Data Fusion Engine (DFE) for the Force Protection Joint Experiment (FPJE)

C.M. Barngrover, R.T. Laird, T.A. Kramer, J.R. Cruickshanks, S.H. Cutler
Space and Naval Warfare Systems Center, San Diego (SSC SD)
53560 Hull Street, San Diego, CA 92152

ABSTRACT

The FPJE was an experiment to consider the best way to create a system of systems in the realm of Force Protection. It was sponsored by Physical Security Equipment Action Group (PSEAG) and Joint Program Manager – Guardian (JPM-G), and was managed by the Product Manager - Force Protection Systems (PM-FPS). The experiment attempted to understand the challenges associated with integrating disparate systems into a cohesive unit, and then the compounding challenge of handling the flow of data into the system and its dispersion to all subscribed Command and Control (C2) nodes. To handle this data flow we created the DFE based on the framework of the Joint Battlespace Command and Control System for Manned and Unmanned Assets (JBC2S).

The DFE is a data server that receives information from the network of systems via the Security Equipment Integration Working Group (SEIWG) ICD-0100 protocol, processes the data through static fusion algorithms, and then publishes the fused data to the C2 nodes, in this case JBC2S and the Tactical Automated Security System (TASS). The DFE uses only known concepts and algorithms for its fusion efforts. This paper discusses the analyzed impact of the fusion on C2 nodes displays and in turn on the operators. Also, this paper discusses the lessons learned about networked control combined with DFE generated automatic response. Finally, this paper discusses possible future efforts and their benefits for providing the useful operational picture to the operator.

Keywords: JBC2S, MOCU, MRHA, command and control, data fusion, networked control, force protection

1. BACKGROUND

The primary objective of the FPJE over the one year period was to evaluate the impact of data fusion and automation on system performance and in turn to evaluate the overall system performance. The experiment began with the sensor selection and the subsequent integration of each sensor into the system of systems (SoS). The integration of these sensors was done in tiers during the first and second Integration Assessment events. Through most of the experiment, the system consisted of dual command and control interfaces through JBC2S and TASS. The integration and testing of all the sensors into the system culminated in the third Integration Assessment (IA-III). During IA-III there were multiple JBC2S stations and a TASS station that shared data in a star topology. It was during this event that it became clear that the shear amount of data coming into the system was a hindrance to the operator. Furthermore, it was clear that having two different C2 applications would increase operator training and decrease operator effectiveness. The focus of this paper is the work that was done in preparation for IA-IV as well as the continuing work that has been done since the conclusion of the experiment. The main efforts for the final IA included work on the DFE and work on a module to replace the functionality of TASS that was not handled by JBC2S, specifically sensor and camera integration. The DFE was built on the infrastructure of JBC2S and its main roles were data distribution, data fusion, networked control, and automation. The module to replace TASS integration capabilities is called the Tactical Device Manager (TDM) and specifically communicates with TASS Sensors, Wide-Area Surveillance Thermal Imager (WSTI) and Long Range Thermal Imager (LRTI) cameras, and Battlefield Anti-Intrusion System (BAIS) Sensors.

2. JBC2S OVERVIEW

2.1 Background

The Joint Battlespace Command and Control System (JBC2S) has a long history beginning with the Multi-Robot Operator Control Unit (MOCU), which was originally developed as a tactical operator control unit for a small man-portable UGV called the URBOT. Since its start MOCU has evolved to support many other unmanned systems including the SPARTAN unmanned surface vehicle (USV) and the iRobot Packbot. MOCU is designed to support new protocols, video CODECS, mapping engines, and other components without modifying the core software through the use of modules with predefined interfaces.

JBC2S branched off of MOCU as a result of a need for integrated control of unmanned systems as well as other sensors including camera, radars, ground sensors, etc. As JBC2S evolved it also incorporated a new protocol based on the Multiple Resource Host Architecture (MRHA), which was developed at SSC Pacific as the command-and-control system for the Army's Mobile Detection Assessment Response System (MDARS) program. MDARS is a fixed-site robotic security system that consists of multiple semi-autonomous unmanned ground vehicles (UGVs) performing automated patrols, as well as intrusion sensors and active barriers. This new protocol allowed for most of the MRHA functionality to exist in JBC2S.

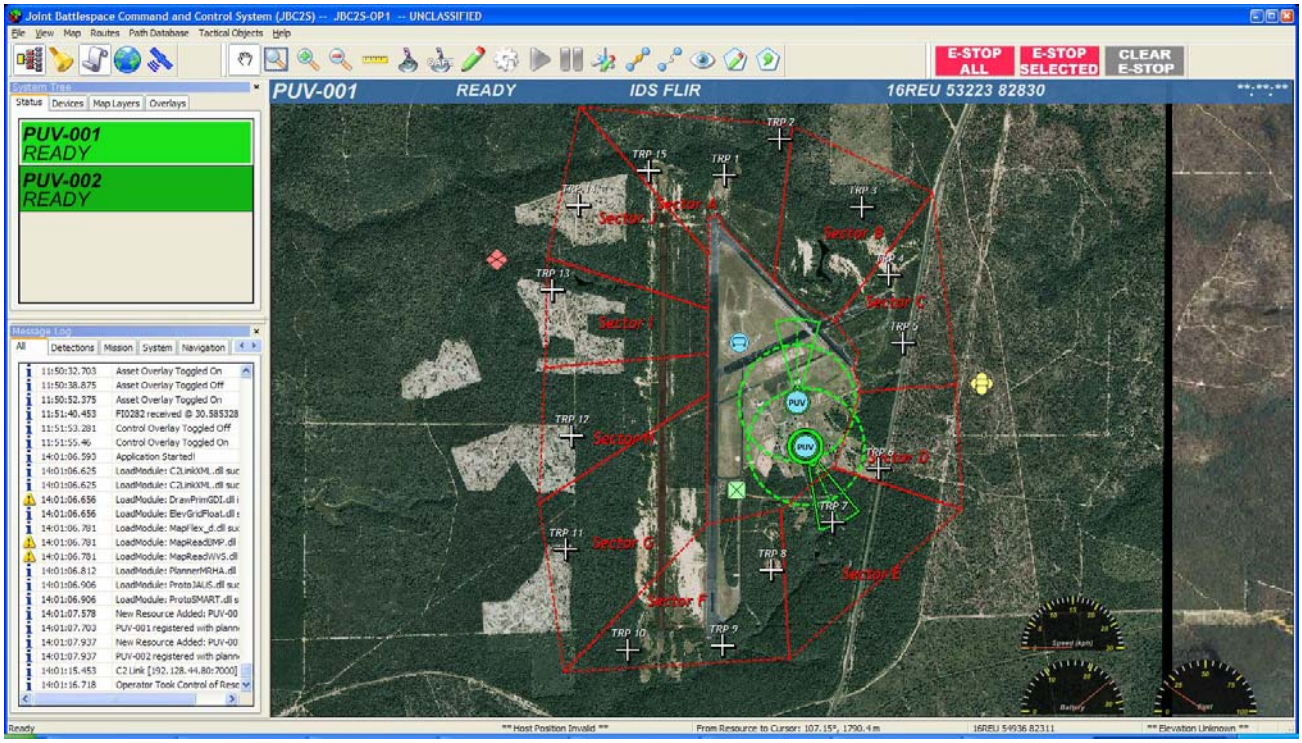

Figure 1: JBC2S software with two MDARS UGVs. This screenshot shows the ability to have sectors, points of interests, and targets. Each UGV also has a camera field of view and a radar coverage arc.

2.2 C2 Link Module

The Command and Control (C2) Link Module is a TCP/IP based framework for communicating protocols between control software. The protocol used for C2 communication during this experiment was the SEIWG ICD-0100 protocol. This XML based protocol was first adopted in 2006 to demonstrate integration with the US Army's Counter Rocket Artillery Mortar (C-RAM) program. In the C-RAM demonstration, JBC2S would deliver and receive basic XML messages to and from the Enhanced Tactical Automated Security System (eTASS), which would then interface with other Army C2 Systems.

The C2 Link Module creates and manages a series of client and server sockets, as well as all persistent TCP sockets from current links. Each persistent TCP socket has its own reference to a protocol parser; in this case all sockets used the ICD-0100 protocol parser. There are two main data flows through the C2 Link Module. First, when data changes within the core of JBC2S (i.e. robot location), the C2 Link Module is notified. Then the C2 Link Module will create the appropriate XML message for each socket and push the message through. Conversely, if a full XML message is received across one of the sockets, the message is parsed and the relevant information is passed through to the core of JBC2S. Entering into the FPJE, this module had been thoroughly tested with regards to connectivity, data integrity, data throughput, and stability.

2.3 Control Methodology

In order to simplify the user interface, JBC2S operates under a very simple rule set. The system is either in Monitor Mode, in which the operator observes and monitors the status of the unmanned vehicles and sensors (known collectively as resources), or the operator is in Direct Control of a single resource, such as a single MDARS robot. In addition, the robot itself carries a number of systems, each of which must also be controlled individually. For example, MDARS may have eight cameras, two radars, and four lights, but only a single camera, radar, and light is controllable at any given time. Multiple cameras can be viewed simultaneously, but the operator cannot pan two of them at the same time. A joystick button or dialog button can be configured to cycle between the various payloads. The control of the resource and each of its selected sub-resources is integrated in such a way that a single joystick can be used to perform all relevant joystick-requiring tasks. For example, the operator can pan the selected camera with the joystick, and then press a button to put the robot in teleoperation mode and drive the whole robot using the same joystick. The concept of modes is used to provide rich control features for each resource. This paradigm reduces the complexity of the software and simplifies the user interface presented, while still giving the operator ample control of each asset.

3. DATA FUSION ENGINE

3.1 Framework

Following the third IA it was clear that we needed a way to remove clutter from the screen to remove burden from the operators. Given our time and funding constraints, we decided to solve this problem in-house, rather than through a contractor, as a proof of concept for data fusion benefits. We decided to build upon the framework of JBC2S and its C2 Link Module. The DFE would consist of an in-memory database of the state of the system, a Data Fusion Module, and a C2 Link Module. The first step was pulling out the in-memory database and C2 Link Module of JBC2S. Next, we focused on implementing the Fusion Engine, Response Engine, and data distribution. The Fusion Engine combined related data so it could be provided to the operator in a more concise format. The Response Engine allowed for automation of the system based on incoming data and the general state of the system. Data distribution was the method by which multiple operator stations could simultaneously receive pertinent fused data.

3.2 Data Distribution

One of the main functions of the DFE was as a data distribution server. The majority of sensors were connected directly to the DFE through the commercial gateway processor, which essentially translated the sensor's native protocol to the ICD-0100 protocol used by the DFE. However, some systems had a more complicated path to the DFE. The unmanned systems, including MDARS and the Air Force Research Lab's (AFRL) Defender vehicle, communicate to specific JBC2S operator stations via the MRHA and the Joint Architecture for Unmanned Systems (JAUS) protocols respectively. The JBC2S operator stations would then be responsible for providing all relevant unmanned systems data to the DFE via the ICD-0100 protocol. There were also a few systems that communicated with the TDM, including BAIS and TASS sensors. The TDM replaced the translation function of TASS by converting the native protocols of cameras and sensors into ICD-0100. This was an important element because, along with additional features to JBC2S, this allowed us to duplicate all of the TASS functionality for IA-IV. As can be seen in Figure 2, all the communication through the DFE was via the SEIWG ICD-0100 protocol. This figure also shows that the JBC2S Battle Captain was also connected to the DFE in much the same manner as JBC2S operator stations.

An important aspect of the data distribution is managing the data flow so that no cycles are created. In order to control the data flow, we needed to know which sockets were providers only and which sockets were also subscribers. The ICD-0100 protocol has a beautiful message specifically for this called SubscriptionConfiguration. Ideally, we would have implemented this message in the DFE and subsequently in all connected systems. However, due to time restraints,

we could not do this implementation and had to resort to a less elegant solution. The solution was to have multiple server ports in the DFE, each representing a certain type of connecting system. There was a port for systems that just provided

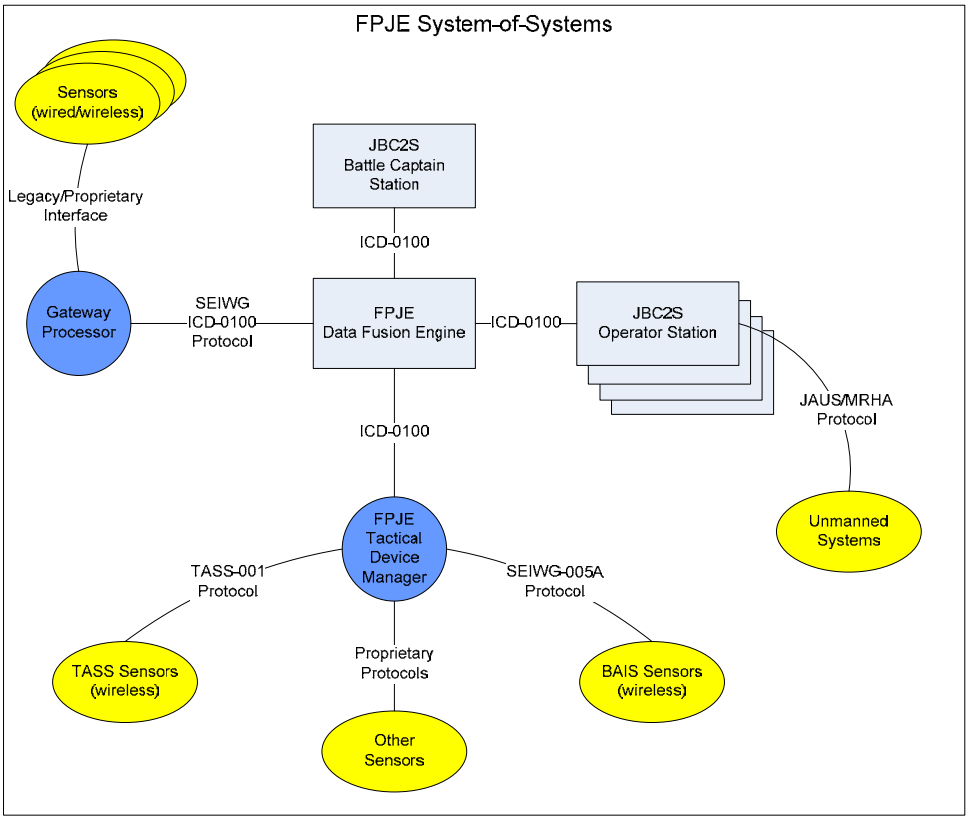

Figure 2: The System View for IA-IV with a focus on how the systems and sensors connect to the DFE. All used protocols are shown, including ICD-0100 as the only protocol through the DFE.

data and did not want to receive any data back. There was a port for systems that would provide data but would also like some information including targets and zones. Finally there was a port for systems that would provide data and would also like to receive all fused data. For IA-IV, the only system that would connect to the full provide and subscribe port was JBC2S since it was the sole C2 software used. Note that there was fourth port that would provide the raw, un-fused data for analysis purposes.

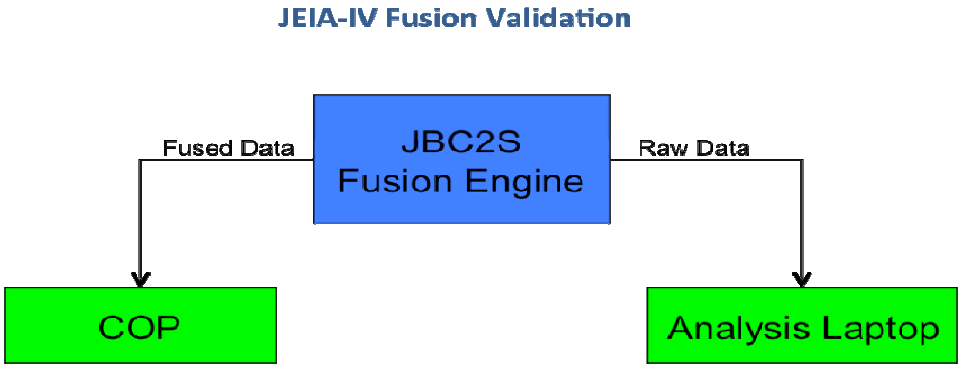

Figure 3: Shows how the fusion was validated following IA-IV via comparison of Fused Data, Raw Data, and known GPS locations of the Opposing Force (OPFOR).

3.3 Fusion Engine

The Fusion Engine component of the DFE is the component that helps us mold the data from a collection of information into a unified knowledge picture. The data fusion done in this section is all static fusion. This means that when a new piece of information enters the system, it is compared to a snapshot of the current state of the system to look for duplications and possible interpretations. Figure 4 shows a screen shot of the Fusion Engine Options Dialog which allows a user to control the settings for the Fusion Engine component of the DFE. The Fusion component is broken into two types of input data, Tactical Object and Detection. Tactical Objects are actual targets whose location, and possibly heading and speed, are determined by a tracking sensor, such as a radar. Detections are point alarms generated by sensors, such BAIS, which only indicate that something in the general vicinity of the sensor may be a threat. All input data is then compared to other state data and to blue force locations determined by UGV GPS and Blue Force Tracking such as Remote Tracking Systems (RTS).

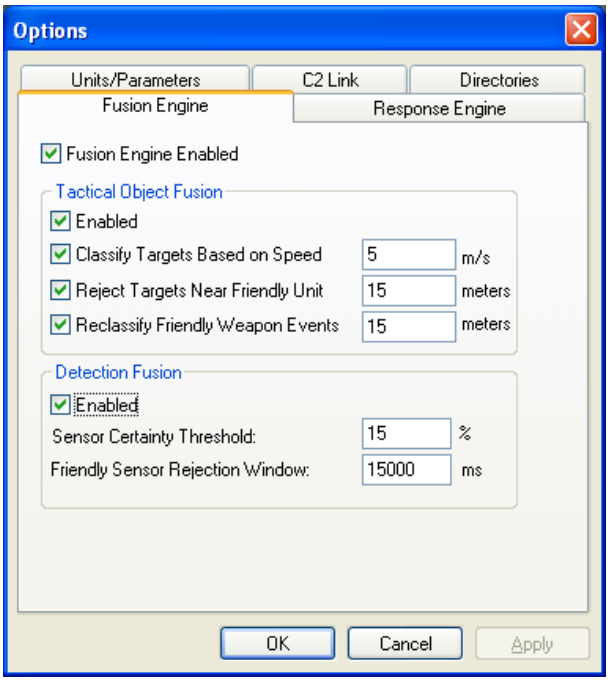

Figure 4: Screenshot of DFE Options Dialog for the Fusion Engine component of the DFE.

The Tactical Object Fusion based on targets includes four fusion elements. First is a situation where the DFE receives a new target with Unknown classification. The Fusion Engine will then compare the new target's location with all known locations of friendly assets. If the new target is within the given threshold of any friendly, 15 meters in Figure 4, then the new target is ignored as redundant. For this fusion element there are two stages of comparison, one for the unmanned system locations and one for the RTS personnel locations.

The next situation handled by the Fusion Engine is when a Weapon Fired target, which is initially classified as Hostile, is received from the WeaponWatch system. In this situation there are three stages of consideration. First the Weapon Fired location is compared to all unmanned system locations, since both unmanned systems have less than lethal weapon capabilities. If the Weapon Fired is within the given threshold of an unmanned system, 15 meters in Figure 4, then the target is reclassified as a Friendly Weapon Fired. The following stage is to compare the Weapon Fired location with all other Tactical Objects in the system. If there is a Tactical Object with Unknown classification within the threshold, then the Unknown Tactical Object is reclassified to Hostile and the Weapon Fired target remains Hostile. If a Friendly Tactical Object is within the threshold, then the Weapon Fired target is reclassified as Friendly. In the final stage for this situation, the Fusion Engine creates a Hostile Tactical Object in the same location as the Hostile Weapon Fired target. It was quickly discovered during testing that the shear number of Weapon Fire targets generated during a fire fight required a decay mechanism in order to declutter the display. All Weapon Fired targets are decayed after a constant period of time, but a Tactical Object remains with this knowledge included in its location and status.

Another situation that the Fusion Engine supports is duplication. When the DFE receives a new Tactical Object, it is compared with all other Tactical Objects to compare classification and location in order to determine if the new one is a duplicate. In the case where the new Tactical Object is determined to be a duplicate, it is simply rejected from the system. Otherwise, the new Tactical Object is distributed on all appropriate sockets.

The final situation handled by the Fusion Engine involves type classification of Tactical Objects. Often the speed of a Tactical Object is included in its data from a radar, which is useful information. If the Tactical Object is over a given speed threshold, 5 m/s in Figure 4, the target is then reclassified as a vehicle. This is based on the assumption that there is a certain speed at which a person cannot travel on foot.

The Detection Fusion element of the Fusion Engine includes only two methods. First is to compare the location of a sensor that is reporting a detection to the locations of all friendly assets, including unmanned system and RTS locations. Because of delays in the system, this comparison is a little more complicated. If any friendly asset passed through the coverage area of the sensor, determined by the sensor type, during a given time period, 15 seconds in Figure 4, then the detection is suppressed.

The second method for Detection Fusion is based on analysis of BAIS detections during the early stages of the FPJE. All of the calculations for the certainty percentage are done in the TDM which communicates directly with the BAIS receiver. The basis for these calculations are discussed in another paper titled, "Force Protection Joint Experiment (FPJE) Battlefield Anti-Intrusion System (BAIS) Sensors Data Analysis and Filtering Metrics." For the Fusion Engine, the rule is to suppress the detections unless they are given with a certainty over the given threshold, which is 15% in Figure 4 but was set at 80% in practice. During this experiment, the certainty threshold was valuable in suppressing noisy data from nearby roads and other unknown causes.

The fusion elements described in this section are basic relative to some of the possibilities out there. The goal of the DFE for this experiment was to show that even basic fusion could make a COP or operator station more usable in a real situation by removing extraneous information. During IA-IV we saw the benefits of the fusion and the possibilities for future development in this realm. The work on the Fusion Engine component from this experiment has been highly utilized in the Joint Force Protection Advanced Security System (JFPASS) Joint Capability Technology Demonstration (JCTD). The underlying goal of the FPJE was to mitigate risk for the JFPASS JCTD through work on similar problems.

3.4 Response Engine

The Response Engine component of the DFE is the component that allows for more efficient utilization of assets, specifically operators. The Response Engine is only able to respond to Tactical Objects currently, due to software development time constraints. When a new target comes into the system and passes through the Fusion Engine, it then enters the Response Engine, where there are many possible outcomes based on software configuration. Whenever the response generated is a controlling response, the DFE then creates a CommandMessage (part of the ICD-0100 protocol) to send to the appropriate asset. Figure 5 shows a screen shot of the Response Engine Option Dialog which allows you to control the settings for the Fusion Engine component of the DFE.

The main response element of the Response Engine is based on Hostile Tactical Objects relative to varying zone types. There are two main responses to Hostile Tactical Objects, each with their own zones. The first is camera response, which has no corresponding configuration elements in the Response Engine Options Dialog of Figure 5. This response is triggered entirely based on the names of the zones saved in the Response Engine directory. So when configuring, a zone is drawn on the map and named for the corresponding camera. When a Hostile Tactical Object is received, its location is compared to all camera zones and, for all matching zones, the DFE sends a CommandMessage to the appropriate cameras commanding them to look at the Hostile target location.

The unmanned system responses are generated in much the same way, but with more complicated options. In the configuration stage of setup, a zone is drawn on the map for each unmanned system. The name of the zone must match the name entered in the robot field of the Response Engine Options Dialog shown in Figure 5. When a Hostile Tactical Object is received, its location is compared to all robot zones, and the DFE sends a CommandMessage to all relevant UGVs commanding them to navigate to the location of the Hostile target. Notice in Figure 5 that there is a checkbox for "Closest Only" and "Hostile Response Track w/ Robot". The Closest Only feature causes the DFE to only send the robot

closest to the destination if the Hostile Tactical Object is in multiple robot response zones. The Hostile Response Track with Robot feature cause the UGVs camera to track the location of the Hostile target as it navigates to the location.

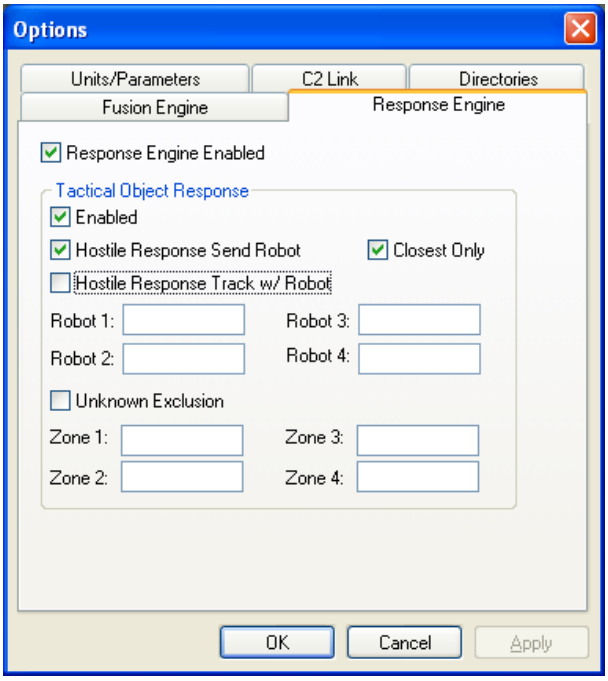

Figure 5: Screenshot of DFE Options Dialog for the Response Engine component of the DFE.

The other response element is a filtration system based on user configured exclusion zones. When an Unknown Tactical Object is received, it is compared to the exclusion zones. If the target is in any of the exclusion zones, then it is automatically rejected. This allows the user to de-clutter high traffic areas where radars sense too much activity.

During the 4th FPJE IA, the response engine was demonstrated to have great utility for the automated slewing of cameras, but the realities of autonomously tasking large UGVs based on radars and ground sensors showed that further configuration options were needed. The most obvious deficiency in the response mechanism described above is that UGVs will be tasked to response to the actual location of an event rather than an over-watch position as a human operator would. The response engine would task all the UGVs too far towards the perimeter, exposing them to hostile fire. What is really needed is a set of pre-defined over-watch positions from which the response engine will select the most appropriate based on the location, speed, and course of a radar target or sensor detection. The goal is to position the UGVs between the threat and high value assets so that a human operator can then take control of the UGV's cameras, microphone, speaker, and lethal/non-lethal weapon systems to respond appropriately based on the Rules of Engagement.

3.5 Networked Control

The concept of networked control of assets was slightly utilized early in the experiment for simple tasks such as "Goto" commands for sending UGVs to locations or for slewing a camera to look at a location. Then, after IA-III, this concept was expanded to allow for much more control. All of this networked control was done through the SEIWG ICD-0100 CommandMessage.

For IA-IV, our networked control involved full control of any asset from any JBC2S station, with the exception of teleoperating the UGVs. At the time this was more of a safety constraint than a capability constraint. This full control from any station meant that operators no longer needed to have assigned assets per se. It turns out that it still makes sense to have primary assets with the option to take control of secondary assets as necessary. This addition to the system was important for work load and response times, because if one operator was busy tracking a potential target with a

camera or UGV, another operator could take control of another asset in that same sector and give redundancy visual of the target.

The problem that arose with all assets being available to four JBC2S Operators, one JBC2S Battle Captain, and one DFE is conflict of control. Since there was no sense of who had control of a resource, the resource would accept and act upon commands from multiple JBC2S Operators at the same time. This caused unusual symptoms, such as a camera jumping around from left to right or shaking. Even if there was only one operator controlling a resource, there was always a chance that the DFE Response Engine component would trigger a response involving the same resource. It became quickly evident that there needed to be some sort of mechanism for locking a resource when an operator takes control, so that no other system could attempt to control it. There also needed to some concept of priority, so that the most important controller could override the locked control by a lower priority controller.

4. FUTURE EFFORTS

There are four elements of the FPJE that are essential to the work being done for the JFPASS JCTD, including integration, data fusion, data response, and networked control of assets. The work done on these areas during the FPJE has placed our team in a great position for success in the next generation work. Not only the experience and techniques used, but also a lot of the actual software can be reused and applied. These elements all involve the ICD-0100 protocol, a fact which highlights the protocol's importance to this system.

The data fusion and data response elements have been taken over by ICx, a contractor, for the purposes of the JFPASS JCTD, but their team is working closely with the engineers from the FPJE, so that all of the accumulated knowledge can be applied. Because of the data fusion and data distribution element of the DFE, we have since implemented the SubscriptionConfiguration message of the ICD-0100 in JBC2S and the new ICx software called CohesionIF. This allows for more robust control of which messages a system would like to receive from the server.

The networked control issues of FPJE led us to add extended commands to the ICD-0100 protocol to allow for requesting control of an asset. We also have incorporated a control status element into the status messages, so that JBC2S or any other controlling entity can show which assets are currently locked. This system fixes the problem of conflicting control of assets while still allowing operators to switch from controlling primary assets to controlling secondary assets with relative ease. Similarly, this fixes the issue of the response engine responding with an asset being controlled by an operator. If the asset is locked, then the automated response will not occur, at least not with that asset. Currently, if an asset is locked, the only option is to verbally ask that operator to release control. The future of networked control for the JFPASS will involve priorities and control overrides.

Another important effort resulting from work during the FPJE is a revamping of the ICD-0100 messages for networked control. The usefulness of this feature was obvious, and so our team reworked the messages to be more consistent, logical, and efficient. In addition to reworking the actual messages, we implemented the SupportedCommands tag set in the status messages, so that an asset could publish which commands it supports. With an ever-expanding list of valid commands, this is important so that a C2 software like JBC2S gives the operator only the valid options via the user interface. This avoids confusion for an operator and possible errors from unknown messages reaching an asset.

There is much work to be done on data fusion, response, and networked control during the remaining time for the JCTD, but it is clear that the work of the FPJE has given this program a head start. Now we are in a position to implement more useful and complex fusion algorithms, add more robust response configuration, and utilize all the benefits of networked control of assets. The resulting system should be an exciting milestone in force protection systems.

REFERENCES

[1] D. Powell, G.A. Gilbreath, *Multi-robot Operator Control Unit (MOCU)*, SPIE Proc. 6230-67, Orlando, FL, 2006.

[2] T.A. Kramer, C.M. Barngrover, R.T. Laird, J.R. Cruickshanks, M. Dinh, *FIRRE Joint Battlespace Command and Control System for Manned and Unmanned Assets (JBC2S)*, SPIE Proc. 6230-82, Orlando, FL, 2006.

[3] C.M Barngrover, R.T. Laird, T.A. Kramer, J.R. Cruickshanks, S.H. Cutler, *Force Protection Joint Experiment (FPJE) Battlefield Anti-Intrusion System (BAIS) Sensors Data Analysis and Filtering Metrics*, SPIE Proc. 7345-20, Orlando, FL, 2009.

[4] Laird, R.T. and H.R. Everett, "Reflexive Teleoperated Control," Association For Unmanned Vehicle Systems, 17th Annual Technical Symposium and Exhibition (AUVS '90), Dayton, OH, pp. 280-292, July-August, 1990