

# Agent Based Approach for Searching, Mining and Managing Enormous Amounts of Spatial Image Data

**Paul Palathingal**

Oak Ridge National Laboratory  
P.O. Box 2008 MS 6085  
Oak Ridge, TN 37831-6085  
palathingalp@ornl.gov

**Thomas E. Potok**

Oak Ridge National Laboratory  
P.O. Box 2008 MS 6085  
Oak Ridge, TN 37831-6085  
potokte@ornl.gov

**Robert M. Patton**

Oak Ridge National Laboratory  
P.O. Box 2008 MS 6085  
Oak Ridge, TN 37831-6085  
pattonrm@ornl.gov

## Abstract

Scientists and intelligence analysts are interested in quickly discovering new results from the vast amount of available geospatial data. The key issues that arise in this pursuit are how to cope with new and changing information and how to manage the steadily increasing amount of available data. This paper describes a novel agent architecture that has been developed and tested to address these issues by combining innovative approaches from three distinct research areas: software agents, georeferenced data modeling, and content-based image retrieval (CBIR). The overall system architecture is based on a multi-agent paradigm where agents autonomously search for images over the Internet, then convert the images to a vector used for use in searching and retrieval. Results show that this system is capable of significantly reducing the time and management effort associated with large amounts of image data.

## Introduction

Enormous volumes of remote sensing image data are being produced on a daily basis throughout the world. This geographical data is then analyzed to create scientific, military, and intelligence information. This image information is critical for scientific and national security purposes. A significant challenge that exists in producing this image information is managing the vast amount of steadily increasing data (usgs 2000). For example, the United States Geological Survey (USGS) has been archiving enormous volumes of data and faces exponential near- and long-term growth in digital data. The typical process for managing geographical image information is to manually create an archive of imagery data. Analysts must continually update this archive with new and better imagery. Within the archive, each image must be normalized so that they can be readily merged, compared, and analyzed with other images. Finally, these images must be manually searched when new information is required.

We have developed and tested a novel agent architecture to address these issues by combining innovative approaches from three distinct research areas: software agents, georeferenced data modeling, and

content-based image retrieval (CBIR). This system addresses the challenges of organizing and analyzing vast volumes of image data, and of automating the existing manual image analysis processes. The overall system architecture is based on a multi-agent paradigm where agents autonomously look for images over the Internet, then convert the image to a vector used for searching and retrieval.

In the next section, we give some background information on related work. Then, we present our multi-agent approach. Finally, we present performance results of two multi-agent architectures used to retrieve and manage a set of images from the Internet.

## Background

Scientists and intelligence analysts are interested in quickly discovering new results from the vast amount of available geospatial data. The key issue that arises in this pursuit is how to manage the steadily increasing amount of available data. Clearly, this is an important problem, and many important results exist.

Christopher et al. were able to show that with software agents, it is possible to gather images from different sensors into one repository (Christopher and William 1995). They make use of agents to look at different vendors and download images that are constantly being produced. However, the approach has a predetermined set of vendors publishing images. It does not take into account that there could potentially be a number of other images on the internet that could be downloaded and is useful for scientific, government or commercial application.

MacLeod et al. describes a search mechanism to look for images using simple server side applications that enable scientists to find and evaluate image data (MacLeod, Amorim and Valteau 2000). A web-hosting server using Map Server technology and Web Map Server (WMS) renders data in response to standard WMS requests. The Data Access Protocol (DAP) protocol is used to abstract different data formats to allow client

applications to work in whatever environment is required. This provides a web services approach to data management, but does not address the issues of analyzing the data that has been retrieved.

Weber et al. present a network image search and retrieval system that addresses the issue of vast amounts of spatial data (Weber et al. 1999). The Chariot architecture stores metadata about images and derived features. From this metadata, similarity searches can be performed. The metadata approach reduces the amount of storage needed, if the image data does not change.

Zhu et al. implemented a neural network based system for creating a digital library for georeferenced information (Zhu et al. 1999). They describe a metadata approach which loosely couple different media sources (textual, image and numerical) for further analysis. This analysis includes a self-organizing map (SOM) to find similar georeferenced data, Gabor Filters for feature extraction, and image compression to allow searching over different image resolution. The metadata concept is extended to other media types, with increased emphasis on automating the analysis aspects. They do not address the challenge of keeping up with enormous amounts of digital data being produced.

Nolan et al. describes an approach for an agent based distributed system for imagery and geospatial computing (Nolan, Sood and Simon 2000). The approach centers on the development of an ontology, agent architecture, and an agent communication language. The Agent-Based Imagery and Geospatial processing Architecture (AIGA) focuses on using agent technology to gather, store, locate, index, process, and transmit imagery and geospatial data. The main emphasis of the work is in resolving issues of managing voluminous data. While an interesting architecture is described, the paper focuses on ontology and agent communications issues, but does not provide detail on the image analysis, searching and retrieval processes.

While significant work has been done towards this problem, a number of challenges remain. First, is the issue of how to retrieve and manage continuously updated imagery data, and secondly, how to distribute the analysis of an image that have been found. The concept of a multi-agent based approach appears to be a viable means of addressing these two issues. Agents can be used to gather data, and to work in parallel to automate various aspects of the retrieval, content-based indexing and analysis process.

### Approach

Historically, image repositories consist of centralized data and processing architectures. For some applications, this is acceptable. However, with the ever changing and growing amount of image data today, a centralized architecture does not work well due to a variety of bottlenecks that occur in

the storage or processing. As an alternative to this centralized approach, agent technology was chosen. Agents provide a novel approach for a variety of reasons. First, agents use flexible peer-to-peer communication and control topology to communicate with one or several other agents, not just to a client or a server. In addition, agents in our architecture send encapsulated messages to each other through a blackboard coordination model, which allows asynchronous communication. Furthermore, agents communicate using a higher-level agent communication language, as opposed to lower-level protocols of traditional technology. Finally, agent technology provides a means to easily distribute lightweight agents on different machines thereby enabling a progression from a more centralized to a distributed architecture.

### Architecture

The overall architecture of our system is an integration of the three main components: software multi-agent technology, geoconformance modeling, and image analysis. A high-level outline of the architecture is shown in Figure 1. For the purposes of this paper, we are going to focus exclusively on the multi-agent component of this architecture. The multi-agent component is responsible for retrieving, storing, and analyzing spatial data from the Internet about the image. In this architecture, agents search for images on the Internet. They store these images and the corresponding metadata. Finally, they create vectors for image analysis and retrieval.

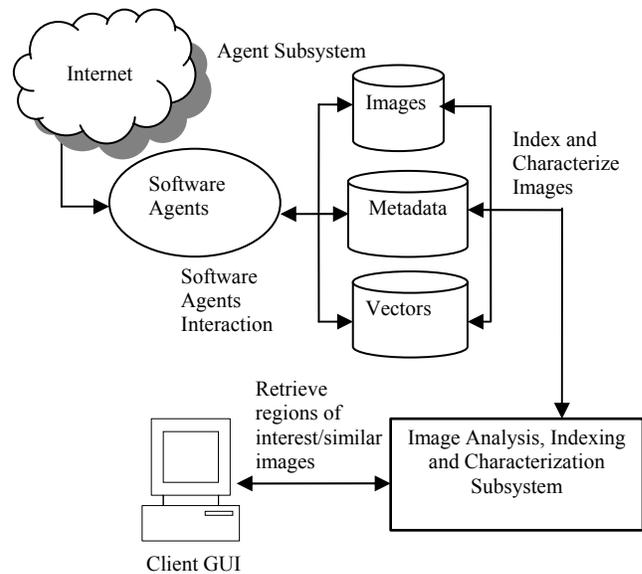


Fig 1. High Level Architecture Overview

Since our approach is based on multi agent technology, we look at our agent development tool in the next section, followed by the agent framework.

## Agent Development Tool

All of the software agents in this system were developed using the Oak Ridge Mobile Agent Community (ORMAC). Figure 2 shows a conceptual view of how ORMAC works. The ORMAC framework allows execution of mobile, distributed software agents, and establishes communication among them. The ORMAC framework provides a peer-to-peer communication and control topology (i.e., one agent can communicate with one or several other agents). This messaging approach provides the ability for communication that is encapsulated and asynchronous with a blackboard coordination model. Messages are passed to a blackboard, and agents that are subscribed to the blackboard receive the messages (Weiss 1999), (Huhns and Singh 1999). ORMAC enables an agent community to be quickly created using a set of machines with each machine executing the ORMAC agent host software. The ORMAC agent host software allows agents to migrate among machines. The ORMAC framework uses the Foundation for Intelligent Physical Agent (FIPA) compliant agent communication language (ACL) messages. This allows any FIPA compliant agent, such as SPAWAR's TIHERA system, to be able to interact with an ORMAC agent (Potok et al. 2003). Within the ORMAC community, each agent host is registered with a name server responsible for tracking where agents are currently being hosted (Reed, Potok and Patton 2004).

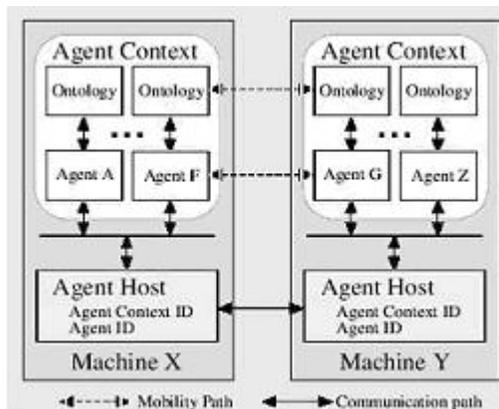


Fig 2. Oak Ridge Multi-Agent Community (ORMAC) framework

## Agent Framework

To develop the image retrieval and management system, the ORMAC framework was used to build a multi-agent system. This system consists of several types of agents: Crawler agent, Download agent, Markup agent, and Extractor agent.

The Crawler agent performs a depth-first search for image links on potential websites (in our case, only URL's ending with .edu, .gov and .net). Najork et al in their paper describes a similar web crawler that is scalable and extensible (Najork and Heydon 1999). When an agent receives a URL to crawl as input, it looks for all potential URL's in the page. However since it does a depth first search, it recursively keeps crawling down every link that it traverses. A depth value is used as a cutoff point to stop the crawling process. At this point, it has a collection of all links that it just crawled.

The next type of agent is the Download agent. This agent downloads images for all the image links generated by the Crawler agent. An element of intelligence is added to the download agent. Before downloading, the Download agent coordinates with the image repository to ensure that the image is not already available. This is done by generating a bounding box of the image to be downloaded. If there is an image in the repository with the same bounding box coordinates, then an image from the same area is already present in the repository. The bounding box coordinates are the coordinate pair values of the four corners of a box that completely bounds the image. If the image is already present in the repository, then the agent checks to see if the image is newer or of a higher resolution than what is already present. To check for newer images the image compares the timestamp on the image URL with the timestamp of the image in the repository. In addition, most spatial data allow extraction of image resolution from the image. This allows the agent to compare the resolution of an image in the repository with an image being downloaded. Therefore, if the image does not already exist in the repository, or if the image is newer or has a higher resolution than the existing one, then the agent downloads the image to the repository. There are a lot of images on the internet from small icons to personal photographs and high resolution satellite images. The download agent checks the size of the images and if it is greater than 2 mega bytes, the image is downloaded. Also most satellite and ortho images have an accompanying metadata file called the world file. If a world file is found along with the image file the image is downloaded with certainty that it is spatial data.

The third type of agent is the Markup agent. This type of agent creates XML files that have images marked up with their properties and metadata (Potok et al. 2002). For each image in the repository, this agent extracts image properties like height, width, bit planes, etc. In addition, this agent extracts geospatial information like the images bounding box coordinates from the accompanying metadata/ world file. After collecting this information, it creates an XML file for each image in the image repository using all of the above-deduced properties. These XML files are stored in a separate XML Repository.

Finally, an Extractor agent performs preprocessing of the images (Gleason et al. 2002). First, this agent monitors the XML repository for new XML files. As new XML files become available, it then begins processing the corresponding image from the image repository. The image is first segmented into 64 X 64 blocks segments. Once the segments are created, a vector file describing each segment is created by making use of the image properties in the XML file. The vector files generated are stored in the Vectors repository.

Figure 3 describes the process and information flow among the four agents:

1. The Crawler agent spiders the Internet looking for new image links.
2. When the Crawler agent has completed crawling, it sends a message to the Download agent containing a set of image links to the download agent.
3. The Download agent downloads these images to a data repository. The repository can either be local to the agent or a central repository as will be described later.
4. The Markup agent monitors the repository and every time a new image is downloaded, it creates an XML metadata file, which contains tags with image properties. The metadata file is stored in an XML repository.
5. The Extractor agent monitors the XML repository and every time a new XML file is created, it reads it and the corresponding image and creates vectors that characterize the image. The vectors are stored in a vectors repository.

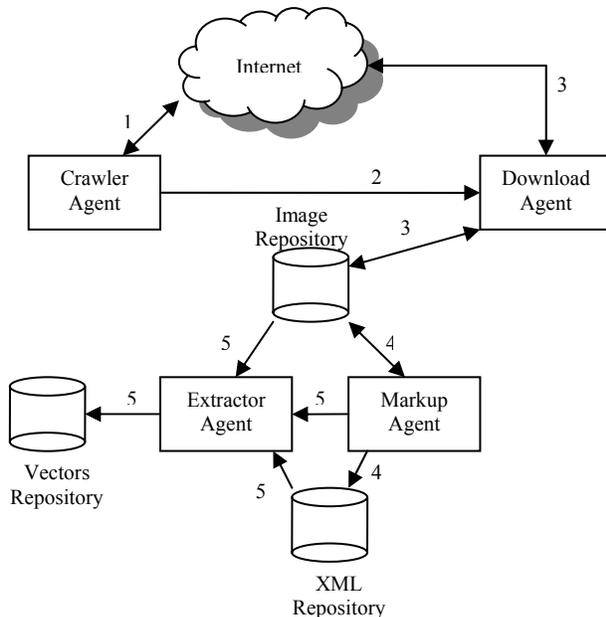


Fig 3. Process Flow Overview

### Data Repository

Two variations of the multi-agent architecture were developed and analyzed to address the problem of managing vast amounts of images in a data repository. The first approach uses a centralized repository as shown in Figure 4. In this architecture, images are downloaded by the download agents and brought into a central repository. The Markup and extractor agents work on these images reading them from the repository and creating metadata files and vectors.

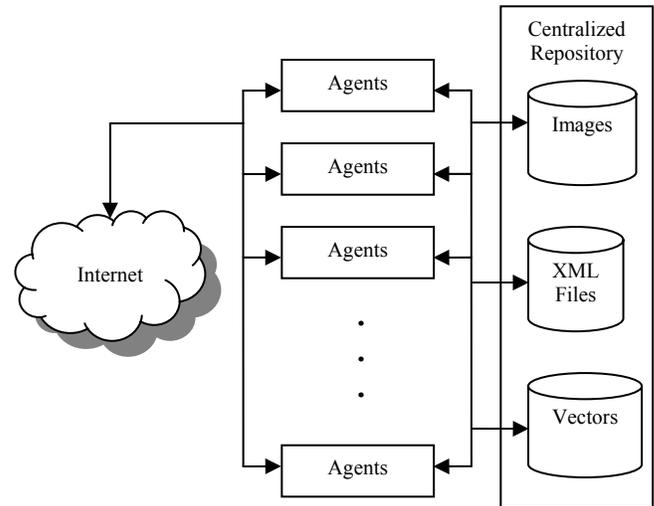


Fig 4. Centralized Repository Architecture

In the second approach, data are stored on separate machines where agents can process the images, as shown in Figure 5.

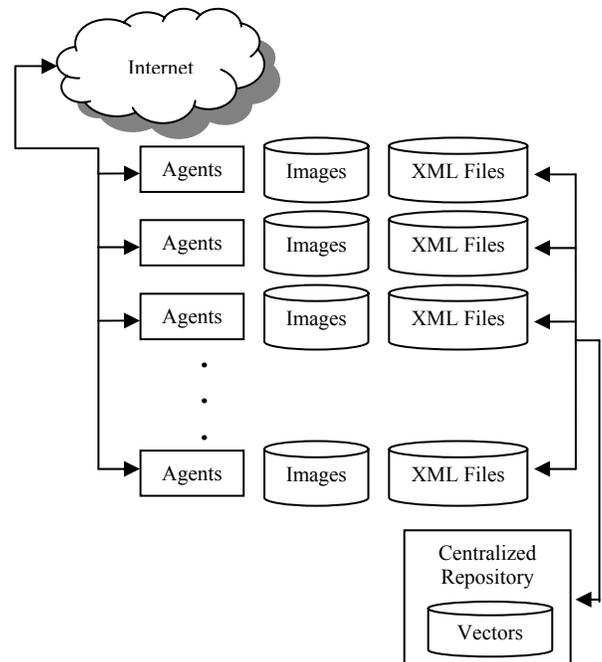


Fig 5. Distributed Repository Architecture

## Results & Analysis

The agent framework for image retrieval and analysis was tested against a set of images from the Oak Ridge Reservation area. A web server hosting eight, 32-megabyte images from the Oak Ridge Reservation area was established. The images are in the .tiff format and represent quadrangles from the reservation area. To demonstrate the scalability and advantages of a distributed system, we ran the multi agent system on 1, 2, 3, 4, 5 processors to compare the times take. One run included crawling, downloading the images, creating XML files and then generating vectors.

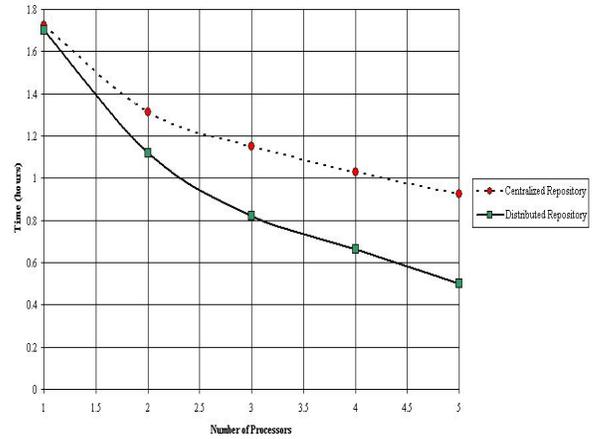
The two architectural approaches were tested. In the first, images are downloaded to a central repository and then distributed to the different processors. In the other approach, images are downloaded directly to a processor and are processed upon locally. The times taken between the two were compared and contrasted. The PCs used are 2 gigahertz and 512 mb RAM.

Table 1 shows the time taken for a single run on 1, 2, 3, 4, 5 processors in the centralized and distributed approaches. Figure 6 shows a graph representing the speedup.

**Table 1.** Times (in hours) for the centralized and distributed approach

Processors	Centralized	Distributed
1	1.725	1.701
2	1.311	1.121
3	1.150	0.820
4	1.029	0.662
5	0.924	0.502

The results show a speedup factor of about 100% when incorporating a distributed processing and storage architecture. In addition, the system is very scalable as the number of processors can be increased with a minimum of effort. The distributed architecture is more reliable with the data being distributed among different machines.



**Fig 6.** Times (in hours) vs. number of processors

## Future Work

This architecture is further being researched in two areas. The first is in trying to reduce data transfer overhead while moving images and metadata. Figure 7 shows how the two architectures vary from the ideal performance curve. This is due to the I/O time it takes for moving large images and their accompanying data around. The ideal curve is determined assuming it would take time proportional to the number of processors.

Another area of research is to break the image into an equal number of pieces depending on the number of processors. A separate agent processes each sub image. This would speed up the overall processing time by almost a factor equal to the number of processors. It would be ideal in a situation where there are not too many new images, but it is essential that the final vectors be generated as quickly as possible.

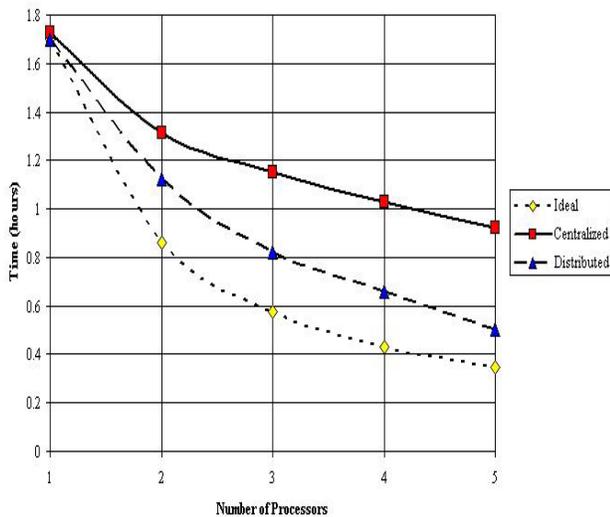


Fig 7. Comparison of the architectures to the ideal state

## Conclusion

Image and remote sensing information is of critical importance to the scientific community and for national security purposes. Staggering increases in the volume of remote sensing image data are being produced and made widely available. We have focused on two key issues in processing this type of data. 1) Retrieving and updating this vast amount of data, and 2) Automating the management process.

The current work in the field has a strong focus on managing and performing analysis on a closed set of images. Many of the proposed techniques do not scale well, and do not consider how new images will be found and incorporated into the system. We have developed a system that uses software agents to gather new images from the internet, update image archives using “better” images, and to parallelize aspects of the image region and feature extraction process. We have been able to demonstrate the ability to manage vast amounts of image data, and to automate the manual processing of this data. This is a significant step forward in image mining and management.

## References

United States Geological Survey. 2000, “<http://vangogh.cse.buffalo.edu:8080/digital/digital/node4.html>”, Multimedia Research Group, Department of CSE, SUNY at Buffalo.

Christopher, T., William, M., October 1. 1995, “*Satellite Image Dissemination via Software Agents*”, IEEE Intelligent Systems 10(05) p.44-51.

MacLeod, Ian N., Amorim, R., Valleau, N. 2000, “*DAP – Large Volume spatial data discovery and distribution over networks*”, Geosoft Inc.

Weber, R., Bolliger, J., Gross, T., Schek, Hong J. 1999, “*Architecture of a Networked Image Search and Retrieval System*”, Eight International Conference on Information and Knowledge Management, Kansas City, Missouri, USA.

Zhu, B., Ramsey, M., Ng, Tobun D., Chen, H., Schatz, B. 1999, “*Creating a Large-Scale Digital Library for Georeferenced Information*”, D-Lib Magazine, Volume 5, Number 7/8.

Nolan, James J., Sood, Arun K., Simon, R 2000, “*An Agent-based Architecture for Distributed Imagery & Geospatial Computing*”, George Mason University.

Weiss, G. ed. 1999, “*Multiagent Systems*”, The MIT Press, Cambridge, Massachusetts.

Huhns, M., Singh, M. 1999, “*Readings in AGENTS*”, Morgan Kaufman Publishers, San Francisco, California.

Potok, T., Elmore, M., Reed, J. and Sheldon, F.T. 2003, “*VIPAR: Advanced Information Agents Discovering Knowledge in an Open and Changing Environment*”, Proc. 7th World Multiconference on Systemics, Cybernetics and Informatics Special Session on Agent-Based Computing, Orlando FL, pp. 28-33.

Reed, J.W., Potok, T.E., and Patton, R.M. 2004, “*A Multi-Agent System for Distributed Cluster Analysis*”, 3<sup>rd</sup> International Workshop on Software Engineering for Large-Scale Multi-Agent Systems.

Najork, M., Heydon, A. 1999, “*Mercator: A Scalable, Extensible Web Crawler*”, Compaq Systems Research Center.

Potok, T., Elmore, M., Reed, J., and Samatova, N. 2002, “*An Ontology-based HTML to XML Conversion Using Intelligent Agents*”, *Proceedings of the Hawaii International Conference On System Sciences*.

Gleason, S., Ferrell, R., Karnowski, T., Tobin, K. 2002, “*Detection of semiconductor defects using a novel fractal encoding algorithm*”, Design, Process Integration, and Diagnostics in IC Manufacturing, Proceedings of the SPIE Vol. 4692.