Deployment of Earth Observation Applications on the European DataGrid Testbed

Luigi Fusco[1], Julian Linford[1], Pedro Goncalves[1], Annalisa Terracina[1], Monique Petitdidier[2], Cathy Boonne[2], Christine Leroy[2], Wim Som de Cerff[3], John van de Vegte[3]

[1]*ESA-ESRIN, Frascati, Italy*

[2]*Institut Pierre Simon Laplace, Université Pierre et Marie Curie, Paris, France*

[3]*Royal Dutch Meteorological Institute, de Bilt, The Netherlands*

Abstract

In the EU DataGrid project, the Earth Observation (EO) scientific community participated in a major European IT research collaboration, whose aim is to develop an enabling new technology that promotes scientific inter-collaboration and organized resource sharing, leading to increased exploitation of many community resources. The initial stage of the project was dedicated to identifying particular requirements of the EO applications and user community. The GOME Ozone Profile retrieval and validation application, a multi-stage processing chain which transforms raw, satellite based observations into high quality, finished products, was selected for Grid deployment. The application is representative of a wide range of EO applications, which perform processing and validation of large volume global satellite observation data. Deployment on the DataGrid testbed required the adaptation of complex algorithms and close collaboration among EO science teams in three European institutes. The deployment approach aimed to evenly distribute both the processing and storage load across the testbed, to fully exploit available resources. Several issues were encountered by the EO team during the deployment and solved in collaborations with the middleware architects and developers. Methods were needed for handling large numbers of files, distributed in various storage locations throughout the Grid. The validation process required locating and retrieving small subsets of the data, depending on specific EO 'areas of interest' being investigated (e.g. data type, time of observation, latitude, longitude). Among several solutions investigated was the extensive use of metadata attributes associated with the replicated data, allowing replicas to be located and retrieved using not just the Logical File Name, but also by metadata searches using the Replica Metadata Catalogue.

1 INTRODUCTION

From the point of view of environmental Earth Observation (EO) applications, there are several benefits a computing Grid can offer to the user community. Applications using remote sensing data from satellite and ground based observation networks are wide ranging, e.g. studies of the earth's mantle, atmosphere, oceans, ice and land formations, agriculture and cartography. Many national and international programmes - both research and operational - build on top of specific application domains to develop large-scale framewroks for monitoring and analysis of dynamic earth system interactions, to better understand and predict prevailing conditions (now casting as well as long term prediction). Also, civil sector applications bring two classes of requirements, the first concerning short-term forecasting of risks (e.g. of pollution, thunderstorms, hurricanes, volcano eruptions), and the second concerning long-term forecasts for climatic trends. Both require quick access to distributed datasets and high performance computing resources.

Many investigations make use of large-scale, high-capacity distributed processing facilities and involve collaborations between numerous actors in EO scientific, operational and commercial sectors. A Grid can facilitate such interactions by providing a standard infrastructure and a collaborative framework to share data, storage and

processing resources, algorithms and data products. A common infrastructure will facilitate collaborations among diverse organizations carrying out data-intensive processing research, and will promote the exchange of information and scientific results in a coordinated way.

EO ground based measurement networks provide a wide variety of data products, although their volumes are relatively small compared to satellite data. Satellite borne sensors are capable of collecting global coverage data over extended periods of time, generating a very large amount of data to be catalogued, archived and processed. For example, ENVISAT, launched in early 2002, has already collected one Petabyte of archived data and is generating 400 Terabytes of data products per year, to be handled by the dedicated ground infrastructure distributed in various European countries. A remote-sensing survey that needs to analyse data collected over a large global area during an extended period of time will consume large amounts of computing time, network bandwidth and storage space and this requires the available Information Technology (IT) infrastructures to be suitably dimensioned to handle peak requirements. However, during non-peak times, high-end resources can be made available for use by outside partnering organizations. Alternatively, where it is preferable to maintain downsized local facilities, the Grid can provide the extra resources needed during peak times.

Together, the Grid Virtual Organisation and Security Infrastructures offer a standard framework for organizations to collaborate by sharing their resources. In the same manner, the infrastructure also facilitates sharing of data, applications, algorithms and other scientific know-how. This leads towards the cross-fertilization of scientific domains, a topic of interest for the integration of cross-domain scientific data in large-scale simulation modelling that is necessary to improve long-range weather and environmental forecasting, etc.

In the first part of this paper, the application deployed on DataGrid by the EO work package is described and the main Grid requirements, representative of EO applications in general, are outlined. In the second part, the DataGrid middleware is evaluated with respect to these requirements. In the third part a Web portal interface that integrates EO operational satellite tools and the DataGrid middleware is described. A summary of achievements and needs for future development are given in the conclusion, together with a future perspective for the EO community.

2 OZONE PROFILE PRODUCTION AND VALIDATION

2.1 The EO Application Use Case

The application selected for deployment on the DataGrid testbed infrastructure represents an ideal candidate for testing the functionalities provided by the EDG middleware.

An end-to-end processing and validation use case has been developed that embodies the typical product processing, refinement and quality control procedures that routinely take place in the EO applications domain. The selected application is fairly representative of the kind of large-scale, collaborative types of EO applications that stand to benefit most from deployment in a Grid scenario. Both the problems encountered and solutions offered while deploying this use case on the Grid testbed may be considered applicable to a wide range of Earth Observation applications.

The application involves processing and validating the entire, seven year global GOME dataset, consisting of atmospheric ozone measurements collected over several years of the ERS-2 satellite mission (launched in 1995).

The data was processed using two different ozone profile retrieval algorithms – one developed by van Oss et al.[12] and deployed on the Grid by KNMI and the other developed by Casadio et al.[13] and ported on the Grid by ESRIN. The resulting Level-2 profiles were then validated by IPSL using ground-based LIDAR ozone observations [14].

This application model (Figure 1) is well suited for assessing the utility of the Grid for processing a large global dataset, as it involves processing large amounts of distributed data using complex algorithms, collaborations between users distributed in different organizations, and sharing of computing resources. Detailed descriptions of the deployment environment and scientific results achieved are available in [15, 16].

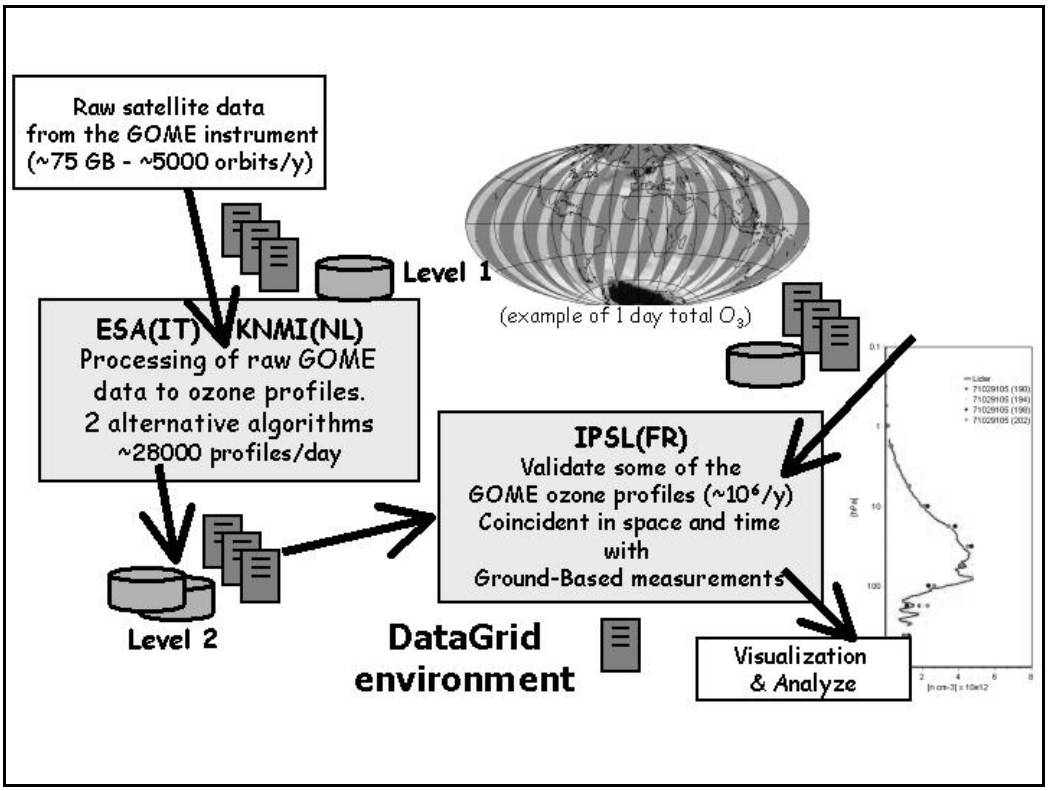

Figure 1. Scheme for GOME profile processing and validation using DataGrid

2.2 Main requirements on the Grid Middleware

Throughout the EDG project, several requirements documents and usecase documents were produced [1, 2, 3, 5, 6, 10, 11], including the results of collaborations between the three application groups in the project, through the Application Working Group (AWG). A common set of requirements coming from all threeapplication groups addressed the general functionalities of the middleware and its stability. We describe here the specific requirements of the EO applications group.

Table 1 illustrates the need for handling large numbers of files and relatively large volumes of data. The volume of data is variable and for new experiments it has a tendency to increase due to progresses in electronics and computer technologies. Packaging the data (i.e. in larger files) can reduce the number of files and time to access, reducing both the search time and transfer time in case of large numbers of files. However, packaging can also introduce overheads that can be a drawback for the user. Tests to explore the limit of the data handling middleware are reported in section III.2.

Dataset	Number of files	File Size
Level 1	4,724	15 Mb
Level2 (NNO)	4,724	19.5 Mb
Level2 (OPERA)	9,448,000	12 Kb
Lidar	12	2.5 Mb

Table 1. One year of GOME data products

A particular characteristic of EO applications being investigated is the need to access both metadata and data. Access controls, restriction and security are needed on both metadata and data. Data may be confidential and accessible only for a given group of users and for a given period of time, for example, around the time of publication of some new results. Data access policy varies depending on the origin of the data and time of production. Some products, e.g. Lidar data, can be made freely available on the web after two years (having informed the data producer), while other products may be used for scientific but not commercial purposes. Certain types of products, e.g. European satellite data, are only made available to users working on approved projects. As a consequence, the EO application community needs secure and restricted access to both metadata and data, although encryption is not required. Furthermore, access control is based on the role of each partner (e.g. read, write, administration, update), and it is therefore necessary to be able to define access rules and capabilities in terms of roles. Tests using Spitfire and the Replica Metadata Catalogue (both made available by EDG) are reported in section III.3.

The EO use case (Figure 2) involves the deployment of two different product-processing algorithms and one validation algorithm. Of the two processing algorithms, OPERA, developed by van Oss et al. [12], is an inversion problem based on an optimization method and NNO, developed by Casadio and DelFrate et al.[13, 15], is based on a neural network approach. The validation algorithm is not very complex and uses a graphics tool to plot the results. Two main issues were encountered concerning the adaptation of these algorithms to run on the EDG testbed, the first concerning code portability and the second concerning the use of Grid functionalities. Porting an algorithm originally developed to run on a workstation depends on the availability of specific languages (and their different versions), that are frequently used in EO applications, such as Fortran 77, Fortran 90, IDL and Matlab. The second issue concerns the availability of required Grid functionalities, e.g. locating products using metadata, access control, as previously mentioned. Some of the required functionalities, e.g. MPI, became available towards the end of the project. An overview of the different algorithms used and the related issues is given in section III.4

The ground segment infrastructure, established over several decades of development by the EO operational satellite community, consists of extensive product catalogues and archives distributed in many large data centres and a large number and variety of user software tools and algorithms for data exploitation. An important issue investigated concerns interfacing the DataGrid infrastructure with the existing EO tools and operational infrastructure. It is especially important to respect local data access policies and to access EO product archives in a secure way. In addition, there is the need to abstract the complexities of the Grid middleware, to allow end users to concentrate on scientific issues. A web portal developed by ESRIN to address these issues is described in section IV.

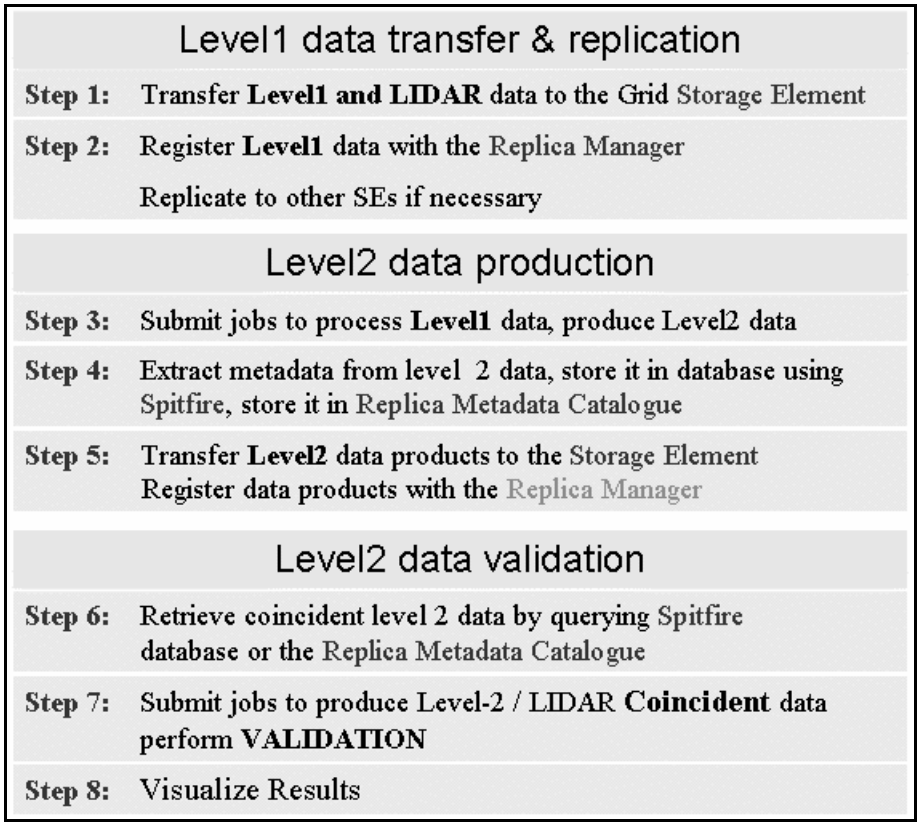

Figure 2. Discrete steps of the end-to-end GOME data processing and validation chain

3 EVALUATION OF THE GRID MIDDLEWARE

3.1 Context

During the EDG project two major releases, EDG 1.4 and EDG 2.0, were formally assessed by the three application work packages (reports on the Earth Observation evaluations are in [4, 7]). We focus here on the EDG 2.0 release and the extent to which it meets the requirements posed by the EO GOME use case as described above.

3.2 Handling large number and volumes of data

A major objective of the EO work package was to assess the suitability of EDG testbeds for handling the class of data-intensive, high throughput applications that are typical in

the EO domain. The GOME ozone profiling application, with seven years of global historical observations to be processed using NNO, a complex algorithm based on Neural Networks, provides an ideal test case for carrying out this assessment.

Considering the deployment of this test case on the EDG infrastructure, and aiming for maximum exploitation (and exercising) of the testbed facilities, it was decided to distribute both the data storage and the processing load evenly over all the available processing and storage resources (Computing Elements and Storage Elements) of the testbed.

This required transferring the Level-1 dataset from EO Mass Storage Archives and distributing it on the available Storage Elements, where it could then be accessed by jobs running on the Computing Elements. An alternative would be to have the jobs access the data at runtime directly in the archive system, using GridFTP. However, the archive has relatively high access latency and this would create bottlenecks when a large numbers of processing jobs were submitted simultaneously. Therefore, it was necessary to carry out a bulk dataset transfer - 'pre-replication' - to Grid storage before scheduling the processing tasks. This was done using both the Replica Manager and Storage Element interfaces.

The initial aim was to run a test using one year of data - roughly 4,724 data files of approximately 15MB each - a total of 70.8 GB. The SE available space information published in the GIS showed that no single SE had the capacity to hold the entire dataset, since the available space was shared with other EDG applications - this was another reason for distributing the dataset over several SEs.

The files making up the dataset were transferred to Storage Elements distributed throughout the Grid and their locations were registered, along with a system-generated GUID, in the LRC (Local Replica Catalogue). Meaningful aliases, or LFNs (Logical Filenames) were associated to the assigned GUIDs using the RMC (Replica Metadata catalogue), thereby allowing the files to be located by the application later on.

When jobs were submitted to the Grid for processing, the LFNs of the files were specified as InputData attributes in the job scripts (JDL files). This exploited the EDG Resource Broker and the Replica Optimisation Service, in order to direct jobs to the Computing Elements closest to the data sources. The EDG testbed computing resources are usually paired with storage resources on a 1:1 basis and the choice of SE during the pre-replication stage is also likely to determine the CE selected by the Resource Broker when the processing jobs are submitted. Therefore, obtaining an even distribution of the dataset over the available SEs is also likely to lead towards an evenly balanced processing load over the CEs.

A bulk file replication procedure was developed to copy the files of the Level-1 dataset from the EO archive system, to distribute them evenly among the testbed SEs and register the locations in the Replica Catalogue. The procedure iterated through a list of files specified as an input parameter, thus allowing the procedure to be run as required on selected parts of the dataset to be replicated (e.g. by year, or month of observation). The procedure maintained a table of relevant SE information (service name and space available) for all available SEs published in the Information System. Before each replication, the next SE was picked from the table and the space available was checked, the SE was skipped if the available space fell below a specified (configurable) threshold.

The procedure used two different Grid middleware functions (Table 2). The first queried the R-GMA information system to retrieve SE details, the second carried out the data transfer and registration in the LRC as an atomic operation. These functions are available both as command-line executables and as linkable library APIs. The procedure executed both functions using the shell command line interface.

Action	EDG Package	Command
Retrieve SE information	R-GMA	edg-rgma -c 'latest select UniqueID, \ AvailableSpace from GlueSA'
Copy data to SE, register in RC	RLS	edg-replica-manager copyAndRegisterFile \ sourceFileName [command-options]

Table 2: EDG commands used for bulk replication procedure

The pre-replication procedure was run for several weeks (24 hr, 7 days) and eventually all 4,724 files were successfully transferred and registered. Among the major issues encountered were slow transfer speeds and frequent service interruptions (the pre-replication script was reinforced with routines for fault detection and recovery). The interruptions were outages either due to R-GMA failures, or service outages due to site upgrades or reconfigurations, often as a result of frequent ongoing testbed releases. Subsequently, the replicas were located as needed by using different commands (Table 3). Given a LFN, any replica could be retrieved using its assigned Logical File Name (LFN) using the following sequence of commands:

1. <storage URL> = edg-replica-manager listReplicas <LFN>
2. <transfer URL> = getTurl <storage URL>
3. globus-url-copy <transfer URL> <destination URL>

Generally, two ways of locating the replicas were used, either retrieving one entry at a time or all entries at once. The performance (i.e. speed) of both methods needs to improve to be really workable.

Command	Action
edg-replica-manager listReplicas	returns GUIDs of all existing replicas, given a logical file name
edg-local-replica-catalog mappingsByPfn	return list (GUID; storage URL) of all entries matching a given search pattern
edg-replica-metadata-catalog mappingsByAlias	return list (GUID; LFN) of all entries matching a given search pattern

Table 3: Commands used for locating replicas

3.3 Access to Metadata and Data

In the EO applications domain, metadata catalogues, organized in databases, are very common. There is a need to select data corresponding to given geographical and temporal

coordinates, or specific data-quality values, algorithm version, etc. By searching the metadata catalogues for the corresponding tuples of the metadata attributes, EO users can identify specific subsets of the data, which cover specific areas of interest being investigated. Using the information returned by the metadata search, the Logical File Name (LFN), the physical data files can be located and retrieved by using the Replica Management (RM) commands.

Different tools and methods were available in EDG to create and access metadata catalogues. The two different solutions experimented with were Spitfire and RMC. Spitfire provides transparent and secure access to databases (e.g. MySQL) for Grid middleware and applications. RMC is the EDG Replica Metadata Catalogue, whose job is to maintain aliases and alias attributes for data files registered in EDG Replica Catalogues (also powered by Spitfire).

The EO community has its operational catalogues outside the EDG, e.g. MUIS, which is the proprietary ESA EO product catalogue (described in section IV).

3.3.1 Spitfire

Description

Spitfire [R8] provides a standard protocol and well-published interfaces for remotely accessing different database (RDBMS) systems across the Internet. It allows Grid users to be mapped to database roles in a configurable way according to different local policies. Three SOAP services are defined: a `Base` service for standard operations, an `Admin` service for administrative access and an `Info` service for information on the database and its tables. There is also the possibility to use a web browser to access the database through HTTPS and canonical XML through a simple SQL Database Service. Client-side APIs are provided for Java and C++ for the SOAP-enabled interfaces. There are also some client-side tools for the SQL Database Service. The Spitfire services abstract the details of accessing different local RDBMS systems.

Spitfire makes use of the EDG security modules to enable full GSI authentication and additional authorization. The new Spitfire (version 2.1.7), including the EDG java security packages was installed on the KNMI and IPSL User Interfaces. The underlying databases were both MySQL databases, containing Ozone profile metadata and Lidar metadata respectively.

Access control

Spitfire provides role based mapping, i.e. a user is mapped to a database role with certain privileges: readrole, writerole, updaterole and Gridadmin. Besides the role-based model of user mapping, Spitfire also supports VOMS (Virtual Organization Membership Service) [R9]. It can be used in parallel with the role-based model. The benefit of using VOMS is that the management of users, roles and capabilities is shifted to the VOMS server, so they can be centrally managed in stead of managing it at each instance of Spitfire. It is also possible to use the VOMS for other authorization purposes (CE, SE access).

Querying data

The Opera metadata database contains the GOME_PROFILE table in which GOME ozone profile metadata are stored. Each record contains the metadata of one pixel file as follows: start and stop time of the measurement, geolocation information (latitude, longitude), LFN of the pixel file, LFN of the input file, processor version, and a free field for text information. The table contains 59,000 entries and will grow when new profiles are processed.

For small result sets Spitfire offers the `SpitfireResult` class, in which the query result is stored. For larger result sets, Spitfire provides the `PartialSpitfireResult` class and a special query method `selectSQLPR`.

Test programs were written to query the database for specific data and for large result sets. The `PartialSpitfireResult` class enables the Spitfire client to retrieve large result sets, i.e. it is possible to retrieve all 59,000 records from the database. A program retrieving all records was written and executed. It succeeded to retrieve all records.

Four types of test were performed: retrieving 100 records, retrieving 1000 records, retrieving 59,000 (all) records and retrieving a specific record. The security features of Spitfire were used. Output was retrieved, but not printed to screen or file. Timing statements were set around the retrieve data methods in the Java programA record consists of 9 columns, containing an average of 200 bytes of data.

The sizes of `ResultSet` and `PartialResultSet` are not defined in the documentation, probably because it is hardware (memory) depended. So it is not clear what is a 'large ResultSet'. Thus one has to test which class to use. When the `ResultSet` is used and a large result set is requested, the error message is not clear:

org.xml.sax.SAXParseException: Document root element is missing.

Which does not really explain what is happening, i.e. a set is requested which does not fit into the `ResultSet` class. To be on the safe side, one should use `PartialResultSet`.

Inserting data

Inserting data was tested. The client program was run from local host, ui02.nikhef.nl and from random Worker Nodes (WN), which provided outbound IP connectivity. The insert script inserts one complete record into the database. The tests were timed and averaged (Table 4).

Creating tables

The creation of tables using the client was tested. Programs were written to create a dummy table with two columns and to drop this dummy table. The client programs were able to create tables and drop them again. The creating/dropping of tables was not timed, as this is an action which is performed only once for our application.

Performance

The timing results (Spitfire performance table) were derived from the tests described above. Each test was run 5 times. This is of course not enough to get real statistics, but it is good enough to give a performance impression. The Spitfire server ran on a machine with a 500 MHz Pentium III processor and 128 MB of memory. Client ran on a machine with a 1 GHz Pentium III processor and 1 Gbyte of memory. Both machines were running the Linux Redhat 7.3 operating system. While testing the database, content was kept constant with 59,000 records.

Action	Average time	Time per record	Comment
Querying data, 100 rows.	5 seconds	0.05 seconds	SpitfireResult class used
Querying data, 1000 rows	19.2 seconds	0.02 seconds	SpitfireResult class used
Querying data, 5000 rows	2 minutes, 17 seconds	0.03 seconds	SpitfireResult class used
Querying data, 59,0000 rows	20 minutes, 44 seconds	0.02 seconds	PartialSpitfireResult class used.
Specific query (4 records returned)	3.8 seconds	0.95 seconds	SpitfireResult class used
Inserting data	3.6 seconds	3.6 seconds	This is how the database was filled

Table 4. Spitfire performance

From the time-per-record column in the Spitfire performance table, it can be concluded that the performance is stable and does not depend on the number of rows requested.

Summary

Spitfire was evaluated in both [R4] and [R7]. The main difference between the two releases tested was the implementation of the security features (full GSI and VOMS support), as was recommended in the evaluation report [R4] and the applications' joint list of recommendations [R10].

Spitfire was installed and configured with the help of the developers, and the EO group was the first application to use it as soon as it was available. Configuring the user mapping can be done either using VOMS or using the Spitfire built in mapping - both provide flexible access control. Performance is sufficient for the EO usecase. Spitfire provides an excellent tool for storing and retrieving metadata in a secure, reliable and stable way, but the documentation and installation procedures can be improved.

3.3.2 Replica Metadata Catalogue

The applications' evaluation of EDG 1.4 [R4] and the joint list of recommendations produced by the AWG [R10] both suggested the need for improved support for application metadata. As a result, EDG 2.0 provided the Replica Metadata Catalogue (RMC).

A single set of RMC attributes can be defined for all the datasets belonging to a VO that are registered in the Replica Manager (RM). This scenario was chosen for the evaluation (a detailed description of the set-up is available in a technical note [11]). Using the RMC as an integral part of the data management middleware provides metadata functionality without the need to install and configure extra software by the user.

A database schema was first defined with 25 different attributes covering the different types of EO data products, specifying the attribute types and allowed value ranges; the attributes were then created in the EO VO RMC using CLI commands inserted in a shell script. When populating the RMC attributes for a given EO product type, only the relevant attributes were filled, fields concerning other products were left empty [R11].

The metadata attribute types supported by the RMC at the time of testing were float, string or integer, although more types could be added in later releases. The date-time type was not available and a float type was used instead. One of the attributes, `CollectionName`, provided a simple way of creating logical collections.

The RMC CLI interface command only allows a single attribute be set each time and takes an average of 4.3 seconds to execute. Registering a single ozone profile requires setting 15 attributes, which takes approximately 64 seconds (add to this the time for copying and registering the file). However, a typical Opera run takes 6 hours to process an average of 1000 profiles, while inserting the corresponding metadata takes 17.7 hours (and this does not consider the time for copying and registering the files). In response to these timings, the RMC developers provided a C++ API capable of setting all the attributes of a file in one go. This was tested and found to take approximately 17.2 seconds for the same update, thus 45 times faster than the CLI.

The RMC provides possibilities for metadata storage and provides easy to use interfaces (CLI and API) without requiring the user to install additional tools. The speed of the RMC needs to be improved in order to be usable in practice. More data types and database functionalities are needed, for instance, support for polygon (spatial) queries, multiple tables and restricted access, not only according to the role (e.g. VO, group, sub-group).

3.4 COMPLEX ALGORITHMS ON THE GRID

3.4.1 NNO

Several methods are available to retrieve ozone profiles from GOME, most of them based on Optimal Estimation techniques, which involve forward modelling by running multiple scattering radiative transfer models. Although the technique results in very high quality products, the computation is very lengthy. An alternative approach, based on neural networks (NN), avoids the lengthy modelling process, since it does not depend on the prior derivation of particular rules or statistical information [R15]. The technique

establishes a set of inverse mapping, input-output discriminant relations by constructing a neural network, by analyzing data presented during a learning phase. The training process only needs to be done once and the resulting neural network can be used to produce new estimations in real time.

The NNO algorithm, supplied by Universitá di Tor Vergata, consists of a pre-compiled IDL runtime executable and several small auxiliary files, including the IDL runtime license. These files, present on the UI (User Interface) machine, are bundled together in a tar file and passed to the RB using the Input Sandbox, together with a wrapper shell script. The total size of the input sandbox sent to the RB, transferred to the selected CE and from there to the local worker node, is roughly 1.2 MB.

Before submitting the NNO jobs, a job preparation script is run on the UI (User Interface) machine, which generates one or more JDL files. The script can be configured to generate *J* jobs, which each process *O* orbit files, depending on the quantity of data to process and number of jobs to run. Typically, *J* may be anywhere in the range 1-1000 and *O* in the range 1-500. To put these figures in perspective, running with *J*=1000 and *O*=500 would process 111 years of global observations. Assuming minimal Grid system overheads, no queue waiting times and no malfunctions or service interruptions, it would take an estimated 25 hours (an ideal assumption, which can be considered practically impossible to achieve today, except in a local supercomputer environment). The JDL files are generated from a common template, they differ only in the InputData attribute, which contains the Logical File Names of the orbits to be processed (previously replicated, as described in section III.2). Among the requirements specified using JDL is the need for the IDL-5.4 runtime environment, a packaged RPM that is pre-installed on all the EDG CEs, wherever the EO VO is authorized.

Once a job starts up on the Worker Node, the wrapper script first un-tars the NNO bundle, sets file permissions, and then uses the BrokerInfo interface to recover the list of Logical File Names that were specified against the JDL InputData attribute. It then uses the Replica Manager `getBestFile` command to locate and retrieve the physical files, so that they are directly accessible by the IDL executable, via the standard file open command (we note here that `getBestFile should` in some way make use of the Replica Optimization Service). Having prepared the environment, the wrapper script runs the IDL executable. The executable takes as input the list of GOME Level-1 orbit files to be analysed and for each one, produces a corresponding Level-2 ozone profile product, writing it to the local working directory (both OPERA and NNO conform to an agreed Level-2 product format specification). When the executable terminates, control is returned to the wrapper script, which then transfers the Level-2 products to the local (i.e. 'Close') Storage Element and registers the Logical File Names in the Replica catalog. Finally, the script extracts the relevant metadata keys from each Level-2 product and registers the values in the Replica Metadata Catalogue, filling the appropriate attributes associated with the Logical File Name. The Level-2 orbit files thus produced, distributed throughout the Grid SEs, can be subsequently located using the RMC, to obtain the set of LFNs associated with a given metadata search, and using the LRC, to obtain the corresponding GUID/SURL for accessing the data on the SEs.

3.4.2 Opera

The Ozone Profile Retrieval Algorithm (Opera) [R12] algorithm uses extracted, recalibrated GOME Level-1 pixel data as input. A preprocessing stage (Figure 3) consists of two steps: first, extract the pixels from the GOME Level-1 data into pixel files containing earth and solar shine spectral radiances, which results in approximately 1500 pixel files per Level-1 file; second, calibrate the pixels using GOMECAL [R17]. The resulting pixels are averaged and used in the calculation, an optimal estimation method and on-line radiative modelling, to produce the ozone profiles.

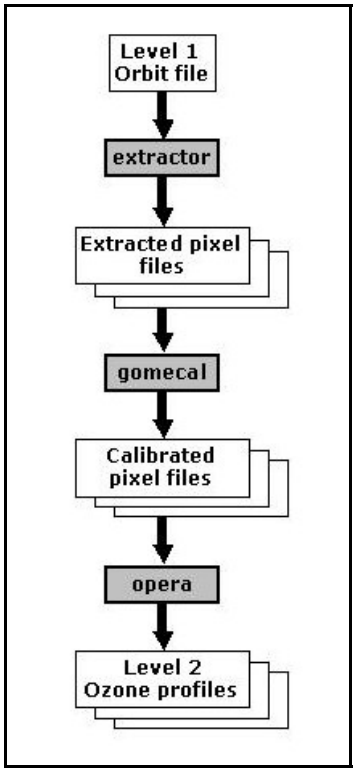

Figure 3. Opera processing sequence

The extraction program is written in C by the Deutshen Zentrum für Luft- und Raumfahrt (DLR), the GOMECAL and Opera programs were written in FORTRAN by KNMI. Shell scripts are used to glue them all together.

The algorithm is executed as a Grid job using the Workload Management commands installed on a Testbed UI machine. This requires writing a description of the job requirements using the JDL scripting language.

In particular, the JDL script specifies: the executable, the Logical File Name of the Level-1 input file to be processed, the names of files on the UI to be transfered to the Worker Node in the Input Sandbox, and optional the CE ranking rules.

The extraction program, GOMECAL and Opera are all put in a single tar gzipped archive file and transfered to the WN just as a normal input data file. Before running the process chain, the main shell script, sent with the input sandbox as the executable, sets up the local environment in the working directory of the WN, it untars the archive and sets file

permissions and environment variables. It will retrieve the Level-1 file from the SE and will start the extraction, recalibration and ozone profile calculations. Metadata is extracted from the resulting Opera pixel file and stored on the Grid, using Spritfire and later also the RMC.

For the deployment of OPERA it was decided to use the CLI (command line interface) commands embedded in shell scripts instead of the APIs, which were also available, in order to keep program changes to a minimum. To get the complete process chain running on the Grid required more effort than expected. Opera consists of a complex chain of science software, which did not make debugging easy. Some odd program behaviour could only be overcome by compiling the sources on the target WN. After the initial debugging the program ran without problems and it was possible to use the Grid for mass production of data.

3.4.3 Validation

Description

The continuous survey of the stratospheric ozone layer requires the use of complementary ground based and satellite monitoring systems. While satellite measurements provide the necessary global coverage of the ozone field, ground based measurements such as those implemented in the frame of the Network of Stratospheric Changes (NDSC), are a prerequisite for the calibration of satellite measurements. Such a validation is necessary for the present European satellite measurements (GOME, GOMOS, Sciamachy). The present application is intended to validate measurements of the ozone vertical distribution data performed by various satellite sensors against ground based DIAL Lidar measurements, performed at different locations.

The critical points in the validation procedure are (1) searching for GOME ozone profiles observed on a given day with geographical location close a given Lidar ground site; the number of satellite profiles per day being around 28,000, and (2) restrict access to the data sets to authorized persons only.

A Lidar metadata catalogue was first created on a local UI machine running at IPSL, using Spitfire as a front end to a MySQL database. The monthly observation data of each Lidar station were copied to the UI machine from the database hosted at the NDSC site http://www.ndsc.ncep.noaa.gov/. The metadata attributes for each day of Lidar observations were extracted from the data and used to populate the Lidar metadata catalogue. The Lidar data were also replicated on several storage elements and registered in the EO Replica Catalogue. Tests focused mainly on the coincident observations of the Haute Provence, Mauna Loa and Table mountain Lidar stations and the GOME satellite obervations made over a 7 year period.

Co-location of Lidar and GOME data

The first step The method used for selecting the GOME ozone profiles covering a given latitude and longitude area, depends on the way the processing algorithm stores the profiles.

The OPERA algorithm stores each individual ozone profile in a separate file. A disadvantage of this storage method is the very large number of files (~10 million per

year), which have to be registered in the RC and RMC. In EDG 1.4 the metadata attributes belonging to each file were registered in a KNMI database. The database was searched using the Spitfire interface and the Logical File Names of the coincident data files were obtained using the Replica Catalogue. In EDG 2.0, the Replica Metadata Catalogue was used instead of the KNMI database.

The NNO algorithm stores all the ozone profiles of a complete orbit together in a single file (about 2000 profiles per orbit). Obtaining the coincident profiles is done in two steps, by first selecting the orbit files using the Replica Metadata Catalogue or the EO web portal, and then extracting the coincident profiles from each orbit file. The present version of the RMC is not suited to the orbit-based storage scheme, since it does not support a polygon data type and therefore cannot perform efficient spatial searches. Due to the almost global extent of each orbit, a spatial search using only the corner coordinates almost invariably returns 10 out of 14 orbits for each day, even if only one or two of those orbits actually passed over the Lidar ground station. The NNO orbit-based storage scheme is interesting for global processing but not so useful for localized, or regional investigations.

An alternative storage scheme has been proposed, based on partitioning by latitude and longitude for each day, which is better suited to atmospheric studies. Preliminary studies and tests have yielded positive results and this method is now being developed further.

3.5 Grid Job execution

The job submission system functions as a large batch system with commands to submit a job, check its status, and retrieve any output. The two main differences between a standard batch system (such as PBS, LSF etc.) and the EDG job submission service are that the job submission service adds a high-level layer permitting uniform access to the resources at different sites and automatically matches the available resources to the requirements of the job.

From the point of view of evaluating the WMS (Workload Management System), the GOME use case involves two different types of jobs: batch processing of Level-1 to produce Level-2 ozone profiles, and validation processing of selected level-2 profiles, which requires a more interactive approach.

Summary

Generally, the WMS was much improved in the EDG 2.0 release; in particular the job distribution among CEs is much better at avoiding bottlenecks.

Automatic, advance replication of job input data is not implemented in the WMS due to the lack of non-blocking services for file replication. However, it can be done on the WN by using the `edg-replica-manager getBestFile` service.

Information system (and consequent failures) represents a single point of failure in the whole system.

Some newer features made available towards the end of the project, such as more detailed error messages, output data registration and job checkpointing, were not evaluated.

The automatic retrying of available RBs by `edg-job-submit` is an excellent example of fault tolerance.

More careful attention to synchronizing the status and error messages written to the stdout and stderr files will facilitate error tracking for the end user. For example, when a series of RM copy and register commands are executed and some of them fail, it is difficult to determine which LFNs failed.

3.6 Conclusion of the evaluation

The Grid application programming paradigm is very different to the conventional model. Our experience shows that the Grid is highly dynamic; it consists of many parts, e.g. middleware, services, sites, resources - and people. The smooth functioning of the Grid as a whole depends on countless automated interactions between these highly dynamic components. Typically these components originate from, or are deployed by many collaborators, e.g. designers, authors, testers, administrators and users, scattered in various organizations throughout Europe and the US.

Often communications and planning suffer from inherent communications latencies, involving a very busy and widely dispersed community, communicating mostly by email or telephone conference. Many dedicated and sustained, close team collaborations are necessary to elaborate the architecture, develop, deploy and test the middleware, to diagnose faults and work out new solutions for complex issues.

During software development and testing real conditions can be simulated to a limited extent, but in a full-scale testbed deployment with many sites and users, entirely new and un-foreseen conditions will inevitably occur. The collective mean-time-between-failure of numerous underlying infrastructure components - networks, CPUs, storage, etc. as well as the human dimension, are also part of balancing the Grid equation.

From the Earth Observation user perspective - and we imagine it is the same for the developer and service provider - the key to successful Grid middleware will be the capacity to detect and recover from numerous faults that are guaranteed to occur due to the inherent dynamic nature of the Grid system.

Therefore, we believe pervasive, built-in fault detection, backup measures, service redundancy and pro-active error recovery techniques will be the key to keeping both services and applications alive on the Grid, while keeping administrative overheads to an absolute minimum. We also noticed that new EDG users (i.e. developers of EDG applications) do not understand the reasons why many of their jobs fail. However, with increasing experience they understand the middleware better, learn the pitfalls and get better at 'second guessing' the system when understanding failures. Only after gaining this experience does EDG application development become easier and more rapid. We believe this is an area that will have to be improved in the future as new user communities come on board.

4 PORTAL DEVELOPMENT

While the Grid middleware provides low-level services and tools, the EO applications need to access the available Grid resources and services through user-friendly application portals connected to back-end servers. The back-end servers then access the Grid using

the low-level Grid middleware toolkits. The ESA proposed Grid Services for Earth Observation defines a generic infrastructure, that allows specific Grid data handling and application services to be seamlessly plugged in. Coupled with the high-performance, data processing capability of the Grid, it provides the necessary flexibility for building an application virtual community with quick accessibility to data, computing resources and results.

The ESA "Grid on Demand" portal demonstrates the integration of several technologies and distributed services to provide and end-to-end application process, capable of being driven by the end-user. The portal integrates:

- User authentication services
- WebMapping services for map image retrieval and data geolocation
- Access to metadata catalogues such as MUIS (ESA Multi-mission User Interface System) to identify the datasets of interest and access the AMS (ESA Archive Management System) archive to retrieve the data
- Access to Grid FTP transfer protocols to stage the data to the Grid
- Access to the Grid Computing Elements and Storage Elements to process the data and retrieve the results - all in real-time

The architectural design of the "Grid on Demand" portal application includes a distinct Application-Grid interfacing layer (Figure 4). The core of the interface layer is implemented by the "EO Grid Engine", which receives Web Services requests from Grid client applications and orchestrates their execution using the available services provided by several different Grids.

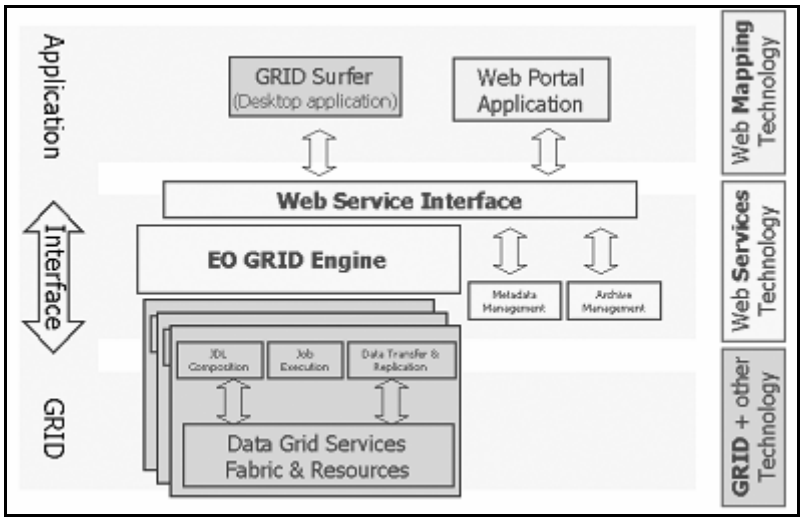

Figure 4. EO "GRID on Demand" Services: architecture model

The underlying GRID infrastructure coordinates all of the steps necessary to retrieve, process and display the relevant images, selected from a vast range of available satellite based EO data products. Using a new generation of distributed Web applications and OpenGIS specifications, the integration of Web Mapping and EO data services provides a

powerful capability to request and display Earth Observation information in any given time range and geographic coverage area.

This Earth Science user-friendly environment allows to:

- Support science users for focused collaborations as needed for calibration and validation, development of new algorithms, generation of high level and global products
- Provide access to computing power – shared processing and data storage resources – and shared datasets
- Provide the reference environment for generation of systematic application products coupled with archives and near real time data access

The application chosen to demonstrate the Web Portal calculates the ozone profiles using the GOME NNO algorithm and performs validation using ground based observation data. The user selects the algorithm, geographic area and time interval, and the web portal retrieves the corresponding Level-1 data orbit numbers by querying MUIS, the ESA EO product catalogue. Using the orbit numbers it is then possible to query a Level-2 metadata catalogue to retrieve the current status of the request orbits. The Level-2 orbits may be already processed, not yet processed, or currently being processed. In the first case, the Service Layer Broker searches the Grid replica catalogue to obtain the Level-2 data Logical File Names, and then retrieves the data from the Grid physical locations. The processed orbits are then visualized by the web portal (Figure 5). In the second case, the EO product catalogue also provides the necessary information to retrieve the Level-1 orbit data from EO archives.

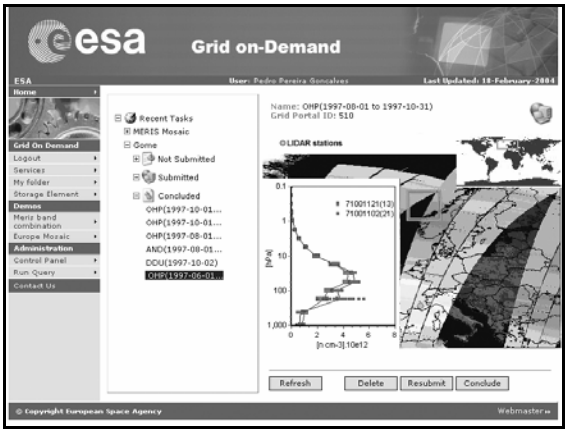

Figure 5. Web Portal Ozone Profile Result Visualization

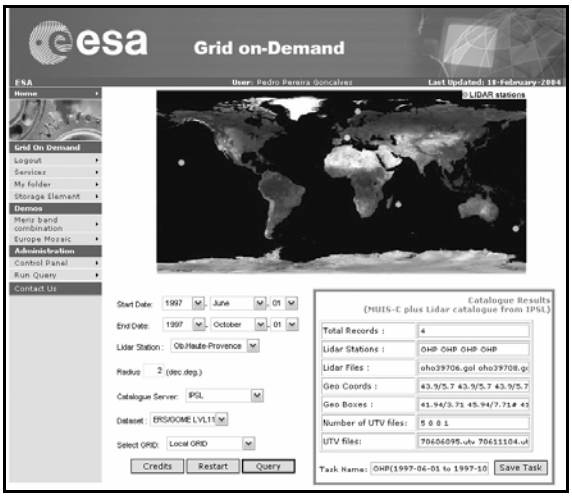

Figure 6. Web Portal Graphical User Interface for the Validation Application

After the Level-1 data has been transferred to Grid storage, jobs are submitted to the Grid to process the orbits. Once the processing has terminated the resulting Level 2 products are also transferred to Grid storage (from the WNs) and the Logical File Names are registered in the replica catalogue. A Level-2 metadata catalogue is also updated. In the third case (orbits currently being processed), the request ID is appended to the current job ID and awaits the job conclusion as in the second case.

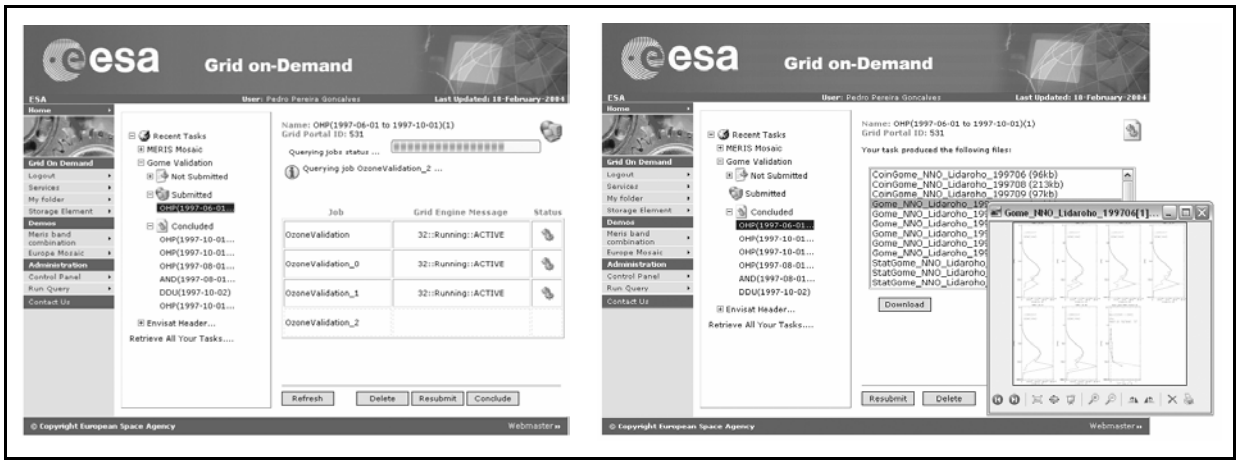

Figure 7. Web Portal Interface for Grid Job Status and Result Retrieval

For the validation application, the web portal has a dedicated graphical user interface (Figure 6) where the user accesses the IPSL LIDAR catalogue and crosschecks that information with the ESA catalogue. It returns the orbit information, LIDAR file names, and calculates the necessary geographical parameters for input to the validation job.

The input parameters are translated into grid job parameters, generating several jobs for each of the corresponding LIDAR files. The status of the different jobs can be viewed using the portal and when all jobs are terminated the web portal is used (Figure 7) to retrieve and view the results.

5 CONCLUDING REMARKS AND FUTURE

Several EO applications have been adapted to run in the DataGrid environment and application Grid interfacing tools have been developed to support their deployment. Several other application Use Cases have been developed, tested and evaluated using the EDG testbeds. The overall result of the project is a substantial body of Grid middleware, tools, knowledge and expertise that will provide a solid base for future developments.

The EU Datagrid project has been a unique opportunity to focus a large European scientific and highly technical community on a common technology framework. The Earth Observation community has been, involved and committed, with interest and dedication, in the process of understanding and testing the potential of the Grid paradigm, as applied in both the science and the routine operational environments.

Throughout its three years the project has taken giant steps forward in several key areas of Grid middleware, applications and integration/deployment. Several new and original Grid middleware services and components have been introduced, e.g. Resource Brokering and Job Submission, Replica Management and Optimization, Mass Storage Management, Grid Information, Networking, Fabric Management. Even considering the limitations of stability, scalability and performance, the hallmarks of an embryonic technology, the DataGrid testbed has been used successfully for carrying out real-world application Use Cases.

Deployment of a working Grid testbed that has been used to deploy several different applications with involvement of end-user groups can be considered a major achievement of the project and a milestone in the development of European collective computing technology.

Our evaluation testing shows that a more a reliable, stable and functional testbed, suitable for dependable Grid data production, is needed. However, the results of our technical evaluation should by no means be interpreted in a negative way. Not just the technical achievement, but also the lessons obtained are extremely beneficial.

On an individual component level, the middleware greatly improved after the v1.4 release, both in terms of new features and functionality as well as usability and reliability. The new Replica manager and Storage Element capabilities are very promising. The WMS is far more robust and functional and RGMA reliability has improved. We believe this trend will continue as the dynamics of a large-scale deployment are better understood.

The progress made so far needs to be consolidated, with further development and testing, guided by the results of our evaluations and feedback, that identify the main areas for improvements. In particular we welcome the implementation of our most outstanding requirements:

- Application metadata capabilities
- Logical collections
- Fine grained security control (including RM commands)

The EO community aims to continue working on Grid developments beyond the EDG project, the results of EDG will be fully exploited as the basis for future developments.

Dissemination and promotion of Grid solutions in the EO community has also been one of our chief aims throughout the project. As a result, Grid solutions are now being considered seriously as an appropriate means for large scale, collective EO scientific data processing, for modelling and simulation and as standard means for collaboration among the different organizations and users in the community. We foresee an enlargement of the potential user community in the near future as more internal investments are made to consolidate the results of RTD gained so far.

Following the interest and expectations raised as a result of several presentations made at workshops and conferences, of some significant scientific results obtained using the Grid for analysis of satellite-based atmospheric ozone data, we expect more EO scientific teams (and also other communities) will begin to deploy their applications on the Grid. For instance, applications in the fields of seismology and climatology need to handle large, regularly updated databases. Data may come from various types of observatories, e.g. ground based, airplane, and balloon measurements, or from simulation and modelling.

ACKNOWLEDGEMENTS:

The work described is funded by the European Commission program IST-2000-25182 through the EU DataGrid. The authors thank the scientists and engineers involved in DataGrid via their expertise, algorithms and their interest all along the project, especially from the University of Tor Vergata (Italy) F. DelFrate, from the University of Graz, (Austria), S. Casadio, from KNMI Ronald van de A, Roeland van Oss and Jos van Geffen, from IPSL Sophie Godin Beekman and Philippe Keckhut, and from Laboratoire de Physique Nucléaire et de Hautes Energies (France), Zaharia Strachman.

REFERENCES:

[1] Requirements specification: EO application requirements for GRID [DataGrid-09-TED-0101-1_0-Requirements]

[2] Earth Observation Use Cases for GRID [DataGrid-09-TED-0102-1_0-UseCases], [DataGrid-09-TED-0120-1_0-KNMI-UseCases]

[3] WP9.4 Use Case, annex to requirements specification: EO application requirements for Grid, [DataGrid-09-TED-0101-1_1], May 23 2002

[4] Results of running EO Applications on EDG 1.4 - Testbed Assessment [EU deliverable D9.3]

[5] Technical Note on EO Application Security Usecases. EDG Technical Note, June 2003. DataGrid-wp9-TN_security_usecases_v0_0_2. EDMS ID 398262

[6] Application Working Group Joint List of Use cases. joint-list-usecases-AWG-v2.1doc EDMS ID 386184

[7] EO application processing testbed demonstration and final report: Results of running EO Applications on EDG 2.0 [EU deliverable D9.5]

[8] W. Hoschek and G. McCance. Grid Enabled Relational Database Middleware Informational Document. Global Grid Forum Frascati, Italy, 7-10 October, 2001

[9] R. Alfieri, R. Cecchini, V. Ciaschini, L. dell'Agnello, A. Frohner, A. Gianoli, K. Lorentey, F. Spataro . VOMS, an Authorization System for Virtual Organizations. 1st European Across Grids Conference, Santiago de Compostela, February 13-14, 2003

[10] Application Working Group, Joint list of recommendations, EDMS ID 386183.

[11] Technical Note on Earth Observation Application MetaData Usecases. DataGrid-wp9-TN_metadata_v_0_0_1_5, September 2003. [EDMS Id 414416]

[12] van der A, R.J., R.F. van Oss, A.J.M. Piters, J.P.F. Fortuin, Y.J. Meijer and H. Kelder. Ozone profile retrieval from recalibrated Global Ozone Monitoring Experiment data. J. Geophys. Res. 107, 10.1029/2001JD000696, 2002

[13] Casadio, S., D. Del Frate, S. Godin-Beekmann, and M. Petitdidier, Grid Technology for the analysis of atmospheric ozone from satellite data. Proceedings of Data Systems In Aerospace (Dasia), Prague, République Tchèque, 2-6 June 2003.

[14] Godin-Beekmann, S., J. Porteneuve, A. Garnier, Systematic DIAL ozone measurements at Observatoire de Haute-Provence, J. Env. Monitoring, 5, 57-67, 2003.

[15] F. Del Frate, M. Iapaolo, S. Casadio, S. Godin-Beekmann and M. Petitdidier, "Neural networks for the dimensionality reduction of GOME measurement vector in the estimation of ozone profiles", soumis à Journal of Quantitative Spectroscopy and Radiative Transfer, 2003

[16] Godin Beekman, S., M. Petitdidier, C.Boonne, C. Leroy, S. casadio and F. delFrate, Validation des profils d'ozone dérivés des mesures GOME avec DataGrid, Proceedings of Experimentation and Instrumentation workshop, Paris, 23-24 mars 2004.

[17] Van Geffen, J.H.G.M., Documentation of the software package GomeCal (Version 1.0), Technical report TR-255, KNMI, De Bilt, The Netherlands, 2003.