Data Fusion for the Apache Longbow: Implementation and Experiences

Steve Jameson[1], Craig Stoneking[2], David G. Cooper[3], Peter Gerken[2], Chris Garrett[2], and Adria Hughes[3]
[1]Manager, [2]Senior Member of the Engineering Staff, [3]Member of the Engineering Staff
Lockheed Martin Advanced Technology Laboratories
Cherry Hill, NJ 08002
{sjameson, cstoneki, dcooper, pgerken, cgarrett, ahughes}@atl.lmco.com

Abstract

Maintaining Situational Awareness and Tactical Decision Making are workload-intensive and time-critical challenges for the crew of the Army's AH-64D Apache Longbow. The Apache crew faces these challenges with extreme mission demands coupled with stressful high-speed, low-level flight. Two technology trends show great promise in addressing these problems: Decision Aiding and Manned/Unmanned teaming with Unmanned Aerial Vehicles (UAV). The US Army Aviation Applied Technology Directorate (AATD) is leading the Airborne Manned/Unmanned System Technology Demonstration (AMUST-D) and Hunter-Standoff Killer Team (HSKT) programs to develop, deploy and demonstrate these technologies in operational evaluations. Data Fusion, the capability to integrate information from multiple sensors and other sources into a consistent Common Relevant Operational Picture (CROP), lies at the heart of both of these capabilities. A reliable CROP is required to support the automated reasoning processes of Decision Aiding, and to automatically combine sensor data from teamed UAV's, reducing the significant human workload that would occur if UAV data were to be manually combined with other fused data representations. For the past 11 years, Lockheed Martin Advanced Technology Laboratories (LM ATL) has been developing a Data Fusion capability specifically designed to support Army Aviation decision aiding systems. In this paper, we describe LM ATL's effort to design, implement, and test a Data Fusion system for the Apache Longbow under the AMUST-D and HSKT programs.

Introduction

The Army Aviation community is promoting the development of technologies and systems that support effective on-the-move command of airborne and ground-based maneuver forces through shared situation awareness and decision aiding technologies. The operational concepts for these technologies and systems are characterized by the extensive use of mobile sensing systems, unmanned platforms, and decision aiding systems in the forward elements of the combat force. This work is exemplified by the Airborne Manned/ Unmanned Systems Technology Demonstration (AMUST-D) and the Hunter Standoff Killer Team (HSKT) Advanced Concept Technology Demonstration (ACTD) programs, led by the U.S. Army Aviation Applied Technology Directorate (AATD) [1].
AMUST-D and HSKT will provide airborne warfighters and mobile commanders with improved situational awareness from the cooperative construction of a shared common operational picture of the battlefield, through the sharing and fusion of information from all exploitable sensors and intelligence data sources that bear on the battlespace. Central to this program is the development of the Warfighter's Associate (WA) decision aiding system for the Apache Longbow, and the Mobile Commander's Associate (MCA) system for the A2C2X Blackhawk.

LM ATL, under contract to AATD, is providing a Multi-Sensor Data Fusion system to provide the necessary Situational Awareness for these decision aiding systems to function. In addition, under AMUST-D, LM ATL has produced a Distributed Data Fusion capability for multi-platform operation to produce a shared CROP representing the best situational awareness available across a network of communicating platforms (Figure 1) [3]. LM ATL's Data Fusion system is integrated into both the MCA and WA decision aiding systems [4]. Implementation, integration, and evaluation of Data Fusion in WA is challenging because of the highly embedded nature of the processing on the Apache, the sensor information available to the Apache avionics system, limitations of the display, and the very tightly focused nature of the Apache Pilot and Co-Pilot/Gunner (CPG) tasks.

This paper provides a brief overview of the general Data Fusion capability LM ATL has produced. We discuss in detail the limitations and challenges inherent in implementing Data Fusion in the Apache Cockpit. We describe the details of the design and implementation needed to overcome those limitations and challenges, presenting results from testing of the WA implementation of Data Fusion, both in engineering tests and pilot

Presented at the American Helicopter Society 61st Annual Forms, Grapevine, TX, June 1-3, 2005. Copyright © 2005 by the American Helicopter Society International, Inc. All rights reserved.

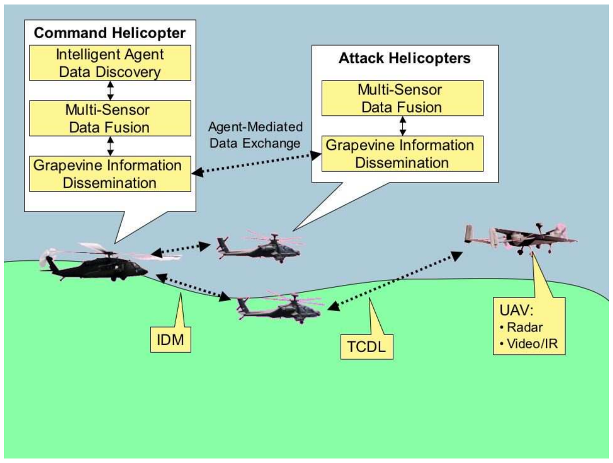
Command Helicopter
Intelligent Agent Data Discovery
Multi-Sensor Data Fusion
Grapevine Information Dissemination
Agent-Mediated Data Exchange
Attack Helicopters
Multi-Sensor Data Fusion
Grapevine Information Dissemination
IDM
TCDL
UAV:
• Radar
• Video/IR

Figure 1. AMUST/HSKT Shared Situation Awareness Architecture.

feedback. Finally, we describe our vision for the future applicability of Data Fusion in aircraft like the Apache, and the developments needed to make this possible.

Multi-Sensor Data Fusion

From 1993 to 1999, LM ATL participated in the Army's Rotorcraft Pilot's Associate (RPA) Advanced Technology Demonstration program, sponsored by AATD. LM ATL developed the multi-sensor Data Fusion system [2] that provides a common fused track picture to the RPA pilot and the RPA decision aiding systems. In the RPA Data Fusion system, data representing as many as 200 battlefield entities, from 14 different types of onboard and offboard sensors, is correlated and fused in real time into a consolidated picture of the battlespace. The RPA system, including LM ATL's Data Fusion system, was successfully flight demonstrated on an AH-64/D in August 1999. The Data Fusion system (Figure 2) consists of four main elements.

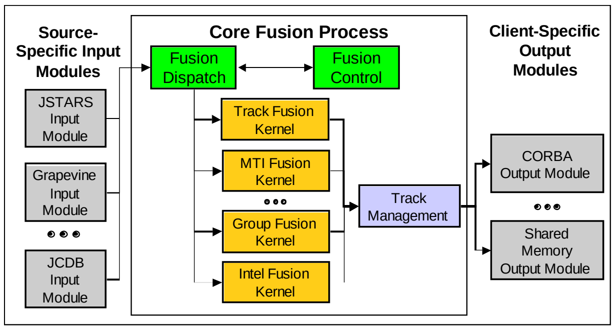

Figure 2. LM ATL's real-time multi-sensor data fusion system.

A top-level control structure, including *Fusion Dispatch* and *Fusion Control* modules, controls the application of fusion algorithms to the input data and ensures that the system meets real-time and resource requirements. The Fusion Dispatch module evaluates incoming data, determines which algorithm set—embodied in a fusion Kernel module—should be applied to fuse the data, and dispatches the appropriate Kernel to process the data set. The Fusion Control module monitors the performance and resource usage of Data Fusion, and applies control measures such as prioritization or down-sampling of input data, in cases where the Data Fusion process begins to exceed resource limitations (such as memory or CPU usage) or fails to meet timing requirements. This separation of the top-level control from the fusion algorithms allows the Data Fusion system to be configured readily to meet different performance and resource requirements in different applications.

A set of *Input and Output* modules manages the real-time interfaces with the sensor systems and the other components of the RPA system. The input modules read input from the sensor systems at a sensor specific rate, using a sensor-specific input protocol, and pre-process the input using tailored sensor-specific routines into a common intermediate input format and information content. The output modules take the fused trackfile and output it to a client, in a client-specified format, using a client-specific protocol. The modular, object-oriented Data Fusion software architecture, including the use of a common intermediate data input format, permits a single core body of Data Fusion code to be easily adapted to multiple input and output formats and requirements, facilitating portability. On AMUST-D, two different versions of the Data Fusion system will be deployed on the Longbow Apache and A2C2S Blackhawk helicopters. Both versions will contain the same common fusion core, coupled with platform- and sensor-specific input and output modules.

A *Track Management* module stores all track data and maintains the relationships among a set of track databases, one for each sensor providing input, and a Central Trackfile that stores the fused picture. Additional databases also provide access to a variety of data about platform, sensor, and weapon characteristics used by Data Fusion.

A set of *Fusion Kernel* modules performs the heart of the correlation and fusion processing. As with the input and output modules, the modular nature of the fusion process makes possible the encapsulation of algorithms tailored to a specific input data type into a kernel module. As input data sets are received, the appropriate Kernel is applied to that data set, with the resulting output passed to the Track Management module to update the fused trackfile.

The general functional flow followed by each kernel follows a similar set of steps. Figure 3 illustrates the steps followed for MTI (Moving Target Indicator) data. The *Prediction* function performs time-alignment across the sets of data being fused to ensure that valid comparisons can be made. The *Clustering* algorithms break down the battlespace into geographically distinct clusters to limit the computational complexity of the following algorithms. *Cost Functions* operate on each cluster to compute a matrix of composite similarity values between each input sensor data item and the candidate fused tracks. The *Assignment* function uses the optimal JVC (Jonker-Volgenant-Castanon) algorithm to compute matches between sensor data and fused tracks. Once the matches are identified, *Fusion* algorithms are applied to update the state of the fused trackfile based on the associated sensor data using an Interacting Multiple Model (IMM) Kalman Filter.

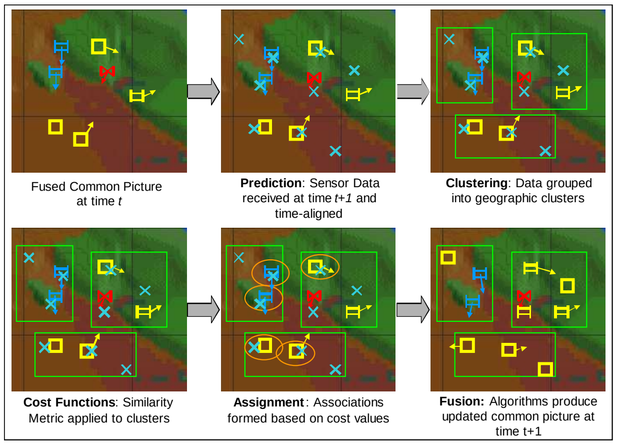
Fused Common Picture at time *t*
Prediction: Sensor Data received at time *t+1* and time-aligned
Clustering: Data grouped into geographic clusters
Cost Functions: Similarity Metric applied to clusters
Assignment: Associations formed based on cost values
Fusion: Algorithms produce updated common picture at time t+1

Figure 3. Data fusion kernel functional flow.

One of the key algorithmic advances of the RPA Data Fusion system was its ability to effectively combine the processing of kinematic (position and velocity) information with the processing of Class (vehicle type), ID (specific vehicle information), and IFF (friend/ hostile) information. This processing takes place during the comparison of sensor data with fused tracks, by the Cost Functions, and during the updating of the fused trackfile with associated sensor data by the Fusion algorithms. The Class Cost Function and Fusion algorithms compare and combine Class and ID information expressed in a class hierarchy (Figure 4). Each sensor report or fused track has a class representation that specifies the confidence of each node in the hierarchy. A set of Modified Bayesian Evidence Combination algorithms developed by LM ATL is used to compare, combine, and summarize this information. LM ATL's work in this area [2] represented a major advance in Data Fusion technology.

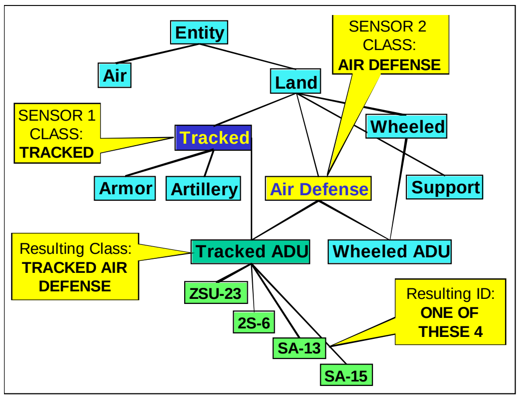

Figure 4. Example of class fusion in RPA data fusion.

Apache Longbow Implementation

Overview of Warfighter's Associate and WA Data Fusion

The Warfighter's Associate (WA) system, developed by Boeing Phantom Works in Mesa, AZ, is the decision aiding system developed under AMUST-D/HSKT to support the pilot of the Apache Longbow. WA provides the following capabilities to the Apache pilot:

- A ***Route Planner*** that allows the pilot to plan a detailed route given a set of high level waypoints, terrain, and time and fuel constraints.
- A ***Route Assessor*** that continually monitors the route for exposure to threat, speed constraints, low fuel, and potential failure to meet time constraints.
- An ***Attack by Fire (ABF) Planner*** that selects the most appropriate location from which to attack a threat based on a variety of tactical heuristics.
- ***Display Aids***, including *Ownship* and *Threat intervisibility*, and *Search Area Covered*.
- ***UAV Management*** functionality, including a TCDL link to the UAV and a variety of UAV management aids.

Integral to many of these functions is the need to have up-to-date, consistent, information on threat and friendly entities in the battlespace, which is the role of Data Fusion.

The Role of Data Fusion in Warfighter's Associate

In order for WA to perform intervisibility calculations with respect to a threat, and to plan a route to best avoid that threat, it is desirable to have an accurate and stable estimate of the threat's location. Data Fusion provides support for these automated decision aiding functions of WA, as well as supporting WA in providing the pilot with better situational awareness, with an operational picture that is more stable, more accurate, more complete, and less cluttered. These benefits are realized through Data Fusion's scan-to-scan tracking, multi-sensor fusion, and false target removal capabilities.

Scan-to-Scan Tracking contributes to the stability, accuracy and completeness of the tactical picture. Without Data Fusion, each time the FCR performs a scan, all of the targets detected in the previous scan are removed from the display and replaced by the results of the new scan. Since the FCR scans only a limited sector in any one scan, targets from a previous scan can be lost from the display when the FCR's scan sector is moved. Even for a target that is detected by both the previous and the new scan, it can be difficult for the operator to determine which target from the new scan corresponds to a target in the previous scan. Data Fusion alleviates this problem with scan-to-scan tracking. The first detection of a target by the FCR establishes a target track, with an identifying track number. With subsequent scans, Data Fusion is able to associate a new detection of that target with the established track, providing continuity of that track across scans. Targets not reinforced by the subsequent scan do not immediately disappear with the new scan. Rather, Data Fusion uses a time-based scheme for eventually eliminating tracks that are not reinforced. Each time a track is reinforced by a subsequent scan, the position location from the new detection can be used to refine the accuracy of the existing track.

Multi-Sensor Fusion increases accuracy and reduces clutter in the tactical picture. Feedback from pilots, during flight test simulations, emphasized that an important situation awareness concern that the pilots have is the unambiguous determination of how many targets are present on the battlefield. Without Data Fusion: if the FCR scanned a target whose position had already been previously stored from another source, such as TADS, then the pilot would see two symbols for the target on the Tactical Situation Display—one for each reporting source. Data Fusion alleviates this problem by recognizing that the reports from the different sources refer to the same object on the battlefield, and fuses them into a single target track, so that the pilot sees a single display symbol per real-world target. Fusing reports from multiple sensors reduces clutter and improves the completeness and accuracy of information on a track, in the same way that scan-to-scan tracking does.

False Target Removal builds upon scan-to-scan tracking to further improve accuracy and reduce clutter in the tactical picture. Data Fusion assists the recognition and elimination of ephemeral and persistent false targets. An ephemeral false target shows up in one FCR scan, but is not reinforced by a subsequent FCR scan of the same area. Tracks that are not reinforced by subsequent scans decay, or "age-out", and are removed, or "dropped", after a certain period of time. A persistent false target represents an actual object on the battlefield, so that it shows up in subsequent FCR scans, but is an object of little or no interest, or a target that has already been destroyed. In this case, once the operator determines that the object is not of interest, the WA system allows the operator to "disallow" the corresponding target track. When subsequent sensor reports on the target are input to Data Fusion, it can associate the new report with a disallowed track, and refuse to process it. This way, a false target that is persistently detected by the FCR can be excluded from the display.

Design Challenges

Design, implementation, integration, and evaluation of Data Fusion in WA are challenging because of the sensor data available on the Apache Longbow, the highly embedded nature of the processing on the Apache, and the need to conform to existing Pilot and Co-Pilot/Gunner (CPG) task expectations. Each of these challenges imposed design constraints that resulted in specific design decisions during the design and development of the WA Data Fusion.

Sensor Data

WA Data Fusion was required to accept and fuse data from a wide variety of sources, for which the original data fusion design was well suited, having been designed to fuse up to 21 different data sources. However, the original design relied on assumptions about the periodic nature of the input sources for input processing and some of the correlation and fusion logic. The fusion algorithms rely on the availability of certain attributes for all sensor reports received.. These include position of the detected entity, time of the detection, the class/ID of the entity, and the uncertainty (error ellipse) associated with the detection. The sensor data available on the Apache posed design challenges because most sources did not report periodically and none of the sources provided all of the information required by the original Data Fusion algorithms (Table 1).

Almost all of these sensors report neither the time of the measurement nor the expected error/uncertainty in the measurement. This information is particularly important for Data Fusion to make a reasonable estimate of the positional uncertainty of any track that it forms from sensor data input. This forced us to accept the time of reporting to Data Fusion as the time of measurement, and to assign to each sensor a default error estimate, based on discussions with pilots and engineers associated with the Apache program, that could be refined, through experience in simulated and actual operations, to achieve reasonable results.

Table 1. The data characteristics of the sensors that are fused on the Apache.

Sensor/Source	Reports Detection Time	Reports Position	Reports Position Uncertainty	Reports Velocity	Reports Bearing	Reports Class	Periodicity
Enhanced GPS/ Inertial Navigation Unit (EGI)	Yes	Yes	No	Yes	No	No	1 Hz
Fly-Over Store	No	Yes	No	No	No	Yes	Crew Initiated
Crew Store	No	Yes	No	No	No	Yes	Crew Initiated
Target Acquisition /Designation System, (TADS)	No	Yes	No	No	No	Yes	Crew Initiated
Fire Control Radar (FCR)	No	Yes	No	Yes	No	Yes	Crew Initiated Scan or continuous scan
Teammate Fire Control Radar (FCR) via IDM	No	Yes	No	Yes	No	Yes	Teammate aircraft crew initiates scan and transmission
Radio Frequency Interferometer	No	No	No	No	Yes	Yes	Reports at 10Hz, but measures at < 1 Hz
Radio Frequency Hand Over (RFHO)	Yes	Yes	Yes	Yes	No	Yes	Wingman Initiated
Electronic Data Transfer Cartridge (EDTU)	No	Yes	No	No	No	Yes	One Time

The reason for the importance of periodicity in input sensor data is twofold. First, Data Fusion makes the assumption that a given sensor will never report twice on the same target within the same reporting interval, the length of which is a property of the sensor. For example, a traditional rotating radar system has a reporting interval equal to the period of rotation of the antenna, and during that period it can reliably be assumed that the radar will not report twice on a target. This allows Data Fusion to assume that two closely spaced reports from the same sensor in that time interval must be separate targets. For a sensor that does not report periodically, we must identify a time interval within which it can reliably be assumed that closely spaced reports are from separate targets. Our design then accumulates input from that sensor over the identified time interval and processes the accumulated data as though that data represented a periodic reporting of data.

The other reason for the importance of periodicity is to determine when a track can be dropped. In previous applications, Data Fusion could expect that if a sensor's regular reporting interval elapsed without a target's track being updated by that sensor, then that track could be considered lost. Once the track is lost, Data Fusion continues to maintain the track, but with increasing uncertainty in the track's location. When the uncertainty in the track location reaches a specified threshold, Data Fusion "drops," (stops reporting on) the track entirely. In the current WA application on the Apache, almost none of the sensors have a regular reporting interval. Many sensors are expected to report only once on a target, and others (most notably the FCR) report only on activation by the pilot/operator. Without a reliable reporting interval for its contributing sensors, the approach to the problem of determining when a track should be declared lost and dropped needed to be modified for WA Data Fusion. Through consultation with Boeing engineers and pilots, we worked out an approach that is consistent with the way the pilots employ the sensors in an operational environment. For those sensors that are expected to report only once on a target, in most cases the target is assumed by the pilots to be stationary and will never be dropped. Data Fusion incorporates this assumption. In the case of the FCR, we implemented a track age-out rate that caused the track to go stale in about 30 seconds. This was selected based on pilot feedback as the period beyond which they did not feel that the data was sufficiently reliable to be shown on the cockpit display.

Radio Frequency Interferometer (RFI)

One of the most challenging aspects of the Apache sensor data is the nature of the data from the RFI. This provides directional indication of radar systems employed by threats such as Surface to Air Missile (SAM) batteries, and a high confidence identification of the threat type. Currently the only information available to a pilot is the directional indicator on the display, along with the threat type. One of the goals of the Data Fusion development was to implement a capability to convert this directional information over time into an estimate of the target's position, and – most importantly for the pilot – the target's range, so the pilot can immediately determine the magnitude of the threat posed by the system.

As part of the development of the WA Data Fusion, we designed and implemented a capability for RFI *Multilateration* (Figure 5).

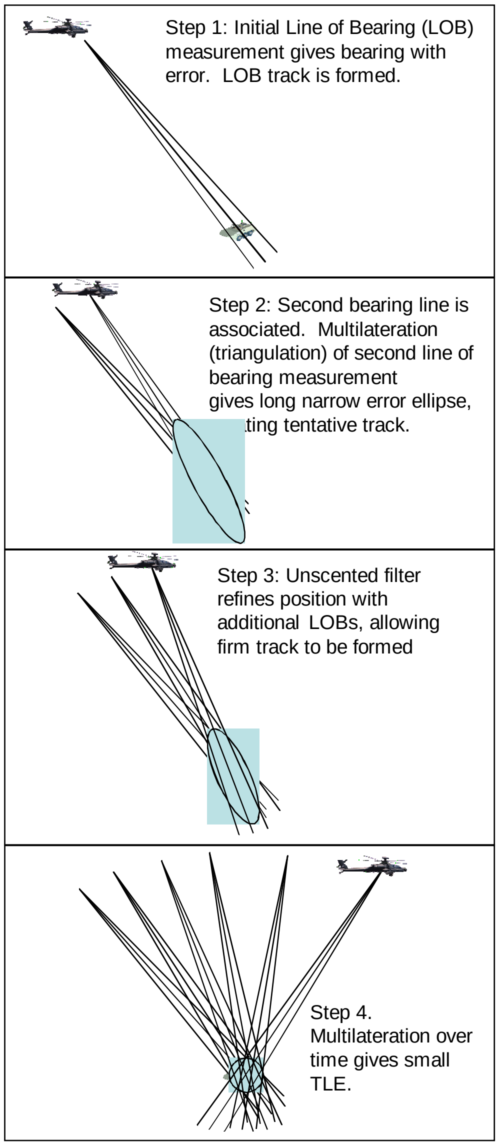
Step 1: Initial Line of Bearing (LOB) measurement gives bearing with error. LOB track is formed.
Step 2: Second bearing line is associated. Multilateration (triangulation) of second line of bearing measurement gives long narrow error ellipse, ...ting tentative track.
Step 3: Unscented filter refines position with additional LOBs, allowing firm track to be formed
Step 4. Multilateration over time gives small TLE.

Figure 5. Generation of positional tracks from bearing measurements enables WA Data Fusion to combine RFI and other bearing data with FCR and other positional sensors.

The RFI Multilateration process consists of four basic steps:

1. ***LOB Track Formation*** – A Line of Bearing (LOB) measurement from RFI or any similar sensor is used to create a track that contains only bearing information and ID.
2. ***Bearing Association*** – A second LOB measurement is associated with the LOB track based on consistency of ID information and geometric feasibility. The geometric feasibility test verifies that if the two lines are crossed, the resulting position represents a feasible target location, i.e., one within the maximum range of the emitters represented by the ID information. The result of the association of the two LOBs gives a triangulated position with a positional uncertainty ellipse that is typically fairly large. This position and uncertainty are used to create a *tentative* track that is not yet reported out of Data Fusion.
3. ***Track Promotion*** – When additional LOB measurements are detected and associated with the tentative track, an *Unscented Filter* is used to refine the position and uncertainty. When the uncertainty reaches a threshold, the track is promoted to *Firm* and is reported out of Data Fusion to WA or other applications using the output of Data Fusion.
4. ***Continued Refinement –*** As additional LOB data, or other reports, such as radar reports, are received on the target, the position is refined further to produce the best possible Target Location Error (TLE) from the available data.

A significant amount of effort was devoted to development and testing to demonstrate the utility and performance of the RFI multilateration. During pilot evaluation we were able to demonstrate that this capability works well for a single target when the RFI sensor is providing *fine*, or high resolution, LOB data. Based on this testing the pilots specifically recognized the particular value of the ability to quickly determine the position of a hostile threat with the passive RFI sensor. Further maturation is in progress to enable this capability to work more effectively with less accurate data and for multiple closely spaced targets so that this capability can be integrated and evaluated in operational testing.

Processing Environment

Data Fusion was originally developed to run in a multiprocessing Unix environment on Silicon Graphics processors, in which it could run continuously as its own process, and rely on the operating system to manage fair sharing of the CPU, and memory space protection, with respect to other processes. The VxWorks operating system used on the Apache's General Purpose Processors provides a very different environment. Under VxWorks, the operating system is the only real process. All applications, like Data Fusion, run as tasks within that process, and share a common data address space. For scheduling purposes, tasks are divided up into real-time tasks and background tasks. Real-time tasks are scheduled to execute in a round-robin fashion, and are expected to guarantee to finish within a small number of milliseconds. As a background task, called periodically by the WA process, Data Fusion is expected to execute for a few milliseconds at a time before returning and relinquishing the CPU, and could be preempted at any time, for any length of time, by one or more real-time tasks.

Data Fusion shares one of the Apache's 375 MHz G4 Power PC – based General Purpose Processors with a number of other tasks. As a subtask of WA, Data Fusion gets called to execute, on average, about 10 times per second. Each time Data Fusion executes, it is expected to execute for no more about 8 ms before returning. This gives Data Fusion a nominal 8% of the total CPU time to perform its processing. However, since Data Fusion can be preempted at any time by a real-time task, the actual portion of the CPU available to Data Fusion is some unknown amount less than 8%. To adapt to this execution paradigm, we needed to provide a means by which Data Fusion could execute for a few milliseconds, save an intermediate state of the processing before returning, then pick up where it left off upon next being invoked, and internally manage memory allocation. The details of this solution are described in a later section.

Interface/Hardware

Integrating Data Fusion onto the Apache required addressing several challenges related to hardware and software platform. The Warfighter's Associate was able to leverage much of the work done under MCA to enable Data Fusion to run on Power PC-based hardware. However, WA had to meet the requirement that Fusion must operate under the real-time operating system VxWorks. Finally, Data Fusion – being written primarily in C++ – was to interface with software written in Ada.

The decision was made early on that it would be more productive and less costly to do the development and testing in a simulated VxWorks environment as multiple instances of the simulator can be run simultaneously. We chose the Tornado™ development environment as it provided both a VxWorks simulator as well as a cross-compiler that can generate PPC-compatible code. We hosted Tornado^{TM} within a development environment that enabled us to compile, test and release software packages in a manner commensurate with our MCA development.

Interfacing the "host" Common Operating Environment (COE) software to Data Fusion was done via two methods – direct procedure calls and data written to common "shared memory" segments. Figure 6 illustrates these interfaces. Most data transfer occurs through the shared memory interface segments that include an Input segment for sensor data, an Output segment for the fused trackfile data from Data Fusion, a Status segment to share information about Data Fusion status, and a Control segment to issue control flags, such as a request to delete a specific track.

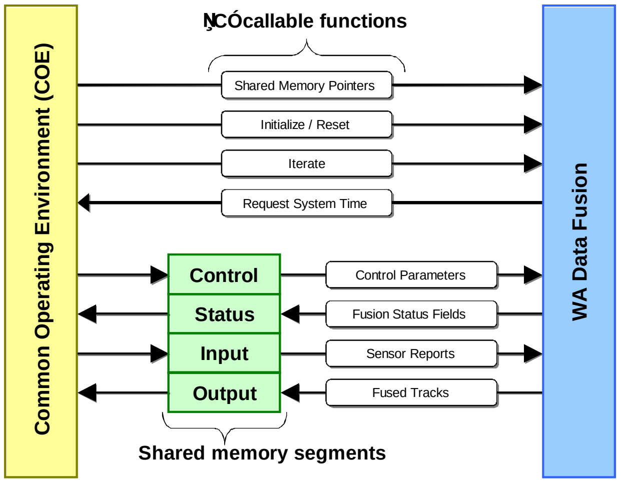

Figure 6. The structure for the Data Fusion-to-WA interface.

Data Fusion Software Architecture

To integrate LM ATL's Data Fusion system into WA, two important implementation challenges were addressed. The first required Data Fusion to split its processing over a series of short time slices rather than running as a continuous process. The second required Data Fusion to avoid the use of dynamically allocated memory.

LM ATL's Data Fusion system was originally developed to execute as a stand-alone continuously executing process, with a well defined set of functions executed in a specified order on the input data to produce the fused output. Integration onto the Apache required that Data Fusion run as a subtask within the overall WA execution process, with a strict limitation of eight milliseconds on the time that it could execute before returning control to the WA process. This meant that it was in general not possible to complete processing on any given set of sensor data during an execution cycle. The solution to this problem was implementation of a *state machine* control structure. The state machine keeps track of the current state of Data Fusion execution (Figure 7), and permits it to resume execution at the appropriate point, with all required contextual information, once Data Fusion execution is resumed.

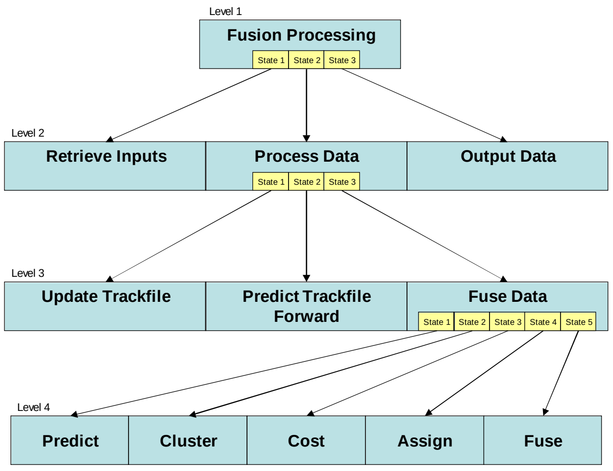

Figure 7. Data Fusion's State Machine control structure allows it to maintain its state of execution over a number of short-duration execution cycles.

The WA processing environment does not support the routine allocation and deallocation of memory that is standard in an object oriented program, such as Data Fusion. To minimize changes to previously developed Data Fusion software and allow commonality between the WA and MCA Data Fusion implementations, we implemented a custom memory manager utility that allocates a single large block of memory at program initialization, and then internally manages the use of this memory to enable the object-oriented software to behave as it would normally. In addition to managing memory, the utility also provides statistical profiling for memory management to ensure that the software has no memory leaks and to support analyzing analyzing memory usage.

Integration and Testing Process

LM ATL worked in collaboration with engineers from Boeing Phantom Works to adapt and integrate Data Fusion to run as part of WA within the Apache mission systems hardware and software environment.

Throughout the development and integration process, effort was made to strike a balance between three overarching goals.

As our primary goal, we worked to ensure that Data Fusion provides the *functionality needed to support WA requirements*. As an example, when a sensor reports the class of a target to Data Fusion, the class ID is mapped from the sensor-specific class reporting taxonomy to a more comprehensive class taxonomy that Data Fusion uses for proper fusion of class IDs from multiple sensors. However, once a fused class ID is determined for a track, special care needed to be taken to map the fused class ID back into an appropriate sensor-specific class taxonomy, to support WA processing logic and display symbology.

While providing new capability, we also worked to meet the goal of *minimizing changes required to the pilots' expectations* of system behavior. One example of this was the adjustment of the age-out behavior of unreinforced tracks to approximate that already experienced by current pilots. In another instance, it turned out that when a track was reported to be unknown, it was meaningful for the pilot to know which particular sensor data source reported the class. To maintain this richness of information in its output, Data Fusion was specifically modified so that when a track was established based on data from a single sensor, Data Fusion reported the track type using the same symbology as the reporting sensor, rather than choosing a single default symbology for representing all tracks of unknown type.

As a third goal, we worked throughout this process to ensure that the *fundamental correctness and accuracy* of the underlying core Data Fusion processing was retained as updates to the software were made in support of the other goals.

The overall development, integration and testing process was divided into four major phases.

Initial Port of Code to VxWorks. In the WA execution environment Data Fusion is called as a subtask of the larger WA task. WA, itself, runs as a task within the Apache's Common Operating Environment (COE), which provides a framework of task management policies and services on top of the underlying VxWorks operating system. To enable development in our laboratory environment, we used the Tornado^{TM} VxWorks emulation, from Wind River Systems Inc. Running on our normal Solaris development platforms, the VxWorks emulation allowed us to initially port Data Fusion so that it could be compiled and run in either VxWorks or Solaris. This allowed us to identify and resolve issues arising from differences between the two operating systems. Most of these differences had to do with differences in the interface to the operating system as represented in the set of system utility functions, (for example, the `time` function), available as C++ function calls. With these issues resolved, most of our subsequent development and testing could be done under Solaris, with a reasonable level of confidence that it would run identically under VxWorks.

Integration with a WA Sub: While Boeing was developing the overall WA software functionality, they were able to provide a stub version that allowed us to test Data Fusion's interfaces with WA, and the execution of Data Fusion within the WA and COE calling context. Boeing was able to compile into the WA stub program a number of test scenarios that provided Data Fusion with input data, and consumed the Data Fusion output. These scenarios provided a thorough and detailed exercise of the Data Fusion functionality. The ability to create a track of each class, from each available sensor source was exercised, as well as the ability to fuse reports from multiple sensors, track moving targets, and manage the ageing and loss of tracks. Logs were collected of the data passing over both the input and output interfaces between Data Fusion and WA, and the contents of the logs were analyzed to verify the correctness of the interface function.

Integration with WA in Boeing Labs. Once the WA software was ready, we began integration testing of Data Fusion with the full-fledged WA software within the actual target hardware and software environment in the Boeing integration labs. During this period Data Fusion was tested on a small number of specific scenarios provided through Boeing's simulation capability. This most lengthy phase of the integration process was characterized by an iterative cycle of development and testing. Running in the Boeing lab, Data Fusion would be observed while simulating a scenario, slightly modified to target a specific aspect of the Data Fusion functionality, and bugs or desired modifications would be noted. Data Fusion modifications would then be developed and tested at LM ATL, running under the Solaris operating system. The compiled image of the modified Data Fusion could then be sent by email to Boeing for targeted retesting and verification in the simulation lab. While many of these test-and-develop cycles could be carried out via email interactions between LM ATL and Boeing engineers, there were also periodic visits by LM ATL engineers to the Boeing labs, for face-to-face interaction with the Boeing engineers, and firsthand observation of Data Fusion operation. With this iterative cycle of discovery and resolution, we were able to resolve the majority of the software interaction and operator usability issues.

Pilot-Driven Operational Focus: Finally, Boeing's Pilot-Vehicle Interface (PVI) engineers, as well as pilots from both Boeing and the Army, were brought into the iterative test-and-fix cycle. Running Boeing's Engineering Development Simulation (EDS) lab, LM ATL and Boeing engineers could observe Data Fusion function, as a part of WA, in operational scenarios with a pilot (or engineer) flying the Apache through the simulated scenario, and operating the sensors in a full mock-up of the Apache cockpit. At times, the pilot was put into a scenario and given the ability to "play" in a relatively free-form manner to uncover operational issues that were unanticipated by the engineers. However, as testing proceeded, a fairly comprehensive checklist of operational tasks was developed to test and verify the proper operational function of Data Fusion as a part of the WA system. For example, a part of this routine checklist would call for the pilot to scan for a target with the FCR and, having acquired the target with the FCR, the pilot would use the Target Acquisition and Designation System (TADS) to find and designate the target to store its location. While the pilot verified and evaluated the expected behavior of the system throughout these tasks, LM ATL and Boeing engineers would verify that the FCR track appeared in the correct geographical location, with the appropriate class, that the TADS store properly fused with the FCR track, and that the track symbol on the pilot's display properly reflected the update to the track. The pilots and PVI engineers provided valuable feedback that led to a number of operational improvements to Data Fusion and WA behaviors, such as the timing and display of tracks going stale, the expected error for certain sources of track data, and the interaction of moving FCR reports with stationary pre-planned targets.

Results

Through this testing process, we were able to validate that Data Fusion addressed all required functionality and performance to support the WA functionality. More importantly, we were able to get feedback from Apache pilots involved in the tests that Data Fusion was a useful capability that should be brought forward into operational use in the Apache and other helicopters. There were several features of Data Fusion that pilots specifically identified as valuable:

- The retention of FCR-generated tracks across multiple scans
- The ability of Data Fusion to take reports on a single target from multiple sources and combine them into a single track, with a single corresponding symbol on the display. That is, they appreciated the use of Data Fusion to reduce "data *con*fusion."
- The RFI Multilateration capability was described as "phenomenal" and something that would be very valuable in tactical situations once it is matured to an operational capability.

Having undergone thorough testing and evaluation in the Boeing simulation labs, we look forward to ground and flight testing in the actual aircraft in the second quarter of 2005.

Other AMUST-D/HSKT work

LM ATL has developed a number of other components to support the AMUST-D and HSKT programs. These include a Data Fusion component to support the Mobile Commander's Associate (MCA) decision aiding system on the A2C2X Blackhawk aircraft. We have provided an Agent-Based Data Discovery (ABDD) component [5] to retrieve and monitor information from the Army Battle Command System (ABCS) C2 applications installed on the A2C2X aircraft. In addition, we developed support for and integrated the PC Improved Data Modem (PCIDM) product that uses the SINCGARS radios on the A2C2X to transfer data between the A2C2X and the Apache. The combination of these components with the WA Data Fusion system enables a *Shared Situational Awareness* capability that spans the multiple aircraft participating in the HSKT experiments. In the remainder of this section, we briefly describe each of these components.

MCA Data Fusion

MCA Data Fusion provides the companion capability to WA Fusion as part of the MCA decision-aiding system on the A2C2X. The underlying functionality and software are the same, but because the data available on the A2C2X is very different from that on the Apache, the behavior is somewhat different. The A2C2X has no onboard sensors, so MCA Data Fusion is responsible for combining sensor data from JSTARS and other sources over Link-16, Friendly position reports received through the Force XXI Battle Command Brigade and Below (FBCB2) system, spot reports and those collected into the Joint Common Data Base (JCDB), and Operator-entered tracks based on video from the UAV. A number of specific behavior logic requirements were incorporated to accommodate the needs of processing data from Link-16 and the UAV. For example, the commander using MCA may request that a strike aircraft, such as an F/A-18, engage a target. While the engagement is in progress, Data Fusion must ensure that it does not drop the target from its trackfile even if no data has been received on the target for several minutes. Another difference is that MCA Data Fusion has a requirement to deal with situations in which there may be as many as 2000 friendly and threat entities, and be capable of adapting its output rate and processing to ensure that it deals as effectively as possible with the very large number of entities.

Another difference between MCA and WA Data Fusion is in the processing environment. Data Fusion for the MCA occurs on a PowerPC Single Board Computer (SBC) running a Real-Time Linux implementation from TimeSys Corporation. All data to and from the Data Fusion system are transmitted over CORBA Event Channels using multicast protocols. Despite these differences, MCA Data Fusion and WA Data Fusion share a common software base, with much of the differences between the two versions accomplished via run-time configuration using an XML configuration file.

Agent-Based Data Discovery

ABDD is a component within the MCA that is responsible for controlling the flow of data between the data sources within the ABCS onboard the A2C2X and the rest of MCA using LM ATL's Extensible Mobile Agent Architecture (EMAA). EMAA agents are tasked to retrieve data from the JCDB and the Live Feed Server (LFS) within ABCS. The JCDB contains information on blue/red unit data and control measures, while the Live Feed Server contains real-time data on friendly force vehicles. The constant blue track feed from the LFS allows the agents to monitor their positions with respect to a set of control measures available from the JCDB. These control measures include: Named Area of Interest (NAI), Engagement Areas (EA), Phase Lines (PL) and Helicopter Routes. By monitoring these control measures, the agents are able to warn the commander when blue forces may be vulnerable if they cross a phase line too soon, cross into a potential engagement area or are near a blue force helicopter route. The agents can also trigger critical decisions when a red force is found in a named area of interest and allow the commander to send an attack to a specified engagement area.

PCIDM Message Service

At the inception of the AMUST-D program, it became clear that a mechanism was needed to enable transfer of Situational Awareness and other data between the MCA and WA aircraft, and LM ATL was tasked to provide this mechanism. Enabling data transfer required both a physical medium, i.e., a set of radios, and a data transfer medium, i.e., a set of protocols and terminal or modem hardware to encode the data for transfer over the radio links. Because the Apache had a previously existing capability to transfer data using the Improved Data Modem (IDM) over its SINCGARS radio sets, we determined that the best approach was to enable the MCA A2C2X aircraft to use this same mechanism. We identified a commercially available product, the PCIDM from Innovative Concepts Inc., as the best route to accomplish this. The PCIDM is a PCMCIA form factor modem that provides the same messaging capability as the IDM on the Apache. Using this product, we developed a PCIDM Message Server software component that enables the MCA software on the

A2C2X to exchange data with the Apache using the Boeing-developed AFAPD message set. Under AMUST-D, this capability is used by MCA to receive FCR target and Apache position data from the Apache, and to send other sensor data, target handover requests, and routes to the Apache from the MCA system.

Conclusions

Through the work designing, developing, integrating and testing Data Fusion for the Warfighter's Associate under AMUST-D and HSKT, we have shown that useful Data Fusion functionality can be performed in the processing environment of the Apache Longbow and with the sensor data sets available on the Apache. Our testing has shown that this Data Fusion functionality is useful not only to the other decision aiding functions that make up the WA system, but to the pilot as well. We expect to continue to mature this capability to support an eventual operational implementation.

Acknowledgments

The work described in this paper was performed under contract DAAH10-01-2-0008 from the US Army Aviation Applied Technology Directorate (AATD). We would like to acknowledge the support of Keith Arthur, Balinda Moreland, Mark Ennis, Ray Higgins, and Tim Condon of AATD in this work. Many other people from Lockheed Martin and Boeing supported this effort. Jim Farrell of LM ATL developed the RFI Multilateration and IMM Kalman Filter algorithms that were used in WA Data Fusion. Alex Artigas, Karl Bielefeldt, Bob Huber, Bob Faerber, Jeff George, and others at Boeing Phantom Works in Mesa, AZ, developed the WA system and worked closely with us in development, integration, and testing of the WA Data Fusion system. Chris Bodenhorn, John Kiernan, Peter Stiles, Janice Hess, and many others at Lockheed Martin Systems Integration – Owego in Owego, NY, developed the MCA system and worked closely with us in the development and integration of the MCA Data Fusion, ABDD, and PCIDM Message Service components. Most of all, we would like to acknowledge the longstanding and tireless leadership, vision, and inspiration of Angela Pawlowski of LM ATL in making this work possible.

References

[1] LTC John C. Wright and Kristopher Kuck, "U.S. Army Modernization: Eyes on the Target," Rotor and Wing, April 2001.

[2] Don Malkoff and Angela Pawlowski, "RPA Data Fusion," 9th National Symposium on Sensor Fusion, Vol.1, Infrared Information Analysis Center, September 1996.

[3] Steve M. Jameson and Craig Stoneking, "Army Aviation Situational Awareness Through Intelligent Agent-Based Discovery, Propagation, and Fusion of Information," American Helicopter Society Forum 58th, Avionics and Systems Session, Montreal, Canada, June 11-13, 2002.

[4] William J. Farrell, Steve Jameson, and Craig Stoneking, "Shared Situation Awareness for Army Applications," Proceedings of 2003 National Symposium on Sensor and Data Fusion, San Diego, CA, June 2003.

[5] Peter Gerken, Steve Jameson, Brian Sidharta, and Joyce Barton, "Improving Army Aviation Situational Awareness with Agent-Based Data Discovery," Presented at the meeting of the American Helicopter Society Forum 59, Phoenix, AZ, May, 2003.