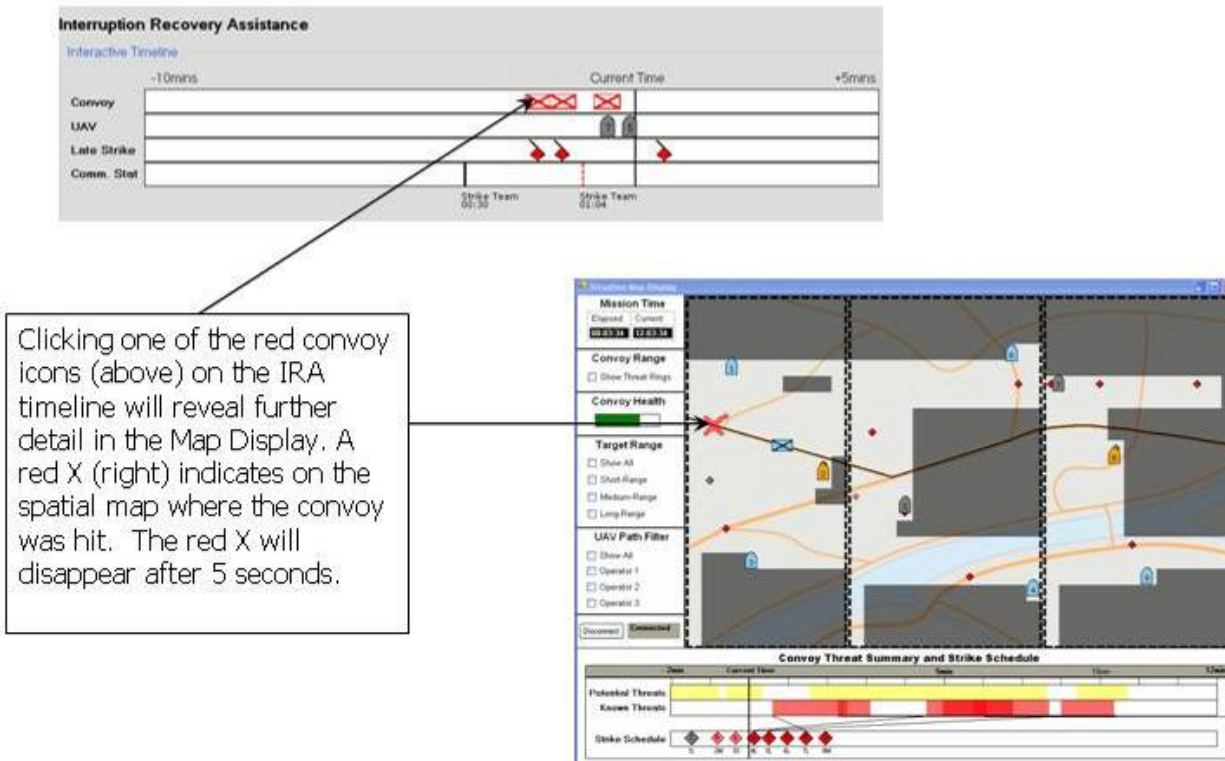
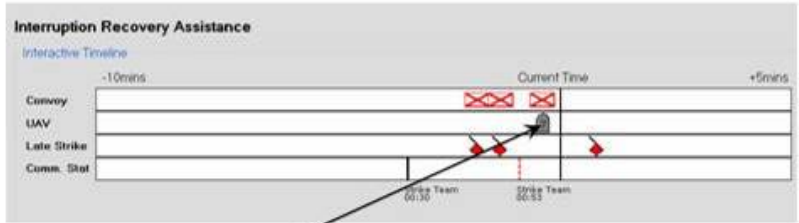


Figure 20. Revealing the Convoy Attacked event information on the Map Display (from Wan, 2007).



Clicking on the gray UAV icon (above) will reveal further details in the Map Display. A transparent red X (right) will appear on the spatial map to indicate which UAV was destroyed. The red X will disappear after 5 seconds.

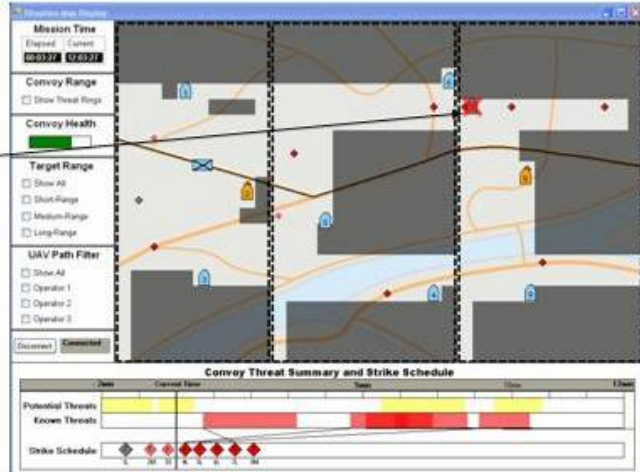
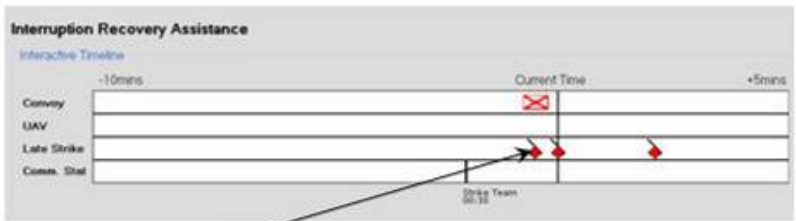


Figure 21. Revealing the UAV Destroyed event information on the Map Display (from Wan, 2007).



Clicking on the late strike icon (above) will reveal further details on the Map Display. A black box (right) will appear on the Strike Schedule panel to indicate which target will be able to attack the convoy if the strike team is not given additional time. Also, the other targets will be dimmed to give emphasis. This box will disappear after 5 seconds.

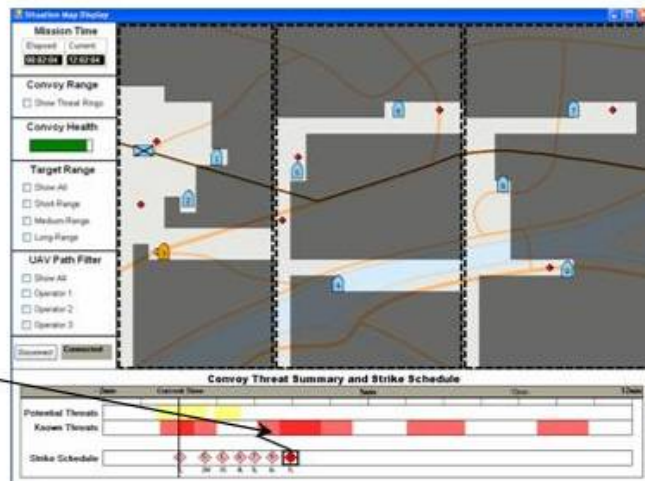


Figure 22. Revealing the Late Strike event information on the Map Display (from Wan, 2007).

Appendix D: Generating a Scenario for the Experimental Platform

This appendix provides instructions on developing a scenario to run in the Simulation Server. If an XML scenario file is already available, it can be loaded into the Simulation Server by selecting the “Load” button at the top of the Build Mode screen (Figure 23) and then selecting the desired file from the Open XML File dialog box. Once loaded, a scenario can be updated by following the scenario development instructions below. Scenario modifications can then be saved by selecting the “Save” button at the top of the Build Mode screen. The scenario file can either be overwritten, if the same filename is provided, or saved as a new scenario by providing a new filename in the Select Save File dialog box.

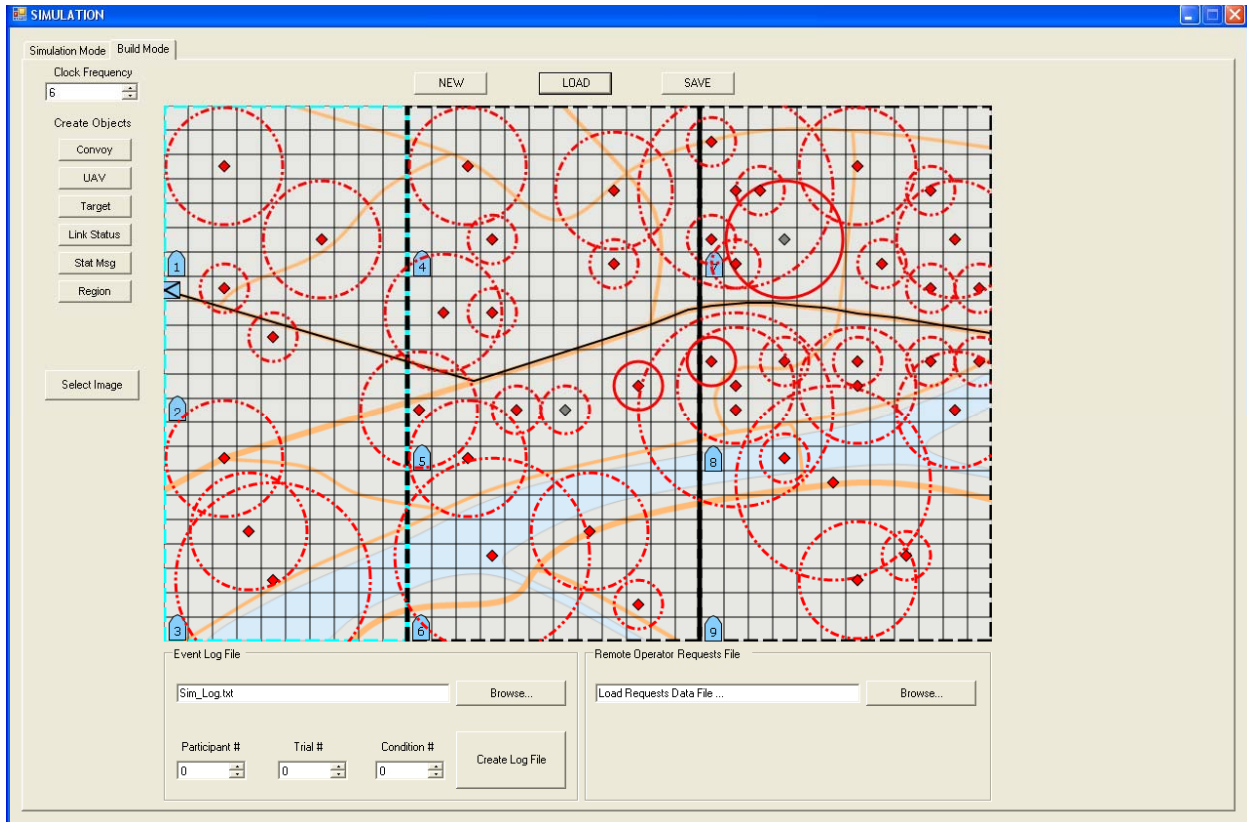


Figure 23. Simulation Server, Build Mode screen with a loaded scenario file.

Scenario Development

To create a new scenario file, select the “New” button at the top of the Build Mode screen. If a scenario file is currently loaded, this action will cancel any unsaved changes and produce a blank scenario. Next, follow the steps below to create the desired scenario entities and events and set their corresponding attributes to produce the desired simulation behavior.

Note, in the following discussion entities are spatially defined (e.g., regions, convoy, UAVs, targets) and events are temporally defined (e.g., status messages, communication link losses). To facilitate real-time calculations while running a scenario simulation, the scenario map is computationally represented as a grid of squares. Each square, called a grid node, currently represents a 25x25 pixel on screen area. During the simulation, entity moves typically occur from grid-square to grid-square, since pixel-by-pixel movement is computationally demanding.

Thus, if an entity, such as a UAV, is defined to move once a clock tick, then it will move the distance of one grid square each clock tick. If an entity, such as the convoy, is defined to move only every 6 clock ticks, then it will move the distance of one grid square every 6th clock tick. This enables relative entity speeds to be simulated.

Regions (Required)

A “region” is an area of interest associated with a specific UAV operator in the UAV team. By default, Region 1 is associated with Operator 1, Region 2 with Operator 2, and Region 3 with Operator 3, regardless of their geospatial size or location. Any UAV defined within a region is assumed to be controlled by the Operator associated with that region, with the exception of a UAV transiting through a region en route to another region during a UAV reassignment.

To define the UAV team’s regions, select the “Region” mode button on the left side of the Build Mode screen (Figure 24). Then, define the location and size of the region by dragging a rectangle in the map panel. Region boundaries are automatically snapped to the map grid lines during this process. A defined Region cannot be modified, but instead must be deleted and recreated. Regions also must be created in order, and the Simulation Server program assumes the existence of **three (3)** regions at runtime. Fewer defined regions will cause runtime errors. This scenario constraint exists to facilitate calculations related to operator performance within each associated region.

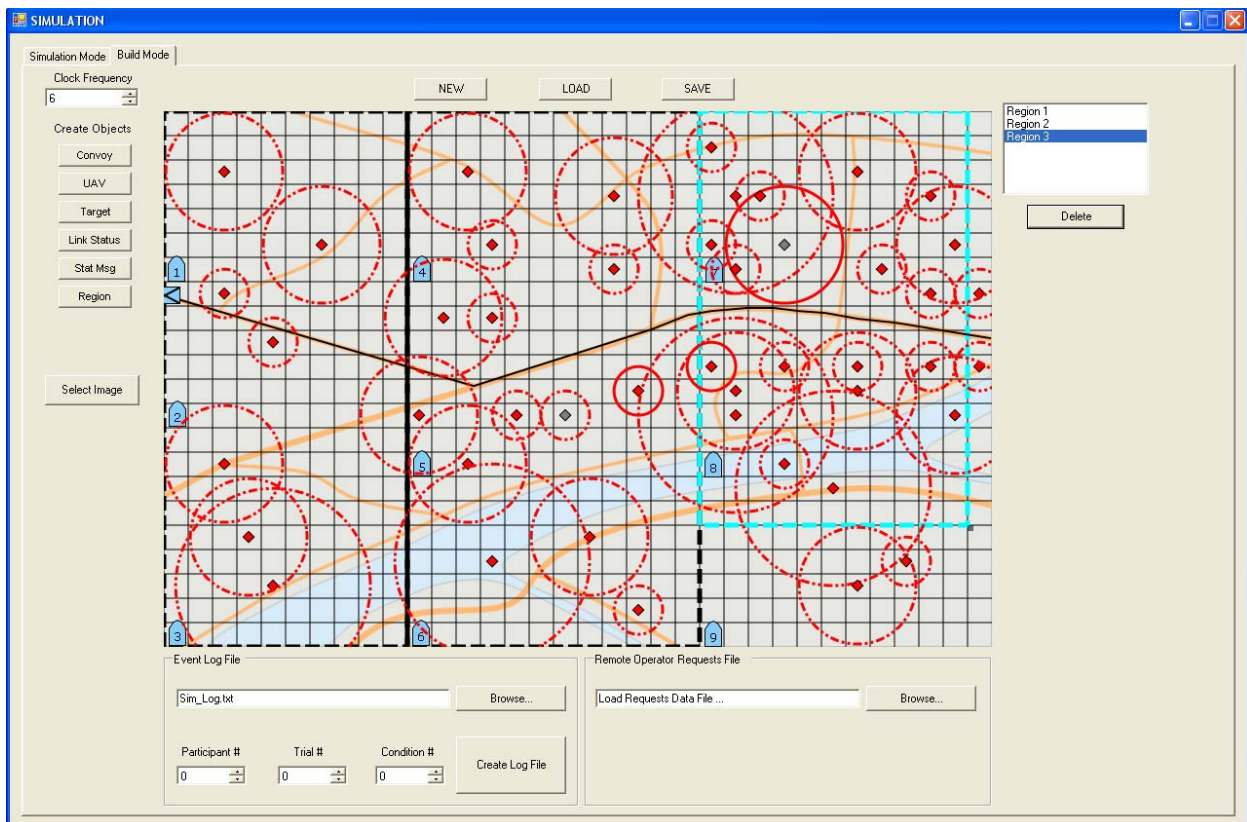


Figure 24. Simulation Server, Build Mode screen in “Region” creation / modification mode.

The Convoy

The UAV team’s primary goal is to protect the Convoy in the current experimental platform task. Thus, a convoy should be defined for each scenario. To create a convoy, select the “Convoy” mode button on the left side of the Build Mode screen (Figure 25). Then, select the “Set Position” button from the Convoy functions panel to the right of the map panel, and select the position on the map at which the convoy will enter the map. Ideally, this position will be on the road depicted on the underlying map image, and near the edge of the map area.

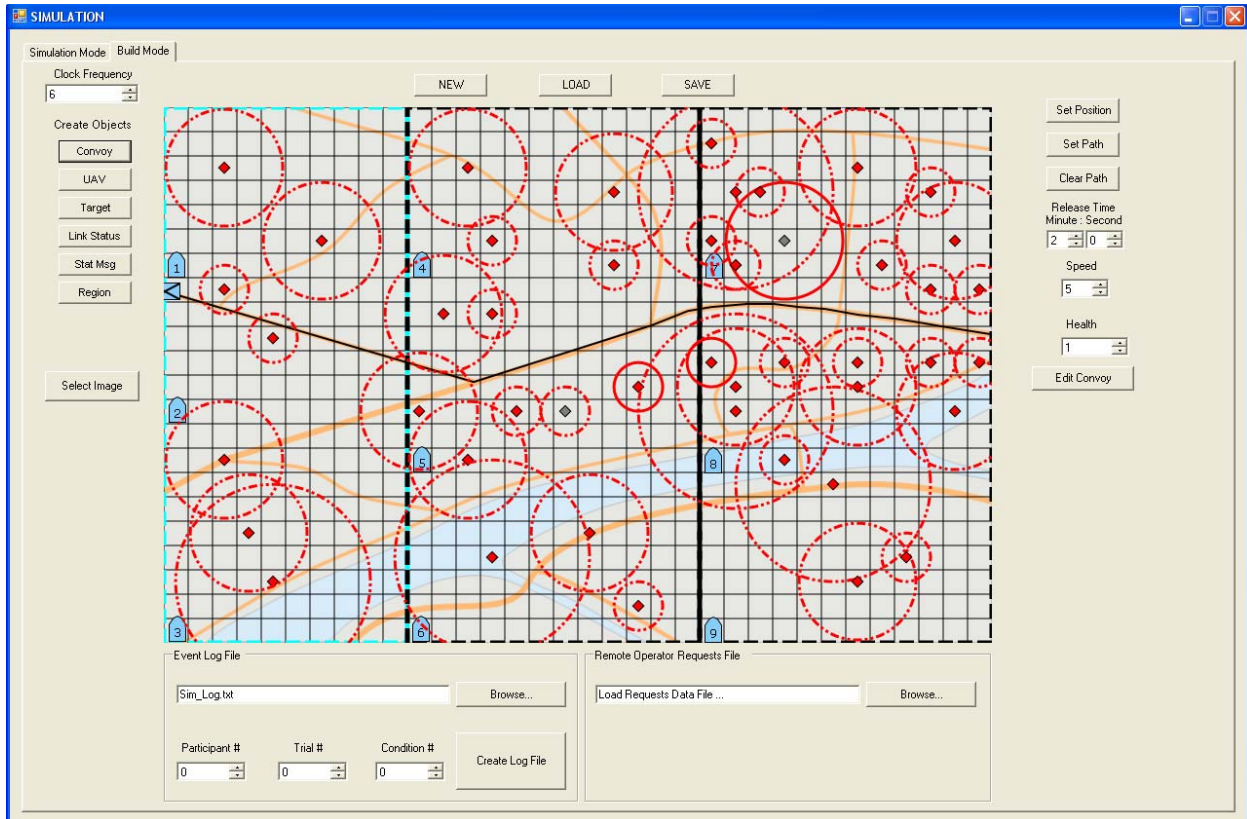


Figure 25. Simulation Server, Build Mode screen in “Convoy” creation / modification mode.

Next, define the convoy’s path by selecting the “Set Path” button from the Convoy function panel, and then select a set of waypoints on the map. These waypoints define the path along which the convoy will travel, beginning with its original entrance point. To make changes to the path, it must first be deleted, using the “Clear Path” button in the Convoy function panel, and then recreated using “Set Path”.

To allow time for experiment participants to become engaged in the simulation environment, and for an interesting and potentially difficult situation to emerge, the convoy can be released (i.e., set to enter the map area) at a predetermined time after the scenario begins. To set this “release time,” set the Minute and Second numeric values under the “Release Time” label in the Convoy functions panel.

Next, to enable scenario events to unfold at a certain rate, the relative convoy speed must be set. This defines how fast the convoy will travel along its defined path during the simulation (as long as the participant hasn't requested that it hold its current position). To set the speed, select an appropriate numeric value under the "Speed" label in the Convoy functions panel. This value represents how many simulation clock ticks should elapse before the convoy's position is moved to the next grid square. Initial studies in the experimental platform have shown that a clock frequency of 6 ticks per second provides a reasonable scenario rate, enabling the Convoy and UAV speeds to be set at a reasonable differential, corresponding appropriately to the fact that a Convoy would travel slower than would a UAV. These studies also showed that also setting the convoy speed to 6, that is the convoy would move once per second, appears to provide a reasonable scenario rate. Setting the convoy speed to a value less than the Clock Frequency will produce a faster convoy pace, and to a value higher will produce a slower convoy pace. The convoy and UAV speed values, and thus modifying the rate at which events unfold during the scenario, should be determined based on the goals of the study.

Finally, an initial convoy health should be set by selecting a numeric value (between 1-100) under the "Health" label in the Convoy functions panel. This value represents the percent of health with which the convoy begins a scenario. Typically this value is set to 100. Setting a lower initial value would increase the urgency of convoy protection during a scenario, likely increasing the intensity / difficulty of the scenario.

Once the above convoy attribute values are set, they must be saved by selecting the "Edit Convoy" button in the Convoy functions panel.

If any of the above attribute values require modification after initially being set, change their values as indicated above, and then apply the changes with the "Edit Convoy" button.

UAVs

To create or modify UAV entities, select the "UAV" mode button from the left side of the Build Mode Screen. The full paths and UAV attribute details will appear for existing UAV entities (Figure 26). To create a UAV, select the "Add UAV" button from the UAV functions panel to the right of the map panel, and then select the initial UAV release position on the map. Experiment participants are told that their UAVs will be air-dropped from a plane at set positions and times during the mission scenario.

The UAV will automatically be labeled by the next available UAV number, and will appear, in order of its label number, in the UAV list at the top of the UAV functions panel. In order to set or modify any UAV attributes, a UAV must first be selected from this list. The numbers to the right of the UAV label in this list indicate the UAV's grid square position in the map. Selecting an existing UAV from the list will display the attribute values for that vehicle in the UAV function panel.

To create a surveillance flight path for the UAV, select the desired UAV from the list, then select the "Set Path" button. Then, select a set of path waypoints, originating at the UAV's initial release position. Similar to the convoy path, a completed UAV path cannot be modified. Instead, the path must be first deleted, by selecting the "Clear Path" button, and then recreated with the "Set Path" function.

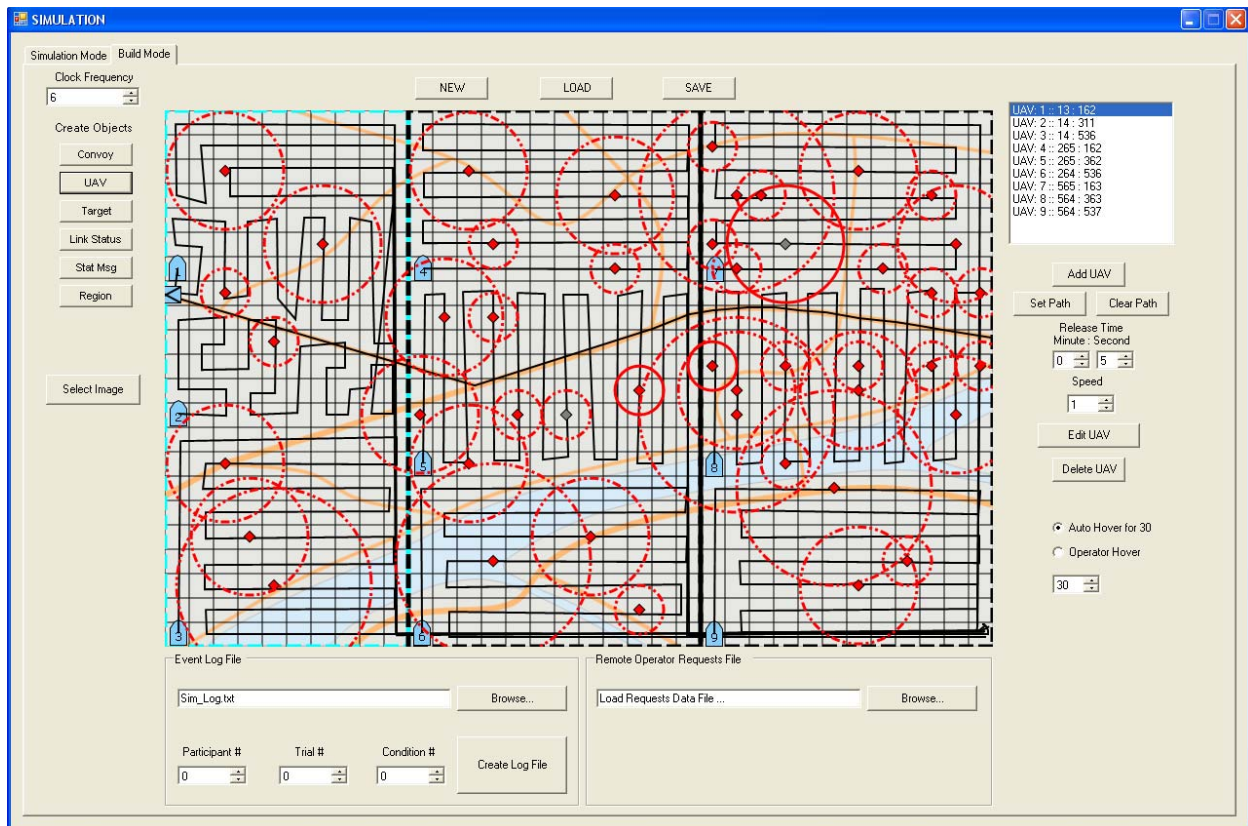


Figure 26. Simulation Server, Build Mode screen in “UAV” creation / modification mode.

To set the UAV’s release time, select the desired UAV from the list, then set the Minute and Second values under the “Release Time” label.

To set the relative speed at which a UAV travels during a scenario, select the desired UAV from the list, then set the numeric value under the “Speed” label. Recall from above that the Simulation Clock Frequency is typically set to 6 ticks per second. Initial studies indicate that setting the UAV speed to 1, with this Clock Frequency (i.e., the UAVs will move 6 times per second), produces reasonable UAV flight speeds.

Once the above UAV attribute values are set, they must be saved by selecting the “Edit UAV” button in the UAV functions panel. Also, if any of the above attribute values require modification after initially being set, change their values as indicated above, and then apply the changes with the “Edit UAV” button.

To delete a UAV from the scenario, select the desired UAV from the list and then select the “Delete UAV” button in the UAV functions panel.

By default, a UAV will automatically hover (i.e., remain in a stationary location, in reality the aircraft would fly in a holding pattern) above a newly discovered potential target for the amount of time specified by the targets “TargetID Time” (see Targets section below), while its operator performs a target confirmation / identification procedure. If the UAV is currently set to perform this behavior, the radio button beside the “Auto Hover for 30” will be selected in the UAV functions panel. (The “for 30” part of this label is now obsolete: Hover time used to be defined

at the UAV level, now it is defined at the Target level and defined per target as described below. Also, changing the numeric value below the Hover selection options is also obsolete.) UAVs should be set to Auto Hover unless the Operator Display (see Section 4.2.7) is being utilized in the experimental platform. When the Operator Display is utilized, which requires non-simulated UAV operators to participate in the experimental trials, the “Operator Hover” Hover selection option should be selected for each UAV created in the scenario, and then saved using the “Edit UAV” button. This Hover attribute setting will ensure that each UAV will hover indefinitely above a detected potential target until its associated operator manually releases the UAV after performing the necessary procedure for target identification / classification in the Operator Display.

Targets

To create or modify Target entities, select the “Target” mode button from the left side of the Build Mode Screen (Figure 27).

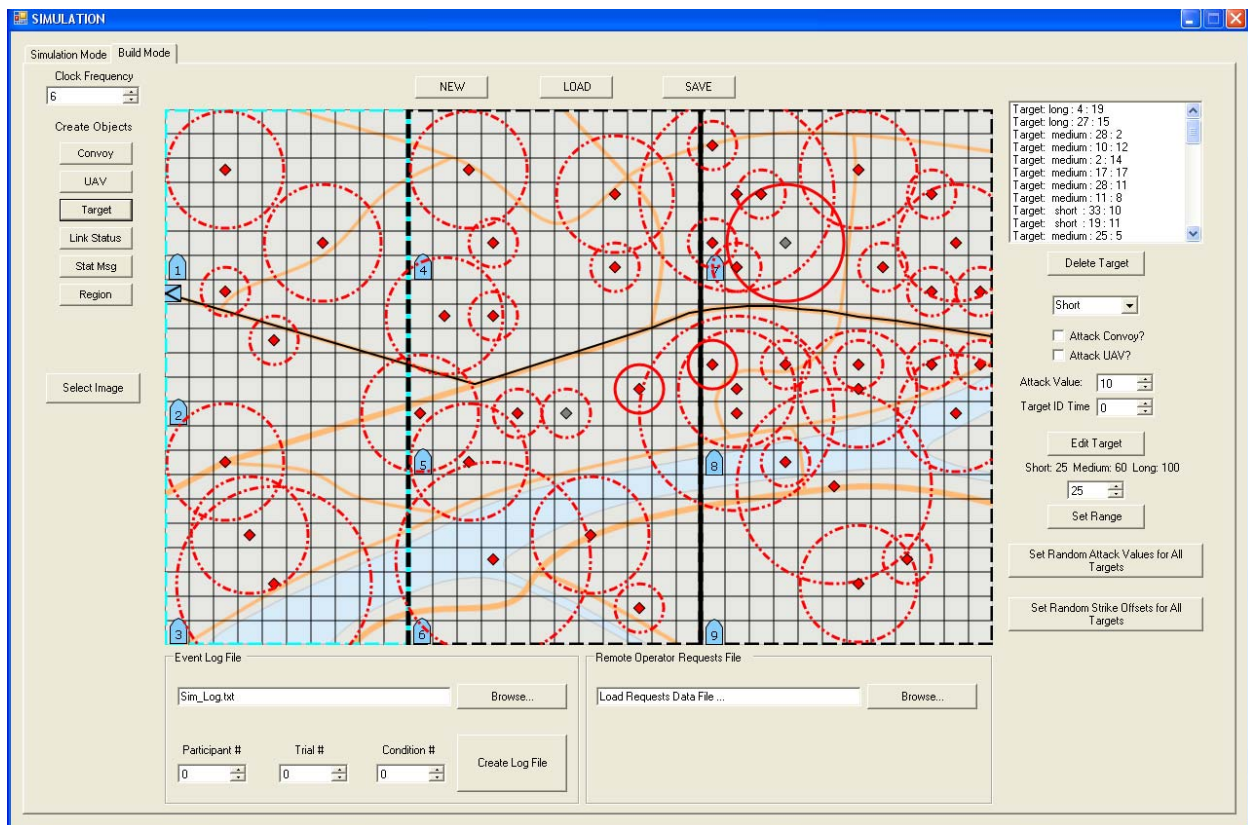


Figure 27. Simulation Server, Build Mode screen in “Target” creation / modification mode.

All existing targets are displayed in the map panel as small diamond-shaped icons, surrounded by a red circle. By default, target icons have a red fill color. A gray target icon fill color indicates the target is set to attack a UAV upon target discovery. The size of the target’s surrounding circle indicates its currently set weapon’s range (Short, Medium, Long). The line type (solid or dashed) indicates whether or not the target will attack the Convoy if it passes

within the target's weapons range (dashed = convoy will be attacked; solid = convoy will not be attacked).

Similar to the UAV functions panel, a list of existing Targets will appear in the top of the Target function panel. Each target line details the current weapons range (Short, Medium, Long) and the grid location of the target. Selecting a target from this list will highlight the associated target diamond icon on the map in blue and display the attribute values for that target in the Target function panel.

In order to add a new Target, simply move the mouse into the map area (a red diamond icon surrounded by an orange circle will appear centered at the cursor) and click the desired location on the map to set the target's position during the scenario. All targets will remain stationary throughout a scenario simulation.

Existing targets cannot be moved. In order to relocate a target, the target must be first destroyed, but selecting the desired target from the Targets list and then selecting the "Delete Target" button from the Target function panel. Then a new target can be added in the new location.

In order to set or modify the target's weapons range, select the desired range (Short, Medium, or Long) from the drop-down list under the "Delete Target" button on the Target function panel (see below for instructions on how to modify the definition of these ranges).

To define whether the selected target will attack the Convoy if it passes within the target's weapons range during a scenario before target is destroyed by the Strike Team, set the checkbox beside the "Attack Convoy?" label in the Target function panel appropriately: checked indicates the target will attack the Convoy, unchecked indicates the target will not attack the Convoy. During the scenario, whenever the Convoy is attacked by a target, a certain level of Convoy Health is lost, representing amount of damage sustained by the attack. See below for instructions on setting how much Convoy Health will be lost during an attack from a target.

Similarly, to define whether the selected target will attack a UAV that passes over the target (i.e., the UAV enters the same grid square and detects the target) during a scenario before target is destroyed by the Strike Team, set the checkbox beside the "Attack UAV?" label in the Target function panel appropriately: checked indicates the target will attack the first UAV that detects it, unchecked indicates the target will not attack any UAVs. During the scenario, when a UAV is attacked by a target, it will be fully destroyed (equivalent to losing 100% health).

To set the amount of Convoy Health value that will be lost during an attack from the selected target, select a numeric value (in percent health) beside the "Attack Value" label in the Target function panel.

As mentioned in the UAVs section above, the amount of time that a UAV hovers above a target upon its discovery is specified by each target. This enables more natural behavior to occur during the scenario, since it is likely that it would take the UAV operator different amounts of time to identify and classify each potential target. To define how long a UAV should hover above each target, the "TargetID Time" numeric value must be set (in seconds) beside the corresponding label in the Target function panel.

Once the above Target attribute values are set for each Target, they must be saved by selecting the "Edit Target" button in the Targets functions panel. Also, if any of the above attribute values require modification after initially being set, change their values as indicated above, and then apply the changes with the "Edit Target" button.

There are several global target attribute values and functions that can also be modified.

To define the available target weapons ranges for a scenario, the ranges for Short, Medium, and Long can be set for all targets by first selecting a target in the Targets list of the desired range (or changing the range in the weapons range drop-down box for the currently selected target), and then changing the numeric values (in pixels) under the “Short: X Medium Y Long Z” label in the Target function panel, and then select the “Set Range” button. This will apply that range value for all targets of the selected weapons range, and will update the label to reflect the current weapons range.

Instead of setting the “Attack Value” for each target individually (recall, this is the amount of Health the Convoy will lose if it passes within weapons range of the target), a global function can be invoked to set to randomly set the Attack Values for all targets between 10 and 15 health points by selecting the “Set Random Attack Values for All Targets” button in the Target function panel.

The final global function relates to the scheduling of target strikes by the Strike Team once a target has been identified by a UAV operator. In order to space out the timing of events (since it would actually take the Strike Team some time to reroute between each target strike), any new target added to the strike schedule will be added at some predetermined period of time past the current time (currently set to 50 seconds, but can be changed in the code, by changing the “Default_Strike_Time” value in the StrikeTeam class). Also, if a target is added to the strike schedule when there are one or more pending strikes, the scheduled strike time for the new target is set to certain timing interval after the last target on the schedule. By default, this value is 20 seconds (this value can be changed in the code by changing the “Default_Strike_Interval” value in the StrikeTeam class). That is, the new target is scheduled to be destroyed 20 seconds after the last pending target strike. In order to add variability to event occurrences during the scenario, some pseudo randomness can also be introduced to these target strike intervals but selecting the “Set Random Strike Offsets for All Targets” button in the Targets function button. This randomizes the strike interval (by default 20) for all targets defined in the scenario to a value between 20 and 30 seconds. If new targets are added, this function needs to be reapplied to the set of targets.

Communication Link Status Events

Another aspect of the simulation environment that can be controlled throughout a scenario is the UAV team’s connection status of the communication links to the external contacts, including the connections to the Convoy, the Strike Team, and the JSTARS (Joint Surveillance and Target Attack Radar System). By default, these communication links remain connected and available throughout the scenario. In order to define period of communication link loss for one or more of these communication links during the scenario, select the “Link Status” mode button on the left side of the Build Mode screen (Figure 28).

A list of any existing status changes defined for the scenario will appear in the Link Status events list at the top of the Link Status function panel to the right of the map panel. Selecting a Link Status event will display the details for that event in the Link Status function panel.

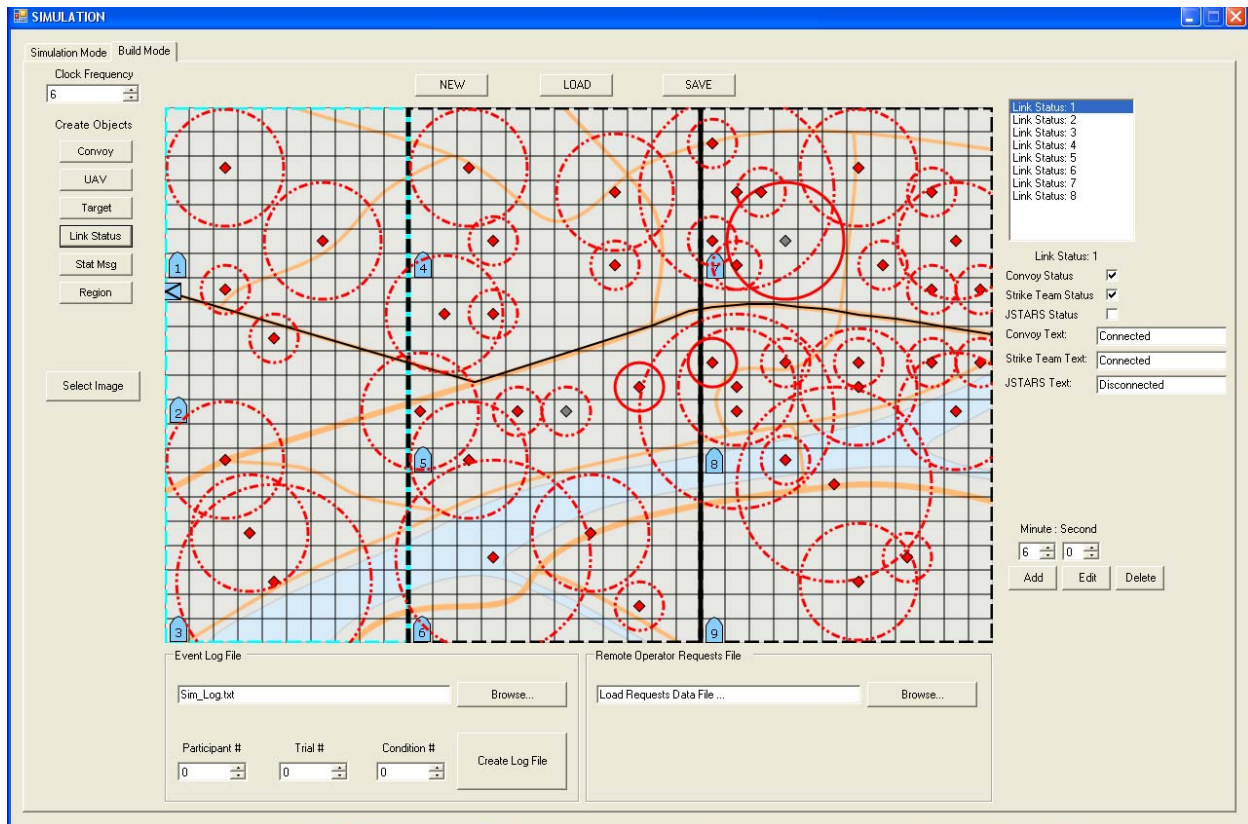


Figure 28. Simulation Server, Build Mode screen in “Link Status” creation / modification mode.

To create a Link Status event, select the connection state for all three UAV team contacts that should occur at that event (and will continue until a subsequent Link Status event alters the connection status) by checking or unchecking the checkboxes to the right of the Convoy, Strike Team, and JSTARS Status labels. Selecting the checkbox (i.e. presence of a checkmark) indicates the communication link to that contact will be activated (or remain active if it was active prior to the event). Unselecting the checkbox (i.e. no checkmark) indicates the communication link to that contact will be deactivated (or remain inactive if it was previously inactive).

The text in the textboxes beside the Convoy, Strike Team, and JSTARS Text labels are automatically updated as the above selections/deselections are made. This text indicates the labels that will be displayed in the Communication Link Status panel associated with each contact in the Mission Status Display (see Section 4.2.4). Alternate labels can be added if desired by replacing the text in these fields.

Next, the time of the Link Status event must be defined in the numeric textboxes below the Minute and Second labels in the Link Status functions panel.

The Link Status event information then must be saved by selecting the “Add” button at the bottom of the Link Status function panel.

To modify an existing Link Status event, select the desired event from the Link Status list, alter the information as described above, and then save the changes by selecting the “Edit” button.

Finally, to delete an existing Link Status event, select the desired event from the Link Status list and then select the “Delete” button from the bottom of the Link Status function panel.

Status Messages

The final type of event that can be defined in a scenario is the sending of predefined messages from either the system or one of the external contacts (Strike Team, Convoy, JSTARS). These messages appear in the Status Message / Message History panels on the Mission Status Display and Mission Commander Interface (see Sections 4.2.4 and 4.2.6) at the predefined time during the scenario. In order to create such messages, select the “Stat Msg” mode button from the left side of the Build Mode screen (Figure 29). A list of any existing Status Messages defined for the scenario will appear in the Message list at the top of the Status Message function panel to the right of the map panel. Selecting a Message will display its details in the Status Message function panel.

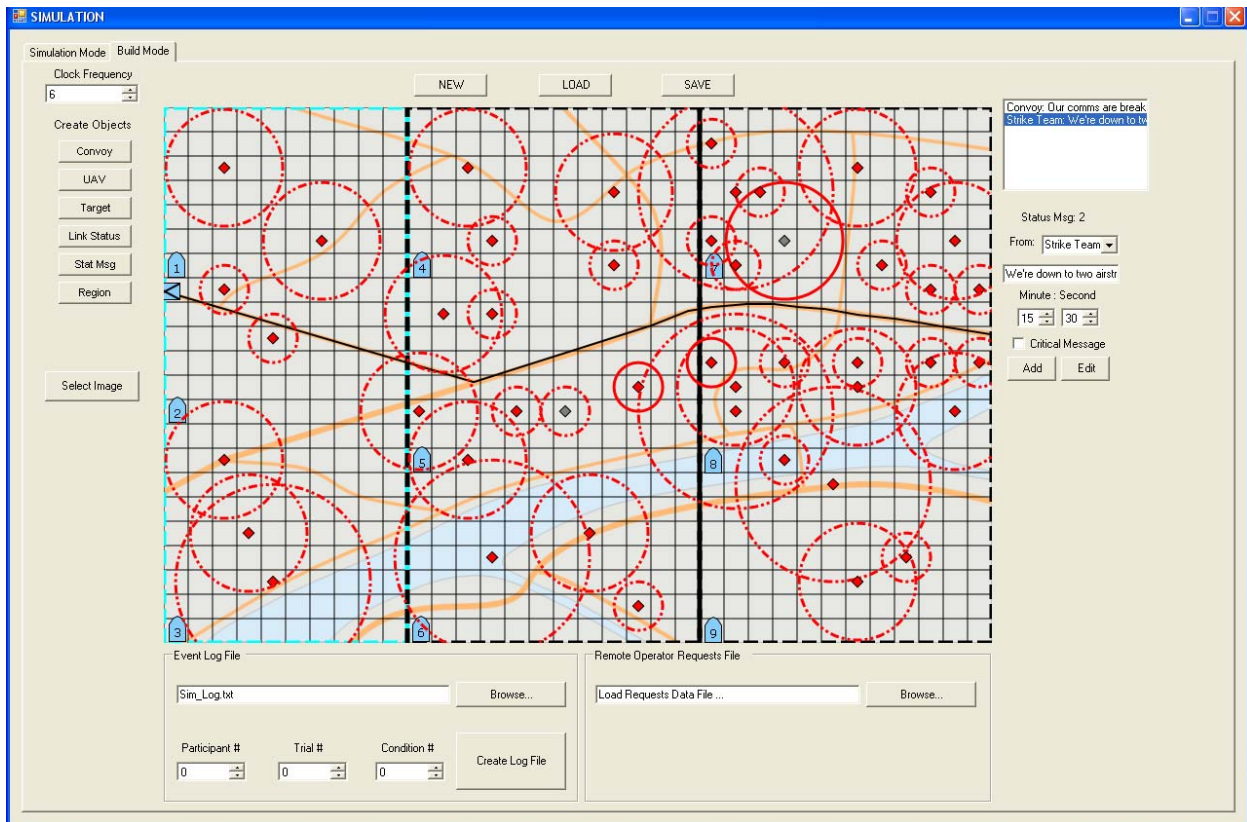


Figure 29. Simulation Server, Build Mode screen in “Status Message” creation / modification mode.

To create a new Message, first select the sender of the Message from the drop-down list beside the “From” label. Then, write the message text in the textbox below the “From” label. Next, set the time at which the message should appear during the scenario in the numeric textboxes below the Minute and Second labels. When the Status Message appears in the team displays, it will appear in the format:

<time> [sender] <message text>

If the Status Message should be highlighted in red on the team displays, the checkbox beside the “Critical Message” label should be selected.

Finally, to save the Message, select the “Save” button at the bottom of the Status Message function panel.

To modify an existing Message, select the desired Message, alter the Message event details as described above, and then select the “Edit” button.

Scenario Background Image

In order to display a different image as the background in the map display, select the “Select Image” button on the left side of the Build Mode screen and select the desired image from the Select Image file browser. Changes to the background image will not be saved once the Simulation Server is closed: this information is not currently saved in the scenario file. If permanent changes to the background image are desired, then a different image should be saved in the code, in the SimulationServer class, and saved to an updated application executable.

Appendix E: Sample XML Scenario File Format

This section contains a sample XML Scenario file. This XML file describes a scenario with the following details:

- A convoy with a simple path, which enters the team's AOI 90 seconds into the scenario. The convoy is set to travel at speed level 6, meaning that it will move one grid square per second, when the simulator is run at 6 clock ticks per second (the standard way to run the simulated scenarios).
- One target, located at (10, 6) (grid square coordinates). The target has long range weapons range capability (weaponsRange 3), will strike the convoy if the convoy passes within its weapons range before the target is destroyed (attackConvoy true), but will not strike a UAV that passes overhead (attackUAV false). If the target attacks the convoy, the convoy will lose 10 points of health (attackValue 10). If the target is detected by a UAV and the UAVs are set to AutoHover in the Simulation (see Appendix D), then the UAV will hover above the target for 30s while its operator engages in target identification (targetIDTime 30). Once the target has been identified, it will be scheduled to be destroyed 20s after the last scheduled target strike (scheduleOffset 20).
- One UAV set to a simple, sweeping surveillance path. The UAV is released at location (4, 183) (on-screen pixel location) at 10 seconds into the scenario. The UAV is set to travel at speed level 1, meaning that it will move one grid square per clock tick. At the standard simulation speed of 6 clock ticks per second, this means the UAV will travel at 6 grid squares a second.
- Three operator regions (required)
- A planned communication link loss between the UAV team and the Strike team from 320s and 620s (5:20-10:20) in the scenario.
- A planned message from the Strike Team of "We will be unavailable for the next 5 minutes." at 320s (5:20) in the scenario.

```
<?xml vrsion="1.0" encoding="utf-8"?>
<World xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <s_range>25</s_range>
  <m_range>50</m_range>
  <l_range>105</l_range>
  <glength>25</glength>
  <clock_frequency>6</clock_frequency>
  <convoy>
    <Xlocal>3</Xlocal>
    <Ylocal>187</Ylocal>
    <current_Region>0</current_Region>
    <Ghost_X>3</Ghost_X>
    <Ghost_Y>187</Ghost_Y>
    <isReleased>false</isReleased>
    <Ghost_isReleased>false</Ghost_isReleased>
    <releaseTime>90</releaseTime>
    <timeUntilRelease>90</timeUntilRelease>
    <Ghost_timeUntilRelease>180</Ghost_timeUntilRelease>
    <currentSpeed>6</currentSpeed>
    <speed_int>6</speed_int>
    <Ghost_speed_int>6</Ghost_speed_int>
```

```
<originalPath>
  <path>
    <Point>
      <X>3</X>
      <Y>187</Y>
    </Point>
    <Point>
      <X>3</X>
      <Y>187</Y>
    </Point>
    <Point>
      <X>0</X>
      <Y>184</Y>
    </Point>
    <Point>
      <X>319</X>
      <Y>284</Y>
    </Point>
    <Point>
      <X>533</X>
      <Y>211</Y>
    </Point>
    <Point>
      <X>631</X>
      <Y>202</Y>
    </Point>
    <Point>
      <X>698</X>
      <Y>208</Y>
    </Point>
    <Point>
      <X>844</X>
      <Y>233</Y>
    </Point>
  </path>
  <current_Local>0</current_Local>
</originalPath>
<plannedPath>
  <path>
    <Point>
      <X>3</X>
      <Y>187</Y>
    </Point>
    <Point>
      <X>3</X>
      <Y>187</Y>
    </Point>
    <Point>
      <X>0</X>
      <Y>184</Y>
    </Point>
    <Point>
      <X>319</X>
      <Y>284</Y>
    </Point>
    <Point>
      <X>533</X>
      <Y>211</Y>
    </Point>
    <Point>
      <X>631</X>
      <Y>202</Y>
    </Point>
    <Point>
      <X>698</X>
      <Y>208</Y>
    </Point>
    <Point>
      <X>844</X>
      <Y>233</Y>
    </Point>
  </path>
</plannedPath>
```

```

    </path>
    <current_Local>0</current_Local>
  </plannedPath>
  <currentWaypoint>0</currentWaypoint>
  <Ghost_Waypoint>0</Ghost_Waypoint>
  <currentHealthLevel>100</currentHealthLevel>
  <isFinishedRoute>>false</isFinishedRoute>
  <Ghost_isFinishedRoute>>false</Ghost_isFinishedRoute>
  <numberOfHolds>0</numberOfHolds>
  <isHolding>>false</isHolding>
</convoy>
<Target_Places>
  <Point>
    <X>10</X>
    <Y>6</Y>
  </Point>
</Target_Places>
<Targets>
  <Target>
    <weaponsRange>3</weaponsRange>
    <attackConvoy>>true</attackConvoy>
    <attackUAV>>false</attackUAV>
    <attack_convoy_value>10</attack_convoy_value>
    <schedule_offset_value>20</schedule_offset_value>
    <targetIDTime>30</targetIDTime>
  </Target>
</Targets>
<UAVs>
  <UAV>
    <Xlocal>4</Xlocal>
    <Ylocal>183</Ylocal>
    <current_Region>0</current_Region>
    <Ghost_X>4</Ghost_X>
    <Ghost_Y>183</Ghost_Y>
    <isReleased>>false</isReleased>
    <Ghost_isReleased>>false</Ghost_isReleased>
    <releaseTime>60</releaseTime>
    <timeUntilRelease>60</timeUntilRelease>
    <Ghost_timeUntilRelease>60</Ghost_timeUntilRelease>
    <currentSpeed>1</currentSpeed>
    <speed_int>1</speed_int>
    <Ghost_speed_int>1</Ghost_speed_int>
    <originalPath>
      <path>
        <Point>
          <X>4</X>
          <Y>183</Y>
        </Point>
        <Point>
          <X>4</X>
          <Y>183</Y>
        </Point>
        <Point>
          <X>11</X>
          <Y>82</Y>
        </Point>
        <Point>
          <X>35</X>
          <Y>85</Y>
        </Point>
        <Point>
          <X>38</X>
          <Y>192</Y>
        </Point>
        <Point>
          <X>62</X>
          <Y>200</Y>
        </Point>
        <Point>
          <X>60</X>
          <Y>88</Y>
        </Point>
      </path>
    </originalPath>
  </UAV>
</UAVs>

```

```

    </Point>
    <Point>
      <X>90</X>
      <Y>88</Y>
    </Point>
    <Point>
      <X>87</X>
      <Y>207</Y>
    </Point>
  </path>
  <current_Local>0</current_Local>
</originalPath>
<plannedPath>
  <path>
    <Point>
      <X>4</X>
      <Y>183</Y>
    </Point>
    <Point>
      <X>4</X>
      <Y>183</Y>
    </Point>
    <Point>
      <X>11</X>
      <Y>82</Y>
    </Point>
    <Point>
      <X>35</X>
      <Y>85</Y>
    </Point>
    <Point>
      <X>38</X>
      <Y>192</Y>
    </Point>
    <Point>
      <X>62</X>
      <Y>200</Y>
    </Point>
    <Point>
      <X>60</X>
      <Y>88</Y>
    </Point>
    <Point>
      <X>90</X>
      <Y>88</Y>
    </Point>
    <Point>
      <X>87</X>
      <Y>207</Y>
    </Point>
  </path>
  <current_Local>0</current_Local>
</plannedPath>
<currentWaypoint>0</currentWaypoint>
<Ghost_Waypoint>0</Ghost_Waypoint>
<currentHealthLevel>100</currentHealthLevel>
<isFinishedRoute>false</isFinishedRoute>
<Ghost_isFinishedRoute>false</Ghost_isFinishedRoute>
<uavID>1</uavID>
<operatorRegion>1</operatorRegion>
<isHovering>false</isHovering>
<isAutoHover>true</isAutoHover>
<HoverTime>30</HoverTime>
<IsRouteReassignedToAlternateUAV>false</IsRouteReassignedToAlternateUAV>
<IsRealTarget>true</IsRealTarget>
</UAV>
</UAVs>
<Regions>
  <Sector>
    <Operator>1</Operator>
    <rootX>0</rootX>

```

```

    <rootY>3</rootY>
    <width>12</width>
    <length>12</length>
  </Sector>
  <Sector>
    <Operator>2</Operator>
    <rootX>12</rootX>
    <rootY>4</rootY>
    <width>12</width>
    <length>12</length>
  </Sector>
  <Sector>
    <Operator>3</Operator>
    <rootX>24</rootX>
    <rootY>1</rootY>
    <width>10</width>
    <length>14</length>
  </Sector>
</Regions>
<links>
  <LinkStatus>
    <strike_status_txt />
    <jstars_status_txt />
    <convoy_status_txt />
    <currentTime>320</currentTime>
    <Strike_last_commm>0</Strike_last_commm>
    <JSTARS_last_commm>0</JSTARS_last_commm>
    <Convoy_last_commm>0</Convoy_last_commm>
    <Strike_next_commm>0</Strike_next_commm>
    <JSTARS_next_commm>0</JSTARS_next_commm>
    <Convoy_next_commm>0</Convoy_next_commm>
    <Strike_avg_cycle>0</Strike_avg_cycle>
    <JSTARS_avg_cycle>0</JSTARS_avg_cycle>
    <Convoy_avg_cycle>0</Convoy_avg_cycle>
    <strike_connected>>false</strike_connected>
    <jstars_connected>>true</jstars_connected>
    <convoy_connected>>true</convoy_connected>
  </LinkStatus>
  <LinkStatus>
    <strike_status_txt />
    <jstars_status_txt />
    <convoy_status_txt />
    <currentTime>620</currentTime>
    <Strike_last_commm>0</Strike_last_commm>
    <JSTARS_last_commm>0</JSTARS_last_commm>
    <Convoy_last_commm>0</Convoy_last_commm>
    <Strike_next_commm>0</Strike_next_commm>
    <JSTARS_next_commm>0</JSTARS_next_commm>
    <Convoy_next_commm>0</Convoy_next_commm>
    <Strike_avg_cycle>0</Strike_avg_cycle>
    <JSTARS_avg_cycle>0</JSTARS_avg_cycle>
    <Convoy_avg_cycle>0</Convoy_avg_cycle>
    <strike_connected>>true</strike_connected>
    <jstars_connected>>true</jstars_connected>
    <convoy_connected>>true</convoy_connected>
  </LinkStatus>
</links>
<stats>
  <StatusMessage>
    <msgType>1</msgType>
    <msgText>We will be unavailable for the next 5 minutes</msgText>
    <msgTime>320</msgTime>
    <Critical>>false</Critical>
  </StatusMessage>
</stats>
</World>

```

Appendix F: Sample XML Targets File Format

This section contains a sample XML Targets Data file. This XML file is necessary whenever the Remote Assistance Display (Section 4.2.5) application is used in the experimental platform. The Targets Data file describes several targets, each of which represent a potential “target” that has been identified by an Automatic Target Recognition (ATR) system onboard one of the team’s UAVs. Within this file, a number of details are provided about each target to facilitate the flexible use of different targets during different simulated scenarios during experimental trials. For each target, this file provides:

- The filename and file location of an image of the target, which represents the image that was captured by the (simulated) UAV (AtrImageFile).
- An initial target classification, which represents the current target classification suggested by the UAV operator when he/she requests classification assistance from the mission commander (see Section 4.2.5 for details of this process) (OperatorClassification). This value must correspond to a defined value in the ATR_Classification enumeration in the TSTEntityLibrary.RemoteOperatorRequest class. If a new classification type (or a set of classifications to be displayed in the Remote Assistance Display) is desired, simply update the ATR_Classification enumeration.
- An initial confidence level, which represents how confident the UAV operator is in the suggested target classification. This value must correspond to a defined value in the ATR_Confidence enumeration in the TSTEntityLibrary.RemoteOperatorRequest class. Again, if new confidence types are desired, the enumeration can be updated.

NOTES:

1. This file must contain enough target definitions to cover all possible Target Classification assistance requests. That is, if the mission commander decides to assist with each target identification the team makes, there must be a different / new target description in this file to populate the data for each request. This requirement is not currently checked during the initialization / file loading process, but will cause a system failure during run-time if the program runs out of target data during a new Assistance Request. A safe rule to follow during experimental preparation is to have at least as many target definitions in this file as there are possible targets defined in the scenario.
2. To facilitate with file organization and storage, the Target image files are currently located in at “Targets” directory in the same directory as the Simulation Server executable. Notice that the filenames of the image files in this sample file are all prefixed with “Targets\” that tells the Simulation Server application to look for the file in the local Targets directory.

```
<?xml version="1.0" encoding="utf-8" ?>
  <AssistanceRequestsFromXML>
    <AssistanceRequests>
      <RemoteOperatorRequest>
        <AtrImageFile>Targets\IED.bmp</AtrImageFile>
        <OperatorClassification>IED_Package</OperatorClassification>
        <OperatorConfidence>High</OperatorConfidence>
      </RemoteOperatorRequest>
      <RemoteOperatorRequest>
```



```
<AtrImageFile>Targets\headquarter_1.bmp</AtrImageFile>  
<OperatorClassification>Munitions_Depot</OperatorClassification>  
<OperatorConfidence>High</OperatorConfidence>  
</RemoteOperatorRequest>  
</AssistanceRequests>  
</AssistanceRequestsFromXML>
```

Appendix G: Sample Data Log File Format

Data log files are created automatically by the Simulation Server, whenever a scenario is executed in the Simulation Mode screen. As described in Section 4.2.1, unless otherwise specified by the experimenter in the Build Mode screen, the log is written to the file called “Sim_Log.txt” in the same folder as the “Simulation Server.exe” executable file. This file is continuously overwritten, unless another filename is specified (see Section 4.2.1 for details of saving log data to a specific file). Data logs are written as ASCII text files, in a compatible format to be imported and parsed into columns by MS Excel. To import a data log file, open Excel, invoke the Open File function (Ctrl-O), and select the data log file (you will likely need to set the file type to “All Files” to see the desired data log text file). The import wizard will automatically begin the import procedure. When asked for a delimiter, deselect the default “space” or “tabs” (whichever your application is set to use), and select “Other,” and type in a tilde (“~”) symbol into the free text box. The Simulation Server automatically inserts a “~” symbol between each value in the data log file entry lines – this makes it difficult to read log files with the human eye, but facilitates the import procedure. The first line of every log file are the column headers for the Excel spreadsheet, which defines the column values.

```
TIME~MESSAGE~CONVOY_MOVING/STOPPED~CONVOY_HEALTH~CONVOY_X_POSITION~CONVOY_Y_POSITION~C
ONVOY_INRANGE_TARGETS~UAV1_REGION~UAV1_X_POSITION~UAV1_Y_POSITION~UAV1_STATUS~UAV2_REG
ION~UAV2_X_POSITION~UAV2_Y_POSITION~UAV2_STATUS~UAV3_REGION~UAV3_X_POSITION~UAV3_Y_POS
ITION~UAV3_STATUS~UAV4_REGION~UAV4_X_POSITION~UAV4_Y_POSITION~UAV4_STATUS~UAV5_REGION~
UAV5_X_POSITION~UAV5_Y_POSITION~UAV5_STATUS~UAV6_REGION~UAV6_X_POSITION~UAV6_Y_POSITIO
N~UAV6_STATUS~UAV7_REGION~UAV7_X_POSITION~UAV7_Y_POSITION~UAV7_STATUS~UAV8_REGION~UAV8
_X_POSITION~UAV8_Y_POSITION~UAV8_STATUS~UAV9_REGION~UAV9_X_POSITION~UAV9_Y_POSITION~UA
V9_STATUS~CONVOY_CONNECTION~JSTARS_CONNECTION~STRIKE_TEAM_CONNECTION~MD_CONVOY_THREAT_
RINGS_FILTER~MD_RANGE_ALL_FILTER~MD_RANGE_SHORT_FILTER~MD_RANGE_MEDIUM_FILTER~MD_RANGE
_LONG_FILTER~MD_UAV_ALL_FILTER~MD_UAV_OP1_FILTER~MD_UAV_OP2_FILTER~MD_UAV_OP3_FILTER~S
D_REGION_ALL_FILTER~SD_REGION_1_FILTER~SD_REGION_2_FILTER~SD_REGION_3_FILTER~ALIVE_UAV
_REGION~DOWN_UAV_REGION~ALIVE_UAV_ID~DOWN_UAV_ID~JSTARS_REGION~BUFFER_VIOLATION_TARGET
_ID~ELAPSED_TIME~IS_TB_VIOLATION~TARGET_ID~CONVOY_ATTACK_TIME~TARGET_CLASS~TARGET_CONF
IDENCE_LEVEL~RAD UAV
```

```
00:00:01~STATUS_MESSAGE: Convoy communication link lost.~moving~100~1~189~~1~13~162~su
rveilling~1~14~311~surveilling~1~14~536~surveilling~2~265~162~surveilling~2~265~362~su
rveilling~2~264~536~surveilling~3~565~163~surveilling~3~564~363~surveilling~3~564~537~
surveilling~connected~connected~connected~hide~hide~hide~hide~hide~hide~hide~hide~hide
~hide~hide~hide~hide~0~0~0~0~0~0~00:00:00~False~~00:00:00~~~0
```

```
00:00:19~MAP_DISPLAY_FILTER_CHANGE~moving~100~1~189~~1~38~152~surveilling~1~63~307~sur
veilling~1~15~511~surveilling~2~265~162~surveilling~2~265~362~surveilling~2~264~536~su
rveilling~3~565~163~surveilling~3~564~363~surveilling~3~564~537~surveilling~down~conne
cted~connected~show~hide~hide~hide~hide~hide~hide~hide~hide~hide~hide~hide~hide~hide~0~0~0~
0~0~0~00:00:00~False~~00:00:00~~~0
```

```
00:00:20~MAP_DISPLAY_FILTER_CHANGE~moving~100~1~189~~1~38~147~surveilling~1~63~302~sur
veilling~1~15~506~surveilling~2~265~162~surveilling~2~265~362~surveilling~2~264~536~su
rveilling~3~565~163~surveilling~3~564~363~surveilling~3~564~537~surveilling~down~conne
cted~connected~show~show~show~hide~hide~hide~hide~hide~hide~hide~hide~hide~hide~hide~0~0~0~
0~0~0~00:00:00~False~~00:00:00~~~0
```

```
00:00:20~MAP_DISPLAY_FILTER_CHANGE~moving~100~1~189~~1~38~147~surveilling~1~63~302~sur
veilling~1~15~506~surveilling~2~265~162~surveilling~2~265~362~surveilling~2~264~536~su
rveilling~3~565~163~surveilling~3~564~363~surveilling~3~564~537~surveilling~down~conne
cted~connected~show~show~show~show~hide~hide~hide~hide~hide~hide~hide~hide~hide~hide~0~0~0~
0~0~0~00:00:00~False~~00:00:00~~~0
```

00:00:20~MAP_DISPLAY_FILTER_CHANGE~moving~100~1~189~~1~38~147~surveilling~1~63~302~surveilling~1~15~506~surveilling~2~265~162~surveilling~2~265~362~surveilling~2~264~536~surveilling~3~565~163~surveilling~3~564~363~surveilling~3~564~537~surveilling~down~connected~connected~show~show~show~show~show~hide~hide~hide~hide~hide~hide~hide~hide~hide~hide~hide~hide~0~0~0~0~0~00:00:00~False~~00:00:00~~~0

00:00:20~MAP_DISPLAY_FILTER_CHANGE~moving~100~1~189~~1~38~147~surveilling~1~63~302~surveilling~1~15~506~surveilling~2~265~162~surveilling~2~265~362~surveilling~2~264~536~surveilling~3~565~163~surveilling~3~564~363~surveilling~3~564~537~surveilling~down~connected~connected~show~show~show~show~show~hide~hide~hide~hide~hide~hide~hide~hide~hide~hide~hide~hide~0~0~0~0~0~00:00:00~False~~00:00:00~~~0

00:00:24~MAP_DISPLAY_FILTER_CHANGE~moving~100~1~189~~1~18~138~surveilling~1~52~286~surveilling~1~15~479~surveilling~2~265~162~surveilling~2~265~362~surveilling~2~263~509~surveilling~3~565~163~surveilling~3~564~363~surveilling~3~564~537~surveilling~down~connected~connected~show~show~show~show~show~hide~show~hide~hide~hide~hide~hide~hide~hide~hide~hide~hide~0~0~0~0~0~00:00:00~False~~00:00:00~~~0

Appendix H: Sample Participant Training Tutorial

Figures 30-42 contain a set of Microsoft PowerPoint slides that was used as the tutorial portion of the participant training in one of the experiments performed in the experimental platform (for details of the experiment see Wan, 2007). These slides detail the mission summary and mission goals for the participant during the experimental scenarios and summarize the functionality of the experimental interfaces they will use during the experiment.

Training Overview

2 of 50

There will be 3 phases of your participant training:

1. Mission Summary & Mission Goals
2. Introduction of the Experiment Displays
3. Practice Sessions

Assisting in Collaborative Time-Sensitive Targeting

MIT Humans & Automation Lab

Experimenter: Stacey Scott & Jordan Wan
Principal Investigator: M.L. Cummings

Spring 2007

Mission Summary

3 of 50

In this experiment, you will be in command of a team of unmanned aerial vehicle (UAV) operators engaging in a time-sensitive targeting (TST) operation. Your team's mission is to secure safe passage for an important political convoy through a hostile region.

Your team's UAVs are equipped with camera sensors only (i.e., they have no weapons capabilities). Therefore, your UAV operators will be coordinating with an external air strike team to destroy any identified threats to the convoy in your team's area of interest (AOI).

Your team's AOI will be divided into three sub-AOIs, each assigned to one of your UAV operators. To surveil these areas, these UAV operators each will be monitoring and interacting with a number of highly autonomous UAVs. You also have **limited** access to additional surveillance capabilities from JSTARS (Joint Surveillance & Target Attack Radar System), a local multi-sensor aircraft, to supplement your team of UAVs.

During the mission, interruptions may occur which will simulate realistic military environments in which a mission commander may be required to perform secondary tasks. The interruption may be short and occur directly in the scenario room, or the interruption may be longer and require you to leave the room.

Mission Goals

4 of 50

The primary goals of this mission are to get the convoy through your AOI as quickly as possible, as safely as possible. Therefore, **clearing a threat-free path** for the convoy and **keeping the convoy moving** are your team's highest priorities.

In order to support the military's larger objectives, your team has also been tasked with two additional mission objectives. The first will be to surveil all the roads in the area for safe passage of future convoys. The second will be to assist remote operators in target classification.

Finally, as the mission commander, another priority of yours will be to monitor the performance of the strike team and to report any instances of unsatisfactory strike support. This will be defined in more detail later.

Figure 30. Tutorial Slide 1 (from Wan, 2007).



Possible Command Interventions

In order to achieve your mission objectives you will have three types of commands at your disposal:

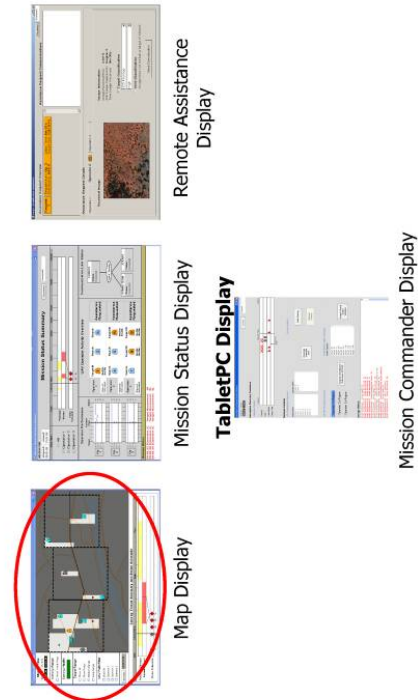
1. Convoy commands: Hold position, release from holding.
2. Requesting additional surveillance information from a nearby intelligence source (JSTARS).
3. Reassigning a UAV from one part of the AOI to another to take over the surveillance activities of a UAV that has been shot down (you may only do this once for each UAV).

The specifics of how and when to use these command interventions will be discussed later in this tutorial.



Map Display

Large-Screen Wall Displays

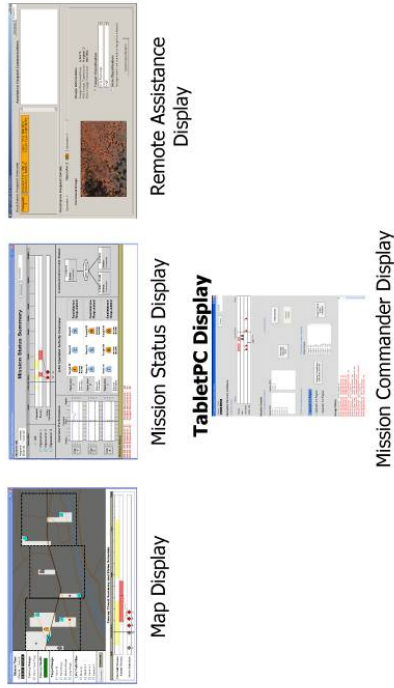


Mission Commander Display



Experiment Displays: TST Supervisor Displays

Large-Screen Wall Displays



Mission Commander Display



Situation Map Display (cont'd)

The **situation map display** visualizes positional information of relevant contacts and assets in the context of your team's geographical area of interest (AOI). It also provides several tools for changing the level of detail displayed on the screen.

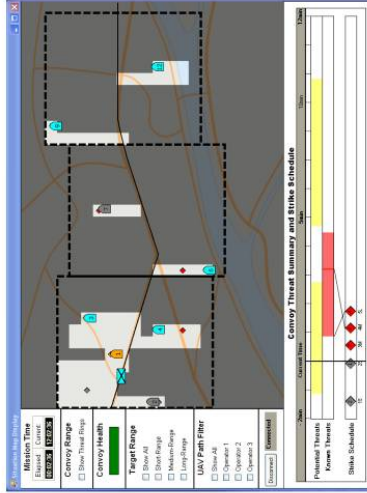


Figure 31. Tutorial Slide 2 (from Wan, 2007).



Situation Map Display (cont'd)

The main components of this display, described in detail next, are:

1. The mission clock.
2. A map of your team's AOI.
3. Filters for toggling certain display information.
4. A timeline indicating the current and expected threat level to the convoy.

1: Mission Clock

2: Map of AOI

3: View Filters

4: Threat Summary & Strike Schedule Timeline



Situation Map Display (1) Mission Clock

MISSION TIME	
elapsed:	current:
00:23:10	12:23:10

The Mission Clock shows the up-to-date mission times, including the **elapsed time** since the beginning of the test session and the **current mission time**.

This mission clock can be found in the upper left corner of all three experiment displays



Situation Map Display (2) Map: Geospatial Mission Activity

The Map shows the up-to-date geospatial mission activity within your team's AOI, see Map Symbology on the right for details. The Map also indicates regions that have not yet been surveilled with a **transparent black overlay**.

Surveillance is the UAV's default state. When a potential target is detected by a UAV's onboard automatic target recognition (ATR) system, an image is sent to the operator for review & confirmation (reviewing ATR imagery state). This process should only take 30 seconds, during which time the UAV loiters at that location. If the TargetID process **takes longer than 30 seconds**, you may choose to assist the operator using the Remote Assistance Display (make sure 30 seconds has elapsed).

If confirmed, the target is added to the strike schedule. If a UAV is lost (i.e. goes down for any reason), it is grayed out at its last known location.

Map Symbology	
Convoy	
Convoy route	
UAV	(default) (reviewing ATR imagery) (down)
Targets	(identified) (destroyed) (nominal) (critical)
Operator AOI boundary	(convoy in WR of air-CP, critically low) surveilled area in AOI

ATR = automatic target recognition
WR = weapons range
OPL = operator performance level



Situation Map Display (3) View Filters

The View Filter panel provides toggle check boxes to enable you to display or hide certain information on the map. By default all toggles are set to the 'off' (i.e., 'hide') position.

To **show** the desired information, touch (with your finger or the wall pen) inside the check box. A checkmark will appear in the box and the corresponding information will appear on the map.

To **hide** the information, touch inside the checkbox again. The checkmark and corresponding map information will disappear.

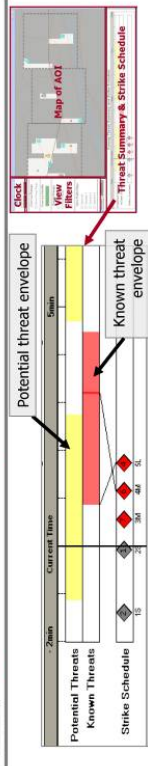
Three types of view filters are available:

1. **Convoy Range Filter:** this toggle shows or hides the distance range rings around the convoy corresponding to the expected range of short, medium, and long-range weapons in the AOI.
2. **Targets Range Filter:** these toggles show or hide the highlight and range rings for any known (i.e., previously identified) short, medium, or long-range targets.
3. **UAV Surveillance Path Filter:** these toggles show or hide the corresponding UAV flight paths.

Convoy Range	<input type="checkbox"/> Show Threat Rings
Convoy Health	<input checked="" type="checkbox"/>
Target Range	<input type="checkbox"/> Show All
	<input type="checkbox"/> Short-Range
	<input type="checkbox"/> Medium-Range
	<input type="checkbox"/> Long-Range
UAV Path Filter	<input type="checkbox"/> Show All
	<input type="checkbox"/> Operator 1
	<input type="checkbox"/> Operator 2
	<input type="checkbox"/> Operator 3

Figure 32. Tutorial Slide 3 (from Wan, 2007).

(4) Threat Summary & Strike Schedule Timeline (I)



The Threat Summary & Strike Schedule timeline displays the temporal relationship between the **scheduled target strikes** and the **known and potential threats to the convoy**.

In particular, the timeline displays red & yellow threat envelopes, corresponding to known threats & potential threats, respectively. The width (duration) of a threat envelope indicates the length of time the convoy is expected to be within range of that threat, based on the convoy's speed and path, and the position and weapons range of that threat.

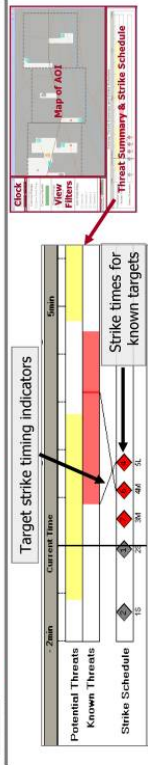
Red/known threat envelopes correspond to periods of time when the convoy will be within range of **identified targets**. Most of these targets should be destroyed by the strike team before the convoy comes within their range (discussed in more detail later). When two or more known threat envelopes overlap, a darker red is shown on the timeline.

Yellow/potential threat envelopes correspond to periods of time when the convoy will be within expected weapons range of an **unsurveilled area**, which may contain a target.

Each time the convoy enters a threat envelope (i.e., comes within weapons range of a known or possible target), the convoy may be attacked. Each time the convoy is attacked, its energy level will decrease by 10-15%. The convoy will report its energy level after each attack. **When the convoy's energy level reaches 0, the mission is over!**

(continued on next slide...)

(4) Threat Summary & Strike Schedule Timeline (II)

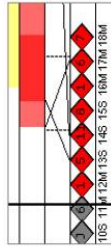


The third (lowest) row of the timeline displays the target strike schedule for identified targets. Each diamond in the strike schedule represents a known target. The numeric label inside a target corresponds to the UAV (or external intelligence team) who discovered it.

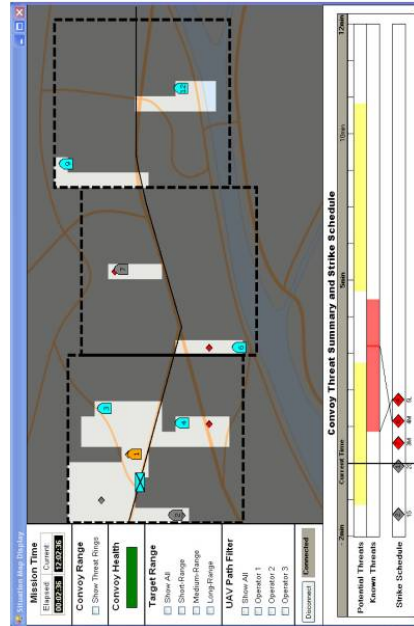
The target ID is provided below each target as an alphanumeric tag indicating the target number and its weapons range capability (S=short, M=medium, L=long). The target's position on the strike schedule represents the expected time the strike team will destroy the target.

A black line connects each target to the **start** of its corresponding threat envelope (i.e., expected time the convoy will be within weapons range of that target if the target is not destroyed beforehand). These lines indicate whether the strike team is prosecuting targets quickly enough to maintain the safety of the convoy.

If a **target is located to the left** of its threat envelope the convoy will be safe. For example, target 135 & 145 in the schedule to the right will be destroyed before they can harm the convoy. However, target 17M will be destroyed slightly after it could harm the convoy.



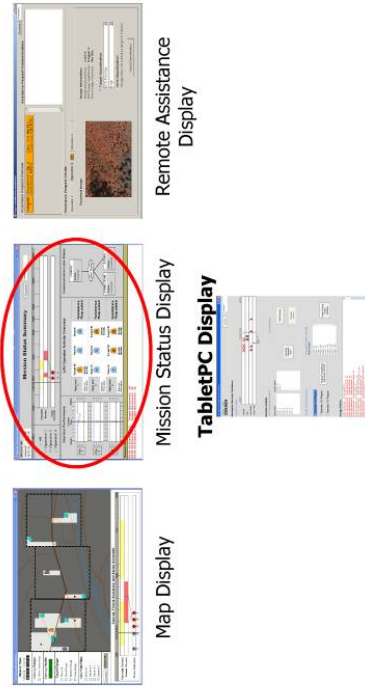
Any Questions?



Advance slide when you are ready ...

Mission Status Display

Large-Screen Wall Displays

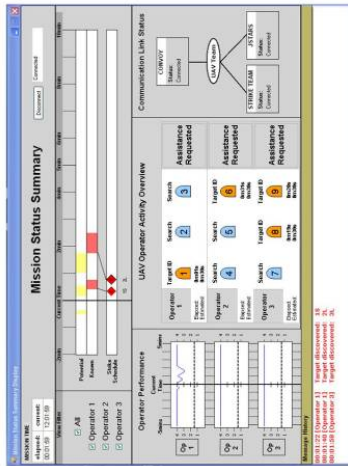


Mission Commander Display

Figure 33. Tutorial Slide 4 (from Wan, 2007).

Mission Status Display

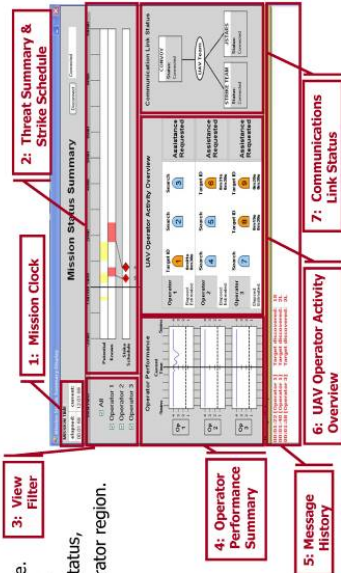
The **Mission Status Display** visualizes current and expected mission status information, which includes surveillance progress for each UAV operator and given areas of interest, communications link status to external resources, and expected target strike schedule for identified targets.



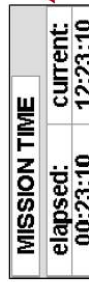
Mission Status Display

The main components of this display, described in detail next, are:

1. The mission clock.
2. Timeline summarizing current and expected convoy threats & the expected time of corresponding target threats.
3. Filter toggles for the timeline data.
4. Graphs summarizing operator performance.
5. Message history box.
6. Panel showing UAV status, broken down by operator region.
7. Current communication link status.



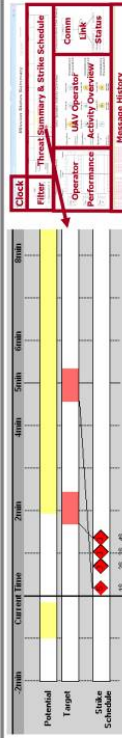
Mission Status Display (1) Mission Clock



The Mission Clock shows the up-to-date mission times, including the **elapsed time** since the beginning of the test session and the **current mission time**.

This mission clock can be found in the upper corners of all experiment displays

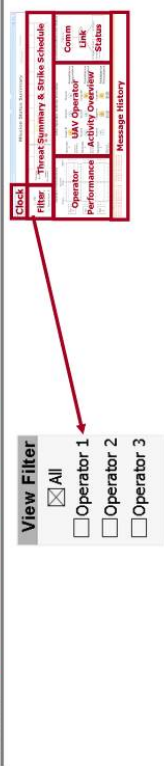
Mission Status Display (2) Threat Summary & Strike Schedule Timeline



The Threat Summary & Strike Schedule timeline displays the temporal relationship between the **scheduled target strikes** and the **known and potential threats to the convoy**. This timeline is a mirror of the one described earlier on the Map Display.

Figure 34. Tutorial Slide 5 (from Wan, 2007).

Mission Status Display (3) Timeline View Filter

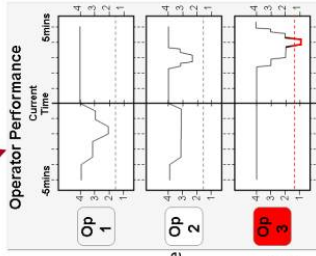
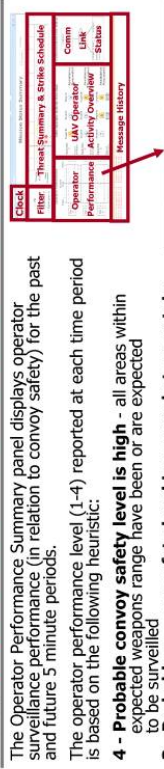


The Timeline View Filter panel provides toggle check boxes to enable you to show or hide target information for all or a subset of the operator regions on the timeline. By default all toggles are set to the 'on' (i.e., 'Show') position.

To **show** target information relating to a particular operator, touch (with your finger or the wall pen) inside the corresponding check box. A checkmark will appear in the box and the threat envelop and target information related to that operator's region will appear on the timeline.

To **hide** the information, touch inside the checkbox again. The checkmark and corresponding timeline information will disappear.

Mission Status Display (4) Operator Performance Summary



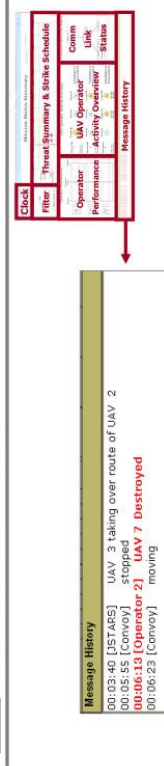
The Operator Performance Summary panel displays operator surveillance performance (in relation to convoy safety) for the past and future 5 minute periods.

The operator performance level (1-4) reported at each time period is based on the following heuristic:

- 4 - **Probable convoy safety level is high** - all areas within expected weapons range have been or are expected to be surveilled
- 3 - **Probable convoy safety level is somewhat uncertain** - convoy within or expected to be within long-range weapons distance to unsurveilled areas
- 2 - **Probable convoy safety level is very uncertain** - convoy within or expected to be within medium-range weapons distance to unsurveilled areas
- 1 - **Probable convoy safety level is low** - convoy within or expected to be within short-range weapons distance to unsurveilled areas

Performance level 4, is ideal. Performance below this is undesirable but minimally acceptable until it reaches level 1, which is considered unacceptable because the convoy is (or will be) entering directly into an unsurveilled area. When performance drops to level 1, the dotted line, the plot line, and the Operator label will be displayed in red to indicate this critical state. This panel is a good indication of past and future operator performance for each region. You may use this information to determine which of the operators need additional assistance.

Mission Status Display (5) Message History

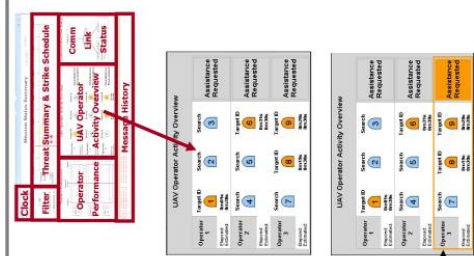


The Message History displays a scrollable list of status and communication messages from team members, external contacts (e.g., the convoy, intelligence sources), and system messages.

Each message begins with a time-stamp indicating when it was received and a label indicating the message source. Critical system messages are displayed in red.

The contents of this message history are mirrored on the TabletPC Display. Sample messages will be provided later in the tutorial.

Mission Status Display (6) UAV Status Activity Overview



The UAV Status Activity Overview panel displays the current state of all UAVs (which can be Search, Target ID, Down).

The timer below the TargetID UAV's keeps track of how long the UAV has been loitering (in order to identify targets).

One thing to keep track of is the Elapsed time when a UAV is loitering (orange). If the elapse time becomes longer than 30 seconds, you may want to assist the operator by using the Remote Assistance Display (see detailed instructions in next section).

Note that when the Assistance Request is accepted by the operator, the Operator panel will turn orange.

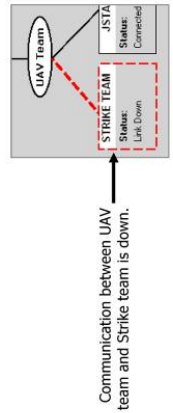
Figure 35. Tutorial Slide 6 (from Wan, 2007).



Mission Status Display (7) Communications Link Status

The Communications Link Status panel displays the current status of your UAV team's communication connections to external contacts:

- Convoy, Strike team, & JSTARS (your external intelligence source)
 - When the line connecting the UAV Team oval and one of the external contacts is solid black, there is a strong connection. When the connection line is broken (i.e., dotted), the communications line is down.
- Each box representing the external contacts displays
- Contact name
 - Current communication status



Communication between UAV team and Strike team is down.



Any Questions?

Mission Status Summary

Operator Performance: Shows graphs for Operator 1, 2, and 3.

UAV Operator Activity Overview: Shows activity for Operator 1, 2, and 3 with search and target ID fields.

Communication Link Status: Shows connections for CONVOY, STRIKE TEAM, and JSTARS.

Message History: Lists messages such as 'Target discovered!', 'Target discovered!', and 'Target discovered!'.

Advance slide when you are ready...



Remote Assistance Display

Large-Screen Wall Displays

Map Display

Mission Status Display

TabletPC Display

Mission Commander Display



Remote Assistance Display

The Remote Assistance Display (RAD) allows the mission commander to assist the remote UAV operators who are taking too long (>30 seconds) to perform the target identification task.

Remote Assistance Display (RAD) interface showing target identification assistance.

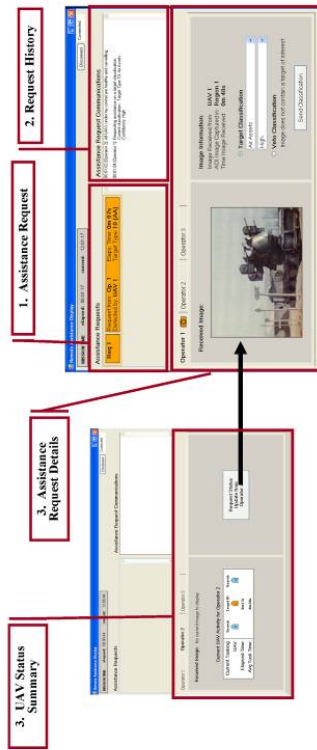
Figure 36. Tutorial Slide 7 (from Wan, 2007).



Remote Assistance Display

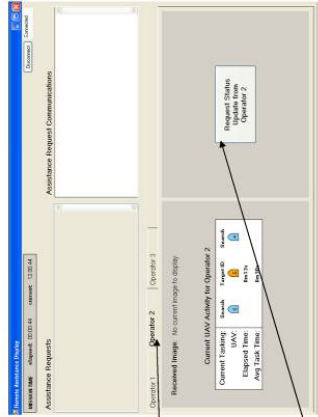
The main components of this display are:

1. Assistance Request.
2. Request History.
3. UAV Status Summary or Assistance Request Details (invoked by clicking "Request Status Update From Operator X" in UAV Status Summary).



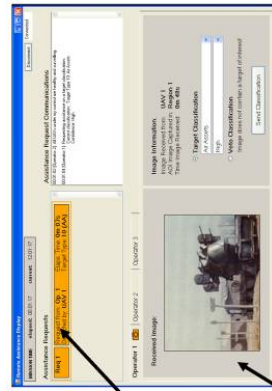
Remote Assistance Display (1) Request Status Update from Operator X

When any of the UAVs are in the process of TargetID, the Mission Commander may assist the operator by selecting the corresponding Tab of the Operator number and clicking on "Request Status Update From Operator X".



Remote Assistance Display (2) Assistance Request

After the "Request Status Update From Operator X" has been clicked, there will be a short delay (communication lag time) and then the request will be added to the Assistance Request box.



By clicking the request box, the target image details will appear.



Remote Assistance Display (3) Request Details

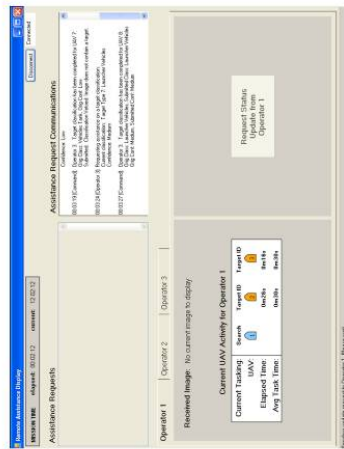
For each assistance request, the details displayed include: UAV, time received, Target Classification, and Target image. It is the responsibility of the mission commander to:

- To the best of your ability, match the target classification to the image.
- Veto any non-military targets (you may assume any target with weaponry threat should be destroyed, ie. no such thing as friendly fire)



Figure 37. Tutorial Slide 8 (from Wan, 2007).

Any Questions?

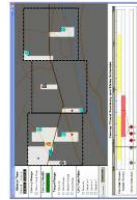


Advance slide when you are ready...

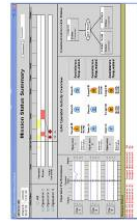
Mission Commander Display



Large-Screen Wall Displays

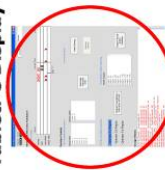


Map Display



Mission Status Display

TabletPC Display



Mission Commander Display



Remote Assistance Display

Mission Commander Display



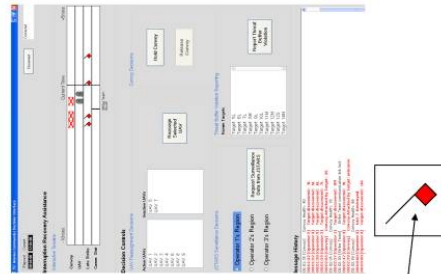
The TST mission commander decision display will be shown on the TabletPC you will use in the study.

All of your command decisions are on this display. As mentioned earlier, the ultimate goal of mission is to **keep the convey safe while moving through the mission as quickly as possible**. In order to achieve this mission goal you will have three types of commands at your disposal:

1. Convey Commands: Hold / Release position.
2. Requesting additional surveillance information from a nearby intelligence source (JSTARS) (limited usage).
3. Reassigning a UAV from one part of the AOI to another to replace a downed UAV (once per UAV).

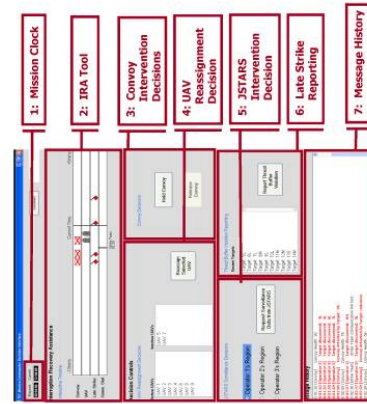
Also mentioned earlier, another part of your job as mission commander will be to monitor the strike team performance and report any unsatisfactory strike support. For the purposes of this study, unsatisfactory strike support will be defined as instances of "late strikes", that is, instances of scheduled target strikes such that the strike schedule target appears to the right of its corresponding threat envelope.

The specifics of this display and the corresponding task functionality will be explained in the following slides.



The main components of the TST mission commander decision display, described in detail next, are:

1. The mission clock.
2. Interruption Recovery Assistance (IRA) tool.
3. Convey decisions panel.
4. UAV reassignment panel.
5. JSTARS surveillance decision panel.
6. Late Strike reporting panel.
7. Text message box.



Mission Commander Display



Figure 38. Tutorial Slide 9 (from Wan, 2007).

Mission Commander Display
(1) Mission Clock



Elapsed: 00:00:00

Current: 00:00:00

Clock

IRA Timeline

Convoys Controls

Late Strike Reporting

Message History

The Mission Clock shows the up-to-date mission times, including the **elapsed time** since the beginning of the test session and the **current mission time**.

This mission clock can be found in the upper right corner of all three experiment displays

Mission Commander Display
(2a) IRA - Overview



Interactive Timeline

Current Time: 05:00:00

Convoys	UAV	Late Strike	Comm. Stat	Comm. Stat	Comm. Stat
[Icon]	[Icon]	[Icon]	[Icon]	[Icon]	[Icon]

The IRA (Interruption Recovery Assistance) panel is an interactive panel consisting of a timeline and iconic bookmarks which are clickable by the user. These interactive bookmarks help the mission commander quickly retain situation/activity awareness of relevant events to decision making. It is designed to be especially useful after returning from an interruption.

The timeline shown above displays the "interactive bookmarks".

The timeline keeps track of four types of recent events. With the exception of "Comm. Stat" events, clicking on the icons will bring up additional information on the Map Display. After 5 seconds, this supplemental information will fade away.

Mission Commander Display
(2b) IRA - Event Detection Descriptions



The four events that will be monitored by the IRA tools are listed below.

When the event occurs, the event bookmark will appear on the IRA timeline. In the first three events, if the user clicks on a bookmark, the corresponding information appears on the Map Display for 5 seconds before fading away. When Comm. Stat events occur, additional information will be presented below the timeline.

Icon	Event	Triggered By:	Click action:
[Red X]	Convoys Attacked	Convoys gets attacked	Draw a red X on the spatial map (Situational Map Display) where convoys was hit.
[UAV]	UAV Destroyed	UAV is destroyed	Draw a red X over the destroyed UAV on the spatial map.
[Red Arrow]	Late Strike	A buffer violation (target will not be eliminated before convoys arrives).	Draw a box around the corresponding target that will not be eliminated on the strike schedule.
[Red Dotted Line]	Communication Status Change	If any of the communication (JSTAR, Convoys, UAV) links changes status (connect/disconnect).	Display red dotted line on IRA timeline if communication was disconnected, show black solid line if communication was reconnected.

Mission Commander Display
(2c) IRA - Convoys Attacked



Interactive Timeline

Current Time: 05:00:00

Convoys	UAV	Late Strike	Comm. Stat	Comm. Stat	Comm. Stat
[Icon]	[Icon]	[Icon]	[Icon]	[Icon]	[Icon]

Clicking one of the red convoys icons (see above) on the IRA timeline will reveal further detail in the Map Display. A red X (see right) indicates on the spatial map where the convoys was hit. The red X will fade after 5 seconds.

Figure 39. Tutorial Slide 10 (from Wan, 2007).

Mission Commander Display (2d) IRA – UAV Destroyed



Mission Commander Display (2e) IRA – Late Strike



Clicking on the late strike icon (see above) will reveal further details in the Map Display. A black box (see right) will be drawn on the Strike Schedule panel to indicate which target will become a threat before the strike team will be able to eliminate it in the near future. Also, the other targets will be dimmed to give emphasis. This box will fade after 5 seconds.

Clicking on the gray UAV icon (see above) will reveal further detail in the Situational Map Display. A transparent red X (see right) will be drawn on the spatial map to indicate which UAV was destroyed. The X will fade after 10 seconds.

Mission Commander Display (2f) IRA – Communication Status Change



A solid black line indicates an event in which the communication link was restored. The timer below the line displays the total length of disconnect.

The dotted red line on the Comm. Stat panel indicates an event in which one of the communication links was broken (JSTARS, Strike Team, or Convoy). The timer below the line shows the duration of the disconnect.

The Convoy Decisions panel enables you to:

- **Hold the convoy** – i.e., send a command to the convoy to hold its current position.
- **Release the convoy** – i.e., send a command to the convoy to resume its progress on its planned path.

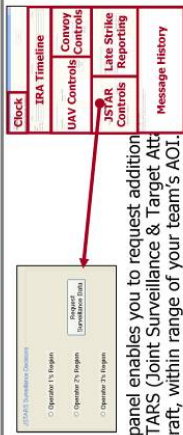
NOTE: When the Convoy Communication link is disconnected, you will not be able to toggle between Hold/Release (i.e. The state of the convoy cannot be changed).



Mission Commander Display (3) Convoy Intervention Decisions

Figure 40. Tutorial Slide 11 (from Wan, 2007).

Mission Commander Display (4) JSTARS Intervention Decision



The JSTARS Surveillance Decision panel enables you to request additional surveillance data from a nearby JSTARS (Joint Surveillance & Target Attack Radar System), a multi-sensor aircraft, within range of your team's AOI.

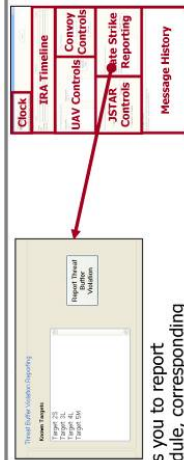
Requesting the JSTARS surveillance information will reveal all areas expected to be within short, medium, and long weapons range of the convoy over the next two minutes within the selected sub-AOI.

Any identified targets will be confirmed and added to the strike schedule by the JSTARS crew. It would be appropriate to request JSTARS data, for example, if the convoy is approaching an area where the UAVs have not (or will not have) time to surveil.

To request the JSTARS surveillance data, select the radio button corresponding to the desired operator region and then select the 'Request Surveillance Data from JSTARS'. A message will be received from JSTARS and the Situation Map and Mission Status Displays will be updated accordingly.

Note, you will only be able to request JSTARS surveillance data **up to 3 times** per mission (once per region). So you should use it **sparingly**.

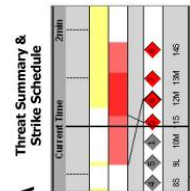
Mission Commander Display (6) Late Strike Reporting Detection



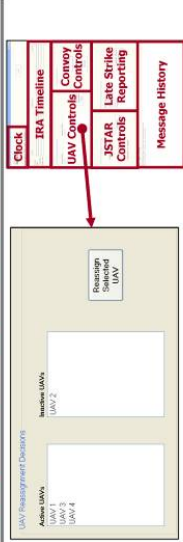
The Late Strike Reporting enables you to report any late strikes in the strike schedule, corresponding to unsatisfactory strike support.

A **late strike** is an instance when a strike is scheduled after a known threat envelope is beginning. A late strike can be determined from the threat summary & strike schedule timeline on both large screen displays. For example, target 12M in the displayed timeline is an example of a **late strike**: 11S is not a late strike because it will be destroyed before the convoy will be within its weapons range. The **IRA Late Strikes panel** should assist you in identifying late strikes.

You are expected to report as many instances of late strikes as possible. To do so, select the respective target from the 'Known Targets' and then selecting the 'Report Late Strike' button.



Mission Commander Display (5) UAV Reassignment Intervention Decision

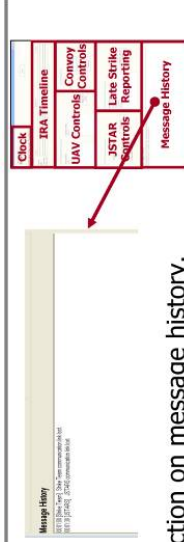


The UAV Reassignment Decision panel enables you to reassign one of your team's UAV to take over the remaining surveillance route of another UAV that has been destroyed.

To reassign a UAV, select the desired UAV in the 'Active' list on the left side of the panel, then select the 'downed UAV' from the 'Inactive' list on the right side of the panel, and finally, select the 'Reassign UAV' button. The reassigned UAV will immediately begin moving toward the downed UAV to take over its route.

Note, reassigning a UAV will cause it to **abandon** its own mission route and assume the mission route of the downed UAV. Due to limited endurance capabilities of the UAVs, once a UAV is reassigned to a new route, **it will not be able to resume its prior mission route**. You will only be able to reassign a UAV **once**.

Mission Commander Display (7) Message History



See previous section on message history.

Sample messages:

12:28:27: [Strike Team] Target 3M scheduled to be destroyed at 12:31:00.

12:30:15: [System] Convoy entering weapons range of target 3M!

12:30:35: [Convoy] Under attack!

12:31:30: [Convoy] Damage report: energy level at 80%.

Figure 41. Tutorial Slide 12 (from Wan, 2007).



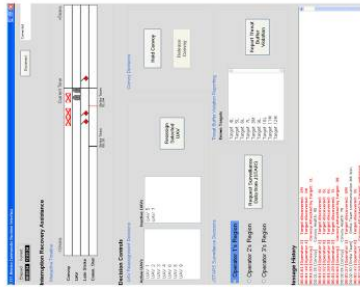
Your Mission Priorities:

Equally high priority:

- Move the convoy through your AOI as quickly as possible
- Keep the convoy safe (i.e., free from target attacks)

Lower priority:

- Assist your operators in completing the assigned set of target designations
- Report late strikes
- Survey all roads in your AOI
- Complete Incidence Report documents after interruptions.



Advance slide when you are ready ...

Figure 42. Tutorial Slide 13 (from Wan, 2007).