J Braz Comput Soc (2013) 19:279–290
DOI 10.1007/s13173-013-0099-5

ORIGINAL PAPER

Challenges for mesoscale climatology execution on experimental grid computing systems

Eugenio Sper de Almeida · Haroldo Fraga de Campos Velho · Airam Jonatas Preto

Received: 5 August 2011 / Accepted: 9 January 2013 / Published online: 22 February 2013

Abstract This paper discusses the challenges of executing a long-term application on a computational grid, which generates the climatology of the atmospheric numerical model BRAMS (*B*razilian development on *R*egional *A*tmospheric *M*odeling *S*ystem) using ensemble members. We have developed a workflow that submits climatology to the computational grid composed by three different grid middlewares (OurGrid, OAR/CiGri and Globus) and three clusters (situated in Porto Alegre, São José dos Campos and Cachoeira Paulista—Brazil). The application characteristics demand a processing grid, rather than a data grid, due to intensive computation and data transfer between the geographically distributed grid nodes. We achieved the goal of generating the climatology using a computational grid. However, we observed problems on application performance due data transfer and non-availability of the computational grid. Questions related to data storage/transfer and grid failures must be better treated to ensure application performance.

Keywords Grid computing · Climatology · Mesoscale numerical atmospheric model · BRAMS · Globus · OurGrid · OAR/CiGri

E. S. Almeida (✉)
Centro de Previsão de Tempo e Estudos Climáticos (CPTEC), Instituto Nacional de Pesquisas Espaciais (INPE), Rod. Pres. Dutra, Km 39, Cachoeira Paulista, SP, Brazil
e-mail: eugenio.almeida@cptec.inpe.br

H. F. Campos Velho
Laboratório Associado de Computação e Matemática Aplicada (LAC), Instituto Nacional de Pesquisas Espaciais (INPE), Av. dos Astronautas, 1758 São José dos Campos, SP, Brazil

A. J. Preto
Serviço Corporativo de Tecnologia da Informação (STI), Instituto Nacional de Pesquisas Espaciais (INPE), Av. dos Astronautas, 1758 São José dos Campos, SP, Brazil

1 Introduction

The computational simulation for weather, seasonal, climate, and environment forecast of ever-increasing range of geophysical phenomena contributes enormously to understand and predict the complex processes in the Earth system, with close relation between the sophistication of prediction models and the availability of computational power [1].

The seasonal forecast refers to a range up to 6 months and requires the monthly or weekly model climatology, the atmosphere state average over a long period of time (typically 30–50 years). To compute the model climatology, two schemes could be cited: the International Research Institute for Climate Prediction (IRI) method executes a single long integration [2], while the European Centre for Medium-Range Forecasts (ECMWF) method executes multiple distinct integrations, beginning in distinct months and with short duration [3].

The ensemble forecasting is a technique to increase forecast skill and consists of multiple simulations of a numerical weather prediction (NWP) model using different initial conditions. Each simulation produces a distinct forecast output (ensemble member), which is a representative sample of the possible future atmospheric states for probability assessment [4].

The mesoscale climatology simulation uses distinct initial and boundary conditions, which comes from lower spatial resolution model ensemble members. After simulation, both schemes calculate monthly or weekly means from the simulation outputs.

The intensive use of ensemble data requires computational resources for data processing and storage, management of long-run simulations and transfer of huge datasets. This kind of computation can exhaust the available computational resources on one institution, requiring the use of grid

computing technique to provide more computational resources from other institutions [5]. Also makes a well suited application for the evaluation of computational grids, a hardware and software infrastructure that provides trustworthy and consistent access to computational resources located at different geographical locations [6].

The G-BRAMS project [7,8] aimed at the mesoscale climatology generation of the BRAMS (*B*razilian development on *R*egional *A*tmospheric *M*odeling *S*ystem) model (http://brams.cptec.inpe.br) for Brazil on a research computational grid. Little emphasis has been placed on implementing and executing meteorological models on computational grids, mainly because they require long executions that mix sequential and parallel execution in grid environments with high performance computing (HPC) clusters.

The "Centro de Previsão de Tempo e Estudos Climáticos" (CPTEC) from "Instituto Nacional de Pesquisas Espaciais" (INPE) uses traditional systems, based on massive parallel processing (MPP) systems and with large support teams, to execute weather, seasonal, climate and environmental prediction models. The CPTEC/INPE generates the seasonal forecast every month to predict how much some meteorological variables differ from climatological normal for the next 3 months. Unlike weather forecasting that requires the model execution in a few hours, since meteorologists need to inform the weather for next days, the seasonal forecast requires a long-term execution with more flexibility and some unavailability can be tolerated without compromising the results divulgation.

Computational failures exist in any computational system and can compromise the application execution, especially on a processing grid with intensive computing demand (months of computation). Even a simple problem in a grid computing system, such as certificate expiration, affects the reliability of the entire system. So, job redistribution features have importance since it permits application re-execution on a free grid node.

This work has the objective of analyzing the viability of grid computing technology as an alternative to traditional systems for operational seasonal forecasting. So, we tested the mesoscale climatology with the IRI methodology on three grid middlewares: OurGrid [9], OAR/CiGri [10], and Globus [11]. Due to requirements of high spatial resolution for climatology generation, we needed to split the climatology computation for three different regions (North, Northeast and Southeastern/South) separately.

In this paper, we evaluated the viability of this methodology and the issues related to grid computing systems using the BRAMS climatology, an application that requires a long execution time (more than 24 h of computation). We also investigated the main points to be considered for a seasonal forecast in a grid environment, focusing on operational use. The next section provides a summary of related work to this paper. In Sect. 3 we describe G-BRAMS project goals. Section 4 describes the G-BRAMS computation, where we include information related to computational grid and climatology workflow. We describe the experiment, considering 3 years of climatology, on Sect. 5. Section 6 presents the results and the analysis of the experiment. We conclude the paper on Sect. 7.

2 Related work

Computational grids have expanded recently to cater to demands on high performance and distributed applications. However, they are inherently unreliable [12,13] and subject to errors that cause failures, with reasons on the geographical dispersion, the large number of heterogeneous resources managed by different organizations and the large variety of applications.

The linked environments for atmospheric discovery (LEAD) created an integrated web service-oriented architecture to support mesoscale meteorological data acquisition, analysis, assimilation, simulation modeling, prediction, mining and visualization [14,15]. The LEAD portal (http://portal.leadproject.org) uses Teragrid infrastructure to execute the simulations.

Kandaswamy et al. [16] have pointed that the execution of complex workflows with reliability is a challenge in computational grids. They executed 165 complex weather forecasting workflows from the LEAD production portal, with a total of 869 applications (workflow steps) in them. Without any fault tolerance and recovery strategies, 268 of the total 869 applications failed on the TeraGrid (application failure rate of 30.84 %). Since a workflow consists of several steps, the failure rate on workflow is higher than on application (79.39 %). They reported that applications failed due to a variety of reasons: GridFTP failures during data transfer, file systems problems, file systems running out of disk space due to very large data transfers, connection timeouts from Globus WS-GRAM, compute node crashes, transient downtimes of core grid services.

Mattocks et al. [17] presented several developments to merge the best aspects of the LEAD portal (ability to submit large simulations across the TeraGrid, multi-level fault tolerance and recovery, drag-and-drop workflow assembly, data mining and feature extraction) with those of the numerical weather prediction model WRF (Weather Research and Forecasting) model portal (desktop Java GUI with domain editors, support for multiple/two-way interactive/moving vortex-tracking WRF nesting, diff tool for comparing workflows).

Medeiros et al. [18] presented a survey, conducted with users of computational environments, regarding fault identification and treatment. Dai and Dongarra [19] analyzed the

grid reliability, presenting and classifying the different types of failures in grid systems. Hofer and Fahringer [13] proposed a grid fault taxonomy describing fault events using eight different characteristics.

Long-running applications that require many resources and must produce precise results are likely to be especially vulnerable to failures [20]. Sander et al. [21] summarized the networking issues identified on grid environments, a potential source of faults when transporting large datasets.

According with [22], the use of grid computing to run long-term jobs is uncommon. Using EELA (E-Infrastructure shared between Europe and Latin America) computational grid, they faced high rate of job failures and CPU-time limitations for the jobs on the local management system (typically only jobs lasting less than 48 h are allowed). They executed 50 serial simulations of Community Atmospheric Model (CAM) during 1 week, with T42 resolution. Each simulation was split into smaller jobs for the period from January 1997 until March 1998 (15 months).

Using a different paradigm, the climateprediction.net uses the distributed computing concept, inviting thousands of participants to download a climate model and executing in a low spatial resolution to simulate 100 years of the Earth's climate [23].

Lagouvardos et al. [24] and Kotroni et al [25] explored the used of ensemble weather forecasting in computational grid to run different weather models, using the South Eastern Europe Grid eInfrastructure for regional eScience (SEE-GRID-SCI, http://www.see-grid-sci.eu) project. In their experiment, only MM5 model was executed in parallel using MPI. They demonstrated the ability of the solution to provide ensemble forecasts at a regional scale, despite technical issues (model, infrastructure or data download failures).

Fernández-Quiruelas et al. [26] presented the advantages and limitations of EGEE (Enabling Grids for E-sciencE, http://www.eu-egee.eu) Grid infrastructure for a climate application experiment involving the execution of 750 19-month simulations of the serial version of the CAM model (T42 resolution). Using six European sites, all 750 realizations were running after 6 h and completed after 3.5 days.

In contrast to previous works, we analyzed the BRAMS model climatology, a long execution parallel application, using dedicated computational grids with different grid middlewares.

3 The G-BRAMS project

The weather forecast reports absolute values and seasonal forecast reports average over time. During the first stages of seasonal forecast, the atmospheric numerical models tend to produce a biased atmospheric state, quite different from the real one. The difference operation between seasonal forecast and the monthly average of atmospheric forecast over a set of years in the past (the "climatology") eliminates this tendency, using the same numerical model, region and period of integration.

The G-BRAMS was a joint project of "Instituto de Informática" (II) of the "Universidade Federal do Rio Grande do Sul" (UFRGS, "Porto Alegre/RS"), "Laboratório Associado de Computação e Matemática Aplicada" (LAC) of INPE ("São José dos Campos/SP") and CPTEC/INPE ("Cachoeira Paulista/SP"). One of its goals was the long-term monthly average of atmospheric forecasts (model climatology), produced by the BRAMS model for the entire country.

It uses grid computing as a strategy to a processing grid, instead of a data grid. Data grids serve to share data and/or allow different users (from different institutions, geographically separated) to promote their data analysis.

The BRAMS is an adaptation derived from version of the Regional Atmospheric Modeling System (RAMS) model [27–29] tailored to tropical regions, with the aim of using in production by the regional meteorological centers and research universities in Brazil [30]. The BRAMS/RAMS is a multipurpose numerical weather model with explicit parallelization, well suited for HPC clusters, designed to simulate atmospheric circulations in operational forecasting and atmospheric research. It serves a broad range of applications, providing outputs from meters to thousands of kilometers.

The IRI (International Research Institute for Climate Prediction) method was selected for model climatology generation, using ensemble members. The generation of ensemble members involves the executing the same model with different initial condition, generating different possibilities of dealing with physical processes, and useful in treating system uncertainty [31]. The ensemble methodology refers to the technique of using a set of members to assure a more accurate characterization of the uncertainty associated to the forecasts [32].

Ensemble inputs for the G-BRAMS come from the CPTEC/INPE's atmospheric global circulation model (AGCM) outputs and provide the lower spatial resolution (200 km) initial/boundary conditions for the integration period. The climatology generation process produces a detailed climatology for three different geographical regions of Brazil (Fig. 1), with spatial resolution of 40 km.

4 G-BRAMS computation

We deployed a computational grid (Fig. 2), using the network infrastructure from RNP ("Rede Nacional de Pesquisa e Ensino"), with three different grid middlewares: OurGrid, OAR/CiGri and Globus. This computational grid provides access to three HPC clusters from G-BRAMS partners:

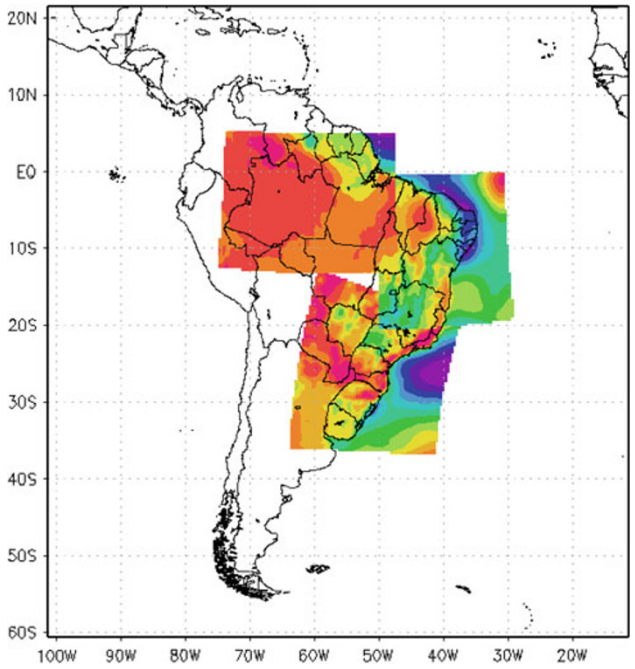

Fig. 1 Areas for climatology execution: Amazon (N); Northeast (NE); South/Southeast (SE)

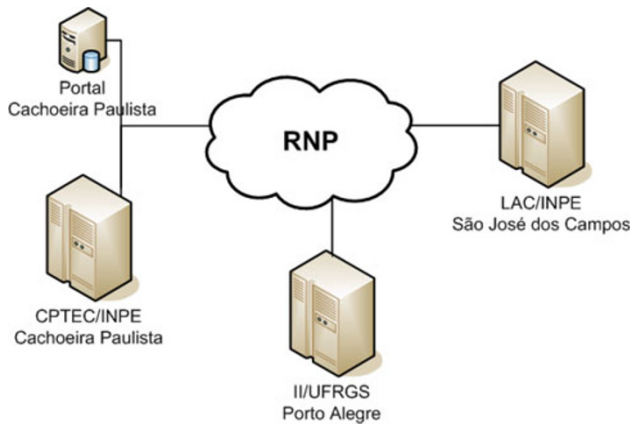

Fig. 2 G-BRAMS computational grid

- *Tatui:* one 18-node HPC cluster with Xeon processors @3.0 GHz at CPTEC/INPE ("Cachoeira Paulista/SP");
- *Paraytinga:* one 12-node CRAY/XD1 HPC cluster with Opteron processors @2.6 GHz at LAC/INPE ("São José dos Campos/SP");
- *xd1:* one 12-node CRAY/XD1 HPC cluster with Opteron processors @2.6 GHz at the II/UFRGS ("Porto Alegre/RS").

The cluster nodes are interconnected with Gigabit Ethernet switch (1 Gbps) on "tatui", and RapidArray network (23.2 Gbps) on "paraytinga" and "xd1". RapidArray is the CRAY interconnection predecessor of SeaStar, Seastar2 and Gemini.

The BRAMS simulation is a parallel application. However, the climatology calculation is a Bag-Of-Tasks (BOT) application, since it deals with the execution of multiple instances of the BRAMS (independent tasks). Therefore, typical BOT middlewares as OurGrid and OAR/CiGri can be used for this purpose.

The OurGrid middleware has three main components: the SWAN security mechanism, the OurGrid Resource Manager Peer and MyGrid Broker. Brasileiro et al. [33] describe four ways to submit applications using MyGrid Broker: script based, embedded, framework based and portal based. Originally OurGrid uses the work queue with replication (WQR) mechanism for scheduling. In this work, we set OurGrid to schedule the applications without replication and used the script-based approach to submit the climatology to the grid.

The CiGri [34] server and the OAR [10] local scheduler compose the OAR/CiGri middleware. The CiGri server acts as a meta-scheduler and manages the execution of BOT applications by submitting individual jobs to each cluster using ssh and a local resource management system (OAR), an open source scheduler. The OAR manages the clusters resources and schedules jobs to clusters as traditional schedulers (as PBS/Torque/LSF/SGE), to successfully handle the applications executions (task and job scheduling, multi-cluster support).

The Globus toolkit implements basic grid services (security, resource management and data transfer). On G-BRAMS, we used Grid Security Infrastructure (GSI) for user identification, Globus Resource Allocation Manager (GRAM) for job submission and control, and GridFtp for secure, reliable and high performance data transfers. We have developed a scheduler, using the round-robin algorithm that schedules jobs to free grid nodes.

The climatology workflow consists of the data transfer, preprocessing, simulation execution and post-processing. The computational grid accepts and treats the workflow as a sequential job. Once defined the HPC cluster to perform the workflow, the sequential applications runs on cluster front-end and parallel applications on a defined number of nodes of a given HPC cluster.

The OurGrid and the OAR/CiGri use a plain text file, a job description file (jdf) and a job description language (jdl), respectively, that contains the job description for grid submission. Including the climatology workflow inside the jdf/jdl file, we can submit it as a single job to the grid using a special submission command from OurGrid and Oar/CiGri. Globus submits the climatology workflow using the scheduler that we have developed.

4.1 Climatology workflow

We have automatized the BRAMS climatology generation by creating a climatology workflow (Fig. 3), used to submit to

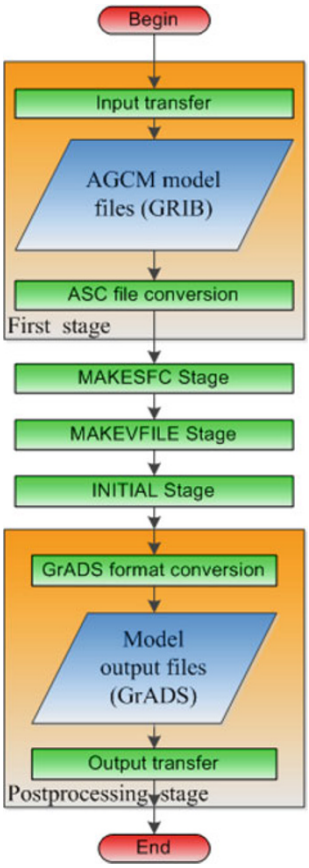

Fig. 3 Climatology workflow

the computational grid. Once the scheduler identifies a free grid node, it submits the climatology workflow.

During the first workflow stage the initial and boundary conditions download to the selected grid node, and the conversion to ASCII format (dp files) occurs.

In the next three stages, the BRAMS executes with parameters defined in RAMSIN (RAMS INput) file, a Fortran namelist format:

- MAKESFC: In this stage occurs the resolution conversion of global data files (sea surface temperature, soil type and topography) that just cover the limited area (and grid), defined by the namelist $MODEL_GRIDS (in RAMSIN);
- MAKEVFILE: This stage produces files with initial and boundary conditions for each grid over the integration time. This is performed by "cutting" the dp files to the desired geographical areas, grids and time interval, as defined by the namelist $MODEL_GRIDS (in RAMSIN);
- INITIAL: Model execution can be serial or parallel (MPI based) in this stage, with input data files originated from outputs of the last two stages: topography, sea surface temperature, vegetation cover, initial and boundary condition files. The simulation generates two kinds of output data files: history and analysis. This stage provides a meteorological forecast for the desired limited area and time period. The BRAMS places history files on a directory, to be used in case of necessary restart.

The post-processing stage includes output format conversion, visualization preparation and output transfers:

- Format conversion: A file format conversion program (RAMSPOST) converts the model output format (RALPH) to GrADS (http://grads.iges.org/grads/grads.html) format, based on rules established by RAMSPOST namelist file (Rampost50.inp). It also generates metadata information about the file contents;
- Visualization preparation: Uses GrADS to generate a few diagnostic meteorological charts to allow the evaluation of the climatology results on the web portal;
- Output transfers: It transfers the simulations results to the portal machine.

After completing all simulations and transferring the simulation to the web portal, the calculation of the climatology on the web portal is done (the average of simulation outputs of each meteorological variable per each atmospheric level and per region).

4.2 Software architecture

A web portal (Fig. 4), designed for the G-BRAMS project, allows users an easy way for the execution of the climatology. It permits the grid middleware selection and has pages for *Job Creation*, *Job Monitoring* and *Output Visualization*.

The fields on *Job Creation* page correspond to the simulation parameters defined on RAMSIN file, such as date, time, duration of the simulation and geographical area. By defining the simulation parameters, users allow their storage on a database, which become available for users submission to the computational grid. The scheduling mechanism defines the HPC cluster that will process the job (simulation).

The G-BRAMS web portal monitors the computational grid and job states, storing information on a database and presenting to the user on the *Job Monitoring* page. The job states starts from editing, waiting, executing and finally ready.

Simulations running on grid nodes receive the executing state. The jobs being edited by the user have the editing state. After edition, the first job can be scheduled, but subsequent jobs must wait (waiting state). Jobs receive the ready state when the job ends its execution and the user accepts it.

The *Output Visualization* page presents the climatology results, generated from the post-processing of the simulation outputs

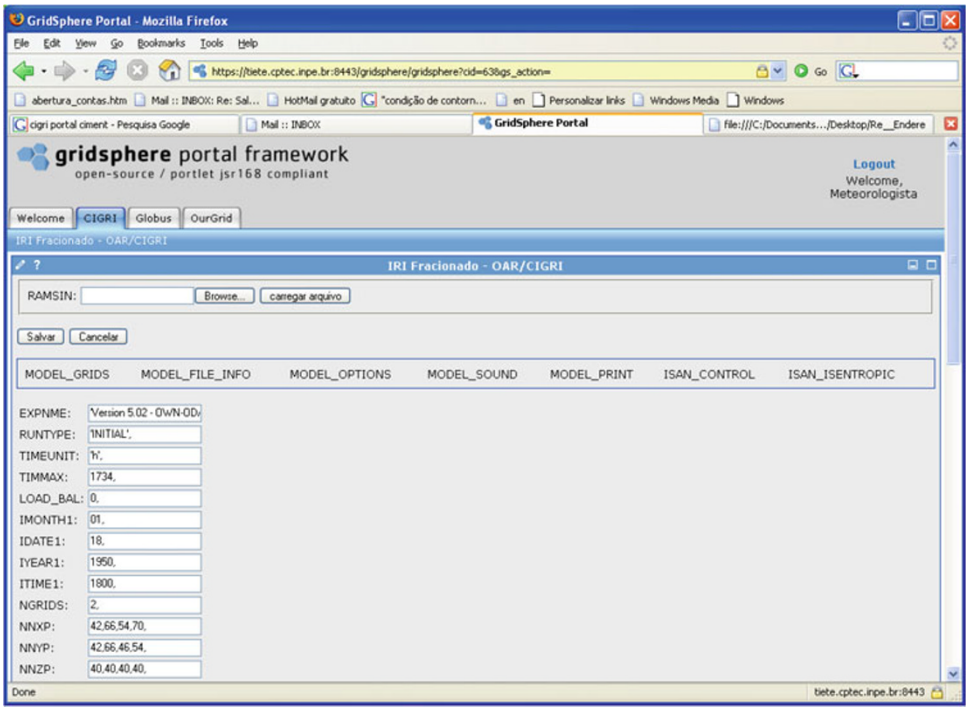

Fig. 4 G-BRAMS web portal

The job scheduling strategy adopted on G-BRAMS project considers a dedicated computational grid, with utilization through a dedicated portal. This strategy allows the BRAMS execution in the shortest time possible, ensuring application performance.

The OAR/CiGri and the OurGrid have their own schedulers. However Globus required the development of a simple job scheduler, since it has no job scheduler. Based on a round-robin strategy, the scheduling algorithm implemented detects a grid node without use and submits jobs automatically.

The climatology calculation consists of the BRAMS simulation for a determined number of years for a given number and ensemble member. The transfer of input data for the simulation occurs from web portal to grid node selected by the scheduler. After processing each 12 months period, the web portal node receives the processed data. The file transfers use gsiftp for Globus, and sftp for OAR/CiGri and OurGrid.

After receiving all simulation outputs, the web portal node calculates the monthly average of the simulation outputs by level, meteorological variable and region. The files holding data for the first 2 months are discarded on this calculation, since their use is restricted to model stabilization. The web portal makes available the climatology for visualization.

4.3 Computational cost of climatology methods

The ECMWF and IRI climatology methods are widely used for the computation of climatology. The ECMWF method has higher computational cost, since it requires individual model simulations of 3 months that repeats each month for a determined number of years, excluding the last 2 months. Considering the use of ensemble mode, its computational cost is presented in equation below:

$$\text{Cost} = ((n\_\text{months} \times \text{integr\_per}) - 2)) \times n\_\text{years} \times n\_\text{members} \quad (1)$$

where n_years represents the number of years of simulation, n_months the number of months, integr_per the integration period, and n_members the number of ensemble members.

The IRI climatology method is less computationally expensive and consists of the model integration for the whole period. It generates one climatology per month/per ensemble member, and its computational cost is given by:

$$\text{Cost} = n\_\text{months} \times n\_\text{years} \times n\_\text{members}. \quad (2)$$

In order to compare the computational costs, we considered the use of three ensemble members to generate the 3-year climatology of BRAMS model. The IRI method requires three integrations of 12 months, with a computational cost of 108 months of integrations. The ECMWF method requires 34 integrations $((3 \times 12) - 2)$ of 3 months, totalling 102 simulations. Considering the three members, the computational cost is 306 months of integration.

The processing of the climatology by itself would maintain the HPC clusters busy for a considerable time (months). So, we decided to use for our analysis the IRI method because

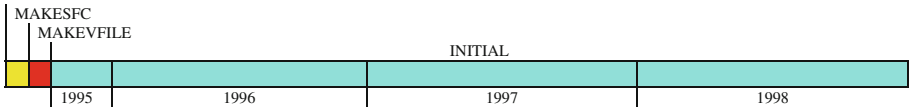

Fig. 5 Three-year simulation of BRAMS

its computational cost is approximately three times less than the ECMWF method.

5 Experiment

We conducted three climatology experiments, each one using a computational grid configuration with a different grid middleware (OurGrid, OAR/CiGri and Globus). In each experiment, we calculated the 3-year mesoscale climatology of the BRAMS model for three different regions of Brazil (see Fig. 1).

For each region, we used three ensemble members of 3 years from the CPTEC/INPE AGCM, plus two extra months for model simulation stabilization. Each ensemble member served as initial and boundary conditions for the correspondent simulation.

The three experiments required 27 independent simulations of 3 years. However, a limitation on BRAMS code, due to hard-coded definition of maximum simulation days, denies the climatology execution for the whole simulation period.

We used a portal to upload the RAMSIN file, which contains the definition of the geographical area of Brazil region, the start/end date of the simulation, the ensemble member and other parametrizations. The portal implementation has mechanisms to split the BRAMS simulations (Fig. 5) for the 3 years into four parts, which became dependent jobs on the computational grid: 1 of 2 months and 3 of 12 months.

- 01 Nov 1995 to 31 Dec 1995—1,464 h (61 days);
- 01 Jan 1996 to 31 Dec 1996—8,784 h (366 days);
- 01 Jan 1997 to 31 Dec 1997—8,760 h (365 days);
- 01 Jan 1998 to 31 Dec 1998—8,760 h (365 days).

This schema uses the BRAMS checkpoint/restart mechanism that allows climatology simulation interruption and resuming at every simulated year. At the end, the computational grid executes 108 serial jobs, considering the use of three different computational grid configurations.

The independent simulations configure the climatology as a BOT procedure and a good example of intensive computing. The portal serves for simulations configuration and submission. Once submitted to the computational grid, a scheduler assigns the simulation workflow to an HPC cluster that executes the sequential and parallel components of the workflow. A database stores information of each job (job execution time and status), with access through the portal.

We organize the job execution times of the experiments, presenting them graphically as a function of:

- Grid configuration, region and ensemble member;
- Region, ensemble member and HPC cluster, per grid configuration;
- The number of jobs processed by each computational grid configuration and HPC cluster.

Our analysis aimed the evaluation of the computational grid capacity of dealing with the meteorological application (mesoscale climatology), the behavior of the G-BRAMS on a computational grid and the management of large amounts of data, looking for patterns on job execution times that we could correlate with hardware and software failures.

We conducted, along the day, an additional experiment to collect data transfer time between portal and grid nodes. Looking at the number of jobs performed by each HPC cluster and the data transfer experiment, we discuss the influence of the RNP network, cluster locality and failures for each computational grid configuration. It is expected that HPC clusters nearer to the data source should produce results faster than farther ones.

For the same region and ensemble member, we compared the job execution times on the HPC clusters in order to identify the influence of grid middleware, when using the same HPC cluster, and the architecture, when using different HPC clusters.

6 Results and analysis

We verified that all grid middleware performed as expected, reaching the goal of producing the mesoscale climatology of the BRAMS. Figure 6 presents the three experiments results, which spent 33.3 days using OAR/CiGri, 22.3 days using Globus and 20.6 days using OurGrid.

For each grid middleware, the climatology execution time was organized by region and by ensemble member. The significant differences on job execution times have relation with data transmission time and hardware/software failures.

We observed that the cluster with INTEL processors (“tatui”) has lower performance than the clusters with AMD processors (“xd1” and paraytinga”), which is around

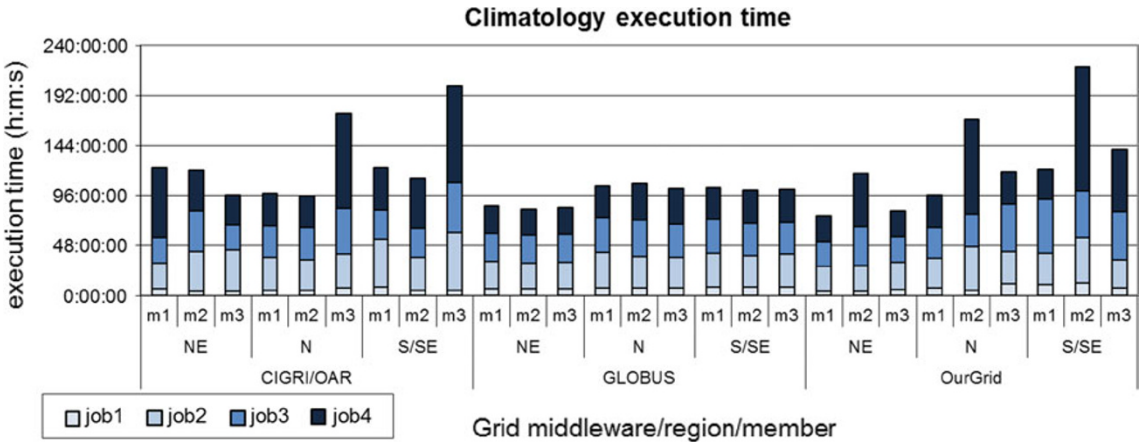

Fig. 6 Climatology execution time for the three grid middleware

1.5 %. Even with fewer processors than "tatui", "xd1" and "paraytinga" performed better due to the interconnection.

The middleware introduces an overhead on each job execution. We verify that the HPC cluster "paraytinga", using the computational grid with OurGrid, has the lowest job execution time in all computational grid configurations. Thus, we can infer that OurGrid have the lowest overhead. On the best case, OAR/CiGRi and Globus have an overhead of 0.8 and 2.8 % higher than OurGrid, respectively.

The distance between the grid nodes and the portal has impact on data transmission time. The farther grid nodes have higher data transmission time than the nearer grid nodes. The reasons are the higher latency, the lower bandwidth and the Internet data traffic variability throughout the day.

The same AGCM output data (2.03 GB) are used as input data for each region simulation, with traffic from the portal to grid nodes. The simulation outputs have traffic from grid nodes to the portal and the size depends on the region: North (175.1 MB), South/Southeast (166.5 MB) and Northeast (154.8 KB).

To receive data from the portal it takes 14 min 29 s in the best case and 3 h 33 min 34 s in the worst case. To send data to the portal it takes 1 min 41 s in the best case and 35 min 56 s in the worst case.

The data receiving time by the grid node is greater than the data transmission time to the portal, since the input files are larger than output files. Both worst cases refer to the grid node "xd1", the farthest grid node in relation to the portal.

The dedicated communication link between São José dos Campos and Cachoeira Paulista makes the communication times between "tatui"-portal and "paraytinga"-portal quite close. However, we observe a small increase of communication time variation due to the interference of other network equipments between "paraytinga" and the portal.

Transmission time affects directly the application performance, with relation to physical location of the grid node. Best case showed that the communication time could represent 1 % of computation time while worst case can achieve 12 %.

Figures7, 8 and 9 show the climatology execution time on each grid node, for a given region and ensemble member, where we can see the differences with more details. The jobs with low execution time correspond to the model stabilization process.

Considering the minimum execution time of jobs 2, 3, and 4 of each region, we have three different reasons for job execution times increase:

- Up to 4 h indicate fluctuation on data transmission time;
- Between 4 and 58 h indicate that a job scheduling occurred due to hardware and/or software failure, and the simulation restarted without human intervention;
- Higher than 58 h represents a serious hardware and/or software failure that required human intervention to fix the problem and to restart the simulation.

As a consequence, the computational grid stayed idle for 17.4, for 10.1 and for 4.7 days using the grid middlewares OAR/CiGri, Globus, and OurGrid, respectively.

The job execution time has variation of 281.3 % (68 h 10 min), when the computational grid middleware uses OAR/CiGri (Fig. 7). The lowest (24 h 14 min) and highest (92 h 24 min) job execution time corresponds to the HPC cluster "paraytinga" and "tatui", respectively.

The job execution time has variation of 401.7 % (95 h 08 min), when the computational grid middleware uses OurGrid (Fig. 8). The lowest (23 h 41 min) and highest (118 h 49 min) job execution time corresponds to the HPC cluster "paraytinga" and "xd1", respectively.

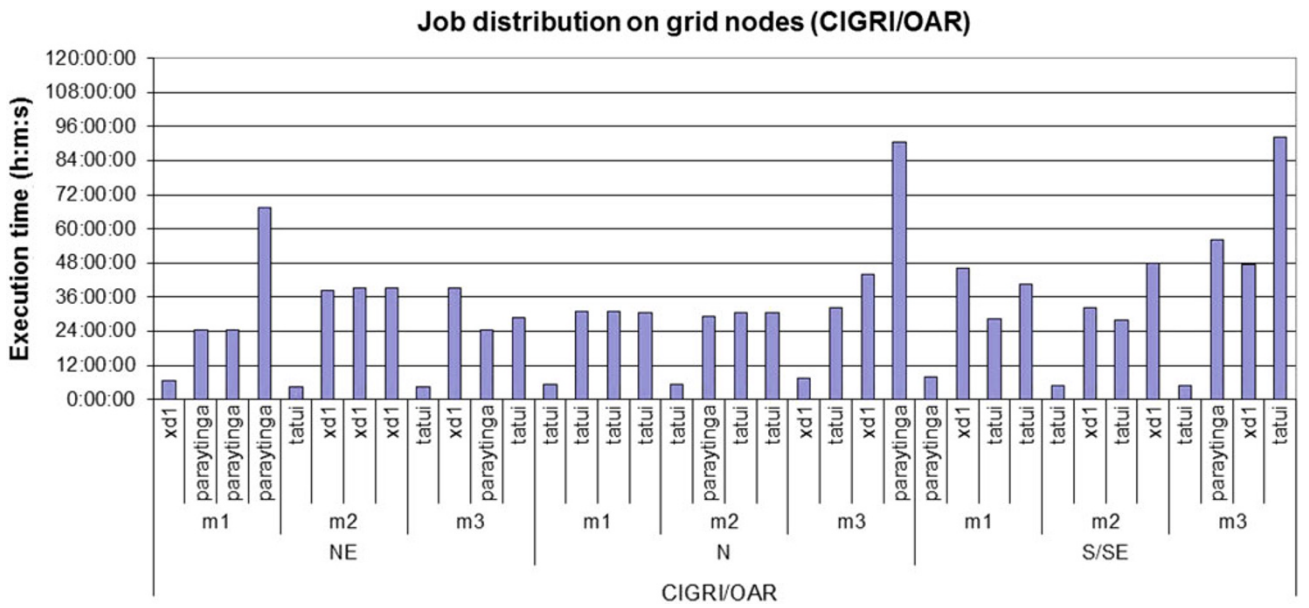

Fig. 7 Job distribution on grid nodes (OAR/CiGri)

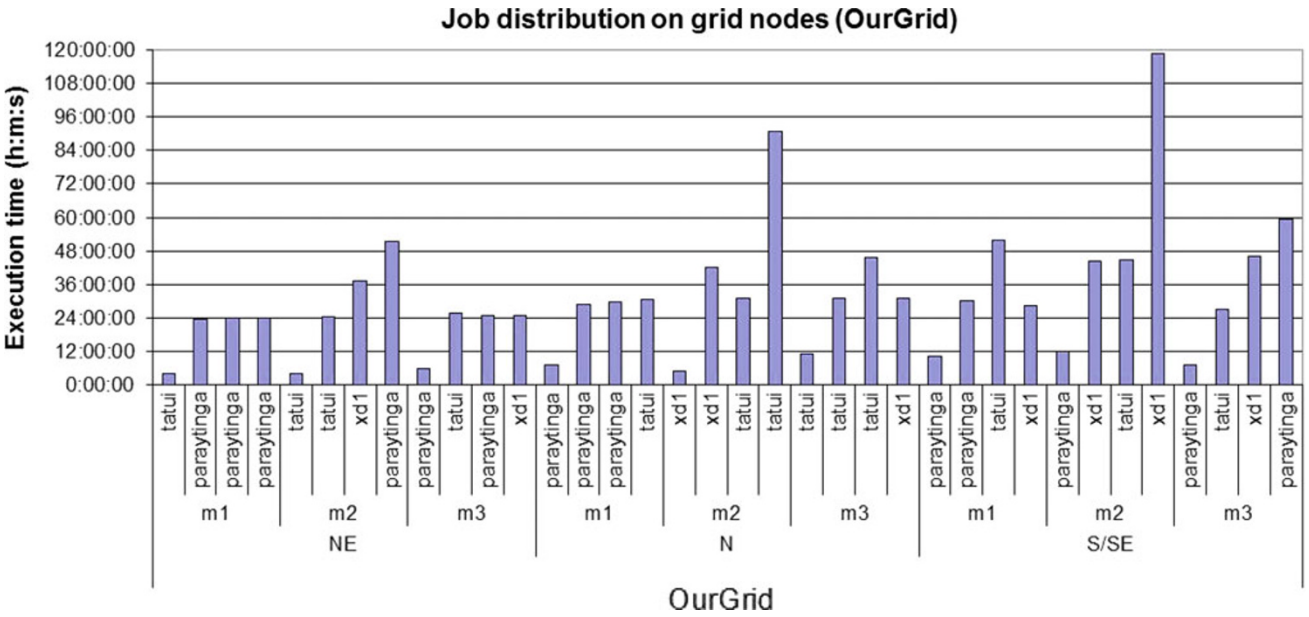

Fig. 8 Job distribution on grid nodes (OurGrid)

The job execution time has variation of 39.0 % (9 h 46 min), when the computational grid middleware uses GLOBUS (Fig. 9). The lowest (25 h 04 min) and highest (34 h 50 min) job execution time corresponds to the HPC cluster "paraytinga" and "tatui", respectively.

Since we have no resubmission features implemented on the Globus scheduler, this variation has direct relation with the model grid size of each region of Brazil. In case of failure, the climatology requires a manual resubmission and the time due to failure appears only as grid idle time.

We observed an unbalanced job distribution (Fig. 10) on grid nodes. The best job distribution between grid nodes, and consequently the lowest grid idleness, occurred when the computational grid used Ourgrid. The cluster "xd1" executed fewer jobs due to high communication costs between "xd1"-portal and high variations on these communication times.

Problems on grid application or grid node failure had more impact on jobs executed on the computational grid with OAR/CiGri e Globus. Using OAR/CiGri, the cluster "paraytinga" performed fewer jobs than cluster "xd1". Using

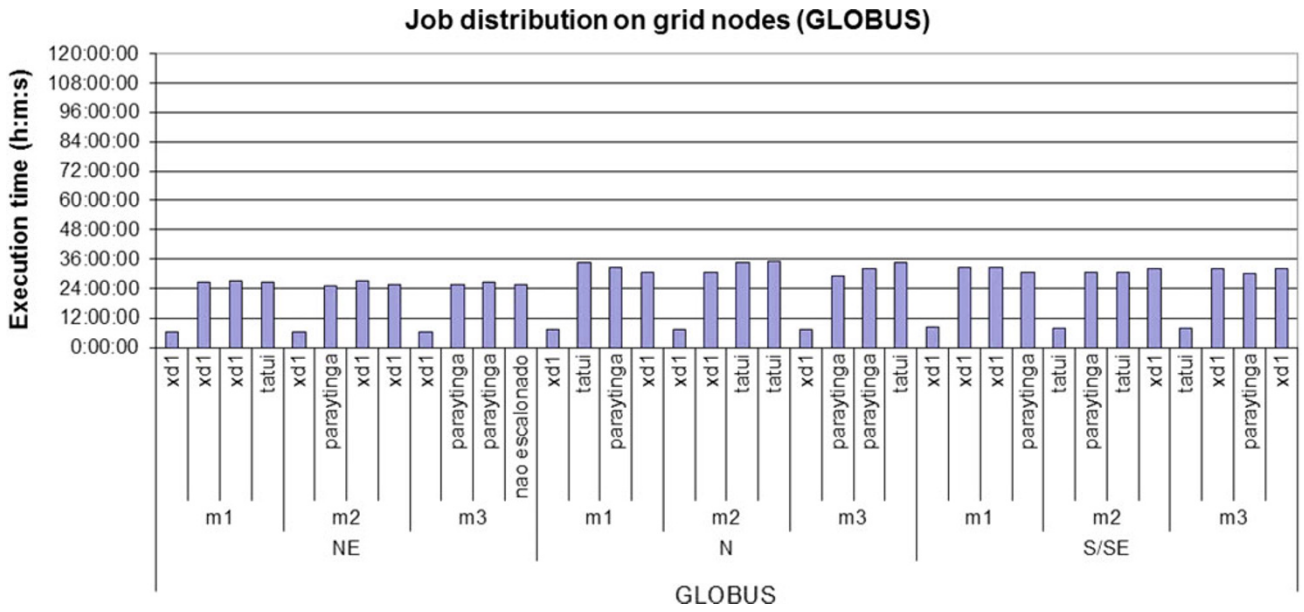

Fig. 9 Job distribution on grid nodes (Globus)

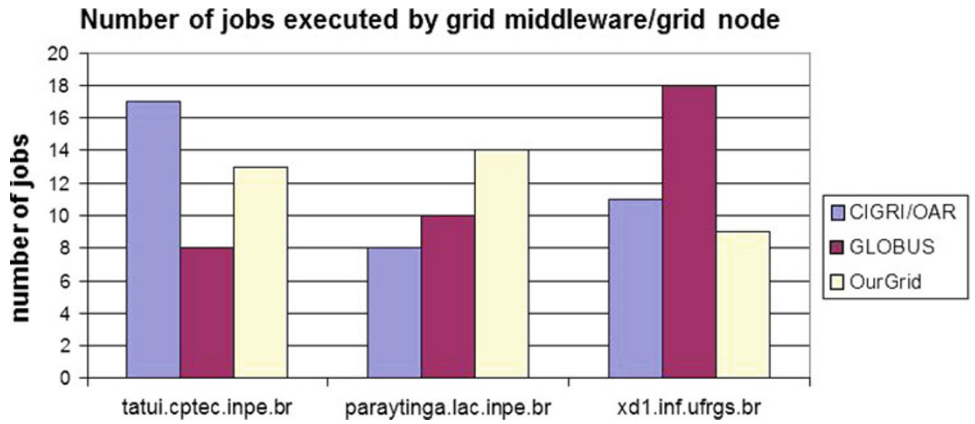

Fig. 10 Number of jobs executed by grid node

Globus, cluster "tatui" executed less jobs than cluster "xd1" and "paraytinga", despite data being available locally.

The job submission features of OAR/CiGri and OurGrid guarantees application execution, even in the case of hardware/software failures. Even when a failure occurred, data movement and climatology execution on another grid node, from the point the simulation stopped, worked as expected.

The problems related to hardware and/or software failures increased (see Fig. 6) the climatology execution time considerably. We have identified different factors that caused those problems.

Instability of the operational system (OS) installed on "tatui" (Linux Fedora Core) caused the interruption of climatology execution (Globus) and increased climatology execution time (OAR/CiGri). Upgrading the cluster OS solved this issue.

Electrical power outages interrupted cluster operation during part of the experiment. In a weekend, it represented an increase of 5 % in climatology execution time. A high-availability electrical power infrastructure can overcome this problem.

The computational grid refused to execute computation during the experiment due to Globus certificates expiration. We solved this problem by generating and installing new grid nodes certificates, signed by the Certification Authority. More definitive solutions could be: (i) start with a certificate with longer duration (1 year in the case of G-BRAMS), (ii) adopt an automatic process to renew the certificates before their expiration.

Data inconsistency during data exchange between grid nodes and portal machine can lead to failure in executing the next job or make results unavailable. Data verification

mechanisms can solve this problem. The "cksum" tool from Linux/Unix implements the cyclic redundancy check (CRC), an error-detecting code designed to detect accidental changes to raw computer data. It can be used to detect a corruption on destination and ask the sender to transmit it again.

Software components failures (scheduler and portal components) contribute to increase the climatology execution time. The scheduler runs on the portal machine as a process and its deactivation, for unknown reasons, caused the non-submission of jobs to grid nodes. To address this problem, the portal can periodically check scheduler status and restart in case of scheduler inactivity. We could not identify the problems related to portal and we re-installed the portal to fix this problem.

Physical problems, as hardware failures (disk, memory, network failures, etc.), depend on human intervention for correction and are harder to address.

Grid monitoring can help the identification of failures and speed up correction, by detecting the exact failure point. Linux utilities or free software monitoring tools could collect system information and publish them at the web portal. Procedures with problems and solutions help human action in restoring the grid environment. Mechanisms for isolating a defective cluster node can also be used to minimize the problem.

7 Conclusion

This paper investigated the challenges of executing the mesoscale climatology, an important meteorological application that requires intensive use of computational resources in a processing grid environment. We explored the application and grid environment limitations for real-world conditions, since our interest includes its use on production.

We executed the climatology in three different computational grids, configured with OurGrid, OAR/CiGRI and Globus middlewares. We used the BRAMS model, configured to compute the climatology of three different regions of Brazil, using three HPC clusters geographically separated. Using a closed and dedicated grid environment, accessed by a web portal, we guarantee the application execution on a minimum time.

The 3-year climatology execution generated the expected results in 20.5, 22.3 and 33.3 days, using the computational grid configured with OurGrid, Globus and OAR/CiGri grid middlewares, respectively.

The grid middleware that introduced the lowest overhead was OurGrid, followed by OAR/Cigri and Globus. The performance difference between the HPC clusters architectures is 1.5 %.

During the long period of execution, we observed a considerable delay on the climatology generation. As a consequence, the computational grid stayed idle for 4.7, 10.1, e 17.4 days, for OurGrid, Globus and OAR/CiGri grid middlewares, respectively.

One of the causes was the data transmission time, which are higher and with more variability on farther than on nearer grid nodes. The hardware/software failures also increase job execution time. Depending on the failure, the job is scheduled automatically to another grid node. However, some cases require human intervention.

The computational grid status monitoring plays a fundamental role, since it detects failures in the system. Failures can be solved by human actuation under well-defined procedures; or without human actuation, in a fault-tolerance designed system.

Transmission time affects directly the application performance, with relation to physical location of the grid node. Best case showed that the communication time could represent 1 % of computation time while worst case can achieve 12 %.

The strategy for the distribution of jobs worked as expected, but increased the execution time of climatology because the checkpoint and data files require movement to a free grid node. The number of failures occurred for climatology generation is acceptable for an operational seasonal forecasting.

Acknowledgments This project has been supported by FINEP under contract CT-INFO Edital Grade-01/2004.

References

1. Lynch P (2008) The origins of computer weather prediction and climate modeling. J Comput Phys 227(7):3431–3444
2. Li S, Goddard L, DeWitt DG (2008) Predictive skill of AGCM seasonal climate forecasts subject to different SST prediction methodologies. J Climate 21:2169–2186
3. Palmer T, Andersen U, Cantelaube P, Davey M, Deque M, Doblas-Reyes FJ, Feddersen H, Graham R, Gualdi S, Gueremy JF, Hagedorn R, Hoshen M, Keenlyside N, Latif M, Lazar A, Maisonnave E, Marletto V, Morse AP, Orfila B, Rogel P, Terres JM, Thomsen MC (2004) Development of a European multi-model ensemble system for seasonal to inter-annual prediction (DEMETER). Bull Am Meteorolog Soc 85(6):853–872. doi:10.1175/BAMS-85-6-853
4. Strazdins PE, Kahn M, Henrichs J, Pugh T, Rezny M (2011) Profiling methodology and performance tuning of the met office unified model for weather and climate simulations. In: 25th IEEE International Parallel and Distributed Processing Symposium (PDESC-11 Workshop). IEEE, Anchorage
5. Sadjadi SM, Fong L, Badia R.M., Figueroa J, Delgado J, Collazo-Mojica X.J., Saleem K, Rangaswami R, Shimizu S, Duran Limon HA, Welsh P, Pattnaik S, Praino A, Villegas D, Kalayci S, Dasgupta G, Ezenwoye O, Martinez JC, Rodero I, Chen S, Muñoz J, Lopez D, Corbalan J, Willoughby H, McFail M, Lisetti C, Adjouadi M (2008) Transparent grid enablement of weather research and forecasting. In: Proceedings of the 15th ACM Mardi Gras conference, Baton Rouge, EUA

6. Foster I, Kesselman C (1999) The grid: Blueprint for a new computing infrastructure. Morgan Kaufmann, San Francisco
7. Souto RP, Ávila RB, Navaux PAO, Py M, Maillard N, Diverio T.A., Campos Velho HF, Stephany S, Preto AJ, Panetta J, Rodrigues E, Almeida ES (2007) Processing mesoscale climatology in a grid environment. In: Proceedings of CCGRID'07, Rio de Janeiro
8. Campos Velho H, Preto A, Stephany S, Rodrigues E, Panetta J, Almeida E, Souto R, Navaux P, Diverio T, Maillard N, Dias PS (2006) Grid computing for mesoscale climatology: experimental comparison of three platforms. In: Proceedings of the 7th international meeting on high performance computing for computational science (VECPAR' 06), Rio de Janeiro
9. Cirne W, Brasileiro F, Paranhos D, Costa L, Santos-Neto E, Osthoff C (2005) Building a user-level grid for bag-of-tasks applications. In: High performance computing: paradigm and infrastructure, chapter 37. Wiley, Hoboken
10. Capit N, Costa GD, Georgiou Y, Huard G, Martin C, Neyron GMP, Richard O (2005) A batch scheduler with high level components. Proceedings of the international symposium on cluster computing and the grid (CCGRID 2005), ACM/IEEE, IEEE Computer Society. Cardiff, pp 776–783
11. Foster I (2005) Globus toolkit version 4: Software for service oriented systems. Proceedings of the IFIP international conference on network and parallel computing, vol 3779 of Lecture Notes in Computer Science. Springer, Beijing, pp 2–13
12. Dabrowski C (2009) Reliability in grid computing systems. Concurr Comput Pract Exp 21:927–959. doi:10.1002/cpe.1410
13. Hofer J, Fahringer T (2008) A multi-perspective taxonomy for systematic classification of grid faults. 16th Euromicro conference on parallel, distributed and network-based processing, PDP 2008, pp 126–130, 13–15 Feb 2008. doi:10.1109/PDP.2008.15
14. Plale B, Droegemeier KK, Mattocks C (2010) The LEAD gateway II: a hardened, persistent community resource for meteorological research and education. In: 26th conference on interactive information and processing systems
15. Droegemeier KK, Chandrasekar V, Clark R, Gannon D, Graves S, Joseph E, Ramamurthy M et al (2004) Linked environments for atmospheric discovery (LEAD): a cyberinfrastructure for mesoscale meteorology research and education. In 20th conference on interactive information processing systems for meteorology, oceanography, and hydrology
16. Kandaswamy G, Anirban M, Daniel AR (2008) Fault tolerance and recovery of scientific workflows on computational grids. In: 8th IEEE international symposium on Cluster computing and the grid. CCGRID'08. pp 777–782
17. Mattocks C, Droegemeier KK, Wilhelmson RB (2009) Integration of LEAD and WRF portal technologies to enable advanced research, operations and education in mesoscale meteorology. In: 23rd conference on weather analysis and forecasting/19th conference on numerical weather prediction
18. Medeiros R, Cirne W, Brasileiro F, Sauvé J (2003) Faults in grids: why are they so bad and What can be done about it? Proceedings of the 4th international workshop on grid Computing, 17 Nov 2003
19. Dai Y, Dongarra J (2012) Reliability and performance models for grid computing. grid and cloud computing: concepts, methodologies, tools and applications. IGI Global, pp 119–140. doi:10.4018/978-1-4666-0879-5.ch803
20. Kola G, Kosar T, Livny M (2005) Faults in large distributed systems and what we can do about them. Euro-Par'05: Parallel processing (Lecture Notes in Computer Science, vol 3648). Springer, Berlin, pp 442–453
21. Sander V, Allcock W, CongDuc P, Monga I, Padala P, Tana M et al. (2004) Networking issues for grid infrastructure. Group
22. Fernandez-Quirelas V, Fernandez J, Baeza C, Cofino AS, Gutierrez JM (2009) Execution management in the GRID, for sensitivity studies of global climate simulations. Earth Sci Inf 2(1–2):75–82. doi:10.1007/s12145-008-0018-z
23. Stainforth D, Kettleborough J, Allen M, Collins M, Heaps A, Murphy J (2002) Distributed computing for public-interest climate modeling research. Comput Sci Eng 4:82–89
24. Lagouvardos K, Floros E, Kotroni V (2010) A grid-enabled regional-scale ensemble forecasting system in the Mediterranean. J Grid Comput 8:181–197
25. Kotroni V, Floros E, Lagouvardos K, Pejanovic G, Ilic L, Zivkovic M (2010) Multi-model multi-analysis ensemble weather forecasting on the Grid for the South Eastern Mediterranean Region. Earth Sci Inf 3:209–218
26. Fernández-Quiruelas V, Fernández J, Cofiño AS, Fita L, Gutiérrez JM (2011) Benefits and requirements of grid computing for climate applications. An example with the community atmospheric model. Environ Model Softw 26(9):1057–1069
27. Pielke R, Cotton W, Walko R, Tremback C, Lyons W, Grasso L, Nicholls M, Moran M, Wesley D, Lee T, Copeland J (1992) A comprehensive meteorological modeling system—RAMS. Meteorol Atmos Phys 49(1–4):69–91
28. Tremback CJ, Walko RL (1995) The regional atmospheric modeling system (RAMS): development for parallel processing computer architectures. In: 3rd RAMS Users' Workshop, Echuca
29. Walko R, Tremback C, Hertenstein R (1995) RAMS—The regional atmospheric modeling system—version 3b—users guide. ASTER Division, Fort Collins
30. Barros SRM (1998) Towards the RAMS-FINEP parallel model: load balancing aspect, towards teracomputing. Proceedings of the 8th ECMWF—workshop on the use of parallel processors in meteorology, Reading
31. Mendonça AM, Bonatti JP (2002) O sistema de previsão de tempo por ensemble do CPTEC. In: Anais do XII Congresso Brasileiro de Meteorologia, Foz do Iguaçú
32. Voorsluys W, Araújo E, Cirne W, Galvão CO, Souza EP, Cavalcanti EP (2007) Fostering collaboration to better manage water resources. Concurr Comput Pract Exper 19:1609–1620. doi:10.1002/cpe.1118
33. Brasileiro F, Araujo E, Voorsluys W, Oliveira M, Figueiredo F (2007) Bridging the high performance computing gap: the ourgrid experience. Seventh IEEE international symposium on cluster computing and the grid, 2007. CCGRID 2007, pp 817–822. doi:10.1109/CCGRID.2007.28
34. Georgiou Y, Richard O, Capit N (2007) Evaluations of the lightweight grid CiGri upon the Grid5000 platform. In: Proceedings of the Third IEEE international conference on e-science and grid computing (E-SCIENCE '07). IEEE Computer Society, Washington, DC, pp 279–286. doi:10.1109/E-SCIENCE.2007.32