

A MULTICAST CAPABLE TUNNEL APPLICATION TO EFFICIENTLY DISTRIBUTE CNS DATA TO DISPERSED USERS OVER NON-MULTICAST CAPABLE SATELLITE LINKS AND NETWORKS

Johnathan Pesce, ERAU ATM Research Laboratory, Daytona Beach, FL

Abstract

There is a need for different broadcast methods and protocols in the expanding aviation information marketplace. At the same time, the availability of these multiple methods and protocols presents an obvious technical difficulty in utilizing communications pipelines, which may not support a protocol to efficiently distribute that information. This paper describes a real world problem with a specific but generalizable solution to the distribution difficulty.

As part of the Global Communications, Navigation, Surveillance System (GCNSS) Program, jointly funded by the FAA and Boeing, Embry-Riddle Aeronautical University's Air Traffic Management Research Lab (EARL) provided display software to show live and simulated radar surveillance data as well as Automated Dependent Surveillance (ADS) data from a Connexion by Boeing (CBB) aircraft operating in and around the Gulf of Mexico outside of both radar and VHF coverage. The displays were part of EARL's Real Time Distributed Simulator (RTDS). Voice communication between controllers and the aircraft was alternatively provided by EARL developed Voice over IP (VoIP) software and a CBB Session Initiation Protocol (SIP) application. The displays also included CPDLC messaging to exchange non-tactical information with the aircraft.

The EARL RTDS is designed to use UDP multicast to distribute data to multiple destinations without replicating the data for each recipient; similar to the way air-to-ground radio broadcasts work. The network, provided by Boeing, was made up of a series of T1 landlines and a satellite link to connect with the CBB aircraft.

One of the main problems encountered constructing this network was that while the hardware connecting the sites supported multicast traffic, the satellite link to the aircraft did not. To solve this problem the author developed a tunnel application, ERoute, that could forward the multicast

traffic between the ground and the aircraft through TCP or UDP unicast based virtual tunnels that encapsulate the multicast packets. Additionally, the ground and satellite links were Internet Protocol Security (IPSec) encrypted for added security.

The ERoute tunnel application successfully carried all the VoIP communications, surveillance data and CPDLC traffic over the satellite links.

This paper describes the applications supported in Segment B of the GCNSS Program and their different communication requirements; the network structure and the problems that were faced in meeting the application requirements; and the development, design and successful implementation of the fully configurable ERoute tunneling application.

Background

The Global Communications, Navigation, Surveillance System (GCNSS) Program, jointly funded by the FAA and Boeing ATM, was created to demonstrate the feasibility of next generation CNS/ATM concepts and a satellite based highly integrated Common Information Network (CIN) architecture. This paper discusses the efforts of Segment B of the GCNSS Program, which included the following requirements [1]:

- Simulated ATC workstations capable of displaying surveillance data fusion of radar and ADS for seamless transition between the non-radar and radar environments.
- A demonstration aircraft for three flights demonstrating a secure, highly integrated CIN and broadband, secure communication and data architectures
- A SATCOM network linking aircraft to ground that can transmit ADS data.
- An ADS system onboard a demonstration aircraft, linked by means of a SATCOM network that can provide accurate aircraft position information to a contractor, ground-based situational demonstration display

- A representative air traffic control display that provides continuous accurate multiple aircraft tracking during transition from oceanic to ground-based environments
- A hybrid secure satellite-to-ground architecture using Ku-band and L-band SATCOM and ground-based communications networks that are capable of two-way communications, including digital voice with measured latency applied to pilot-to-controller communications
- A subset of the CPDLC message sets transmitted by means of the secure SATCOM network to the ground-based demonstration display

To fulfill these requirements, Embry-Riddle Aeronautical University's Air Traffic Management Research Lab (EARL) provided display software to show live radar surveillance data and Automated Dependent Surveillance (ADS) data from a Connexion by Boeing (CBB) aircraft operating in and around the Gulf of Mexico including outside of both land based radar and VHF radio coverage. The displays, placed in Houston Center for the controllers, the Boeing ATM laboratory in McLean, VA for observers and onboard the CBB aircraft for the pilots, were part of EARL's Real Time Distributed Simulation (RTDS). CPDLC messaging for non-tactical information exchange was integrated with the display software. Voice communication between controllers, the CBB aircraft and pseudo pilots controlling EARL simulated aircraft were provided alternatively by EARL developed Voice over IP (VoIP) software and a CBB provided SIP application. The Live surveillance and ADS data was received via MIT Lincoln Labs' Surveillance Data Network (SDN) and overlaid with surveillance data from simulated aircraft 'flown' by pseudo pilots in the EARL laboratory.

EARL Real Time Distributed Simulation (RTDS)

To support the demonstrations, the EARL real time simulator was expanded from a LAN based system in the EARL laboratory to a widely distributed system [2]. The EARL real time simulator was originally designed for "human-in-the-loop" simulations. Unlike legacy simulators, RTDS is built around a detailed object model in order to develop and execute customized efficient simulations, largely directed toward ATC and controller training

applications and new technologies and concepts. In addition to simulation, the RTDS is capable of linking external aircraft sources into RTDS aircraft objects. This allows for integration of aircraft simulators, live radar and ADS feeds or even software flight simulators. It is designed to use UDP multicast to distribute data to multiple destinations without replicating the data for transmission to each recipient, as would be the case in a client/server architecture using TCP or UDP unicast protocols. Also, UDP multicast is similar to the way air-to-ground radio broadcasts work as used by transponders, ADS-Broadcast, VHF data-link and VHF radios in that one copy of the data is transmitted for all recipients to receive.

Simulated aircraft used in the Segment B demonstrations were created in the RTDS target generator application and introduced into the simulation to provide computer generated traffic that could interact with the CBB aircraft while exercising satellite-based party-line voice and data link communication systems. Pseudo pilots controlling simulated aircraft at the EARL lab were able to communicate with Houston Center pseudo controllers using VoIP during the demonstration. The Pseudo pilots maneuvered their computer-generated aircraft from pseudo pilot displays in conjunction with the in-flight operations of the CBB aircraft.

The controller display design was never intended to be a high fidelity DSR clone. Instead it was designed to be DSR-like with the look and feel of a DSR display using similar symbology and data block formats and using the same command inputs. The display symbology was extended to display ADS positions of equipped aircraft. Additionally the data block was extended to show CPDLC responses on the forth line of the data block as well an indicator showing the availability of CPDLC for an aircraft.

The controller displays were closely similar to the Ocean East and Ocean West oceanic DSR displays showing the Gulf of Mexico with video maps of the coast, oceanic control sectors and the demonstration area. Each controller position was also equipped with a touch panel communication display. This panel allows a controller to establish two-way communication with another controller utilizing the underlying VoIP protocol. Controller-to-controller audio is heard through one ear of a stereo headset. Pilot-to-controller audio, which also uses VoIP, is heard through the other ear of the headset. A voice-activated switch (VOX) triggers transmission over an established controller-to-controller connection. Controller-to-pilot transmission is triggered by a

hand or foot switch, which interrupts the VOX if a controller-to-controller connection is established.

The GCNSS Network Structure

The network (Figure 1), provided by Boeing, was made up of a series of T1 lines between Houston Center, TX; Boeing ATM in McLean, VA; EARL in Daytona Beach, FL; the CBB Enterprise Operations Center (EOC) in Irvine, CA; and MIT Lincoln Labs. Additionally CBB provided the satellite connectivity between their EOC and the CBB aircraft using both low earth orbit L-band and geostationary Ku band satellites. Each site's router was equipped with a local loop T1 physical circuit connected to Sprint's frame relay "cloud". Each frame relay switch port was configured for T1 speed; a data rate of 1.544 Mbps. Permanent Virtual Circuits (PVCs) between the remote sites and the hub router at Boeing ATM headquarters in McLean, VA connected each of the sites together.

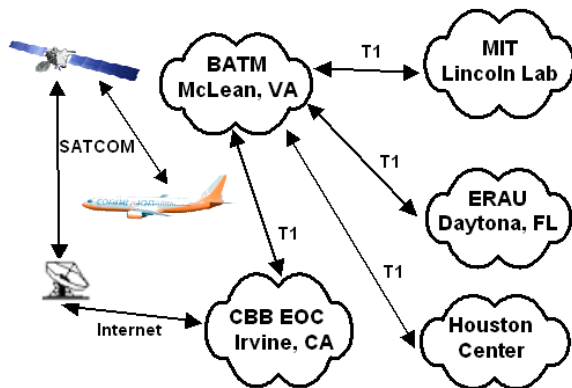


Figure 1. GCNSS Network Layout

FAA regulations mandated that all T1 links and the SATCOM link, which includes the path to the uplink station, be encrypted. The encryption method chosen was Triple DES (Data Encryption Standard, 168 bit) using the Internet Protocol Security (IPSec) extensions to the IP protocol. Each T1 link between the sites and the link between the EOC and the CBB aircraft was configured with a Generic Router Encapsulation (GRE) tunnel. All packet headers and payloads traveling between sites were encrypted as they passed through these GRE tunnels.

Each site's Cisco router was enabled with the Protocol Independent Multicast – Dense Mode (PIM-DM) routing protocol to forward the RTDS and VoIP UDP multicast packets between sites through the encrypted GRE tunnels. This allowed the underlying hardware to distribute the multicast packets without any application level overhead. The exception was

the SATCOM link between the CBB aircraft and the EOC in Irvine, which was not multicast capable; this is the main subject of this paper and is discussed later in detail.

The Multicast Issue

When an application is started that uses a multicast group, it will inform the operating system's kernel, which will in turn advertise on the network the desire to receive multicast traffic for the particular multicast group. A multicast enabled router on the LAN will receive the request and in turn advertise a desire to receive that multicast traffic on all of its multicast enabled network interfaces on behalf of that computer that requested it. Other routers receiving those requests will each in turn advertise that desire to eventually create a network tree connecting all interested parties. In this way the routers distribute the RTDS data and VoIP data with no effort on the part of the applications wishing to communicate.

Although the Cisco routers onboard the CBB aircraft and in the EOC supported multicast, the satellite and network connections between the EOC and the CBB aircraft did not. This presented a problem in that the multicast traffic on the EOC LAN could not reach the aircraft LAN; nor could multicast traffic on the aircraft LAN reach the EOC.

The ERoute Solution

The solution to multicast issue with the SATCOM link was for the author to create the ERoute tunnel application that could forward TCP and UDP data packets, including UDP multicast packets, through virtual tunnels.

This allowed the use of a reliable TCP based virtual tunnel over the satellite link to pass the multicast packets for the more critical control data flows and those where transmission order was important. A connectionless UDP unicast based tunnel was used for the VoIP audio packets as the VoIP protocol tended to handle the occasional lost or jumbled packets acceptably.

The ERoute tunnel application acts like a multicast forwarder. Modern day routers internally support standard multicast protocols like PIM-SM and PIM-DM and multicast routing protocols like Distance Vector Routing Protocol (DVMRP) to handle the forwarding of multicast packet between LANs. The problem with the current hardware trend is that most

multicast traffic on the Internet today is exchanged with protocols that are geared for one server and many clients like streaming audio and video. These protocols make use of a spanning tree from the server fanning out to all the clients, which use a rendezvous point to locate the closest branch to join the tree. This does not work as well with a distributed peer-to-peer system like the RTDS where each participant is both a source and client of the data being exchanged, such as VoIP packets and CPDLC messages.

The end result was the requirement to write an application, ERoute, which creates user-defined tunnels to encapsulate multicast packets for transport between the CBB aircraft and the EOC within the encrypted GRE tunnel connecting them. The ERoute tunnels are defined in a user configuration file read at startup.

The ERoute Architecture

The ERoute application makes use of two types of “tunnels”, internal tunnels and virtual tunnels (see Figure 2). ERoute internal tunnels exist within an instance of the ERoute application to pass data between two socket endpoints and act as conduits for data packets within that instance. Internal tunnels are synonymous with an internal switchboard.

ERoute virtual tunnels may exist between two instances of the ERoute tunnel application separated by any routable path. A routable path may be any IP network such as a LAN, WAN or the Internet.

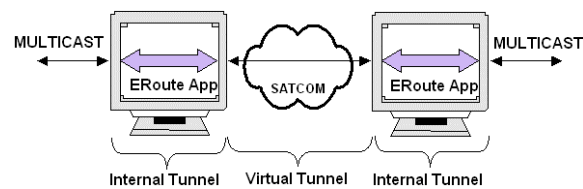


Figure 2. ERoute Application Tunnel Types

Each tunnel socket endpoint has an associated type, IP address and port. The available socket endpoint types include TCP_STREAM_SERVER, TCP_STREAM_CLIENT and DATAGRAM.

The TCP_STREAM_SERVER socket type may only connect to a TCP_STREAM_CLIENT socket type to form a reliable TCP client/server connection with all the benefits of the TCP protocol including automatic handshaking, error detection, resending of lost or corrupt data packets and guaranteed packet order on reception as they were sent.

The DATAGRAM socket type uses the UDP protocol and is therefore connectionless and does not

guarantee packet delivery or order. The DATAGRAM socket type may use discrete IP addresses for unicast point-to-point transmission or multicast IP addresses for mass distribution without replication.

The tunnels are resilient in that they will automatically connect and remain connected so long as there is a routable path for the packets. If the connection between a TCP_STREAM_CLIENT and TCP_STREAM_SERVER socket endpoint is broken the application will continuously try to reconnect.

It is important to note that when using the DATAGRAM type socket with a multicast IP that multicast loopback is disabled on the network interface used. Multicast loopback is a standard UNIX socket feature that loops back all outbound multicast packets back into the system so that they are available to other applications listening to the same multicast group. Since this application is meant to forward any incoming multicast packets on one end of a tunnel to the other end of the tunnel, the outgoing packets would come back into the system and be forwarded again in an endless loop. To prevent this looping the multicast loopback must be disabled. This also means that this application cannot be run on the same system as another application wishing to receive and send to the same multicast group. The reason is that any multicast packets leaving the system from a tunnel would not be looped back for the other application to receive.

The GCNSS ERoute Configuration

For the GCNSS demonstrations, an instance of the ERoute tunnel application was started on both a computer in the EOC and a computer onboard the CBB aircraft. The virtual tunnels between them were used to transmit the ERAU VoIP data and the RTDS data.

The VoIP software used three channels during the demonstrations:

- The VoIP control information channel
- The Oceans East VoIP voice data channel
- The Oceans West VoIP voice data channel

The RTDS uses five data channels: ELS, ADS, VDL, VDL2 and SimControl.

- The ELS channel is used to exchange information with the Electronic Library System (ELS). The ELS keeps track of unique IDs for all simulation participants

including aircraft and displays as well as aircraft specific data such as flight plans.

- The ADS channel distributes aircraft object ADS-B position reports and radar beacon reports to controller and pilot displays. For pilot displays this is effectively Traffic Information Service - Broadcast (TIS-B).
- The VDL and VDL2 channels carry VHF Datalink broadcasts between aircraft objects and the 'ground'. The VDL channel carried CPDLC control commands from the pseudo pilot stations to the simulated aircraft used in the demonstration. The VDL2 channel was separated for CPDLC text messages between the live CBB aircraft and controllers due to the mix of CPDLC standards demonstrated and ease of analysis.
- The SimControl channel is used by the RTDS control GUI for such things as time synchronization between applications.

An ERoute virtual tunnel was created for each of these channels to connect the EOC to the CBB aircraft while using the Ku-band satellite. However, due to limited bandwidth, while using the Iridium satellite connection, only one virtual tunnel was established to carry only the VDL2 channel CPDLC data. While using the Iridium satellite the VoIP data was directed to an analog bridge device patched via a telephone line to an Iridium phone in voice mode equipped with a headset onboard the aircraft. The analog bridge device was custom designed by EARL and includes a built in VOX circuit that triggers VoIP transmission when audio was heard over the Iridium phone onboard the aircraft. In this way any conventional form of analog communication, such as a telephone or radio, can be interfaced with and take full part in the VoIP system. The analog bridge device also supported the EARL VoIP application's pilot step-on prevention and controller priority features.

To ensure reception of surveillance data and RTDS and VoIP control data, all the channels except the two used for VoIP voice data were connected by TCP type virtual tunnels. Because packet loss on the two VoIP data channels could be tolerated better, those channels were connected by UDP unicast type virtual tunnels.

For illustration purposes the IP address of the ERoute tunnel computer in the EOC shall be designated EOC_IP. The IP address of the ERoute tunnel computer onboard the aircraft shall be designated AC_IP. The multicast IP address used by

the RTDS shall be designated RTS_M_IP. The multicast IP address used for VoIP data shall be designated VOIP_M_IP. The network ports used for the RTDS and VoIP channels are listed in Table 1.

Table 1. ERoute Tunnel Addresses and Ports

Channel	IP Address	Port
ELS	RTS_M_IP	10050
ADS	RTS_M_IP	10053
VDL	RTS_M_IP	10054
SimControl	RTS_M_IP	10055
VDL2	RTS_M_IP	10057
VoIP control data	VoIP_M_IP	50100
Oceans East VoIP voice data	VoIP_M_IP	50101
Oceans West VoIP voice data	VoIP_M_IP	50104

The ERoute Config File Explained

Each line in an ERoute configuration file, see Figures 3 and 4, represents an ERoute internal tunnel and contains the socket type, IP address and port for each end of the internal tunnel. Each internal tunnel specified works like a switchboard forwarding packets coming in one end of the internal tunnel and pushing them out the other end of the internal tunnel. Additionally, the ERoute application is not just for tunneling multicast packets. Any application can be allowed to connect to a tunnel end point to provide or accept data. In this particular use of the ERoute tunnel application for the GCNSS Segment B, all TCP_STREAM_SERVER type sockets from one ERoute tunnel application were connecting to TCP_STREAM_CLIENT type sockets of another ERoute tunnel application to form virtual tunnels over the SATCOM link.

It should be noted that the IP address for a TCP_STREAM_SERVER type socket is the IP address that the internal tunnel should listen on for incoming TCP_STREAM_CLIENT connections on the specified port. Conversely, the IP address for a TCP_STREAM_CLIENT type socket is the IP address of the TCP_STREAM_SERVER that the internal tunnel endpoint should connect to on the specified port. Because DATAGRAM type sockets are connectionless, there are no separate client and server types. A DATAGRAM type socket will listen for incoming UDP packets on the specified port on all available network interfaces, but will send

outgoing packets to the IP address and port listed for the internal tunnel endpoint.

For example, consider the internal tunnels specified on the first line on each file in Figures 3 and 4. The left side of each line contains the same IP address and port number. For the aircraft side, this means the TCP_STREAM_CLIENT socket will connect to the TCP_STREAM_SERVER socket on the EOC side at the EOC_IP on port 10050 to form a virtual tunnel. From the EOC side, this means that the TCP_STREAM_SERVER socket will listen for

incoming TCP_STREAM_CLIENT connections to the EOC_IP on port 10050 to form a virtual tunnel. The right side of each line contains the same IP address and port number of the multicast group for the RTDS ELS channel. Once the TCP connection is established a virtual tunnel between the two ERoute applications exists over the satellite connection. Each line represents an internal tunnel within the application that will pass packets bidirectional between the multicast group and the virtual tunnel connecting the ERoute applications.

```
TUNNEL = TCP_STREAM_CLIENT, EOC_IP, 10050, DATAGRAM, RTS_M_IP, 10050
TUNNEL = TCP_STREAM_CLIENT, EOC_IP, 10053, DATAGRAM, RTS_M_IP, 10053
TUNNEL = TCP_STREAM_CLIENT, EOC_IP, 10054, DATAGRAM, RTS_M_IP, 10054
TUNNEL = TCP_STREAM_CLIENT, EOC_IP, 10055, DATAGRAM, RTS_M_IP, 10055
TUNNEL = TCP_STREAM_CLIENT, EOC_IP, 10057, DATAGRAM, RTS_M_IP, 10057
TUNNEL = TCP_STREAM_CLIENT, EOC_IP, 50100, DATAGRAM, VoIP_M_IP, 50100
TUNNEL = DATAGRAM, EOC_IP, 51104, DATAGRAM, VoIP_M_IP, 50104
TUNNEL = DATAGRAM, EOC_IP, 51101, DATAGRAM, VoIP_M_IP, 50101
```

Figure 3. The Tunnel Application Configuration File Used Onboard The Aircraft

```
TUNNEL = TCP_STREAM_SERVER, EOC_IP, 10050, DATAGRAM, RTS_M_IP, 10050
TUNNEL = TCP_STREAM_SERVER, EOC_IP, 10053, DATAGRAM, RTS_M_IP, 10053
TUNNEL = TCP_STREAM_SERVER, EOC_IP, 10054, DATAGRAM, RTS_M_IP, 10054
TUNNEL = TCP_STREAM_SERVER, EOC_IP, 10055, DATAGRAM, RTS_M_IP, 10055
TUNNEL = TCP_STREAM_SERVER, EOC_IP, 10057, DATAGRAM, RTS_M_IP, 10057
TUNNEL = TCP_STREAM_SERVER, EOC_IP, 50100, DATAGRAM, VoIP_M_IP, 50100
TUNNEL = DATAGRAM, AC_IP, 51104, DATAGRAM, VoIP_M_IP, 50104
TUNNEL = DATAGRAM, AC_IP, 51101, DATAGRAM, VoIP_M_IP, 50101
```

Figure 4. The Tunnel Application Configuration File Used On The EOC Server

Overall what this means is that any multicast packets arriving on the EOC LAN, destined for the RTDS_M_IP address on port 10050, will be picked up by the EOC ERoute tunnel application and sent over the satellite connection via the virtual tunnel and be output on the aircraft LAN by the ERoute tunnel application onboard the CBB aircraft as a multicast packet destined for the RTDS_M_IP address on port 10050. The reverse is also true. The end result is that the virtual tunnel acts like a virtual network switch and the two ERoute tunnel applications are the switch's network interfaces.

Now consider the last two lines of the files. The left side contains the IP address of the other ERoute tunnel application since the socket type is DATAGRAM. The IP address represents the destination address to send outgoing packets from that end of the internal tunnel. The tunnel DATAGRAM socket endpoints will listen on both computers for incoming UDP packets on port 51101. Therefore, on the EOC computer a packet leaving the

left side of the tunnel will travel over the satellite to the AC_IP on port 51101. Likewise, on the aircraft computer a packet leaving the left side of the tunnel will travel over the satellite to the EOC_IP on port 51101.

Overall this means that any multicast packets arriving on the EOC LAN, destined for the VOIP_M_IP address on port 50101, will be picked up by the EOC ERoute tunnel application and sent over the satellite via the UDP virtual tunnel and output on the aircraft LAN by the ERoute tunnel application onboard the CBB aircraft as a multicast packet destined for the VOIP_M_IP address on port 50101. The reverse is also true. The end result is that DATAGRAM type virtual tunnels act similarly to the TCP type virtual tunnels.

The Future

The tunnel application is fully configurable for multiple tunnels of different types and hides the details of the connections between the tunnel end points from the user. The tunnels work over any IP routable connection, such as the Internet, without the need for multicast capable routers between the end points. Optionally, as was the case with the CBB satellite link, the tunnel can pass through an IPsec encrypted path for added security.

Since the completion of the Segment B GCNSS demonstrations, the ERoute tunnel application has already been expanded to include the option of requiring authentication upon connection to restrict access to sensitive data flows. Future planned additions include native support for data encryption.

The ERoute tunnel application is not limited to forwarding multicast packets over non-multicast enabled networks. The application can also be used to easily pass multicast feeds through a firewall, which can be challenging to configure to safely pass multicast traffic. Sky marshals could send video streams from onboard an aircraft to a tunnel endpoint on the ground, which could then forward or multicast the feed to multiple agents in the air or on the ground. The ERoute application could also be used to forward an UDP unicast data stream over a TCP based virtual tunnel for added reliability over a noisy communication link. Many other possibilities exist.

The tunnel application successfully carried all the VoIP communications and surveillance data traffic over the satellite link while meeting all end-to-end transmission and latency requirements for Segment B of the GCNSS Program.

References

- [1] Boeing Air Traffic Management, 2004, GCNSS Demonstration Segment B Report, D794-1014101
- [2] Wilson, Ian, Johnathan Pesce, 2004, Provision of Distributed Integrated Air Traffic Management Displays For The Global Satellite Communication, Navigation and Surveillance System (GCNSS), Proceedings of NASA ICNS Conference 2004.

Biography

Johnathan Pesce received his Bachelor of Science in Engineering Physics in 1996 from Embry-Riddle Aeronautical University. He worked as a Research Engineer in the Space Physics Research Lab from 1996 to 2002 studying Arctic and Antarctic Auroral optical emissions. In 1999 he earned his private pilot certificate and became more interested in aviation. In 2002 he joined Embry-Riddle Aeronautical University's ATM Research Laboratory. In 2003 he received his Masters of Software Engineering from Embry-Riddle Aeronautical University. Current research projects include enhancing the Real Time Distributed Simulation, further stages of the FAA/Boeing GCNSS Program, and managing the capture and processing of data for the EARL data warehouse of ASDI and weather data.