CACR Technical Report

CACR-184 **March 2000**

Web Access to Supercomputing Using the Grid

Giovanni Aloisio, Massimo Cafaro, F. Paolo Falabella, Carl Kesselman, 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

Web access to Supercomputing using the Grid

Giovanni Aloisio[*], Massimo Cafaro[*], F. Paolo Falabella[*], Carl Kesselman[+], Roy Williams[o]

Abstract

Earth Observation Systems (EOS) have provided over the years a tremendous amount of image data covering the surface of the Earth. These images are a potential source of information for lots of users working in various fields, like archaeology, geology, drawing of maps, ecology and others. In order to access this precious source of knowledge, the user needs to search for images of the zone of interest in a distributed database and to submit them to an elaborate processing to extract the required information. Various "static" approaches have been created to allow a user to search for an image with some specific requisites in the distributed database from a Java-enabled web browser. Our approach is "dynamic" because our system can autonomously make certain decisions in order to look for images and to process them according to a high-level request by the user. The construction of this system has been possible thanks to the Globus "computational grid" toolkit.

1. Introduction

To manage the tremendous amount of image data covering the Earth's surface that has been acquired during the years by remote sensing space missions, several Earth Observation Systems (EOSs) have been built by the National Space Agencies. Remote sensing imaging based both on traditional sensors and on more sophisticated radar technologies (SAR-Synthetic Aperture Radars) [1] can provide a potential source of information for lots of users working in various fields, like archaeology, geology, drawing of maps, ecology and others. It is worth noting that after several EOS missions dating since the early eighties almost the whole surface of the Earth has been remotely sensed.

A Dynamic Earth Observation System (DEOS) is made of a sensor mounted on a satellite or a space shuttle that gathers images of the Earth's surface, of a distributed database in which these images are stored and of heterogeneous distributed computing resources which process and distribute them via web to the final users (Fig. 1). Computing resources are needed to transform the raw data gathered by the sensor in an actual image. This preliminary step of processing is what is called *pre-processing* to distinguish it from the *post-processing* that may be needed to extract knowledge from the image. The real-time pre-processing of SAR raw data requires the use of high performance computing resources

While for a given sensor there's just one kind of pre-processing, different post-processing algorithms can be activated by the final users: some will give an output that is a transformed image, some will extract some quantitative information and return a graph or numbers.

The work presented in this paper was supported in part by ISUFI (Istituto Superiore Universitario di Formazione Interdisciplinare) and by CACR (Center for Advanced Computing Research.

[*] University of Lecce/High Performance Computing Lab, Lecce, Italy
[+] USC/Information Sciences Institute, Marina del Rey, CA
[o] Caltech/Center for Advanced Computing Research, Pasadena, CA

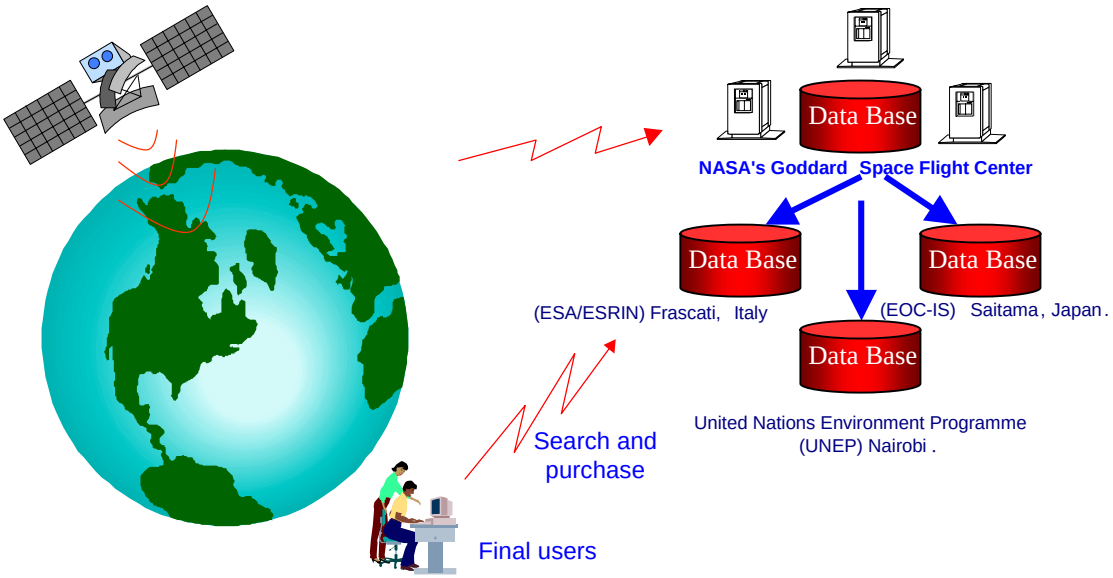

Fig.1 Earth Observation System

The traditional "Static" EOS systems allow a user to request a pre-processed image of a place[1] (if the required image is found in the distributed archive!) but they lack the intelligence needed to start a specific post-processing in response to high level user requests. On the contrary, a dynamic EOS should be able to manage high level requests like "show me the corn fields in the surroundings of that place" or "give me the image of that place, and find the computer that can post-process that image in the fastest way and with the lower cost".

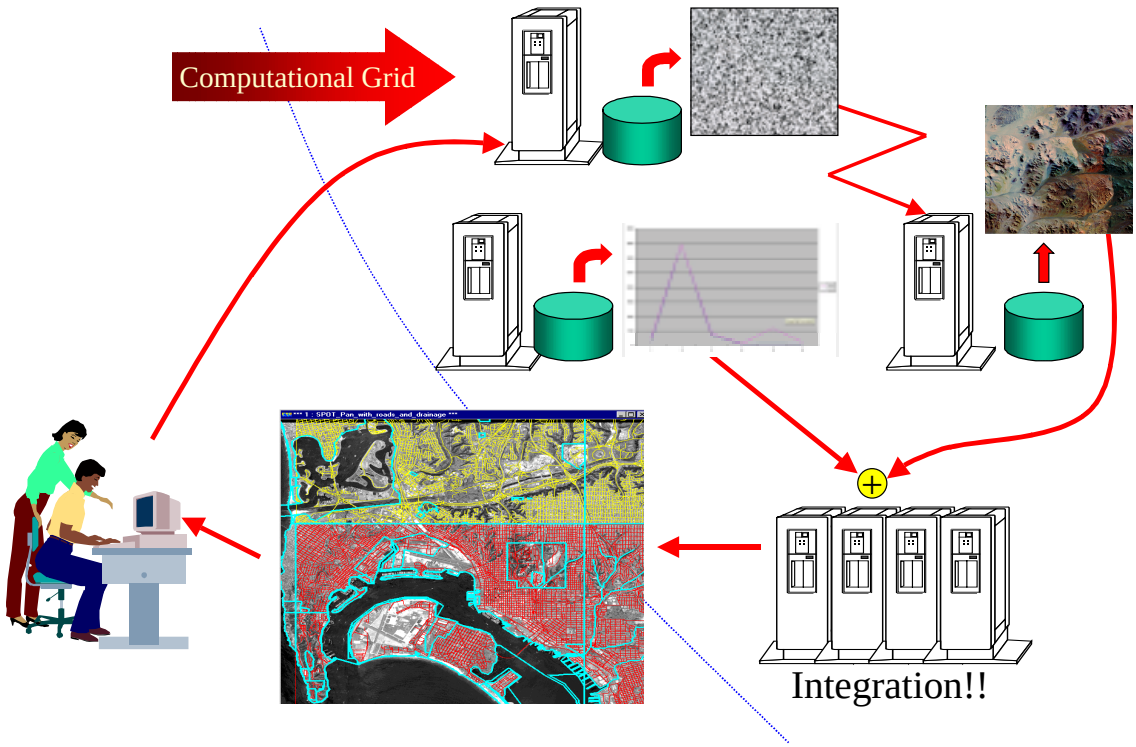

Fig.2 Data Integration

[1] In this paper we adopt the definition of the term "place" given in [2] to refer to a location of interest on or near the Earth's surface.

To satisfy these requests, a DEOS should have a flexible management and control of the distributed resources. Further challenges for a DEOS are represented by

- the integration of information coming from different sources eg. the integration of maps and images of the Earth's surface with any information associated with the geographic location;
- the integration of the functions of a friendly web browsing with those of GIS and related technologies (Fig.2).
- A DEOS must also provide a secure access to the distributed information and the steering of remote applications.

In this paper we describe a dynamic evolution of our system, SARA-Digital Puglia (Synthetic Aperture Radar Atlas-Digital Puglia Project), which implements a digital library of SAR images of the Salento zone in Italy that's been chosen as a testbed [3-6]. The extension of our project to the case of generic (i.e. not only SAR) EOS images is absolutely straightforward, as we're not interested in the part of acquisition of the images but in the following operations of storing, searching and distributing them which remain the same whatever the sensor is.

The purpose of the SARA/Digital Puglia dynamic system is to allow a user to choose from a web browser a geographic area of interest and to specify, using high level requests, the information to be extracted from the selected images. High level user requests usually require not only the retrieving of the image of the selected place but also further post-processing on it which must be transparently activated on some of the remote machines belonging to the "Computational Grid".

An intelligent resource management is needed to manage problems like:

- Is it worthwhile to move data to a remote machine for processing?
- how can the extracted information be delivered to the user in the fastest way?
- how to minimize the overall cost given a maximum user waiting time?
- how to minimize the waiting time given a maximum cost the user can afford?

The steering of remote applications via web should also be provided to allow the user an interactive control of the program in execution on the remote machine. Finally, the high level user request could also require the integration of information coming from different distributed sources.

To build such a dynamic EOS the web technologies were integrated with the Globus Computational Grid toolkit [8].

In section 2 the issues related to dynamic EOSs are presented, section 3 describes the static version of SARA while its dynamic evolution is described in section 4. Future developments and conclusions are in section 5.

2. Issues of a dynamic system

Several problems come out of the dynamic approach we're developing. The most important are the following:

1. *Search of the images*: we need to look for the images that suit the user's needs. That means we need to store some *metadata* about the images, saying what we need to know about the geographic location, the format (raw data, pre-processed, post-processed in some way) and the position in the distributed database.
2. *On-demand processing*: we want to let the system start processing if it's explicitly or implicitly required by the user. An implicit request is when the user asks for some information concerning a place that can be extracted by some form of processing on images of the selected place. To translate this high level request in a sequence of system level operations, the system is required to map the request with specific programs, to be able to find those programs on the machines of the computational grid and to start them remotely.

3. *Resource management*: this is a very general problem when dealing with Computational Grids. Besides the normal problems of managing the resources of as complex a system as a computational grid, here we would like to have dynamic info concerning the performance of both the machines and the network. This is required to build a system that can autonomously choose the best course of action to satisfy the users request at the lowest cost and in the minimal amount of time.
4. *Integration:* different information related to the same place and coming from different sources stored in different archives have to be combined to infer new knowledge.
5. *Security*: this includes the problem of letting the system recognize the user who's trying to access it from the web and the problem of securing the system against unauthorized users.
6. *New forms of E-commerce*: the images of the Earth are not freely available and the computation time on the computers that have to do the processing is not for free either. The system should tell the user in advance the cost associated with a request. In the case of a static system this has a quite straightforward solution: the price of the image is stored among the metadata about it. When the system is dynamic, instead, we have to evaluate the processing time according to the user's request. This is an open problem. We would like eventually to be able to provide a user with a choice of this kind: "you can have what you requested in this time and at this cost or in a lower time at a higher cost".
7. *Quality of Service*: As we're trying to allow potentially a lot of users to access the power of a computational grid by a web browser, the Quality of Service will be a more and more important issue.

3. SARA: from static to dynamic

The first version of SARA-Digital Puglia allowed a user to select a place on a map and to see the available pre-processed tracks. The user could select one or more of the tracks (the tracks may be overlapped, so that more than one can cover the point chosen by the user) and see a thumbnail of the corresponding image (Fig. 3). The problems regarding the search for an image in a distributed database and the use of metadata were explored in this version.

In its second "semi-dynamic" version, it was possible to manually choose the nearest data-server and a post-processing (Fig.4). In this approach there was no search for remote program and remote execution, as the processing programs were already all on the web server. What we wanted to add was the intelligence needed by the system to start on demand post-processing.

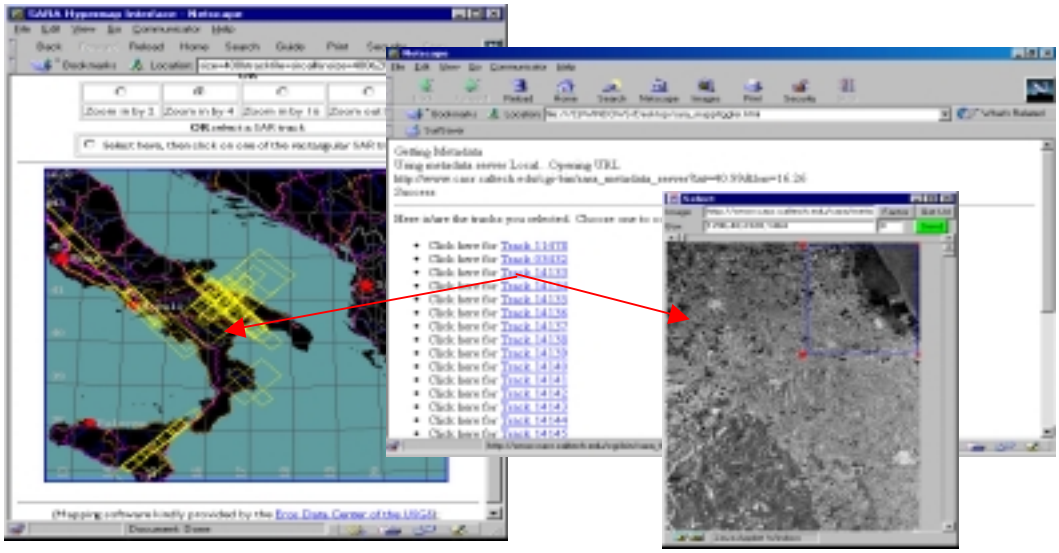

Fig. 3 SARA v.1, static version. The user chooses from a set of pre-processed tracks

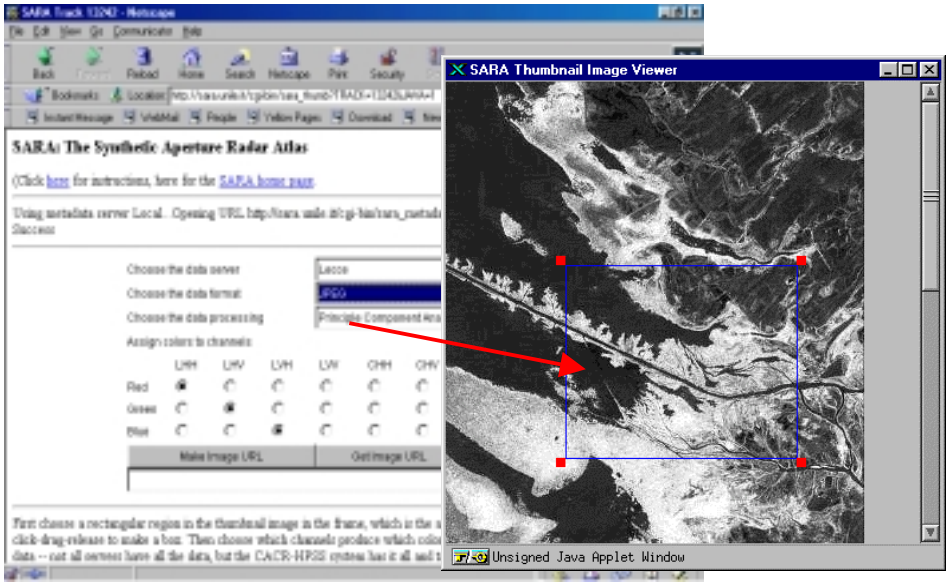

Fig. 4 SARA v.2, semi-dynamic version. The user can choose a data-server and a post-processing

The dynamic version, has a three tier architecture, the first tier being the client browser, the third tier a (Globus) computational grid and the second tier a secure web server that can communicate with the web browser on one side and with globus on the other side.

Our intention with this system is to separate completely the user from the implementation of the grid and from Globus. We were thinking about the analogy between the computational grid and the electric power grid [7]: when somebody plugs a radio or a toaster in the outlet she doesn't need to be aware of what's happening behind the wall. There might be a simple generator or a complex system that distributes electricity from various heterogeneous power plants to a huge number of users. In our system, the web server in the middle tier is the analogous to the wall, leaving the user unaware of Globus on the other side.

In other words, the second tier should allow a seamless access to the resources of the Computational Grid. For this purpose we have focused on two problems, the interaction between the web style authentication methods and the Grid Security Infrastructure, and the management of user high level requests.

Since we are allowing the user to access a set of valuable resources (considered as both datasets and computing times), a limitation of the access to a set of trusted users is required. At the same time, however, we do not want every user to be a Globus user, so the Globus Security Infrastructure works only on the "computational grid" side of the wall; on the left side there is a standard login/password authentication scheme. From the Grid point of view, all the trusted users will run the remote processing applications using the same Globus certificate.

To translate high level user requests into low level system requests, resource brokers must be provided (fig. 5). In our system, the resources are the images available in different formats (raw data, pre-processed and post-processed images), the machines of our Grid and the processing programs. To manage the high level requests, the brokers need information on the machines (stored in the Globus Meta Directory Service) and on the images (stored on a Metadata server). It should be noted that in the Globus MDS the locations of the applications available on a machine can be added in the entry for that machine.

The resource broker we have actually implemented is the part of the system which performs the search for the SAR tracks. The user issues a high level request to the system (for example: "find the tracks that are closed in this polygonal line"), the broker consults the information stored in the Metadata server and gives a result in machine-level terms ("these files have been found on these machines").

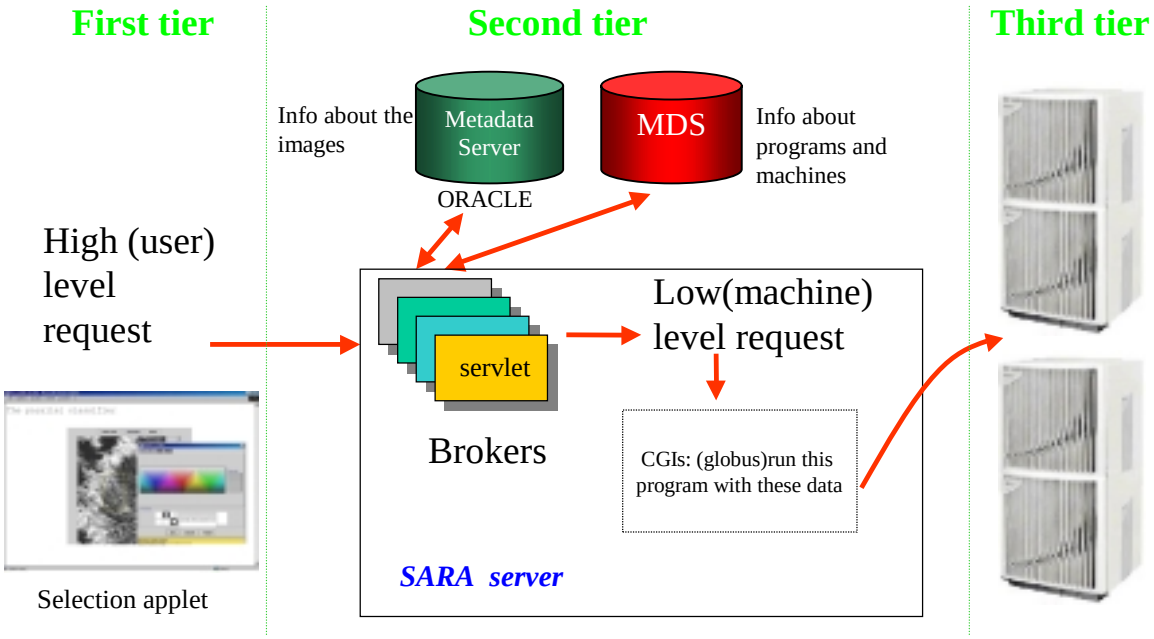

Fig. 5 Resource Brokers

In the future we will develop a resource broker (or several interacting brokers) to perform an analogous task for the choice of the processing programs.
Once we have stored in the MDS the name of the processing programs that can run on the machines belonging to the SARA Grid, a query to the MDS will allow the broker to find the program needed to perform the required processing. The most basic level of brokering is a simple mapping of a application name understandable by the user (like "Parallel Classification Analysis") with the low-level information the system needs to start an actual program (name of the machine, location of the executable). Eventually we want to give the user the possibility to make a higher level query like "color in yellow the corn fields in this zone", instead of "start a pattern recognition and a supervised bayesian classification on this track". Clearly this will require more sophisticated levels of brokering.

4. Sara version 3

The current implementation of SARA has started to move toward the idea of dynamic system we have described before. With respect to the final system we want to develop, an additional simplifying constraint has been added: we are assuming that each of the sites has the same processing programs but not necessarily the same images (Fig.6). When a site wants to be part of the SARA project, it provides one or more machines to perform the processing and one machine to hold the images (this is the most general setting, but the same machine can also be used both as a processing machine and as a database).

When the processing application are installed on the chosen machine, they are configured to fetch the images from the local database. So, it is the image that migrates to meet the processing program, but the transfer happens on a local high-speed network, so it does not slow down the system too much.

Now the search for the processing application is not an issue because every site has its own copy of all the processing programs, but we still have to let the server in the second tier know on

which machine in the Grid it will have to run the program. The possibilities are restricted by the search of the image, i.e., the processing has to be performed on a machine which is on the same site as the chosen image. In case of an image replicated on more than one site, we currently ask the user which site has to be used. The distinguished name of the machines that hold the processing programs are currently held statically on the SARA server, but in the future we are planning to use the Globus MDS to store this information to allow a more dynamic system management.

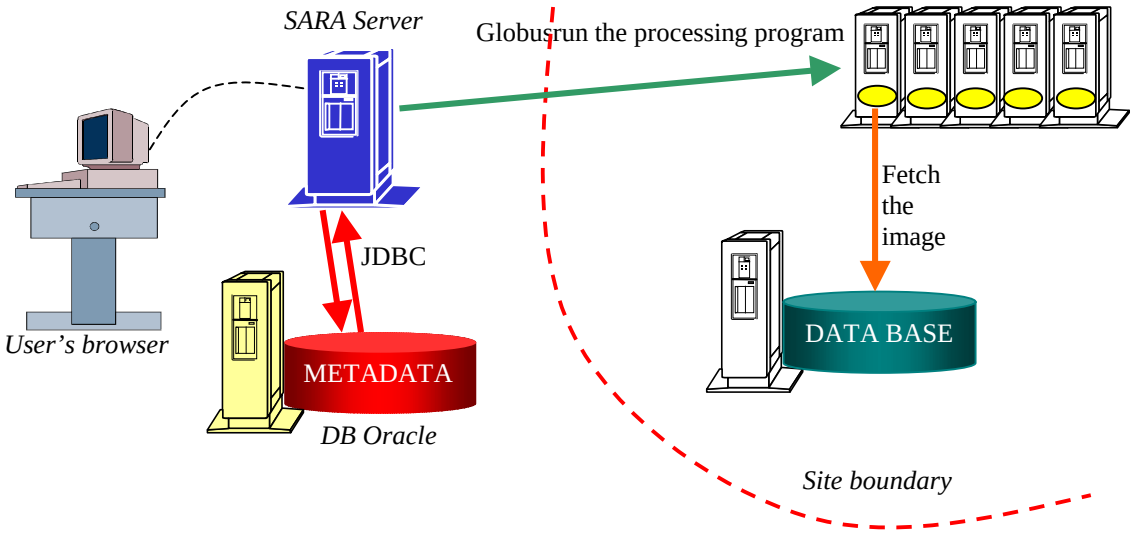

Fig. 6 In SARA v.3 every site has all the processing programs

Here is an overview of how the system works (Fig.7). The user selects a geographic area where she is interested in finding image data. This is done through an applet on which the user can draw a closed polygonal delimiting the desired place (zooming capabilities are also included). In SARA v.2, there was a number of CGIs collectively called *SARA hypermap interface* which served the same purpose of letting the user select a track: the tracks were drawn as small rectangles directly on the map, linked statically to the relative metadata, and were chosen with a click. This system was quite straightforward but it exhibits poor scaling with an increasing number of tracks which would have eventually covered the map with lots of overlapping small rectangles.

Now the selection of an area starts a search (performed by a Java servlet with JDBC) in an Oracle DB where the metadata are now stored and returns a number of results. Clicking on one of them, the user can see a thumbnail of the image and the available processings.

One of the possibilities the user can request is to see the metadata themselves converted in XML and rendered on the browser with a XSL style sheet (which dynamically creates a HTML page). Since at the moment only Microsoft Internet Explorer 5 supports XML this option has been left separated from the thumbnail, but in the future the user will have the information about the image on the same page as the thumbnail [6].

If the user chooses to start a Grid-enabled post-processing on the image, a password is required, and if the authentication is successful, the chosen processing is started using the "globusrun" command [8] at the remote site and passing as arguments the number of the chosen image and the user name.

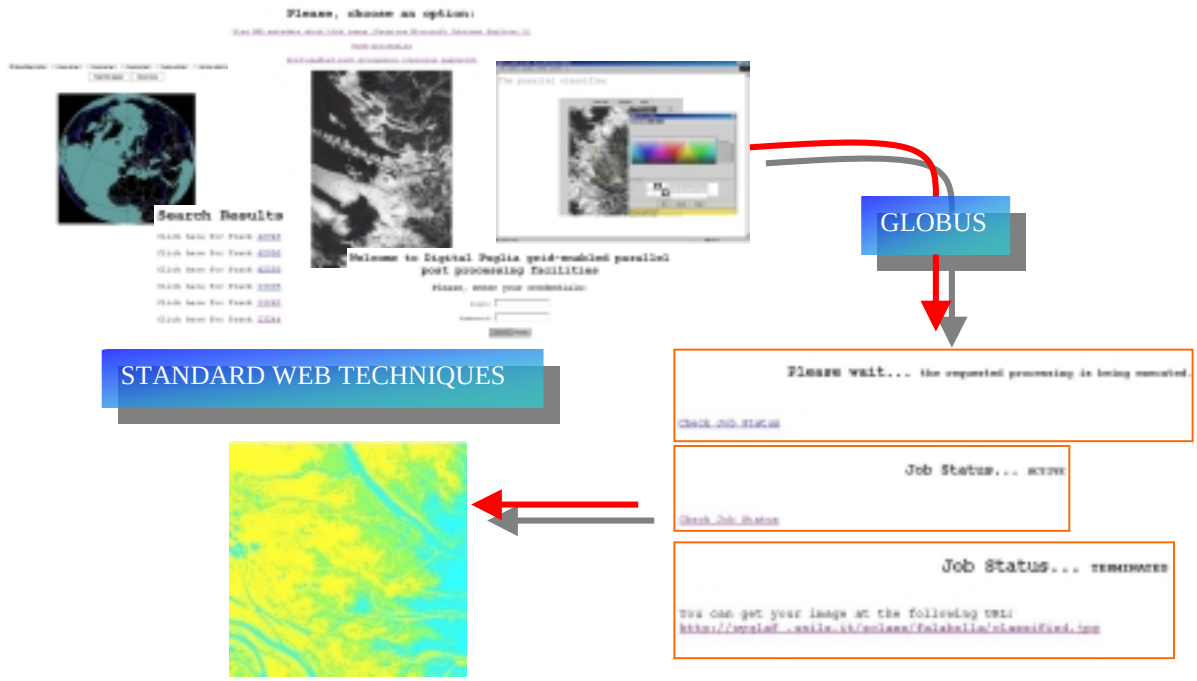

Fig. 7 Screenshots of the SARA v.3. The Globus part of the system is CGI based. It runs the remote job and checks its status.

The processing program that is started tries to fetch the image corresponding to the image ID it has been passed as an argument from the local machine containing the data. As we have pointed out before, when the programs are installed, they are configured to tell them which machine they should use as a local database. As a matter of fact it would not be strictly necessary for the images to be on a local machine (meaning a machine that is in the same site), but the transfer time for the data will become sensibly higher if the machine with the processing and the machine with the data are not connected by a high-speed network, the images being in the multi-megabyte range.

Instead of statically configuring the programs to fetch the image from the local machine, it is possible to pass the name of the machine containing the data as an argument to the program and release the constraint that the processing programs should be replicated at each site. That would not require major changes to the system, but it would make it slower. There are two reasons to do this: first, when we introduce the costs of the computation and of the images in the system (which by now have not been considered), a user could choose the cheapest combination of image and processing regardless of the total time it takes to get a result. Another reason is that some applications could be not so easy to port on different architectures and they may be available only on some of the participating sites, so putting the above constraint would make those programs usable only on the images available at the same site.

Currently, the choice of the program is automatically made once we have chosen the site from where we want to fetch the image. Another CGI is used to check the Job Status and to return the URL to obtain the requested image once the job is terminated. The image is saved under a directory with the name of the user who requested it, to save it from being overwritten by somebody else's result.

The machine on which the result is saved is the one on which the processing was performed, assuming that it has a web server installed on it. Otherwise some minor changes to the system have been designed to move the image on yet another machine, on the same site which has a web server. At first the system was designed to move the result on the web

server at the second tier (the SARA server). This seemed to close the chain nicely because the SARA server is a secure server and would have granted security in this phase too. Anyway this solution was discarded for two reasons: the image, as already said, can be several megabytes in size, so minimizing its movements decreases sensibly the time required to complete the user's request; besides, security at this stage might be not so important because a possible thief would steal information extracted by some unknown processing from an image of an unknown place somewhere on the Earth. A demonstration of our dynamic EOS was presented in the NPACI Exhibit at the Supercomputing Conference-Portland November 99.

5. Conclusion and Future Development

The issues related to the development of a dynamic on-line Earth Observation System were analyzed and some of them were solved using the Globus Computational Grid Toolkit. A definition of Dynamic EOS systems was given and the evolution of the previous version of SARA in a dynamic direction was presented.
SARA v.3 implements a digital library of SAR images of the Salento zone in Italy and provides the users with a high level web interface to search for the images and start an on-demand post-processing on them. Globus was used to allow a flexible management and control of the distributed resources and a secure access to the distributed information.

Further developments are in progress to enhance our system, in particular to provide the steering of the remote applications via web and to achieve the integration of information coming from different sources.

References

[1] C. Elachi, T. Bicknell, R.L. Jordan, C. Wu: "Spaceborne Synthetic-Aperture Imaging Radars: application, techinques and technology", Proc. IEEE, Vol.70, 1982, pp. 1174-1209.
[2] *Workshop on Distributed Geolibraries* (NRC) held in Washington on June 15-1998 http://www.nap.edu/catalog/9460.html
[3] R.D. Williams, G. Aloisio, M. Cafaro, G. Kremenek, P. Messina, J. Patton, M. Wan, *"SARA: The Synthetic Aperture Radar Atlas":*, http://www.cacr.caltech.edu/sara
[4] G. Aloisio, M. Cafaro, P. Messina, R. Williams, *"A Distributed Web-Based Metacomputing Environment"*, Proc. HPCN Europe 1997, Vienna, Austria, Lecture Notes In Computer Science, Springer-Verlag, n.1225, 480-486, 1997.
[5] G. Aloisio, M. Cafaro, R. Williams, *"The Digital Puglia Project: an Active Digital Library of Remote Sensing Data"*, Proc. HPCN Europe '99, 12-14 April 1999, Amsterdam, The Netherlands Lecture Notes In Comp. Science, Springer-Verlag, n.1593, 563-572, 1999.
[6] G. Aloisio, G. Milillo, R. Williams, *"An XML Architecture for High Performance Web-Based Analysis of Remote Sensing Archives"*, to appear on FGCS Int. Journal, North Holland
[7] Ian Foster, Carl Kesselman, *"The Grid. Blueprint for a new computing infrastructure"*, Morgan Kaufmann, San Francisco 1999
[8] Ian Foster, Carl Kesselman, *"The Globus Toolkit"*, http://www.globus.org