# Benefits and Requirements of Grid Computing for Climate Applications. An Example with the Community Atmospheric Model

V. Fernández-Quiruelas[a], J. Fernández[a], A.S. Cofiño[a], L. Fita[a], J.M. Gutiérrez[b]

[a] *Dpto. Matemática Aplicada y C.C. Universidad de Cantabria. Santander, Spain*
[b] *Instituto de Física de Cantabria, CSIC-UC. Santander, Spain*

**Abstract**

Grid computing is nowadays an established technology in fields such as High Energy Physics and Biomedicine, offering an alternative to traditional HPC for several problems; however, it is still an emerging discipline for the climate community and only a few climate applications have been adapted to the Grid to solve particular problems. In this paper we present an up-to-date description of the advantages and limitations of the Grid for climate applications (in particular global circulation models), analyzing the requirements and the new challenges posed to the Grid. In particular, we focus on production-like problems such as sensitivity analysis or ensemble prediction, where a single model is run several times with different parameters, forcing and/or initial conditions. As an illustrative example, we consider the Community Atmospheric Model (CAM) and analyse the advantages and shortcomings of the Grid to perform a sensitivity study of precipitation with SST perturbations in El Niño area, reporting the results obtained with traditional (local cluster) and Grid infrastructures. We conclude that new specific middleware (execution workflow managers) are needed to meet the particular requirements of climate applications (long simulations, check-pointing, etc.). This requires the side-by-side collaboration of IT and climate groups to deploy fully ported applications, such as the *CAM for Grid* (CAM4G) introduced in this paper.

*Keywords:* Grid computing, Community Atmospheric Model (CAM), El Niño, sensitivity analysis, Workflow management

## 1. Introduction

Earth Science (ES) applications —in particular weather and climate models— are among the most computer-power and storage demanding disciplines; thus, they are key users of High Performance Computing (HPC) infrastructures, favoring their continuous growth and improvement. For instance, ES-dedicated supercomputers such as the Earth Simulator (`www.es.jamstec.go.jp`) rank at the top of the list of the world's most powerful computers[1]. However, during the last two decades new computing paradigms have emerged, such as Grid computing [15] and volunteer computing [3]. They provide an alternative to HPC for different problems facilitating the access to high capacity production-quality computing infrastructures to small groups or institutions.

Grid computing consists of a geographically distributed infrastructure gathering computer resources around the world in a transparent way [15]. Unlike volunteer computing projects, such as climate-prediction.net [2], where the applications (a global climate model in this case) need to be simplified and most of the results thrown away to avoid the overloading of the volunteer hosts, the Grid allows running a full state-of-the-art model and store the regular output information. This is done through a software layer, referred to as *middleware*, which allows for the transparent use of the distributed computing and storage resources which are seen as a single infrastructure. Thus, the most complex tasks of the Grid (security, authentication, resource discovery and allocation, storage, job execution) are managed by the middleware built on top of the infrastructure providing a simple and transparent interface for users.

In the last two decades, a number of computer-demanding applications in fields such as High Energy Physics (HEP) and Biomedicine have migrated towards Grid technologies as a complementary way to fulfil their increasing CPU power and storage requirements. Most of the problems and applications in these fields correspond to the so-called *production tasks*, where a single application is run many times with different parameters and/or input files. In those cases, parallel capabilities are used for the different realizations of a serial application, instead of the parallel execution of a single application. Many challenges have been achieved using Grid infrastructures to run production tasks; see, e.g. [23] in Biomedicine or the LHCb computing data challenge [30] in HEP. Although the Grid was initially though for both production and heavy parallel tasks, nowadays parallel execution is still dependent on the specific Grid infrastructure. This makes the process of migrating a parallel application to the Grid harder than migrating a serial one.

The ES Grid community, unlike the above mentioned fields, has been mainly concerned with data access and management. There are efforts aiming to develop Grid services for transparent discovery and access to heterogeneous data such as satellite data, model simulations or observations [see 9, and the documents of the DEGREE project `www.degree-eu.org`]. However, less effort has been devoted to the deployment and execution of applications such as a global climate model either for parallel or production tasks. Note that although the main need of a climate science user would be the parallel execution of a climate model, modern problems that involve large amounts of independent simulations such as ensemble prediction [32] and sensitivity analysis experiments [29, 4]

[1] The Earth Simulator ranked first of the world since its creation in 2002 until 2004. Moreover, computers at different national weather services can often be found at the top 10; see `www.top500.org`

correspond to *production* tasks appropriate to be deployed and run in Grid infrastructures. These problems have received increasing attention in the last decades due to their connections with the study of uncertainties, such as those related to seasonal prediction or climate change and its impacts on the different socio-economic sectors [33].

In this paper we give an up-to-date and user-oriented view of the Grid for the Climate community where the different applications have common needs. As an illustrative application, we describe an experiment with the popular Community Atmospheric Model [CAM; 8, 6] to test the sensitivity of the precipitation simulated in South America to sea surface temperature variations over areas affected by the El Niño phenomenon. As shown in Fernández-Quiruelas et al. [12], unlike other areas of research, the particular characteristics and requirements of climate applications become a challenge for actual Grid middlewares, posing new problems to the Grid: long execution times, multiple jobs with complex interdependencies, huge input files, etc. These particular applications need to be managed in terms of *ad hoc* implementations of execution workflow frameworks, building on the available middleware. For instance, in this paper we describe *CAM for Grid* (CAM4G), a port of CAM to the Grid including an execution workflow implemented using existing middleware services to organize and manage the execution of the climate model. This paper extends the capabilities of the prototype port of the CAM model to the Grid presented in [12] and provides a successful proof-of-concept experiment solving the problems which affected [12].

This paper is structured as follows. Section 2 describes the Grid including its main components, different solutions and the most important infrastructures available. It is intended for a potential user from the climate community and only covers the most basic concepts from the user's point of view. Section 3 provides an overview of both the benefits of the Grid for the climate community and the special requirements that a climate application poses on the existing Grid solutions. As an example, Section 4 describes *CAM for Grid* (CAM4G), a port of CAM to the Grid solving the special requirements of the climate model on the Grid. Finally, Section 5 presents a sample experiment using CAM4G to perform a sensitivity test consisting of 750 simulations successfully run on the Grid and summarises the statistics of the execution in the Grid environment compared to the execution on local resources.

## 2. The Grid Technology

Grid computing has recently emerged [15] as an alternative for flexible and secure access to heterogeneous and geographically distributed resources (computing clusters, storage units, etc.). Thus, for instance, in order to create a collaborative virtual community, several institutions that collaborate in a project with different resources (a computing cluster, storage units or databases) could agree to share them, granting access to users from other institutions. The way of optimising these synergies could be the creation of a Grid infrastructure that aggregates all the resources allowing the users to transparently access to a macro-system composed by all the processors and storage units of all the associated centres.

The analogy for this infrastructure is the power grid, where users plug their equipment obtaining energy in a transparent form, regardless of where and how it is produced.

This approach has several advantages for the users:

- Users take advantage of resources not fully used. In some institutions, clusters are used just a few hours per day or during some months in the year. Sharing the resources among several institutions will improve the usage capacity of the system. Institutions that have access to Grid will not have to be sized on peak load but can cleverly share the burden.
- Users are provided access to an enormous amount of storage space and computing resources difficult to reach by a single institution. This allows the research community to face new challenges that could not be achieved with traditional computing paradigms.
- Accessing geographically distributed heterogeneous resources in a homogeneous way make it easier for the user working with data or computing resources of other institutions. As we will see in section 2.1, Grid technology provides security mechanisms that manage the access to shared resources. System administrators find Grid technology helpful because they can rely on its security mechanisms to grant access to users. On the other hand, users can discover and access a vast amount of data sets distributed in several locations as if they were stored on a single computer.

### *2.1. Main components of the Grid*

In this paper we describe the Grid from a user's point of view. Technical details about Grid can be found in Foster and Kesselman [15]. A typical user from ES is accustomed to local cluster environments, where all resources are homogeneous and access to them is done through a unique account. In a Grid environment, each resource has its own users and may have different policies and systems. In order to provide the users transparent access to these distributed resources, Grid technology uses some services called middleware, that aggregates heterogeneous resources and present them as a single homogeneous system.

The most important part of Grid middleware are the core services, in charge of centralising the management of all the resources (see Figure 1). There are two basic services, authentication and authorisation (AUTH) and information (manages resource characteristics and status, INFO). These basic services are used by other core services in charge of centralising the access to each kind of resource or service (e.g. the data and execution services, labelled as DATA and EXEC, respectively, in Fig. 1). These resource-specific core services rely on the authentication and information services to make their decisions. In order to communicate with the core services some middleware has to be installed in the resources. Finally, some middleware user tools need to be installed in the user interface, in order to access the Grid services and infrastructures (as schematically depicted in Figure 1).

One of the main differences between working in Grid and in a traditional computing system is the authentication method. Grid users have a personal *certificate* instead of the traditional *user name* and *password*. This aspect of Grid often constitutes the task most difficult to understand by a non-experimented user, but because of it, all processes developed within a Grid infrastructure are highly secured. Personal *certificates* are X509 certificates [41] signed by Certification Authorities (CA) that have previously checked that the user belongs to the institution he claims to be part of. This certificate is password protected to ensure that only the owner of the certificate can use it to access the resources. To avoid typing the password every time the user carries out a transaction, a time-limited proxy —which is a self-signed copy of the certificate [43]— is used automatically in all the transactions for a limited time period. This security infrastructure is known as Grid Security infrastructure [GSI; 16].

Grid users are organised in so-called Virtual Organisations [VO; 17], where they register their *certificates*. A VO is just an entity that maintains a list with the *certificates* of all the users that belong to it along with their roles and groups. The VO is queried by the resources in order to determine if a user can access it or not. Usually, VO members share something in common (work in the same project, organisation or research topic) regardless their physical location. In large Grid infrastructures, where there are many VOs and institutions, not all the resources are shared among all the VOs (e.g. a meteorological center may only share its resources among the Earth Science VO). In several cases, such as when confidential data sets are shared, other VO features such as groups and roles may be used for fine-grained access to the resources.

From the user's point of view, the job submission to a Grid infrastructure works the same as in a local cluster or a supercomputer: the user fills a template with the job requirements and the executable to be run and submits the job to a queue using the middleware user tools. The storage and access to data is done through a virtual file system which maintains a relationship between the logical names in a virtual structure and the sites where the data are stored (multiple copies). This way, the data is replicated and distributed through the different sites and the Grid middleware selects the particular copy to be used for a particular execution according to, for instance, proximity to the execution node. Furthermore, to avoid data loss and to improve efficiency, the virtual filesystem can automatically manage replicas of the files. The user can transfer/download files to/from the Grid though GridFTP, a new protocol based on FTP [34] and GSI [1] for the Grid.

The technical requirements to take advantage of a Grid infrastructure depend on the level of involvement. There are at least 3 levels of involvement in a Grid infrastructure:

1. The minimum requirement for a new user to start using a Grid infrastructure is to have a personal certificate and join a VO. If the infrastructure provides (traditional) access to a user interface (a machine with the user

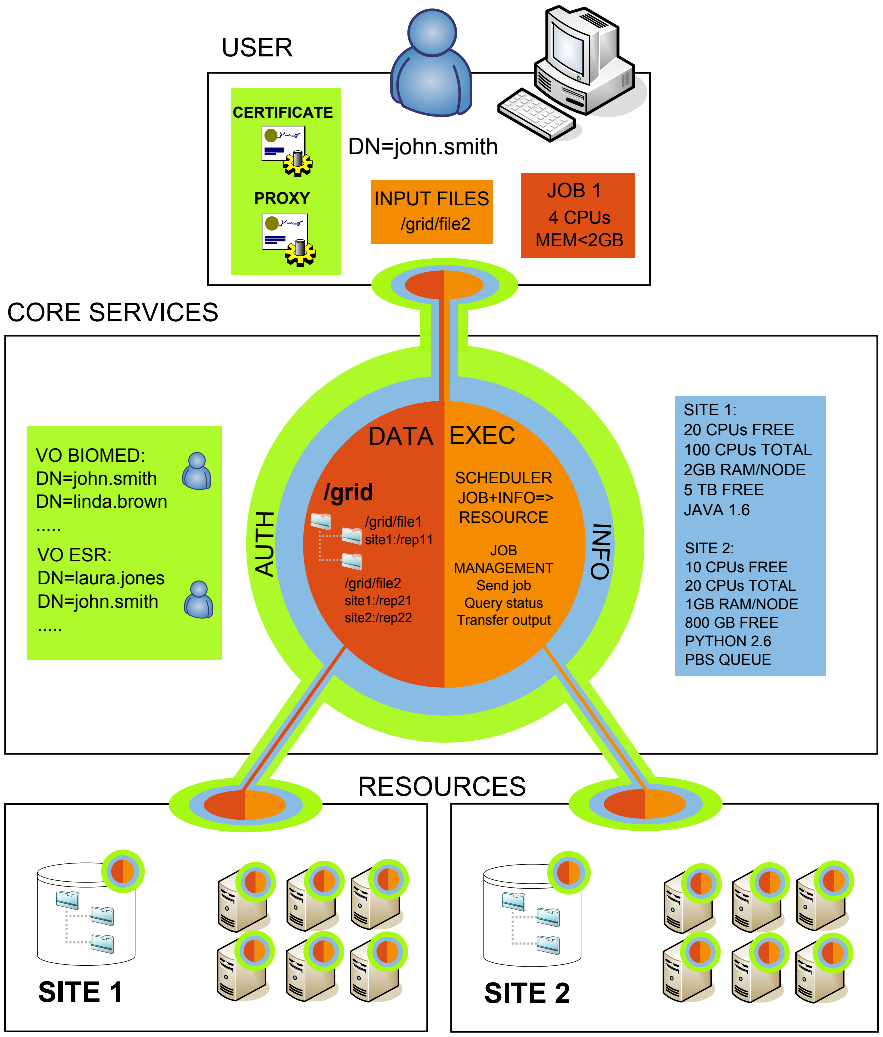


Figure 1: Schematic representation of the Grid. There are 3 main layers: resources, core services and user environment. All of them make use of a given middleware in order to communicate with the others (see Section 2.1).

middleware installed) this would be enough to start using the infrastructure. Otherwise, the user must install the user middleware and configure it to use this infrastructure (see Fig. 1, top).

2. If an institution wanted to share their resources in the Grid infrastructure, they would have to install the resource middleware in their resources and configure them to interact with the core services of the infrastructure (see Fig. 1, bottom).
3. If the institution wanted to create a new Grid infrastructure (e.g. joining all the resources from all the departments), in addition to installing the user and resource middleware, they would have to install the core services in charge of giving transparent access to the resources (see Fig. 1, middle).

New users interested in using Grid resources may start by contacting the national Grid initiative of their respective countries.

### *2.2. Middleware implementations*

Nowadays, there are several Grid middleware implementations that provide seamless access to distributed resources.

The first Grid middleware, Globus Toolkit [`www.globus.org`; 14], was developed in the 90's in the United States and it is currently one of the most used implementations among the academia and industry. A middleware based on Globus Toolkit, gLite (`glite.web.cern.ch`), was created under the scope of the EGEE project in Europe (Enabling Grids for E-sciencE, `www.eu-egee.eu`). It is the middleware used in most of the European Grid initiatives. The application workflow presented in this study has been deployed using gLite.

gLite defines middleware packages or roles for each service. It provides 4 different roles for the core services. The Berkeley Database Information Index (BDII) is the information core service, Virtual Organizations Management system (VOMS) is the authorization service and the Large Hadron Collider Grid File Catalog (LFC) and Workload Management System (WMS) are the data and execution core services respectively. The users interact with them through a computer where the User Interface (UI) role has been installed. Users can install their own UI (usually UIs can be downloaded as a virtual machine) or access the UI of the infrastructure. Each institution can join its computing cluster to a Grid infrastructure by installing the Computing Element (CE) and Worker Node (WN) roles in the head (the single point of management and job scheduling for the cluster) and computing nodes of their cluster respectively. Note that in order to ensure an easy installation and configuration, gLite only supports certain platforms and Operating Systems for each role. Currently, the WN middleware can only be installed on x86_64 computing nodes with the Scientific Linux 5 or Debian 4. In order to interface with the local storage system the Storage Element (SE) role can be used.

There are many other special-purpose middleware implementations, such as UNICORE (`www.unicore.eu`), which was initially developed to join German supercomputing centres.

### 2.3. Grid infrastructures

Although Grid middleware can be used in several scenarios to join different resources (in some cases just 2 or 3), in this paper we focus on large heterogeneous Grid infrastructures that join several institutions geographically distributed.

The largest Grid infrastructure in the world is the one created under the European project Enabling Grids for E-sciencE (EGEE, `www.eu-egee.eu`). It started in 2004 with the goal of aggregating as many as possible computing and storage resources from different organisations in order to face the challenge of storing and analysing the data produced by the CERN's Large Hadron Collider (LHC). Nowadays, it aggregates 150.000 processors and 41 PB of storage distributed in 260 sites all over the world using the gLite middleware. The use of the EGEE infrastructure is not only limited to the HEP community. Today, there are thousands of users distributed in more than 200 VOs that comprise several disciplines (Biomedicine, Earth Sciences, Astrophysics, etc ...).

As EGEE, other EU-funded projects have aggregated European resources within Latin America (EELA projects, `www.eu-eela.eu`), Asia (EUAsiaGrid project, `www.euasiagrid.org`), South Eastern Europe (SEE-Grid, `www.see-grid.org`), etc.

Apart from EGEE, that joins commodity data and execution resources, there are other large infrastructures more focused on joining supercomputing centres. For instance, DEISA (Distributed European Infrastructure for Supercomputing Applications, `www.deisa.eu`) puts together 11 of the most important supercomputing centers in Europe using the UNICORE middleware. As DEISA, TeraGrid [`www.teragrid.org`; 5] interconnects 11 American institutions using high performance networks and has, nowadays, a computing capacity over 1 PetaFlop and 30 PB of storage.

With respect to the climate science community the most representative infrastructure has been the Earth System Grid [ESG 44]. ESG is focused on facilitating the access to more data for climate scientists. This data comprise more than 200TB of climate data and is distributed to more than 10000 users registered in the ESG portal.

## 3. Grid for the Climate Modeling Community

Climate science community already benefits from technologies like the Web and is starting to benefit from the Grid to manage the increasing amount of data produced. For instance, Web services were rapidly adopted and nowadays provide data from many international climate initiatives. Successful examples are ESA G-POD [18] and ESG [44], `earthsystemgrid.org`). Renard et al. [37] and Cossu et al. [9] offer recent reviews mainly focused on data. However, the use of Grid infrastructures to perform large experiments that make intensive use of the computer power is in a more incipient status. Only a few efforts have been reported to adopt the Grid technology to execute applications [26, 28, 39]. An updated overview of this problem has been analysed in the DEGREE project

(Dissemination and Exploitation of GRids in Earth sciencE, www.eu-degree.eu).

The computer power and storage provided by a huge Grid infrastructure such as EGEE allows the climate science community to face new challenges. This is particularly important for emerging countries (e.g. in South America and Asia) which could easily use the existing Grid infrastructures, such as those of EELA (www.eu-eela.eu) and EUAsiaGrid (www.euasiagrid.org). Moreover, due to the complexity of the climate model applications there is an inherent difficulty of migrating these applications to other computing infrastructure. One benefit of Grid technology is that once an application has been migrated to a Grid infrastructure, the user will find very easy running it in every computing element of this Grid infrastructure or the new ones joining in the future.

However, further research is necessary in order to adopt the applications from the climate modelling community due to their high productivity and high performance requirements. The specific characteristics and requirements of climate modelling applications pose new challenges to the Grid. Today, the existing Grid middleware does not meet many of the requirements climate models demand to properly run in Grid infrastructures. To overcome this situation, particular *ad hoc* solutions are developed to adapt each experiment to run in Grid [39, 11].

Considering that most climate models face the same problems to run in Grid, the development of a generic framework that meets these requirements would be desirable. With this aim, Fernández-Quiruelas et al. [12] devised a first prototype of the framework and performed some experiments using the CAM model. This helped us to detect the weaknesses of our prototype and to establish the requirements the framework had to fit. The following Section summarises these requirements.

### *3.1. Requirements for climate modeling*

One of the main issues of the Grid is the heterogeneity of computing resources, which may be a critical fact in order to properly run long executions managing large amounts of memory and data [see, e.g. 12]. Moreover, most clusters in the Grid have limitations regarding: CPU time (the processor time spent, not counting the time waiting for input/out operations or for the availability of resources), wall time (the real time spent running in the queue), disk usage, memory usage, etc. These limitations may force the premature end of a job. Furthermore, it is also common to find missconfigured resources, due to the large number of sites and administrators involved. Regarding data transfer, when sites are scattered all over the world, network bandwidth becomes critical.

Some typical applications from disciplines such as bio-medicine or HEP are short-time simulations that do not manage large datasets nor need a huge amount of memory or disk space to be run. Thus, if a simulation fails, it is sent again to the infrastructure with minimum impact on the whole experiment. By contrast, ES applications usually require running complex models during days, consuming a lot of memory and generating large amounts of data. If these simulations were sent directly to the Grid, it may happen that none of them finished due to the limitations explained before (memory, CPU, disk limits). Moreover, climate models highly interact with data resources requiring the data sets to be intelligently replicated; otherwise, models may expend more time downloading and uploading data than running. This is why it is necessary to do some changes in the workflow of the applications in order to adapt them to overcome these limitations.

The most important requirements for a successful climate Grid application are [12]:

- Failure awareness: The application has to foresee all the possible sources of failure (including wall time and CPU time limitations) being able to face them or at least detect them and act in consequence.
- Checkpointing for restart: In case of failure, due to the computational cost of climate applications, one would want to restart the simulation in a different working site from the point it was interrupted (or as close as possible). This is done by writing intermediate recovery files to disk at a given frequency.
- Monitoring: Since climate simulations last for a long time, the user requires to know the current status of the experiment and their associated simulations: which percentage of the experiment is complete, whether there are simulations running, which time step is being calculated by a simulation, which data sets have been produced and in which storage elements are they, which is the last checkpointing/restarting point, etc.
- Data and Metadata storage: The goal of the climate model experiments is the generation of (large amounts of) simulated climatic information. This information needs to be post-processed and analysed by the different tools used by the climate researcher. Therefore, the data has to be easily accessed by users. A data and metadata management system has to be developed to handle all the information generated.

The above requirements made necessary the development of a goal-oriented workflow manager in order to run the experiments with a minimum of human intervention.

As mentioned, the current Grid middleware does not fulfill these requirements. Therefore, the development of a new framework is necessary to use the current Grid resources and infrastructures by climate modeling applications. This framework has to address all the previous requirements which, at the same time, must be transparent and easy to use for the end user (usually not a Grid expert). With these ideas in mind, the CAM4G application has been developed, which is a *Grid workflow management layer for the climate simulation with CAM*. CAM4G is described next as an illustrative example of how a state-of-the-art climate application has been ported to the Grid.

## 4. CAM for Grid: CAM4G

The Community Atmospheric Model [CAM; 8, 6] is the atmospheric component of the Community Climate System Model [CCSM; 7], which is a coupled atmosphere-ocean global climate model (AOGCM). CCSM3 is a state-of-the-art climate model developed at the National Center for Atmospheric Research (NCAR) of the U.S. and used e.g. to simulate future scenarios in the latest (4th) assessment report of the Intergobernmental Panel of Climate Change [IPCC; 35]. We deal only with the atmospheric component (CAM3) coupled with the land surface model (CLM3). A relatively coarse T42 (approx. $2.8° \times 2.8°$) resolution is used in order to simulate our experiment in a reasonable time and to be able to use the largest amount of grid resources. The CAM3 model is open-source, it is coded in Fortran and is available from `http://www.cesm.ucar.edu/models/atm-cam`.

Fernández-Quiruelas et al. [12] (hereafter referred to as FQ09) presented an initial prototype of a framework to run the CAM model on a Grid environment. In this first attempt, the gLite middleware was used to build the framework. The data management was controlled by the LFC server, and the monitoring system was handled by AMGA (gLite Grid Metadata Catalogue). With this prototype FQ09 discovered that the implementation had a bottleneck in the data management and that the monitoring system had to be improved.

CAM4G is a new implementation of CAM for Grid, improving the FQ09 execution workflow by adding new data management and monitoring capabilities, as described in this section.

From the user's point of view, CAM4G has 3 hierarchical components: (1) The *experiment* to be carried out with the model, designed to answer some scientific question, usually by means of an ensemble of (2) *realizations*, that will be carried out in a single or, most probably, several (3) *Grid jobs*. The term *realization* refers to the independent pieces an experiment can be divided into. A Grid job cannot be related one to one with a *realization* since realizations cannot be guaranteed to finish in a single job. In general, a *realization* requires several Grid jobs to be completed, each one restarted from the previous one. Thus, from the point of view of the workflow, the *realizations* are independent tasks to be carried out on the Grid and the jobs spanning a *realization* are dependent tasks.

In order to submit an experiment with CAM4G, the user only has to fill the experiment details in a configuration file, prepare the input data and submit the experiment to the Grid using the CAM4G user tools or the web portal. During the execution of the experiment, the user can check the status of the realizations conforming the experiment and access the output data while they are produced by the running jobs. Failing jobs are restarted in an unattended way until each realization is completed. To achieve this transparency for the user, a complex execution framework has been designed. This framework has been built from scratch by adapting well-known Grid services to our needs and creating new modules for the tasks that the existing middleware could not manage (see Figure 2). In order to provide the monitoring capability, we retrieved all the events in the workflow and consolidated them in a self-developed monitoring system based on MySQL (the database system used by FQ09 did not fulfill our requirements [ ►OJO: porque? ] ). Regarding the new data management in CAM4G, after analysing the middleware solutions provided by gLite, we decided to create a replica service with the aim of optimising the data transfers (using a system that finds the nearest replica of a file). The job execution is managed by GridWay [21], a flexible job meta-scheduler.

The monitoring system is fed with the information retrieved by two monitors (execution and data) that are started with the job in the computing node. Apart from giving the realization status to the monitoring system, the execution monitor interacts with GridWay to overcome all the possible job failures and reschedule the jobs. The data monitor detects when new output or restart data are created and uploads it using the replica service.

The framework presented here has been applied to other models such as the Weather Research and Forecasting limited area atmospheric model [WRF4G; 13]. Although CAM4G provides a precompiled serial version of CAM3, thanks to this framework, users could run their own compiled code in the Grid. It is important to note that although there are some production experiments using this framework, currently, it is just a prototype. At the moment, CAM4G has been tested in Globus and gLite infrastructures, and it supports x86 and x86_64 systems running Linux (tested on Scientific Linux, CentOS, RedHat and Debian). Further efforts are being made in order to adapt the framework to other arquitectures and operating systems. As soon as CAM4G is fully documented, it will be launched under an open-source license. All the components of the framework, including the model itself, are open-source.

Figure 2 shows an schematic illustration of the CAM4G components, using the Grid representation shown in Fig. 1. The top of the figure shows the web environment from where the user can submit and monitor the realizations and manage the data. In order to carry out these tasks, the user's web environment make use of the CAM4G core services (see the middle part of the figure): The job execution workflow is managed by Gridway, the data by the replica service and the job status is retrieved from the monitoring service. For instance, in the example, the user has submitted an experiment composed by 3 realizations that have been scheduled and sent by Gridway to 3 worker nodes in 2 different sites (each site shares a cluster with one or more worker nodes). The bottom of the figure shows how the jobs run in the computing resources (WN001, WN002, ...) wrapped by the data and execution monitors. These monitors transfer the relevant information to the monitoring core service, upload the output and restart data produced by the job and interact with Gridway to overcome the possible job failures. If a job fails in a computing node, the execution monitor will detect it and will notify Gridway and the monitoring service. Then, Gridway will send the job to another site and download the data for restarting the job from the nearest replica.

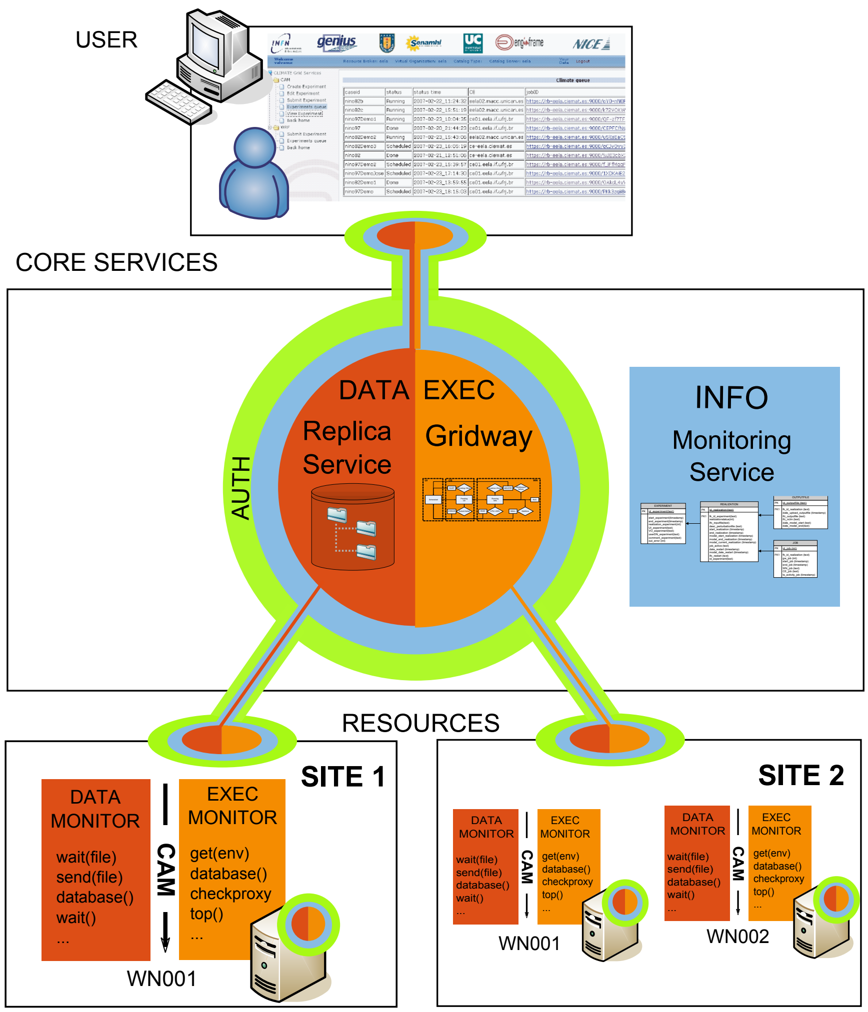


Figure 2: CAM4G framework components. From the user interace the user manages the data and jobs and monitors the experiment. This is done thanks to the core services in charge of managing the jobs (Gridway), data replicas (Replica Service) and the experiments information (Monitoring Service). Jobs are executed in the WN wrappered by 2 monitors (execution and data monitors).

## 5. An Illustrative Experiment with CAM4G

In order to illustrate the performance of state-of-the-art Grid computing for the climate community, in this section we present the results obtained using the EGEE Grid infrastructure (Section 2.3) to run a sensitivity experiment involving the execution of 750 19-month simulations of the CAM model (T42 resolution) with varying prescribed sea ice and sea surface temperature (SST). The goal is to analyze the effect of El Niño SST forcing in the accumulated precipitation. The El Niño phenomenon consists of an anomalous heating of the eastern pacific ocean, which has an associated atmospheric circulation counterpart known as the Southern Oscillation (both oceanic and atmospheric components are referred to as El Niño/Southern Oscillation or ENSO). ENSO events occur every 2 to 7 years and affect the global circulation, changing e.g. the rainfall patterns in distant regions. This phenomenon has huge social impact since it is related to flood and drought events in different regions (e.g. in several south American countries). CAM3 has already been used in previous works to study El Niño responses with the same T42 resolution uncoupled version [24] and also comparing different resolutions [45] or the coupled and uncoupled versions [20].

### *5.1. Description of the experiment*

As a first step, we computed an El Niño SST perturbation pattern using the mean SST anomaly in the tropical Pacific ocean given by the two strongest events recorded (1982 and 1997), with respect to the long term SST climatology. This SST pattern was scaled by its maximum grid value ($2.5K$). The resulting normalised anomalous SST pattern (hereafter NAS pattern, Figure 3) was applied to generate perturbed SST fields that were used as boundary conditions in our CAM4G ensemble. For instance, if the NAS pattern is multiplied by -2.5 and added to the observed SST, the El Niño anomalous signal will be removed. If it is multiplied by a negative scaling parameter $-2.5 < s_n < 0$, the El Niño signal will be weakened. Values above zero intensify the SST anomaly producing record-breaking ENSO events. We generated 750 perturbed SST distributions by randomly selecting scaling parameters $s_n$ in the range $[-2.5, 2.5]$ from a uniform distribution:

$$\mathrm{SST}_n(t,x) = \mathrm{SST}_{obs}(t,x) + s_n \mathrm{NAS}(x), \qquad n = 1 \dots 750.$$

That is, we are sampling SST distributions from normal conditions ($s_n \approx -2.5$) to an ENSO-like SST anomaly around twice as strong as that observed in 1997-98 ($s_n \approx 2.5$). The large number of simulations allows a quantification of the internal variability of the model (sensitivity to small variations in the boundary conditions) as a reference for the changes observed as the SST changes. We focused on the eastern tropical pacific, where most of the circulation variability can be explained by the SST variability in AMIP-type simulations [25].

The atmospheric and soil initial conditions for all the ensemble realizations were the same. They were obtained from a previous model run which started from climatological conditions on 1st January 1990, and was forced by the observed SST and sea ice for one decade, in order to properly spinup the soil

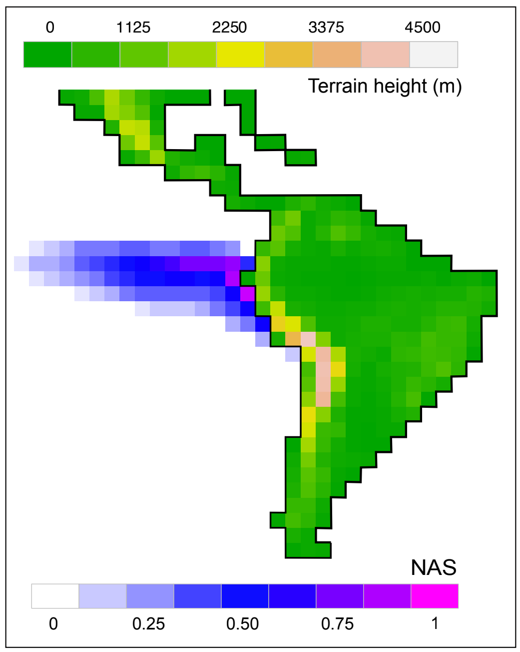


Figure 3: Terrain elevation as seen by the CAM model (over land) and the normalised anomalous SST (NAS) pattern used to perturb the SST (over sea). The insets show the sensitivity of precipitation to the SST perturbation at different grid points.

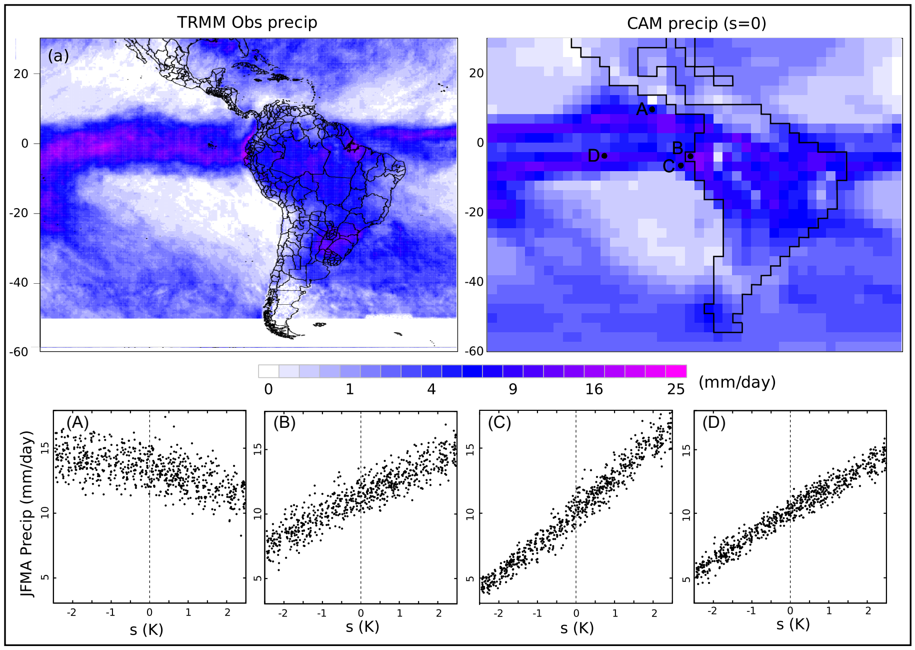


Figure 4: (a) Mean observed precipitation (mm/day) from January through April 1998 according to TRMM data vs. (b) CAM simulated precipitation for a realization with s=0 (unperturbed simulation). The scale is square-root, to appreciate low precipitation areas. Panels A-D show scatter plots of the precipitation vs. the perturbation for the four different grid points shown in the figure.

component [AMIP-type simulation, 19]. The resulting SST-assimilated atmospheric and soil state on January 1st, 1997 was used as initial condition in our experiment. The simulations run for the 19-month period, up to July 1998. This period includes one of the strongest El Niño events observed to date [27].

As a sample analysis we focused only on precipitation averaged from January through April, when the largest precipitation in coastal Peru occurred [40]. This region is specially sensitive to ENSO events, carrying floods to places where usually there is few or no rainfall [42]. Figures 4a and 4b compare the observed precipitation according to TRMM [36] data with the precipitation simulated by the model when the SST is as observed (not perturbed). CAM simulations underestimate the observed mean precipitation in the period considered. Although the main precipitation pattern is well reproduced, there are several deviations. The tropical rainbelt associated to the ITCZ appears in the simulation split into two. This is a recurrent problem in coupled and uncoupled GCMs [46, 10, 22] which is not related to their execution on the Grid. Also, the precipitation maximum north of Paraguay was not reproduced in the simulations.

In order to analyze the changes produced by the intensity of the SST ENSO perturbation ($s_n$) on the simulated precipitation in this region, we chose four illustrative locations (labeled as A, B, C and D in Figure 4b). For each of these grid points we considered the ensemble of 750 simulations performed using the Grid and displayed the scatter plots showing the precipitation value vs. the perturbation (see Figures 4A-D). These figures exhibit a linear trend in most of the cases, although there are also some nonlinear responses (see Panel A). Therefore, in order to test the sensitivity of the results we did not consider the slope of the corresponding fitted regression line, but the Spearman correlation coefficient, as shown in Fig. 5 for the global domain. Thus, positive values indicate increasing precipitations with higher perturbations, whereas the negative values indicate decreasing trends. This figure shows a complex sensitivity pattern with highly correlated regions (both negative and positive; see the scatter plots) as well as intermediate positive ones particularly in midlatitudes. The highest responses are mainly located over the region where we added the SST perturbation. However, there is also a significant response over southwestern Africa; this ENSO sensitivity over Africa was documented long ago [38], and has been related to SST variations over the Atlantic and western Indian ocean [31]. In the model, this sensitivity appears even though only the eastern pacific SST was modified.

Fig. 5 also shows the scatterplots of precipitation change vs. the perturbation intensity for those grid points with largest positive (panels E and F) and negative (G and H) correlation values. From this figure it can be clearly shown the existence of nonlinear responses (panel F). Further analysis is needed for a detailed comprehension of this pattern, but this work is out of the scope of this paper.

The internal variability of the model due to small variations of the SST is related to the thickness of the scatterplots. A measure of this thickness was obtained both globally, by removing the linear trend at each grid point and computing the standard deviation, and locally, computing the standard deviation of those points in a window of $\pm 0.1K$ around the zero perturbation value ($s = 0$), obtaining similar results. Figure 6 shows the variability obtained with the later approach. Again, three scatterplots with the grid points with largest variabilities are also shown. Note that the variability is not directly related to the precipitation intensity. For instance, Figures 6 J-L correspond to grid points with very different precipitation amounts, but exhibiting similar variability.

Therefore, Figures 5 and 6 provide different sensitivity information about the relationship of precipitation amount and SST perturbation intensity. The former provides an estimation of the increasing or decreasing trends associated with large perturbation values ($\pm 2.5K$), whereas the later provides information about the variability of the result for a small perturbation ($\pm 0.1K$). Note that a high number of simulations (750 in this example) is required to appropriately estimate both quantities. In a previous attempt [12], the number of successful simulations were not enough to distinguish the signal of the response to the SST perturbation from the noisy internal variability. This stresses the need for a large number of simulations and the benefits of Grid computing for this kind of experiments.

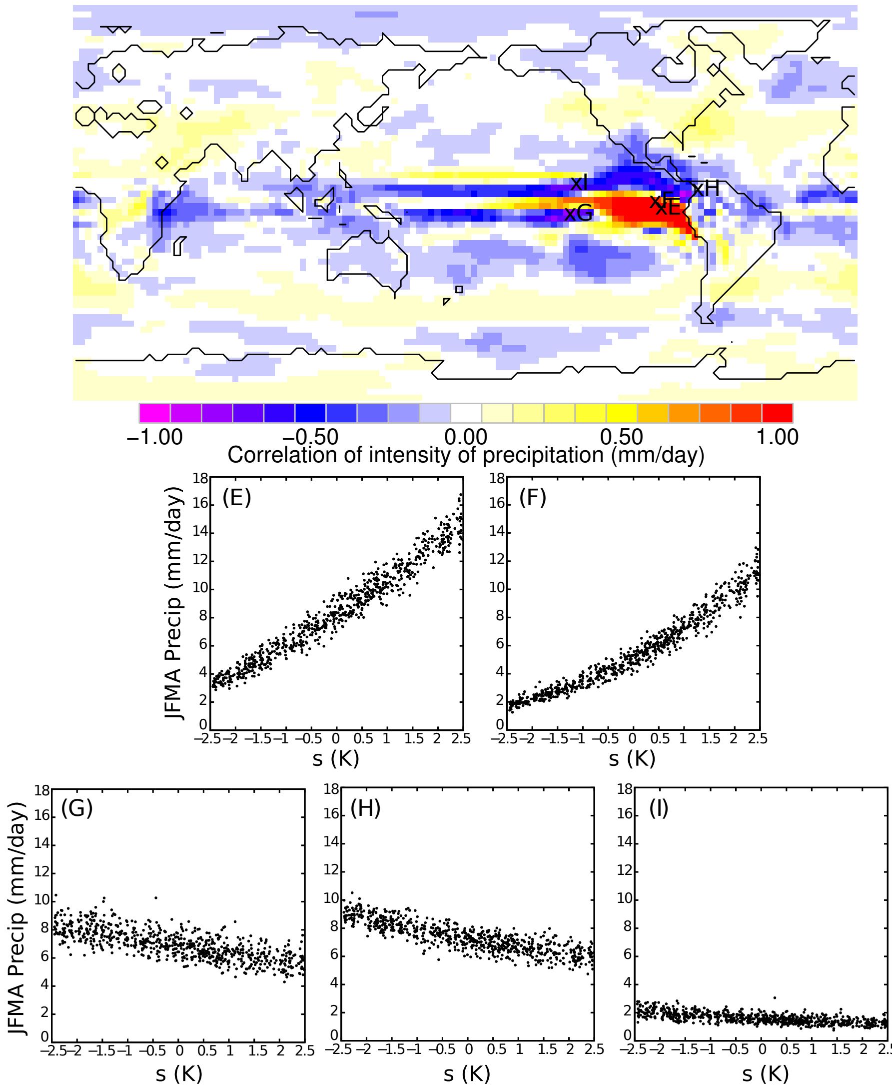


Figure 5: Spearman correlation between January–April mean precipitation (mm/day) and the perturbation intensity (K) for each model grid point (see text). The scatterplots correspond to the 750 simulated values for five illustrative locations shown in the map.

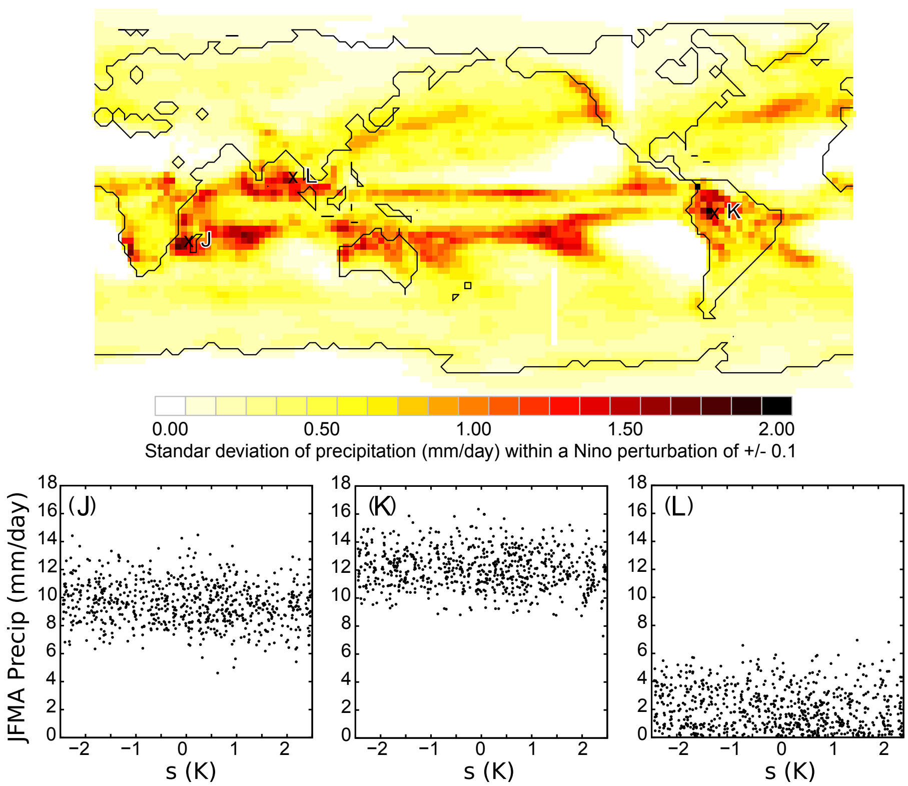


Figure 6: Internal model variability, obtained as the standard deviation of the points in a window of $\pm 0.1K$ around the zero perturbation value ($s = 0$).

### *5.2. Job execution statistics*

The above example consisting of 750 realizations was run on the EGEE Grid infrastructure (Section 2.3). The serial version of CAM was used to perform this experiment (each realization only required one core to execute). In order to compare the efficiency of the Grid with traditional computing resources, we run a realization using one core of our local cluster (16 nodes with 2 Intel Xeon E5410 CPUs –totalling 128 cores– and 8GB of memory). This realization took 44 hours to complete and, thus, if we would have send the 750 realization to our cluster, the whole experiment would have taken all of our computing resources for 11 days.

A pre-screening of the available sites was done to meet the requirements of the experiment. Each realization runs for about 44 hours and requires around 200MB of input data. Sites with short job wall time should be avoided since, in that case, most of the time would be spent by the download of the input data and resubmitting the jobs instead of running. We established a requirement on the wall time to be over 12 hours. Given the large amount of input data, we selected sites with large bandwidth and, among those, we chose the 9 sites with the faster cores.

Once the sites were chosen, the input data were replicated in 6 European sites. In this way, the network load would be distributed when the 750 realizations started to run. One of the main advantages of the CAM4G framework is that it is prepared to locate the nearest replica from a given location. This feature is very useful also when the output data and restart files are uploaded to the storage elements.

The 750 realizations were submitted at the same time. As shown in Figure 7, in half an hour the first 400 jobs had started to run (i.e. the input files were already downloaded) in the computing nodes. The rest of the jobs started to run as resources were available. Some of them were queued for some time in the clusters and others spent a long time to download the input files. After 6 hours, all the 750 jobs were running.

The first set of realizations finished in 42 hours. Each of the realizations that finished before 48 hours was run in a single job (the realization did not need to be restarted and ran in a single attempt). This implies that the sites where they were running had a wall time larger than 42 hours and no problem was found during the execution. The rest of the realizations had to be restarted at least once and spanned at least 2 Grid jobs. In these cases, the realization started to run, and before finishing, the job failed (usually the wall time limit had been exceeded). Then, the framework detected it and submitted another job that continued the simulation from the last restart point stored by the failed job. In the worst case, a realization required 6 jobs to complete, but most of the realizations spanned less than 4 jobs (Figure 8a). The realizations took between 40 and 85 hours to complete (Figure 8b). The computing time differences among realizations were due to several factors including the CPU speed, queue wall time (increases the number of restarts required), errors during the execution and bandwidth differences.

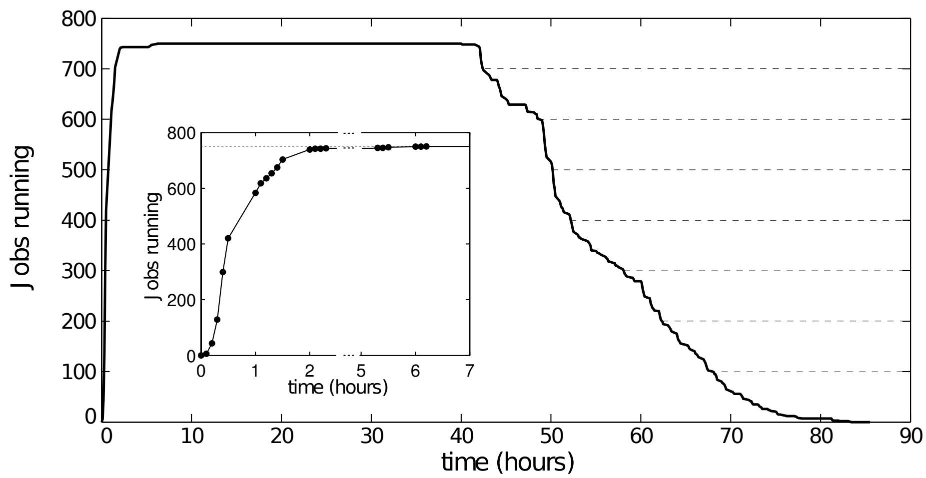


Figure 7: Number of jobs running at a given time (in hours from the experiment submission). The inset shows a zoom of the initial 7 hours.

The experiment was finished in 3.5 days and all the 750 realizations were successfully completed.

## 6. Conclusions

In this article, a wide and general introduction to Grid computing is presented having in mind a climate science researcher used to work with local clusters. Thus, we focus on those aspects which are different when using the Grid than when using a local cluster (e.g. login, transferring data, submitting jobs, etc.). For this purpose, the three main layers of the Grid (user, core services and resources) have been presented and the new protocols of security and authentication services to access a Grid infrastructure have been described (certificates, virtual organizations, etc.). It has also been introduced the new concept of *middleware*, which provides services for a transparent and clear access to the heterogeneity of resources of a Grid, and *GridFTP*, as a protocol to distribute and access data in the Grid.

Moreover, it has also been described the existing state-of-the-art Grid infrastructures (such as the European EGEE initiative, with more than 10000 CPUs from 50 different sites) and the different ways to aggregate them through Virtual Organizations (VOs). A review of the work done within the climate community (included in the Earth Science VO) has been done, pointing out that most of the attention has been focused on data management. Thus, further work is needed to deploy and run climate applications (such as climate models), which is still a challenge for actual Grid infrastructures.

This paper has focused on this problem, analysing the main requirements of climate applications (failure awareness, checkpointing for restart, monitoring and data and metadata management, etc.). As a result of this analysis it was

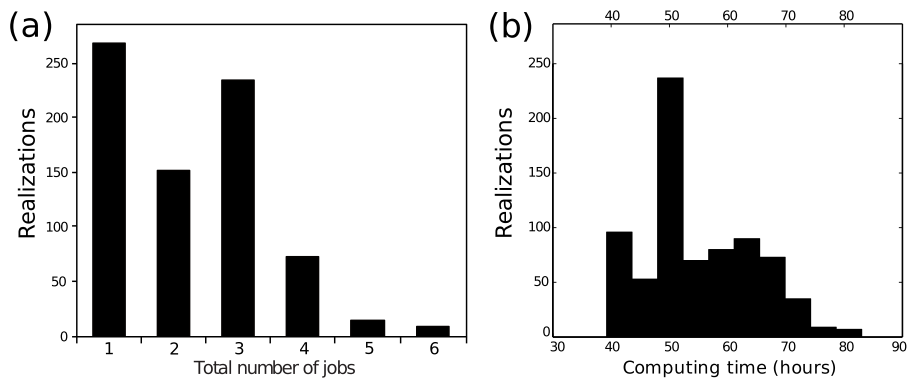


Figure 8: (a) Number of jobs required by each realization to complete. (b) Histogram of the time (in hours) required by each realization to complete.

concluded that new middleware components (execution workflow managers) are needed to cope with the particular requirements of these applications to run efficiently on the Grid. For instance, as an illustrative example, we described a new Grid execution workflow for the Community Atmospheric Model (CAM) wrapping the model and allowing to restart interrupted jobs, to manage the data and to monitor the running experiments. Moreover, in order to demonstrate the performance of this Grid-deployed model, a real computing challenge of a climate research experiment has been designed. The experiment consisted in a sensitivity analysis of global precipitation to perturbations in the sea surface temperature (SST) in the Niño region, considering a total of 750 perturbed simulations (realizations). Results show that precipitation sensitivity is higher in the areas where SST was modified. However, world-wide teleconnections of El Niño signal are found, since some sensitivity is also found in some other places like in West Equatorial Africa. It is also illustrated that precipitation shows both linear and nonlinear responses to the strength of the Niño signal.

To quantify the benefits of Grid computing, statistics of Grid execution of the experiment are given and compared to the computational cost on a local cluster. It was shown how the 11 days required to finish the experiment (all 750 simulations) in the authors' cluster is reduced just to 3.5 days in the Grid. At the same time, statistics of the different realizations are also shown describing that in general individual realizations require about 2 or 3 Grid jobs to be finished, consuming an average of 50 hours, a 114% of the 44 hours spent in the local cluster. As an example of the technical realism of our experiment, in a study published last year, Jin and Kirtman [24] performed with this same model and resolution, four 72-year simulations totalling 288 simulated years. Our simulations, carried out in less than 4 days in the Grid, are the equivalent of 1187 simulated years (750×19 months).

For the sake of illustration and fast deployment, we ported only the atmospheric and land components of an state-of-the-art coupled GCM. The experiment was designed to finish in a reasonable time by choosing a relatively coarse resolution. The state-of-the-art resolution of a GCM varies wildly with the particular experiment to be carried out. For example, for century-long simulations in the last IPCC assessment report, they used this model with T85 resolution. This is twice the resolution used in our experiment (i.e 4 times more memory demand and 8 times slower). Moreover, the model was coupled to the ocean component and ran for a 100-times-larger period. Such an application would necessarily need to take advantage of the parallel capabilities of the resources contributed to the Grid. We also tested successfully the parallel execution in the Grid, but it restricts the number of usable sites and requires a more specific treatment for each site, unlike the serial execution example shown in this work.

Due to the complexity and high demanding computational resources of climate modeling, Grid infrastructures have some aspects that should be improved in order to be completely useful for the climate modeling community. Some of these lacks have been found during the development and use of the CAM4G middleware presented in this paper. The main issues are related to data storage and access. Climate models need a large amount of data in order to be run and at the same time, produce a large amount of data. That fact reduces the Grid resources where a climate application such as CAM can be used. An improvement on bandwidth on Grid infrastructure would be desired. At the same time fast and stable management and replication of the data is also needed. Finally, an effort to allow parallel execution on Grid infrastructures should also be done, allowing the design of more high demanding experiments, in terms of memory and CPU resources, than the one presented in this article.

## References


[1] Allcock, W., Bresnahan, J., Kettimuthu, R., Link, M., Dumitrescu, C., Raicu, I., Foster, I., 2005. The Globus Striped Gridftp Framework and Server. In: Supercomputing, 2005. Proceedings of the ACM/IEEE SC 2005 Conference. IEEE Computer Society, 54–54.

[2] Allen, M., 1999. Do it yourself climate prediction. Nature 401 642.

[3] Anderson, D. P., Fedak, G., 2006. The computational and storage potential of volunteer computing. In: Proceedings of the Sixth IEEE International Symposium on Cluster Computing and the Grid. IEEE Computer Society, 73–80. URL http://portal.acm.org/citation.cfm?id=1134996.

[4] Board, I. A., 2009. Impact assessment guidelines. Technical Report 92, European Commission.

[5] Catlett, C., 2002. The philosophy of TeraGrid: building an open, extensible, distributed TeraScale facility. In: 2nd IEEE/ACM International Symposium on Cluster Computing and the Grid, 2002. 8–8.

[6] Collins, W., Rasch, P., Boville, B., Hack, J., McCaa, J., Williamson, D., Briegleb, B., Bitz, C., Lin, S., Zhang, M., 2006. The formulation and atmospheric simulation of the Community Atmosphere Model version 3 (CAM3). Journal of Climate 19 (11) 2144–2161.

[7] Collins, W. D., Bitz, C. M., Blackmon, M. L., Bonan, G. B., Bretherton, C. S., Carton, J. A., Chang, P., Doney, S. C., Hack, J. J., Henderson, T. B., et al., 2006. The community climate system model: CCSM3. Journal of Climate 19 (11) 2122–2143.

[8] Collins, W. D., Rasch, P. J., Boville, B. A., Hack, J. J., McCaa, J. R., Williamson, D. L., Kiehl, J. T., Briegleb, B., Bitz, C., Lin, S.-J., Zhang, M., Dai, Y., 2004. Description of the NCAR Comunity Atmospheric Model (CAM 3.0). Technical Report NCAR/TN-464+STR, National Center for Atmospheric Research. URL `www.cesm.ucar.edu/models/atm-cam/docs/description/description.pdf`.

[9] Cossu, R., Petitdidier, M., Linford, V. e. a., 2010. A roadmap for a dedicated earth science grid platform. Earth Science Informatics .

[10] Dai, A., 2006. Precipitation characteristics in eighteen coupled climate models. Journal of Climate 19 4605.

[11] Davidovic, D., Skala, K., 2010. Implementation of the WRF-ARW prognostic model on the Grid. In: MIPRO, 2010 Proceedings of the 33rd International Convention. IEEE, 220–225.

[12] Fernández-Quiruelas, V., Fernández, J., Baeza, C., Cofiño, A. S., Gutiérrez, J. M., 2009. Execution management in the GRID, for sensitivity studies of global climate simulations. Earth Science Informatics 2 (1) 75–82.

[13] Fernández-Quiruelas, V., Cofiño, A. S., Fernández, J., Fita, L., 2009. WRF4G: enabling WRF on the Grid. In: Proceedings of the Second EELA-2 Conference. Centro de Investigaciones Energéticas, Medioambientales y Tecnológicas, Madrid, 149–154. URL `http://www.ciemat.es/recursos/doc/Redes_Cientifico_Tecnicas/422103760_17112009151715.pdf`.

[14] Foster, I., 2006. Globus Toolkit Version 4: Software for Service-Oriented Systems. In: International Conference on Network and Parallel Computing LNCS 3779, volume 15. Springer-Verlag, 2–13.

[15] Foster, I., Kesselman, C., 1999. The grid. Blueprint for a new computing infrastructure. Morgan Kaufmann Publishers, San Francisco.

[16] Foster, I., Kesselman, C., Tsudik, G., Tuecke, S., 1998. A security architecture for computational grids. In: Proceedings of the 5th ACM Conference on Computer and Communications Security. ACM New York, NY, USA, 83–92.

[17] Foster, I., Kesselman, C., Tuecke, S., 2001. The anatomy of the grid: Enabling scalable virtual organizations. International Journal of High Performance Computing Applications 15 (3) 222.

[18] Fusco, L., Cossu, R., Retscher, C., 2008. Open Grid services for Envisat and Earth observation applications. High Performance Computing in Remote Sensing 237–280.

[19] Gates, W. L., 1992. AMIP: The atmospheric model intercomparison project. Bull. Amer. Met. Soc. 73 (12) 1962–1970.

[20] Hack, J. J., Caron, J. M., Yeager, S. G., Oleson, K. W., Holland, M. M., Truesdale, J. E., Rasch, P. J., 2006. Simulation of the global hydrological cycle in the ccsm community atmosphere model version 3 (cam3): Mean features. Journal of Climate 19 (11) 2199–2221.

[21] Huedo, E., Montero, R., Llorente, I., 2005. The GridWay framework for adaptive scheduling and execution on grids. SCPE 6 (8).

[22] Hurrell, J. W., Hack, J. J., Phillips, A. S., Caron, J., Yin, J., 2006. The dynamical simulation of the community atmosphere model version 3 (CAM3). Journal of Climate 19 (11) 2162–2183.

[23] Jacq, N., Salzemann, J., Jacq, F., Legré, Y., Medernach, E., Montagnat, J., Maass, A., Reichstadt, M., Schwichtenberg, H., Sridhar, M., et al., 2008. Grid-enabled virtual screening against malaria. Journal of Grid Computing 6 (1) 29–43.

[24] Jin, D., Kirtman, B., 2010. The extratropical sensitivity to the meridional extent of tropical ENSO forcing. Climate Dynamics 34 (7) 935–951.

[25] Kushnir, Y., Robinson, W. A., Bladé, I., Hall, N. M. J., Peng, S., Sutton, R., 2002. Atmospheric GCM response to extratropical SST anomalies: Synthesis and evaluation. J. Climate 15 2233–2256.

[26] Lagouvardos, K., Floros, E., Kotroni, V., 2010. A grid-enabled regional-scale ensemble forecasting system in the mediterranean area. Journal of Grid Computing 8 (2) 181–197.

[27] McPhaden, M. J., 1999. Genesis and evolution of the 1997-98 El Niño. Science 283 950–954.

[28] Mineter, M., Jarvis, C., Dowers, S., 2003. From stand-alone programs towards grid-aware services and components: a case study in agricultural modelling with interpolated climate data. Environmental Modelling & Software 18 (4) 379–391.

[29] Murphy, J., Sexton, D., Barnett, D., Jones, G., Webb, M., Collins, M., Stainforth, D., 2004. Quantification of modelling uncertainties in a large ensemble of climate change simulations. Nature 430 (7001) 768–772.

[30] Nandakumar, R., Jimenez, S., Adinolfi, M., Bernet, R., Blouw, J., Bortolotti, D., Carbone, A., M'Charek, B., Perego, D., Pickford, A., et al., 2008. The LHCb computing data challenge DC06. In: Journal of Physics: Conference Series, volume 119. Institute of Physics Publishing, 072023.

[31] Nicholson, S. E., Kim, J., 1997. The relationship of the El Niño–Southern Oscilation to african rainfall. Int. J. Climatol. 17 117–135.

[32] Palmer, T. N., 2002. The economic value of ensemble forecasts as a tool for risk assessment: From days to decades. Quarterly Journal of the Royal Meteorological Society 128 (581) 747–774.

[33] Parry, M., Canziani, O., Palutikof, J., van der Linden, P., Hanson, C., (Eds.) , 2007. Climate Change 2007: Impacts, Adaptation and Vulnerability. Contribution of Working Group II to the Fourth Assessment Report of the Intergovernmental Panel on Climate Change. Cambridge: Cambridge University Press.

[34] Postel, J., Reynolds, J., 1985. File transfer protocol.

[35] Randall, D. A., Wood, R. A., Bony, S., Colman, R., Fichefet, T., Fyfe, J., Kattsov, V., Pitman, A., Shukla, J., Srinivasan, J., Stouffer, R. J., Sumi, A., Taylor, K. E., 2007. Climate models and their evaluation. In: Climate Change 2007: The Physical Science Basis. Contribution of Working Group I to the Fourth Assessment Report of the Intergovernmental Panel on Climate Change. Cambridge University Press, 589–662.

[36] Rasmusson, E., Chandrasekar, V., Clayson, C. A., Hawkins, J. D., Katsaros, K. B., McCormick, M. P., Steiner, M., Stephens, G. L., Velden, C. S., Williamson, R. A., 2006. Assessment of the benefits of extending the Tropical Rainfall Measuring Mission: A perspective from the research and operations communities. The national academies press. Available on-line at: http://www.nap.edu/catalog/11195.html.

[37] Renard, P., Badoux, V., Petitdidier, M., Cossu, R., 2009. Grid Computing for Earth Science. EOS Trans. AGU 90 (14).

[38] Ropelewski, C. F., Halpert, M. S., 1987. Global and regional scale precipitation patterns associated with the El Niño/Southern Oscilation. Mon. Wea. Rev. 115 1606–1626.

[39] Sulis, A., 2009. GRID computing approach for multireservoir operating rules with uncertainty. Environmental Modelling & Software 24 (7) 859–864.

[40] Takahashi, K., 2004. The atmospheric circulation associated with extreme rainfall events in Piura, Peru, during the 1997–1998 and 2002 El Niño events. Ann. Geophys. 22 3917–3926.

[41] Tuecke, S., Welch, V., Engert, D., Pearlman, L., Thompson, M., 2004. Rfc 3820: Internet X.509 public key infrastructure (PKI) proxy certificate profile.

[42] Waylen, P., Caviedes, C., 1986. El Niño and annual floods on the north peruvian littoral. Journal of Hydrology 89 (1-2) 141 – 156. URL http://www.sciencedirect.com/science/article/B6V6C-487D1S0-D/2/5729b7151f79430fe2966e17ab7f3649.

[43] Welch, V., Foster, I., Kesselman, C., Mulmo, O., Pearlman, L., Tuecke, S., Gawor, J., Meder, S., Siebenlist, F., 2004. X.509 proxy certificates for dynamic delegation. In: 3rd annual PKI R&D workshop, volume 14.

[44] Williams, D. N., Drach, R., Ananthakrishnan, R., Foster, I. T., Fraser, D., Siebenlist, F., Bernholdt, D. E., Chen, M., Schwidder, J., Bharathi, S., Chervenak, A. L., Schuler, R., Su, M., Brown, D., Cinquini, L., Fox, P., Garcia, J., Middleton, D. E., Strand, W. G., Wilhelmi, N., Hankin, S., Schweitzer, R., Jones, P., Shoshani, A., Sim, A., 2009. The earth system grid: Enabling access to multimodel climate simulation data. Bulletin of the American Meteorological Society 90 (2) 195–205. URL http://journals.ametsoc.org/doi/abs/10.1175/2008BAMS2459.1.

[45] Zhang, T., Sun, D., 2006. Response of water vapor and clouds to El Niño warming in three National Center for Atmospheric Research atmospheric models. Journal of Geophysical Research 111 (D17) D17103.

[46] Zhang, X., Lin, W., Zhang, M., 2007. Toward understanding the double Intertropical Convergence Zone pathology in coupled ocean-atmosphere general circulation models. Journal of Geophysical Research 112 (D12) D12102.