CACR Technical Report

CACR-186 **March 2000**

Mobile Agent on the SARA Digital Library

Yanyan Yang, Omer F. Rana, David W. Walker, Christ Georgousopoulos,
Roy Williams

Mailing Address: CACR Technical Publications, California Institute of Technology,
Mail Code 158-79, Pasadena, CA 91125. Phone: (626) 395-6953 Fax: (626) 584-5917

Mobile Agent on the SARA Digital Library

Yanyan Yang, Omer F. Rana, David W. Walker, Christ Georgousopoulos

Department of Computer Science,

Cardiff University, PO Box 916,

Cardiff CF24 3XF, UK

email: (l.yang, o.f.rana, david.w.walker, C.Georgousopoulos)@cs.cf.ac.uk

Roy Williams

Center for Advanced Computing Research

California Institute of Technology, CACR 158-79

Pasadena, CA 91125, USA

roy@caltech.edu

Abstract

Remote-sensing data about the Earth's environment is being created at an ever-increasing rate and distributed among heterogeneous remote sites. Traditional models of distributed computing are inadequate to support such complex applications, which generally involve a large quantity of data. In this paper, we explore an approach based on mobile agent techniques for autonomous data process and information discovery on the Synthetic Aperture Radar Atlas (SARA) digital library, which consists of distributed multi-agency archives of multi-spectral remote-sensing imagery of the Earth. Our goal is to enable automatic and dynamic configuration of distributed parallel computing and to efficiently support on-demand processing of such a remote-sensing archive. The architecture design and implementation status of the prototype system , which applied the approach, is reported here.

Keywords: *Digital Libraries, Mobile Agents, Parallel and Distributed Computing, XML, Remote Sensing*

1. Introduction

The explosive growth in remote-sensing data has generated an urgent need for new techniques and access mechanisms that can intelligently and automatically process distributed data, and transform the processed data into useful information and knowledge. Efficient support for scientific collaborations for both professional and casual users is also needed within this domain. Among many different paradigms and architectures of distributed computing systems for a remote-sensing archive [Williams98, James98, Coddington 98,Wims 99], the mobile agents paradigm appears to be the most promising due to the following reasons:

The Work presented in this paper was supported in part by the UK CLRC (Central Laboratory of the Research Councils.

- Mobile agents are programs that encapsulate data and code, which may be dispatched from a client computer and transported to a remote server for execution [Chess93, Harrison95, White96]. When large quantities of remote-sensing data are stored at distributed remote hosts, moving the computations to data is a more realistic and feasible approach, compared with migrating data to the computations. Instead of gathering data distributed in remote sites at a centralized site, users can dispatch mobile agents to a destination site to perform information retrieval and filtering locally, and to return to a user the results of analysis. Thus, the information transmitted over the network is minimized, especially when using a low-bandwidth access network. A theoretical analysis of the trade-off between mobile agent migration and the remote procedure call paradigms can be found in [Straoer 97].
- Mobile agents execute asynchronously and autonomously. Once a user has created an agent, it can run autonomously and asynchronously, without intervention from the user. The agent performs its task and saves any results until its connection to the user is re-established. Mobile agents provide a reliable transport between a client and server without necessitating a reliable underlying communications medium.
- Mobile agent-based transactions are also robust and flexible. They can react autonomously to changes in their execution environment, and are therefore more flexible in their operation. Multiple mobile agents have the ability of distributing themselves among the hosts in the network to maintain the optimal configuration for solving a particular problem.

The above observations motivate us to study the mobile agent paradigm to retrieve, process, fuse and mine distributed Synthetic Aperture Radar Atlas (SARA) remote-sensing data. In this paper, we report on the extended design of SARA digital library for a mobile environment and its initial implementation. The goal of this paper is twofold. First, we give a general overview of the architecture of SARA digital library and of the important concerns of our design. Second, we focus on our system support for autonomous and dynamic data process and information discovery based on agents collaboration and show how our scheme is in compliance with the general architecture and design concerns.

The rest of this paper is organized as follows. In section 2, we introduce the SARA digital library, present a mobile agent-based architecture for on-demand processing of the remote-sensing archives, and introduce the main considerations of the design. Section 3 describes our scheme for information discovery and dissemination on the SARA digital library. The novelty of this scheme is the existence of two types of agents that enable automatic and dynamic configuration of distributed parallel computing. In section 4 we report on the implementation status of our system. We conclude in section 5 with remarks on the future work as well as other related work in brief.

2. Overview and Architecture of System

SARA is a digital library of multi-spectral remote-sensing imagery of the Earth, which provides web-based online access to a library of data objects at Caltech, the San Diego Supercomputer Center, and the University of Lecce in Italy [Williams98]. SARA is freely available from the Internet [SARA]. A user may use SARA's specialized web browser software to view and initiate supervised processing on Synthetic Aperture Radar images from the archives.

As the knowledge economy broadens, this old-fashioned model applied in the first version of the SARA digital library has become increasingly inadequate. Our approach is based on the use of Java-based mobile agents. As mentioned in section 1, mobile agents offer an important technique for performing transactions in a distributed computing system. We propose to deploy Java-based mobile agents between a client and the SARA server, and use XML (eXtensible Markup Language) for encoding agent communication. Java's platform independence, object serialization, multithreading, remote method invocation, secure execution, and dynamic class loading are essential properties for implementing a mobile agent system. XML is an industry-standard mechanism for tagging information between heterogeneous applications [XML]. XSIL [XSIL] is based on XML, and used to represent collections of scientific data objects, which can either be small objects with data explicitly contained in a file, or large objects represented by the salient metadata, with references to binary files elsewhere. XSIL is aimed at supporting, (1) Digital Puglia Synthetic Aperture Radar Atlas, an archiving and processing facility for knowledge discovery in remote-sensing databases, (2) Digital Sky, a prototype confederation of astronomical surveys, (3) Center for Simulation of the Dynamic Response of Materials, a multi-disciplinary consortium at Caltech for simulations at multiple scales, and (4) Interferometric SAR digital library, a facility to improve the usability of the Synthetic Aperture Radar Atlas.

We extend the web based client-server model of SARA digital library to include agents that allows users to dispatch their tasks as mobile agents and that can autonomously migrate from one machine to another according to a predefined itinerary. In mobile agent based SARA, once a user has created the agent, it can securely move with its data, code and state to an itinerary of servers that may have relevant data and services. After being dispatched, the agents become independent of the user that created them, they can survive intermittent or unreliable network connection. The agents can connect to the server in native language and native protocol and perform information retrieval, dissemination and mining etc. SARA mobile agents are persistent, which can wait for resources to be available. Finally, the agents come back to the user with the results. The mobile agents based SARA digital library provide a sophisticated view in which users can extract useful knowledge rather than simply retrieve; where the process can be dynamic, incremental and constrained by resource limitations.

Figure 1 illustrates the general architecture of the SARA remote-sensing library. The SARA architecture has been re-designed to fulfill several important goals as follows:

- Modularity. The system component should be composed of interchangeable modules, each providing some of the required functionality. For example, Local Retrieval Agents (LRAs) can translate a query task and retrieve information from a local archive, which could be a database system or a file system. If the local archive system changes, we only need to change the LRA.
- Scalability. The system component must deal with a large number of requests coming from many users simultaneously. As the load grows, the system should scale gracefully. For example, in our architecture, the Local Management Agent may intelligently assign computation resources.
- Decentralization. The system should be open and evolving. There should be no global administrator agent, because agents are submitted from various remote sites and it would be inefficient to route all agents through a central site. Control is therefore distributed.
- Extensibility. The system components should allow new elements, such as new services and new archive systems to be easily added.

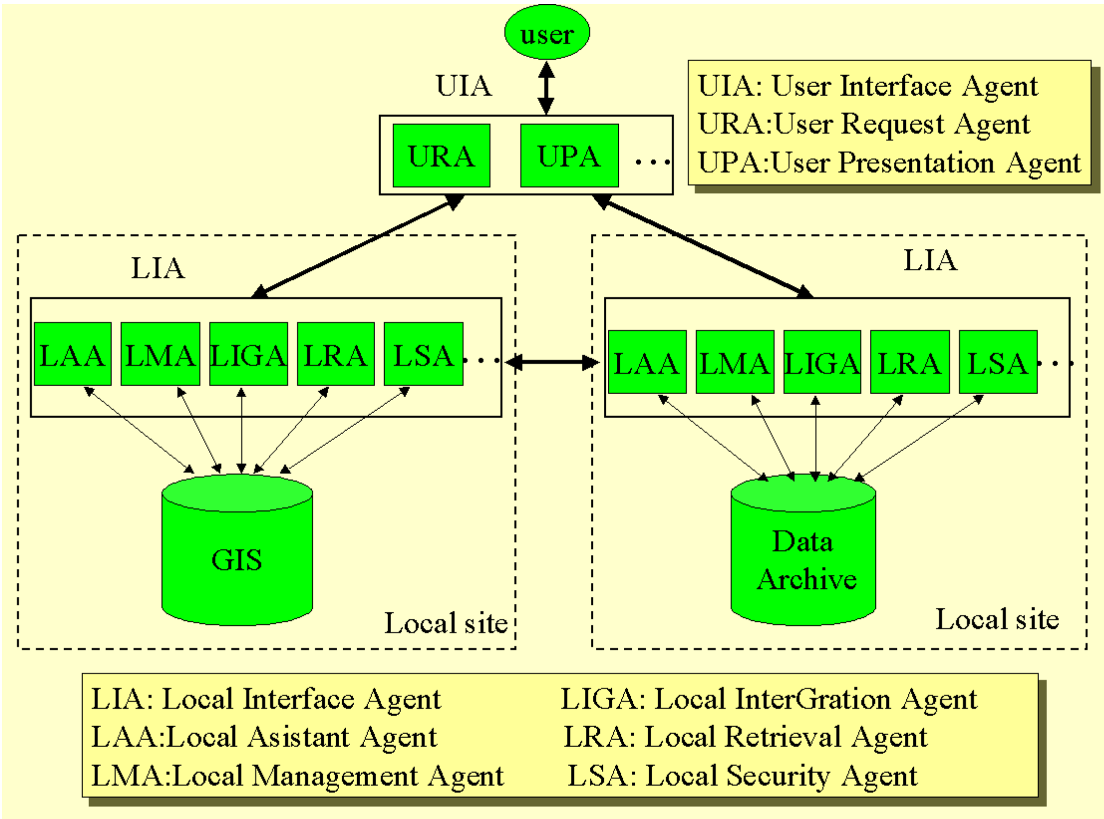

Figure 1 Mobile Agent-Based Architecture

As illustrated in Figure 1, agents in the SARA system can be divided into two main types:

1. User Interface Agents (UIA) provide a front end to the end user, for checking user input and displaying results. There are two kinds of UIA:
   - A User Request Agent (URA) is responsible for holding a user's request, and carrying it to the appropriate local archive sites, interacting with a LIA at each remote site visited, and either returning back to the user site with the results of analysis, or with a URL reference.
   - A User Presentation Agent (UPA) enables the visualisation of results according to the user's choice and environment.
2. Local Interface agents (LIA) provide an extensible set of services, of which there are many types:
   - A Local Assistant Agent (LAA) supports interaction with any visiting URAs and informs them about the available data and computing resources, and supports the completion of the task carried by the URA.
   - A Local Management Agent (LMA) coordinates access to other LIA and support negotiation among agents.
   - A Local InteGration Agent (LIGA) provides a gateway to a local workstation cluster, or a parallel machine.

- ➢ Local Retrieval Agent (LRA) can translate query tasks and performs real-time information retrieval from the local archive.
- ➢ Local Security Agent (LSA) is responsible for authenticating the incoming URA.

As the system is developed, specialized agents will be added to the system as needed. For example, we can create a user profile agent, which is of type UIA, to manage the profile of the user and publish useful information for use by the appropriate agents. We can also create a knowledge mining agent, of type LIA, to extract useful information from local sites.

Metadata of the system are replicated on every server. Metadata contains the available service in each server and the status of the connections of each server etc. Metadata maintains also index of the data that results from a time consuming query involving significant processing and that results from intraday and up-to-day query tasks. The data are cached on the corresponding server. Thus the same task that requires those results can execute much more quickly.

3. Collaboration Scheme of Mobile Agents

In this section we describe in detail collaboration scheme of the two types of agents in SARA digital library. Figure 2 illustrates a simple example of query processing and show the interaction and collaboration of the agents.

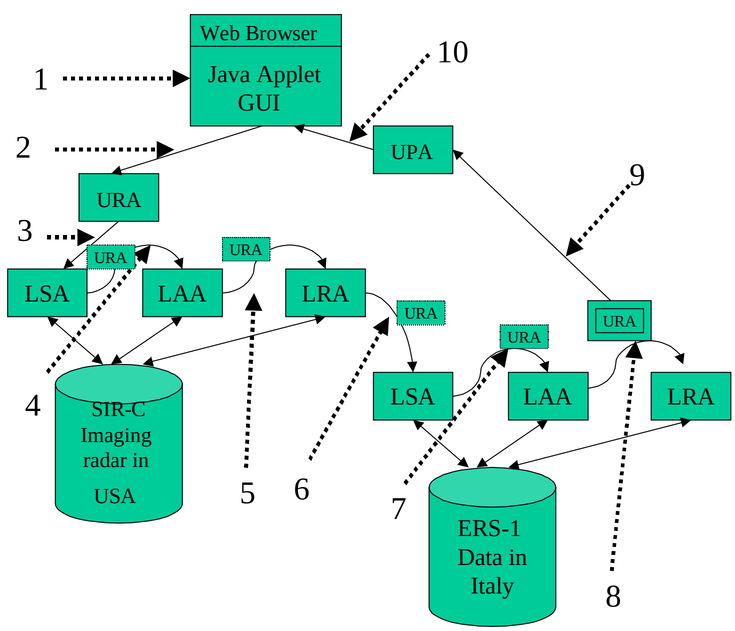

Figure 2 Transfer processing of the query

As shown in figure 2, the user downloads a web page contained a mobile agent from a Java-enabled web browser (Step 1) and then initiates and dispatches a URA (Step 2). The URA then takes a user's query request encoded in XML to the first stop of its itinerary – SIR-C imaging radar in USA and interacts with the LSA (Step 3). After authenticating the sending user by LSA, the URA interacts with the LAA to find out about the available data source and services (Step 4). Once the required resources in the local sites are available, a URA communicates with the LRA, and the LRA completes the retrieve task at the local sites, saves results to a file, then the URA creates a URL that points to that file(Step 5). The URA then dispatches itself to the next stop of its itinerary and communicates with the agents of the next site – ERS-1 data in Italy (Steps 6, 7, 8). Finally, the URA communicates with the UPA and returns URLs pointing to the results files to UPA when the itinerary is completed, then URA dies (Step 9). UPA then presents the results to the user through the graphical interface according to the rules defined in the UPA (Step 10). In this way, a user can connect to the network at a low bandwidth while the data archive is connected to a powerful computer server over a high bandwidth network connection, and the user can also engage in other tasks or shut down the machine after dispatching the URA.

In SARA mobile environment, each URA contains an itinerary, in which each itinerary element consists of a task, the destination server, and the reaction to task success and failure events. Each server has its own LMA. URAs arriving at a server check in with the LMA before starting work, giving the LMA its itinerary. The LMA examines the coming URA's plans, and may return a possibly altered itinerary and redirect the agent to another server where the result data has been stored. URA check in again before they leave the server. The LMA add the corresponding index to metadata. LMA is also responsible for monitoring the status of the other servers and informing the URA hosting on local server when other server is unavailable or available again.

For example, URA1 has the following itinerary : visit server Sl, perform task Tl , and result data are stored in S1; then visit server S2, and perform task T2, visit server S3 and perform task T3, the merging result data of T2 and T3 are stored on S3. Next, suppose that after URA1 performs its work, URA2 is sent out with the itinerary: visit server S1 and perform task T4, visit server S2, perform task T2 ; then visit server S3, and perform task T3. When URA2 arrives S1, LMA1 on S1 will recognize the results of T2 and T3 can be found at S3. Rather than send URA2 to S2, the LMA1 changes URA2's itinerary so it need not visit S2 and S3 and just return back with URL reference on S3, or it just has to visit S3 and return back with the result data.

The above information sharing and agents collaboration scheme reduces the number of costly server requests as well as enable automatic and dynamic configuration of distributed parallel computing.

4.Status of Implementation

Mobile agents based SARA digital library is in its early stages of development. The system is being developed in the Java programming language, using the Voyager mobile agent platform from ObjectSpace Inc.[Voyager]. Voyager is chosen because it supports CORBA services, JDBC, transaction, security and also a number of different types of collective communication. Especially, Voyager 3.1 provides direct support for XML. The basic functions of URA, LAA, LRA and UPA described in the previous section have been implemented.

The following IUra interface enables clients to create agents and send them to corresponding server.

```
import java.sql.*;
import java.net.*;
public interface IUra
{
 URL travel(String client_ip,String sql_query);
 void execute(String client_ip,String sql_query,String host1a);
 String resolve_hostname(String hostname);
 void save_errors(String errors,String client_ip);
}
```

The **travel** method enables the Ura agent to move itself to a server by loading its class from the client. The **execute** method starts the execution of the agent object: contacting the Laa agent, which creates a JDBC connection with the server's database, then contacting the Lra agent, which executes the client's SQL statements. The **resolve_hostname** method resolves a host name to corresponding IP address. The **save_errors** method saves the errors information received by the 'errors' parameter in the client's file specified by 'client_ip' parameter.

The ILaa interface declares three primary abstract methods connect(), connection_close(), and save_errors(). The **connect** method creates a connection with local server's database. The **connection_close** method closes the connection specified by the 'Connection' parameter. The **save_errors** method accomplishes the similar function as in IUra.

```
import java.sql.*;
public interface ILaa
{
 Connection connect(String client_ip) throws ClassNotFoundException;
 void connection_close(Connection connection);
 void save_errors(String errors,String client_ip);
}
```

The ILra interface declares methods for executing the client's SQL query and saving the results to a file whose name specified by the 'client_ip' parameter.

```
import java.sql.*;
import java.io.*;
public interface ILra
{
```

Upa provides methods for synchronous or asynchronous communication between Ura, user and servers. The Upa is able to create the GUI, start a Voyager server as a client, enable this Voyager server to be able to serve other agents with any resources that it can load, and create a proxy for the Ura agents. The proxy of the Ura agents is named as: client's host+address+port+recent time (with milliseconds). The client's filename that holds the client's results has the same name as the Ura agent so conflicts between different client's files or different Ura agents on the same server are going to be avoided . Each name is unique. This will be useful in the future where a lot of Ura agents launched from a lot of different clients are going to reside in the same server. The Upa contacts the Ura by calling its 'travel' method. After the 'travel' method has been executed, the Upa receives a URL address (sending by the Ura) where the client's results will be retrieved. A basic implements of the Upa is as follows.

```
  public class Upa extends Frame
  {   ……
     public static void main(String[] args)
     { try
       {Voyager.startup();
        ClassManager.enableResourceServer();     }
       catch(Exception e)   { ……  }
      (new Upa()).setVisible(true);
     }
    public Upa(){……}
    public Upa(String title){……}
    public void setVisible(boolean b){……}
    public void addNotify(){……}
    void button2_ActionPerformed(java.awt.event.ActionEvent
event){……}
    void button3_ActionPerformed(java.awt.event.ActionEvent
event){……}}
```

The current prototype was presented as a proof of concept and has not been deliberately refined. A full-fledged implementation of mobile agent based SARA digital library is expected to be released in the end of 2000.Figure 3 and figure4 illustrates a query example in the prototype system.

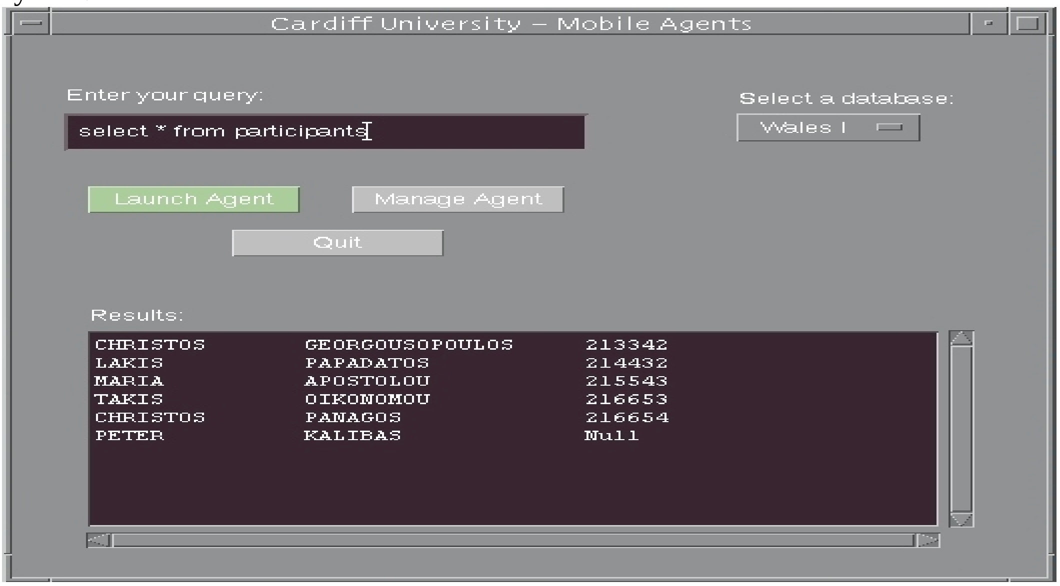

Figure3 Graphical user interface

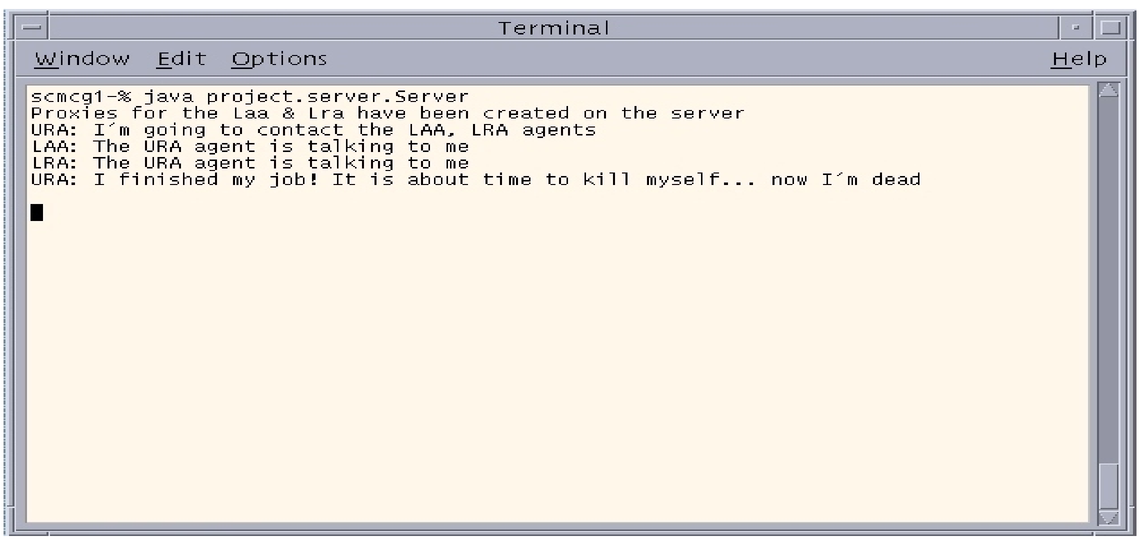

Figure 4 Server console

5. Conclusions

Remote-sensing data about the Earth's environment is being created at an ever-increasing rate and distributed among heterogeneous remote sites. Traditional models of distributed computing are inadequate to support such complex applications, which generally involve a large quantity of data. Some research work has been focused on technologies to support agent-based models for distributed applications [Papastavrou98, Rasmusson98, Hamard99]. In this paper, we have described the extended design of SARA digital library based on mobile agent, which is a distributed digital library of multi-spectral remote-sensing imagery of the Earth. The contribution of this paper is twofold.

First, we propose a general mobile agents based architecture, which allows distributed access, concurrent querying, and parallel computing over multiple and heterogeneous remote-sensing archives in a modular and scalable fashion. The main concerns in designing for a mobile environment are identified. We show how the functionality required for on-demand processing of remote-sensing archives can be decomposed into different classes of agents to achieve our desired goals. The SARA can grow while adding new remote-sensing archives or encapsulating new local services within an agent.

Second, we have shown how these general principles can be used to provide system support for automatic and dynamic configuration of distributed parallel computing based on mobile agents collaboration.

Finally, we have briefly reported on the status of our system.

References

[Aloisio97] Aloisio, G., Cafaro, M. , Kremenek, G., Williams, R.D. , Messina, P. A Distributed WEB-Based Metacomputing Environment, Proc. HPCN Europe 1997.

[Chess93] Chess, D. ,Grosof, B., Harrison, C. ,Levine, D., Parris, C., Tsudik ,G., Itinerant Agents for Mobile Computing. Journal IEEE Personal Communications, Vol. 2, No. 5, October 1993.

[Coddington 98] Coddington, P.D., K.A. Hawick, K.E. Kerry, J.A. Mathew, A.J. Silis, D.L. Webb, P.J. Whitbread, C.G. Irving, M.W. Grigg, R. Jana, K. Tang, Implementation of a Geospatial Imagery Digital Library using Java and CORBA . Available from web site http://www.cs.adelaide.edu.au

[Hamard99] Hamard, K., Nie J.,Bochmann and G., Godin, R. et al.., A Multi-Level Agent System for Digital Libraries, See web site at http://www.iro.umontreal.ca

[Harrison95] Harrison, Colin G., Chessm, David M., Kershenbaum, Aaron. Mobile Agents: are they a good idea? Research Report, IBM Research Division.

[James98] James H.A and Hawick K.A., A Web-based Interface for On-Demand Processing of Satellite Imagery Archives, In Chris Mcdonald, editor, Proc. 21st Austraasian Computer Science Conf. ACSC'98, volume 20 of Australian Computer Science Communictions. Springer-Verlag Pte Ltd, Februry 1998

[Nelson99] Nelson, J., Programming Mobile Objects with Java, John Wiley & Sons, Inc, 1999

[Papastavrou98] S. Papastavrou, E. Pitoura, G. Samaras. Mobile Agents for WWW Distributed Database Access. (extended version). Technical Report TR 98-12, Univ. Of Cyprus, Computer Science Department, Sept. 98.

[Rasmusson98] Rasmusson, A & Olsson, T & Hansen, P. 1998. A Virtual Community Library: SICS Digital Library Infrastructure Project. Research and Advanced Technology for Digital Libraries. Second European Conference, ECDL'98, Heraklion, Crete, 19-23 September, Proceedings. Lecture Notes in Computer Science, Vol. 1513. pp 677-678. Springer-Verlag.

[SARA] SARA, the Synthetic Aperture Radar Atlas. See web site at http://www.cacr.caltech.edu/sara

[Straoer 97] Straoer, M. and Schwehm, M. A Performance Model for Mobile Agent Systems. Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications PDPTA'97, Volume II, (Las Vegas NV, 1997), 1132-l 140.

[Voyager] Voyager 3.1, ObjectSpace Inc. See web site at : http://www.objectspace.com

[White96] White, J. E.. Mobile Agents. General Magic White Paper. See web site at http://www.genmagic.com/agents, 1996.

[Williams98a] Williams, R. , Bunn , J., Moore, R., Interfaces to Scientific Data Archives, Report of a Workshop sponsored by the National Science Foundation, May 1998. See web site at http://www.cacr.caltech.edu/isda

[Williams98b] Williams, R.D. , Sears, B., A High-Performance Active Digital Library, Parallel Computing, Special issue on Metacomputing, November 1998

[Wims 99] Wims, B. and Xu, C., TRAVELER: A Mobile Agent Infrastructure for Wide Area Parallel Computing, In Proc.of First Joint Symposium : Int. Symp. on Agents and Applications (ASA'99) and Third Int. Symp. on Mobile Agents (MA'99), October 1999.

[XML] XML, Extensible Markup Language. See web site at http://www.xml.com

[XSIL] Blackburn, K., Lazzarini, A., Prince, T. and Williams, R. XSIL: Extensible Scientific Interchange Language. HPCN'99, pp 513-524, Amsterdam, February 1999.