

# Distributed Image-Sensor Web Management for Bandwidth Constrained Environments

**Daniel Marconett**  
dmarconett@gmail.com

**Skylar Bemus**  
nautskye@gmail.com

**Samuel Johnson**  
samueljohnson@acm.org

**Ryan Jarvinen**  
ryan.jarvinen@gmail.com

**Daniel Potter**  
potterd64@gmail.com

*California State University, Sacramento, California 95819*

**Lawrence C. Freudinger**  
l.freudinger@dfrc.nasa.gov

*NASA Dryden Flight Research Center, Edwards, California 93523*

Remote and in situ observation is an essential ingredient of understanding of our Earth and its processes. Observed data greatly improve the ability of Earth and Environmental scientists to model and predict various events and patterns which transpire in the environment and around the globe. Unmanned aircraft, with onboard instruments which enable such data collection, are an essential part of the future remote observing infrastructure. These platforms will contain imaging instruments capable of producing far more data that can be transferred to decision-makers for near real-time use in a cost-effective manner although narrowband communications can be assumed. In this paper, we discuss a prototype system to manage metadata for mobile cameras similar to systems that could be installed on suborbital Earth observation platforms. Network cameras simulate a distributed system of imagers on one or more platforms. A software system is described that integrates the webcam imagery with time-varying metadata. This paper offers an overview of design and performance of this system in a benign communication environment, and discusses data management challenges as it applies to implementation of such a system on unmanned airborne observation platforms for Earth science missions.

## Key Terms

KML = Keyhole Markup Language  
OTH = Over-the-Horizon network communications  
RBNB = Ring Buffered Network Bus by Create  
UAV = Unmanned Aerial Vehicle  
XML = Extensible Markup Language

## Introduction

Images from digital cameras are two-dimensional observations of a small part of a three-dimensional physical world that varies in time (four dimensions). Webcams add repetition to the observations. If attribute data such as camera location, orientation, and field of view were available, it would be possible to project webcam images into a three-dimensional simulated world that varies in time. In many scenarios of interest here, the cameras themselves could be moving around, so we recognize that some of the aforementioned attributes themselves might also vary in time. Given a network-distributed set of possibly mobile or articulated imaging sensors such as webcams, along with the ability to acquire the required

time-varying attribute data (or metadata), it should be possible to combine these images into a common visualization environment. As with other forms of data fusion, the value of effectively presenting an integrated whole should be greater than the sum of its parts.

Essentially, we are interested here in constructing practical imaging sensor webs for various synthetic vision applications. Whereas traditional and commercially available cameras typically act as passive producers of data today, future systems of cameras will increasingly need to intelligently produce more useful data. Examples of the types of systems envisioned here include intelligent video security systems for large and small interior and exterior spaces, futuristic three-dimensional video teleconferencing centers, and synthetic vision systems for vehicles like aircraft and cars. We can categorize these systems into three application classes:

### ***Roaming Observation***

This application class is characterized by one or more imaging instruments on a mobile vehicle with at least occasional network connectivity via one or more data links. We assume availability of deep storage capacity for still and video imagery and the availability of embedded computing resources for relevant components of our application.

### ***Interior space observation***

This application class represents observation of interior spaces, and is characterized by multiple cameras with a high degree of overlap in their field of view. With a high degree of overlap, 3-D scene reconstruction, synthetic aperture (enhanced focusing ability) and freeze-frame 3-D video teleconferencing is possible.

### ***Exterior space observation***

In contrast to remote observation from an aircraft and in contrast to the heavily redundant scene coverage of an interior space, this application class tends to consist of geographically separated groups of possibly well-covered areas with large amounts of coverage gaps in between. In general this class of application is monitoring environments where some images may but not necessarily overlap. Given a single camera in each of  $n$  locations, this application class might be (ironically) the traditional video teleconferencing application, security monitoring of urban regions, residential security monitoring, or the traffic camera network we sometimes see on the evening news. Object tracking across the set of imagers (through time and space) is a common goal in this application class.

One particular application in the roaming observation class that is useful to the airborne science community and illustrative of certain technical challenges involves one or more autonomous unpiloted air vehicles (UAVs). Multiple cameras on a vehicle collect data useful to remote terrestrial decision-makers (airborne science researchers or platform operators), but the wireless communication links may be intermittent, and when those links are operating nominally they do not provide adequate bandwidth to move more than a small fraction of imagery acquired.

A specific example of this would be an UAV searching the Antarctic coast in winter for Emperor penguins. With geosynchronous satellites not visible and line-of-sight telecommunications with human-inhabited ground stations not feasible, the only available data links are either very low bandwidth (such as Iridium) or highly intermittent (TDRSS satellite or other relay scenarios could be available for a fraction of a given flight.). In order for terrestrial researchers to effectively use the imagery for situational awareness and decision support in this scenario, it is desirable to have the ability to “TiVo” compressed versions of the source imagery in a manner where the imagery is automatically integrated with live weather and previously known terrain data sources.

Analysis of these types of problems leads to a few observations: Metadata and image data in a sensor web do not in general come from the same place. A camera does not know where it is or where it's pointed. Metadata for the images is time varying, and online solutions must dynamically marry image data with its metadata. Effective bandwidth management involves buffer management on both sides of an unreliable data link. Effective bandwidth management for online and near realtime use also requires the ability to implement flexible on-the-fly processing, possibly at many geographically disparate locations, and likely on both sides of any unreliable or otherwise constrained data link. In the UAV example discussed above, random access of time-indexed cached data, coupled with compression and decimation represent some of the required processing, but more sophisticated processing can also be envisioned and may be required as the total number of source image pixels increases.

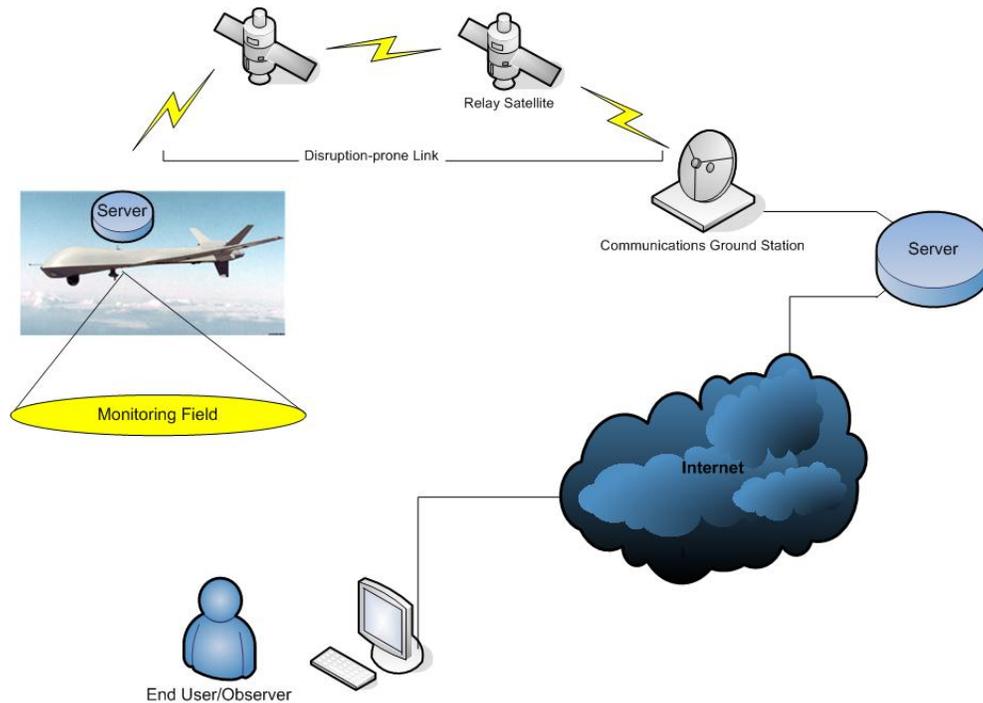
Clearly a general solution is a formidable undertaking. The project discussed in this report is a modest effort to prototype a partial solution. This prototype focuses on the dynamic marriage of data and metadata and leverages existing tools wherever possible. The objective of this partial solution is to aggregate multiple webcams with their respective metadata into a single geographic reference frame, and demonstrate the ability for those images to automatically move in this reference frame based on dynamic updates to the metadata. The remainder of this report discusses the resulting system architecture and provides an initial look at performance and discusses, where possible, known performance bottlenecks and recommended approaches to follow-up activity.

## Project Background

The I-Scan project was conceived as a project to fulfill academic requirements for undergraduate computer science coursework at Sacramento State University. The focus of the project is derived from ongoing research and development within NASA's suborbital Earth observing science program. Among the needs of the Earth science communities are flexible and affordable operations leveraging interactive (network-oriented) communications with instruments carried in piloted and unpiloted vehicles deployed potentially anywhere on the planet. The Over-the-Horizon Networks (OTH) component of the Earth Science Capabilities Demonstration Project is interested in maturing affordable, disruption tolerant, and network-centric communications components as well as the application-layer tools and services that accelerate the emergence of useful sensor web capabilities that enable greater productivity for researchers with airborne science instruments. The project targets a core operational capability by 2010 that facilitates emergence of increasingly intelligent observation instruments.

Aligned with this technology vision, it turns out that within the international Earth science communities there is a concerted effort to conduct research in the Earth's polar regions in 2007-2008. The first designated International Polar Year in 1882-83 and all similar initiatives since have all had a major influence on growing our understanding of Earth's processes. One of the projects being considered was the use of UAVs to autonomously patrol coastal Antarctica in the winter night to study one of the planet's leading indicator species on global climate change, the Emperor penguin.

A UAV with imaging systems - visible and infrared cameras, synthetic aperture radars, etc, could easily generate many gigabytes of useful imagery during a typical flight. Unfortunately, it's not possible to move all that data in real-time back to decision-makers in their (significantly warmer!) laboratories. The only continuous connectivity is low bandwidth. While studying this operations challenge, the concept of being able to scroll back and forth through time on thumbnails and "filmstrips" of the onboard imagery was noted as an enabling capability. It is this scenario that precipitated the idea for the I-Scan project discussed here.

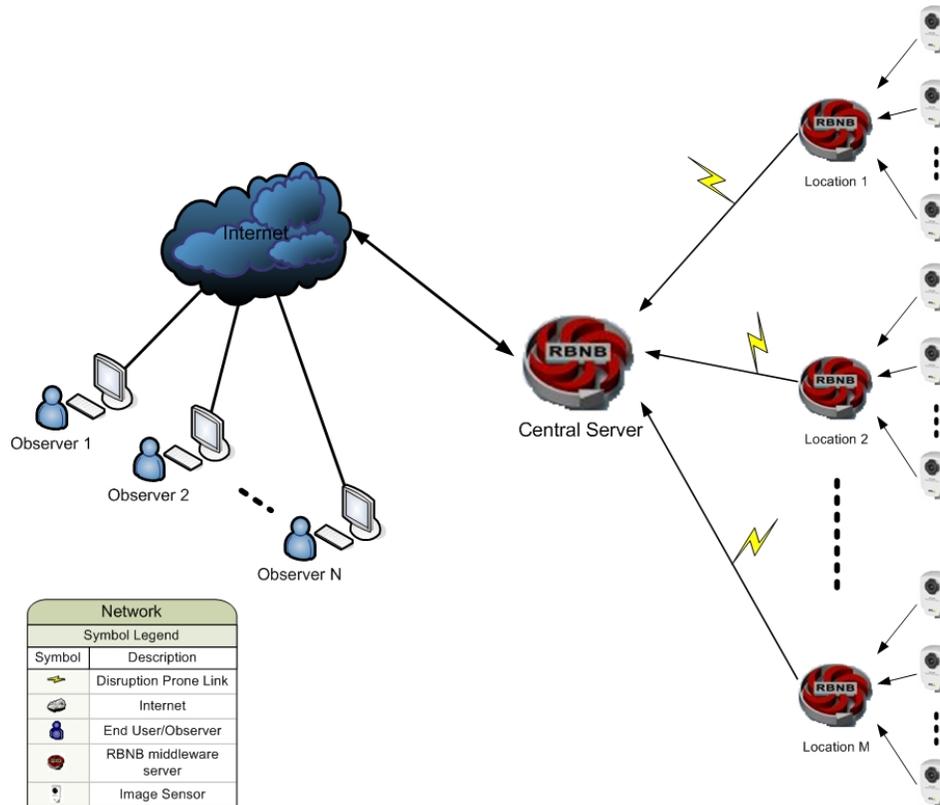


**Figure 1 - End Use Implementation**

Disruption and delay-prone links demand the system to be tolerant of data starvation for substantial periods of time. This fact presents challenges when designing a video-on-demand network computing system. This constraint is necessary since implementation is to occur onboard a NASA Unmanned Aerial Vehicle (UAV) in the particularly harsh environment of Antarctica. The only feasible communications infrastructure to leverage for such a mission in this part of the world is the Iridium satellite constellation, which is significantly limited, 2.4kbps to be exact. Thus, topology and link control are crucial to optimal system performance. Figure 1 is an over view of the desired end-use implementation and system structure.

## System Architecture

The main task of this project is to create this sensor monitoring application as an interoperable part of Create's RBNB software distribution, and to augment the built-in capabilities of this system for NASA Earth Sciences. We are doing so by utilizing the basic functionality provided by the Create RBNB API. Our goal is to create a more defined, extensible, and robust version of this application to support a hierarchical structure of sensors, or more abstractly, data acquisition devices, which will monitor multiple geographically distributed locations and broadcast this information to a central collection server. Figure 2 is a graphical overview of this project's multi-tiered distributed network architecture.



**Figure 2 - System architecture**

The first tier is comprised of the observation locations, sensors, and their respective local ring buffer servers. Each observation location corresponds to an aforementioned “monitoring field” in figure 1. Essentially, each source location can be thought of as an independent UAV, monitoring its own respective observation location. Alternatively, the UAV can be abstracted as a “source location” or a “sensor collection” for the purposes of system development.

The middle tier is the central server, which services end-user requests and receives data broadcasts from source locations. The central server will have knowledge of the sensor collections at the various locations as well as the types of sensors that each collection is composed of. Furthermore, it is to have full knowledge of the channel mapping and system/metadata configuration and tagging at each location.

The last tier is the observer(s), who make(s) requests of the central server over the internet for relevant data, and viewing them via a media player. This will be comprised of an existing prototype media player client application which is included in the standard RBNB package distribution, including Google Earth.

## A. Data Acquisition Layer

The fundamental purpose of a sensor web is to facilitate data acquisition. Each monitoring field (or data source location) is observed by a sensor web in our architecture. The testbed and system upon which it runs is specifically designed to monitor and process data from multiple geographically distributed monitoring fields, and have them accessible to end-users via the internet. The data acquisition layer is comprised of a local monitoring application which acquires data from imaging and metadata sensors and feeds them to the local RBNB caching mechanism.

As with the Central Server layer, the main data store for the data acquisition application at each sink location is the RBNB. At each source location, LocalMonitor polls the local sensors for updated information. One of our tasks was to create this LocalMonitor application by defining an XML data hierarchy which defines the system configuration and metadata dependencies. The XML files which conform to this predefined hierarchy are used by the monitoring application as a configuration file, specifying the local sensors from which data will be acquired. Additionally, RBNB's at source locations communicate with the Central Server via TCP/UDP connection(s).

## B. Central Data Staging Layer

The Central Server of this system acts as a data-staging area, whereby the end-user can query and receive relevant data based on the location and camera(s) specified. The ability to view multiple image sources simultaneously depends on which end-use application the client is running locally. This layer's main data store mechanism is the Ring Buffer Network Bus (RBNB) Data Turbine. The Central Server accepts client requests over http through the various end-user applications mentioned in the previous section.

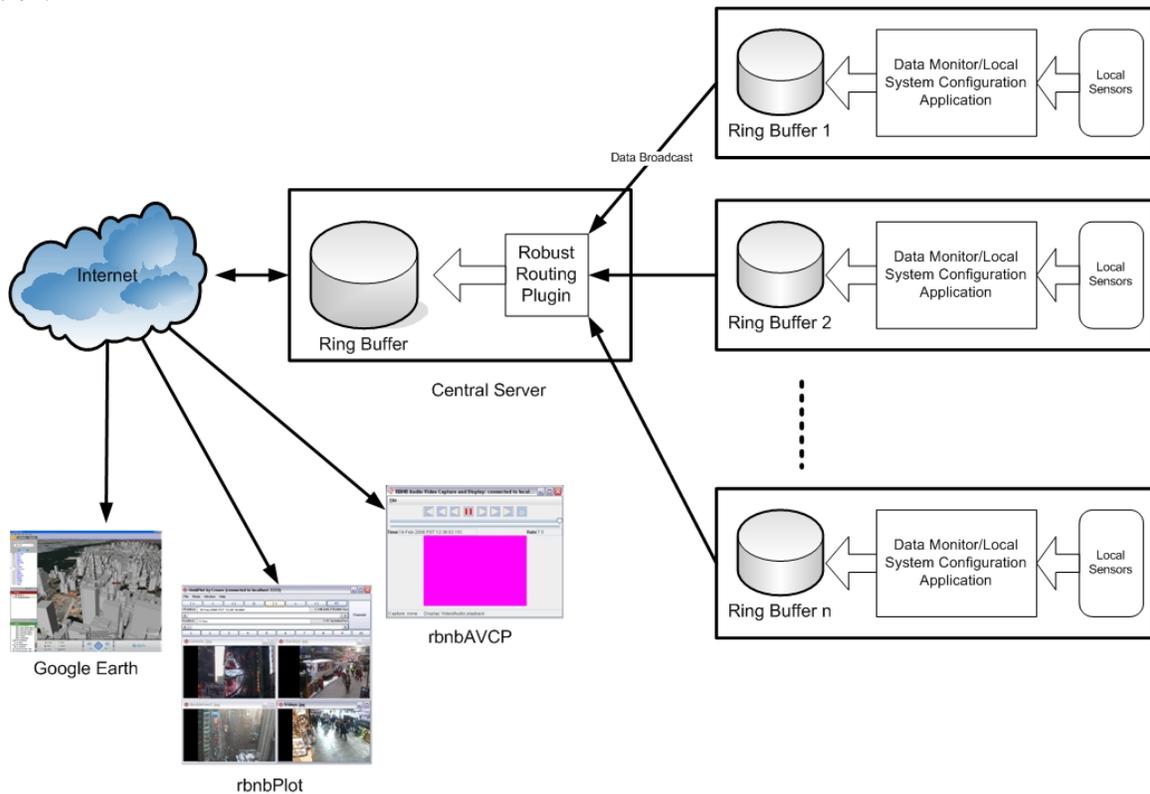


Figure 3 - Component Process Interactions

The Central Server is the repository of all sensor data from all locations, stored in channels which correspond to the sensor which produced the data. The  $i^{th}$  location has  $n(i)$  channels. The central location therefore has  $m$  channels, where

$$m = \sum_{i=1}^n n(i).$$

Each source location ring buffer  $i$  will have  $n(i)$  sensors and  $n(i)$  channels, each channel corresponding to a particular sensor, the Central Server ring buffer will have “ $m$ ” channels, where  $m$  is equal to the sum of the total number of sensors of all the distributed source locations. Furthermore, the data in these “channels” are stored in time oriented fashion. Each channel has an associated data type which corresponds to the data stored in the channel, i.e. image data is stored as byte arrays, each one representing an image from a particular camera. Each camera, and for that matter, each sensor, has its own channel.

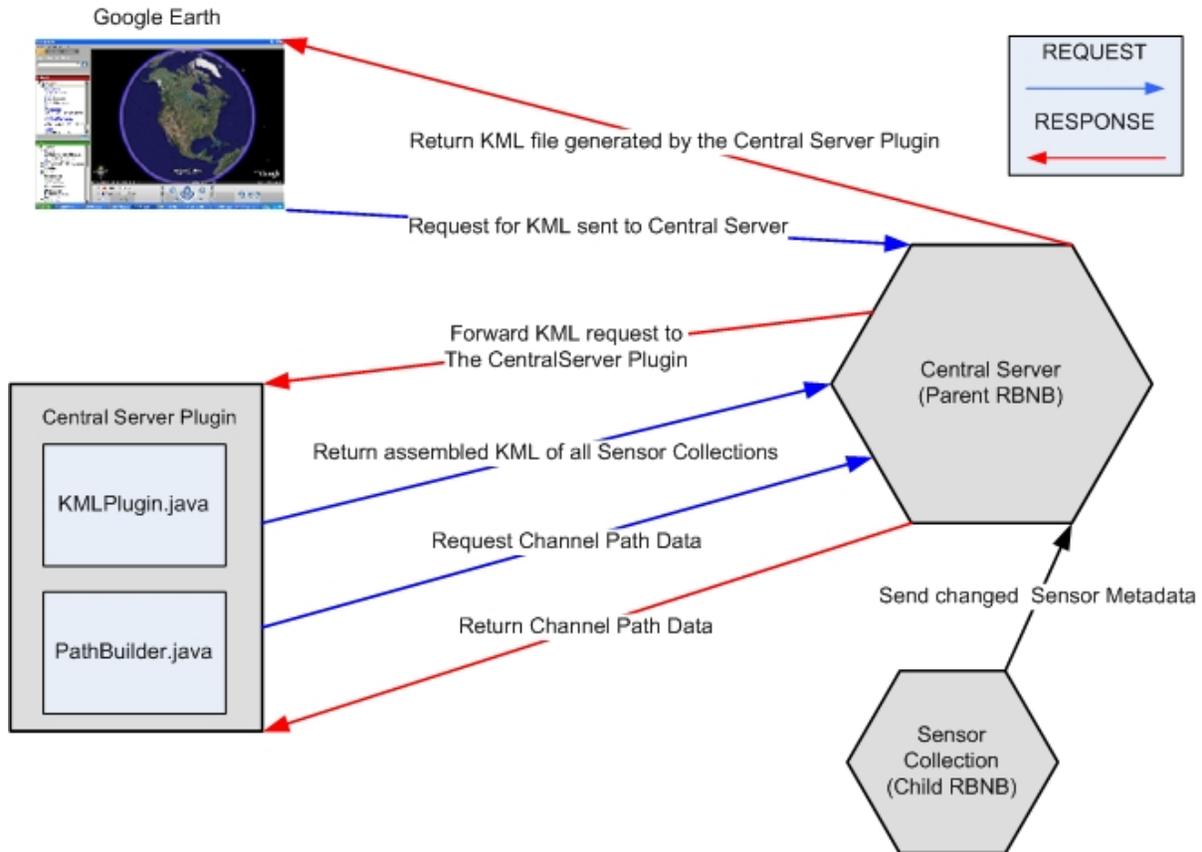
### C. End-User/Observer Layer

This layer is comprised of several possible media applications which will reside on the client machine. These applications are already existing prototypes and will be leveraged to view the image data as the main foci of this project are the issues of system configuration, data classification, data acquisition, and network topology concerns. These applications are `rbnbAVCP`, `rbnbPlot`, and Google Earth.

The Audio-Video Capture and Display (AVCP) application from Creare is a media player which is capable of capturing audio and (more importantly for our purposes) video as a stream of jpeg files and allow the viewer to stream back and forth through time in a time-oriented fashion. The end-user merely connects to the appropriate ring buffer server and selects the desired channel to view the collection of images as a video stream.

The `rbnbPlot` application, also from Creare, provides a mechanism by which the user can view numerical and/or image data, also in a time-oriented fashion. Again, the end-user merely connects to the appropriate ring buffer and selects the desired channel(s) from the structure to view through the plot interface. This application is suited to view multiple channels at once and is thus uniquely suited for the viewing of multiple image devices, connected to our system, in tandem.

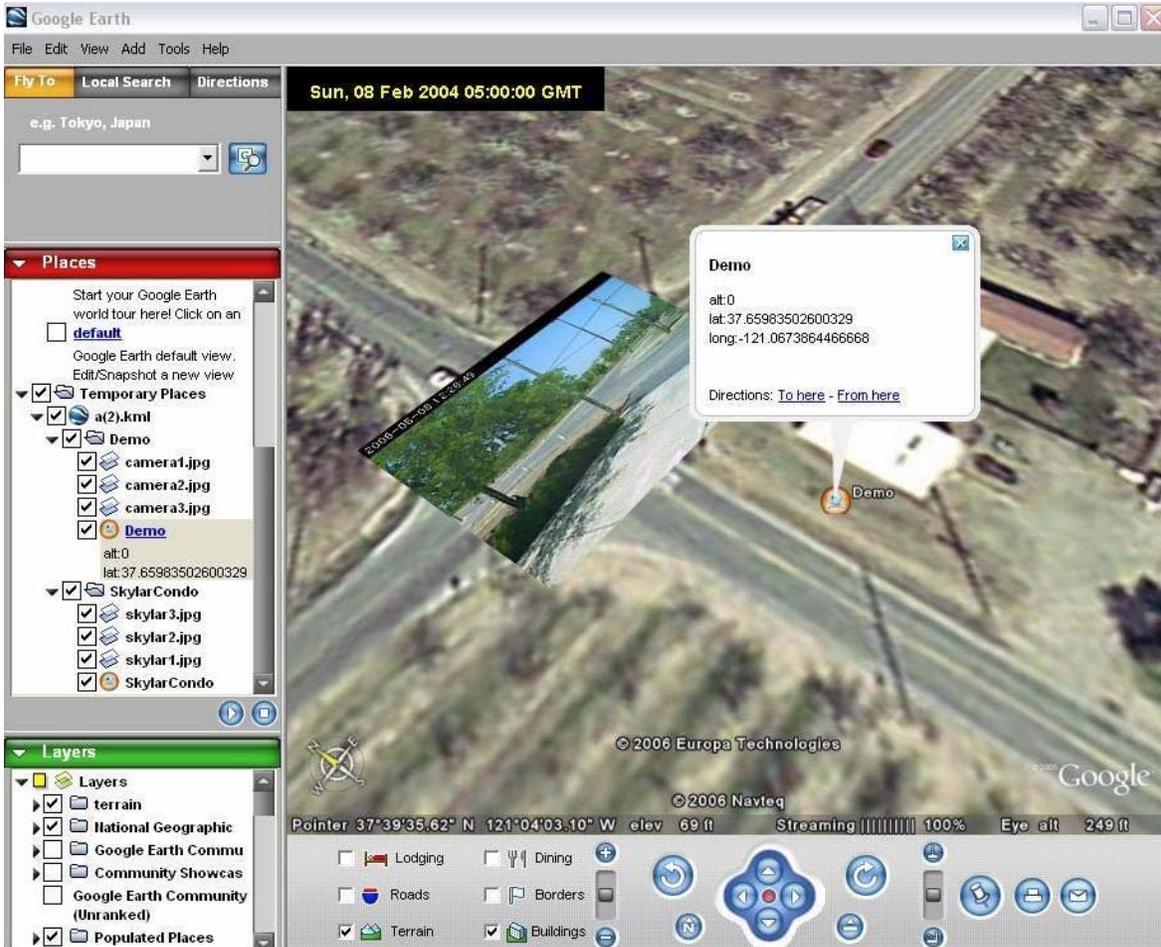
The final (and most important to this project) end-user application mentioned is Google Earth. Creare has provided a plug-in whereby any jpeg imagery can be fused with the Google Earth application relative to its GPS coordinates. The plug-in – also a prototype – helps perform data fusion of images and metadata for display in the Google Earth visual browser. The additional functionality introduced by this plug-in will allow an end-user to fly to any location near the earth, watch live data, and scroll back and forth through time. The optimal functionality would be to click on an aircraft while in transit, virtually go inside of it, and look out through the windows via onboard cameras. The I-SCAN Project is an initial framework for testing this hypothesis of using network cameras to achieve temporally sensitive virtual visits of the monitoring field. The data interaction diagram below conveys the basic process of .kml file request and generation with respect to this three-tiered testbed architecture.



**Figure 4 - KML Generation**

The bidirectional communication for KML file generation is handled strictly by the central server. The only other task for the local sensor collection(s) to handle (besides that of periodically broadcasting the latest sensor data) is that of creating a KML template which corresponds to the sensor hierarchy of the XML configuration file at the local level. This happens a single time, at initial system configuration, were shortly after creation, the template is dedicated its own channel, and broadcasted to the central server. It is the central server's job to concatenate these templates into a single master KML file which can then be accessed over the internet by end-users through the Google Earth application.

Moreover, the ability of this Google Earth application to view data based on a desired time of acquisition, in addition to the functionality of viewing the latest data, is governed by the 'Timedrive' plug-in provided by Creare. Timedrive provides a mechanism by which an end-user can move through archived data by specifying the desired time and interval through a web browser interface before acquiring the imagery/scalar data. In essence, Timedrive serves as a proxy between the central server and applications acquiring data over the web from the central server, such as Google Earth. Unless otherwise requested, the base case for this interaction is viewing the data in real-time. The afore mentioned time-sensitive request capability enabled by Timedrive provides a hitherto unrealized potential of Google Earth as an application which not only can be used for real-time data capture/observation, but also observation based on a specified fourth dimensional time aspect, which is a value added objective of this project



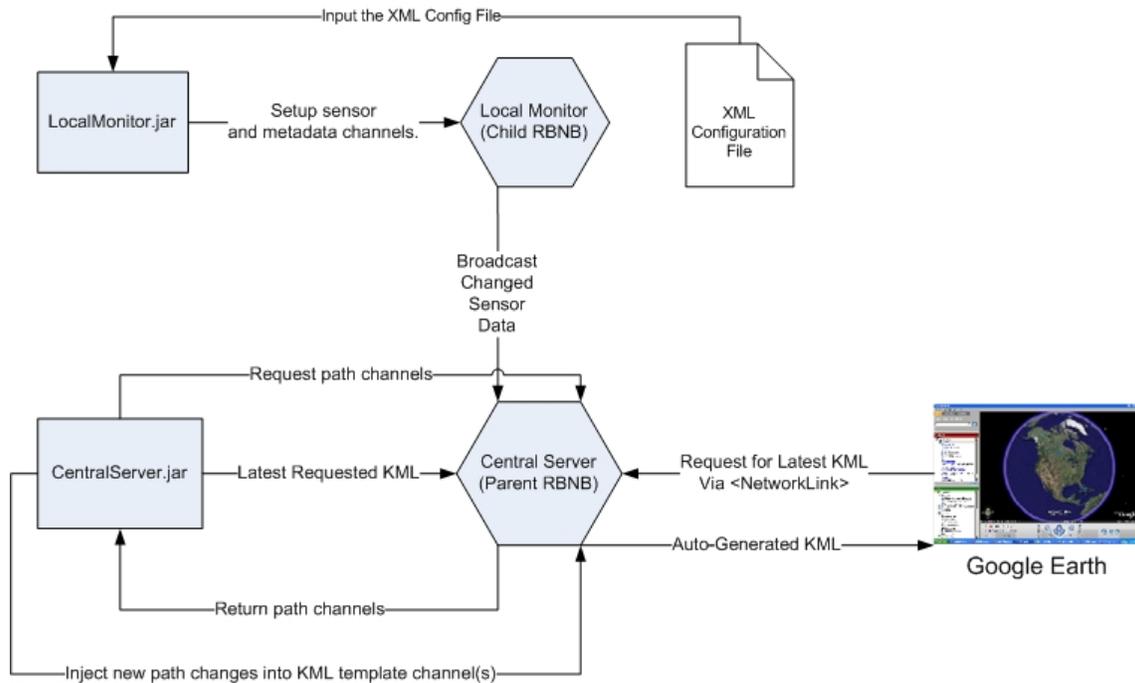
**Figure 5 - Google Earth Sensor Web Observation**

Figure 5 displays the user interface view of a particular monitoring location named “Demo.” The metadata visible in the “balloon” window is that of altitude, latitude, and longitude of that particular location. The image overlay is an image sensor’s view of that particular intersection, which is updated through the Timedrive proxy server which residing on the central server. The KML file data which makes this interface possible is viewable in the left pane of the application, and is named “a(2).kml.” The intersection image overlay is camera3.jpg and is also viewable in the left pane under the “Demo” location directory.

## Data Management

### A. Intelligent Management

Three key components of intelligent management are autonomy, survivability, and interoperability. Autonomy is applicable to an intelligent data acquisition system in that, if the acquisition process is being automated, priority must be granted to the agent which is directing data delivery between system components and the edges of the system for the end-user. Survivability is key to this automated system in that critical real-time data acquisition must take place at regular intervals, in a variety of environments, otherwise the data loses much of its value. Interoperability is the crucial piece of the puzzle whereby all components of the system are allowed to interact successfully. This is especially important when dealing with heterogeneous internetworking and multiplatform distributed systems such as ours. A key point of successful interoperability in our case is not only the compatibility of the system components, but the flow of data between these components and how this data is interpreted, especially when it pertains to how the data is categorized. These interactions are characterized at the system level by the abstract flow diagram below.



**Figure 6 - System Data Flow**

While the above data flow mapping depicts the interactions of the various components at the logical level, there is an explicit semantic relationship between the entities and data which is exchanged between them. This relationship is defined by the hierarchy of the XML configuration file at the data acquisition layer, and defines whether or not the scalar data from other local sensors (such as temperature, air speed, GPS, etc) will be visible within the Google Earth application. Since the non-image sensors produce scalar values over time, and the main focus of this project is implementation of an imaging sensor web, we have found it convenient to define the data from these secondary sensors as ‘metadata.’ In the following section, we provide additional commentary on metadata and how exactly it fits into the scheme of a given local sensor web.

## B. Data Hierarchy and Metadata

The collection of data by itself is generally without context. Metadata provides context for data. With respect to a distributed sensor web, the metadata could provide such information as geographical location of the sensor, the relationship of that sensor's data collection, the environment in which the sensor resides, and the time each sensor provides newly acquired data. Image sensors can be supplemented with important metadata parameters. Besides geospatial metadata, it is beneficial to record the orientation of the camera's view, specifically pitch, yaw, and roll. Additionally, the angle of the lens' parallax could be important when the cameras are at a significant distance from their target.

By using location and view-related metadata, our system is able to display image data from many sources on a single shared display (Google Earth). Without this metadata, it would not have been possible to place the images in their correct geospatial locations. The diagram below depicts the XML schema hierarchy as a tree.

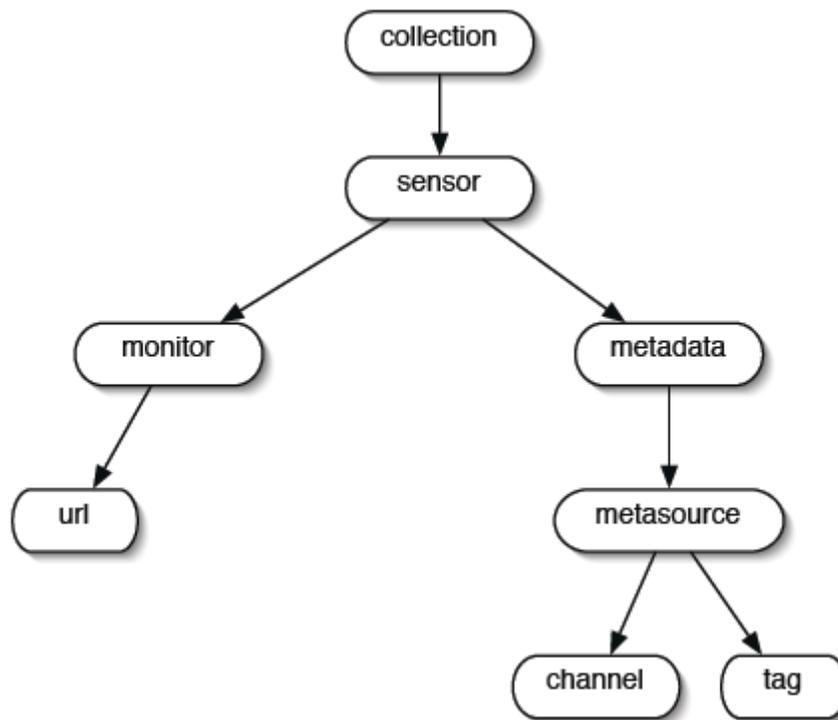


Figure 7 - Schema Hierarchy

- Starting at the root of the tree in the above diagram is the tag "collection". This tag has a three attributes: "name" is a unique name given to a specific collection of sensors, "destURLPath" is the path to the local RBNB server, and "static" should contain a value of *false* if the collection is moving, or *true* if the collection is static. A collection can have any number of sensors.
- At the next level is the "sensor" tag which has a single attribute, "kml". This kml attribute should be assigned the boolean value "true" if the data associated with this sensor is going to be an image overlay in Google Earth, or it should be assigned "false" if it is not an image to be overlaid in Google Earth.

- The next level below sensor contains "monitor" and "metadata" tags. The monitor tag is used to describe from where the sensor will get its data. The attributes for monitor are "destURLPath", "minimumInterval", and "mkcolQuery". These attributes describe how monitor behaves.
- Below monitor is the tag "url" which contains the url of the data that monitored.
- The "metadata" tag contains no attributes. Below this is "metasource" which is used to describe a source of metadata. The metasource needs a "tag" and "url". Some special "tag" values are listed below. A metadata source can also contain a "balloon" attribute which, if set to *true* enables a description balloon to appear over the placemark corresponding to the collection's location.

If the "kml" attribute is true, the following metasource tags are required:

- alt – the altitude value associated with the view altitude in Google Earth
- lat – the latitude value (decimal) associated with the view location in Google Earth.
- long - the longitude value (decimal) associated with the view location in Google Earth.
- heading – the heading value (decimal) associated with the view location in Google Earth.
- tilt - the tilt value (decimal) associated with the view location in Google Earth.
- range - the range value (decimal) associated with the view location in Google Earth.
- overlay-north – the location of the north edge of the image overlay in Google Earth.
- overlay-south - the location of the south edge of the image overlay in Google Earth.
- overlay-east - the location of the east edge of the image overlay in Google Earth.
- overlay-west - the location of the west edge of the image overlay in Google Earth.
- overlay-rotation – the rotation of the image overlay in Google Earth.

This section contains a skeleton of the schema with descriptions in the form of comments to further describe the purpose of each item.

```

<collection name="" static="" destURLPath="">
  <sensor kml="">
    <monitor destName="" minimumInterval="" mkcolQuery="">
      <url> </url>
    </monitor>
    <metadata>
      <metasource balloon="">
        <tag> </tag>
        <channel> </channel>
      </metasource>
    </metadata>
  </sensor>
</collection>

```

Generally speaking, choosing the types of sensors for a particular deployment depends on the domain of the information being observed. That being the case, the primary data type would require metadata which lends context to provide a more complete depiction of the monitoring field. Although this schema is used with regards to image sensors, it was designed with the concerns of a general sensor web in mind.

## Performance

Performance considerations were primarily with regard to data management. It was necessary to make all attempts at minimizing network congestion by not transmitting redundant data over the constrained link between sensor locations and the central server. Features such as Google Earth path building required the appendage of new data onto a list of old data. To minimize link saturation, this path was built at the central server tier by monitoring specific channels for changes as data is being retrieved from the acquisition tier. This approach accomplishes the objective of efficiently utilizing link capacity while maintaining temporal state information. However, by building path information at the central server tier and constantly monitoring for newly acquired data, the central server must allocate more of its resources in order to take on this added processing.

However, regardless of the effectiveness of the strategies employed for efficient bandwidth utilization, the single central server and the link it shares with the monitoring server remains the network traffic bottleneck of this particular approach. Not only is all data being broadcasted over constrained links to this central system, but all of the data processing, from caching and archiving multiple channels at a time to servicing potentially multiple KML requests, is being carried out at this tier area as well.

The RBNB mechanism is the essential data store which ties this entire system together. Since certain metadata relationships can yield new information through additional analysis, under constrained network conditions it may be desirable to postpone this additional processing to later tiers. It is therefore desirable to interact with data as it moves from tier to tier. Specifically, the CSUS I-Scan Group recommends the implementation of an event-driven mechanism for data processing between tiers, allowing the ability to modify data upon arrival at the central caching location.

## Conclusion

In summation, the project detailed in this paper has been an attempt to show the viability of network cameras as imaging sensors in a geographically distributed imaging sensor web. The applications designed to run at both the central server and data acquisition tiers were created to facilitate a network testbed subject to connectivity constraints by efficiently defining and managing acquired data and their respective relationships. A star topology architecture was employed to buffer data at a central location to better its availability to client applications, such as Google Earth, over the Internet.

Performance bottlenecks were identified as the bandwidth-constrained links between the various distributed monitoring localities and the central server data repository, as well as the fact that most, if not all, of the data processing occurs at this central server. Future investigation into the possibility of introducing one or more intermediary data processing tiers into the architecture to provide computational speedup might be warranted.

The system described manages all stages of data processing from acquisition to end-user observation. By defining and maintaining data-metadata relationships, context is preserved, yielding a more complete depiction of the monitoring environment. Furthermore, the added approach of acquisition in near real-time enables end-users to accumulate temporally-sensitive knowledge within a reasonable time frame.

## **Acknowledgements**

The I-Scan project was sponsored by NASA Dryden Flight Research Center and the AERO Institute under Task Number 28, "Network-Enhanced Telemetry and Global-Reach Test Range Development. Industry participants included Creare, Inc (Hanover, NH) and SemQuest, Inc (Colorado Springs, CO).

The CSUS I-Scan Group would like to thank Creare, Inc for the time and resources contributed to this project, as well as the endless consultation regarding their RBNB middleware package. Furthermore, this project would not have been possible without the funding provided by the Aero Institute (Palmdale, CA). Finally, this project was conceived by coauthor and project sponsor Larry Freudinger. The group is truly grateful for the opportunity afforded by him to engage in system research and development.

## References

- [1] C. E. Sorenson, S. K. Yarbrough, L. C. Freudinger, P. T. Gonia, "Research Environment for Vehicle Embedded Analysis on Linux," Paper No. 03-11-04, Proceedings of the International Telemetry Conference, Vol. 39, 20-23 October 2003, Las Vegas, Nevada. ISSN 1546-2188
- [2] M. J. Miller; L. C. Freudinger, "A Web Compatible File Server for Measurement and Telemetry Networks," Paper No. 03-11-05, Proceedings of the International Telemetry Conference, Vol. 39, 20-23 October 2003, Las Vegas, Nevada. ISSN 1546-2188
- [3] A. Dietz, E. Friets, W. Finger, J. Bieszczad, M. Miller, L. Freudinger, "Online Distributed Vehicle and Machinery Health Management," Paper No. 03-04-04 Proceedings of the International Telemetry Conference, Vol. 39, 20-23 October 2003, Las Vegas, Nevada. ISSN 1546-2188
- [4] M. Miller, W. Baschnagel, J. Bieszczad, E. Friets, W. Finger, R. Kaszeta, J. Wilson, "Intelligent Observation Study," TM-2528 Final Report Task 1.5.2, Creare Inc., January 18, 2006
- [5] M. Miller, W. Baschnagel, J. Bieszczad, E. Friets, W. Finger, R. Kaszeta, J. Wilson, "Intelligent Observation Study Near Term Capabilities Demonstration," TM-2528 Final Report Task 1.5.3, Creare Inc., January 18, 2006